Vrije Universiteit Brussel

KONINKLIJKE MILITAIRE SCHOOL

FACULTEIT INGENIEURSWETENSCHAPPEN
Departement Elektronica en Informatica

POLYTECHNISCHE FACULTEIT
Departement Mechanica

VARIATIONAL METHODS FOR DENSE DEPTH RECONSTRUCTION FROM MONOCULAR AND BINOCULAR VIDEO SEQUENCES

Proefschrift voorgelegd voor het behalen van de graad van
Doctor in de ingenieurswetenschappen door

Geert De Cubber

Promotoren: prof. H. Sahli
Prof. Y. Baudoin

March 2010

VUBPRESS

VRIJE UNIVERSITEIT BRUSSEL
FACULTY OF ENGINEERING
Department of Electronics and Informatics (ETRO)
Image Processing and Machine Vision Group (IRIS)

Variational methods for dense depth reconstruction from monocular and binocular video sequences

Thesis submitted in fulfilment of the requirements for the award of the degree of
Doctor in de ingenieurswetenschappen (Doctor in Engineering)
by

Geert De Cubber

Address
Pleinlaan 2, B-1050 Brussels, Belgium
Tel: +32-2-6292858
Fax: +32-2-6292883
Email: gdcubber@etro.vub.ac.be

Brussels, March 2, 2010

Print: Silhouet, Maldegem

VUBPRESS is an imprint of ASP nv
(Academic and Scientific Publishers nv)

Ravensteingalerij 28

B-1000 Brussels

Tel. ++32 (0)2 289 26 50

Fax ++32 (0)2 289 26 59

E-mail: info@vubpress.be

www.vubpress.be

ISBN 978 90 5487 710 3
NUR 958 / 984 / 919
Legal deposit D/2010/11.161/031

Acknowledgment

A PhD. work doesn't just happen in isolation. This research project could not have been terminated successfully or the results would have looked quite different without the kind contributions of many people whom I cannot thank enough for helping me solve the numerous problems encountered while pursuing this PhD.

First of all, I have to be grateful to my promotors, Prof. Hichem Sahli of VUB-ETRO and Prof. Yvan Baudoin of RMA-MECA. It has always been my dream to perform scientifically valuable research and apply this to real-life systems. They made this possible by giving me the opportunity to investigate an open research question within the computer vision domain, namely dense robust 3D reconstruction from moving camera images, and to seek for applications in the domain of intelligent mobile robotics.

For all theoretical aspects, I could always rely on the extensive knowledge of Prof. Hichem Sahli. His investigative spirit showed me the way to do scientific research, while the countless corrections to my work he provided me, helped me to improve my work.

For the practical aspects, I have to show great gratitude to Prof. Yvan Baudoin who gave me the opportunity to work with a whole range of mobile robotic systems to apply partial aspects of my fundamental research.

Seeking the subtle balance between theory and practice has been a crucial aspect of this PhD. work, which has sparked lively discussions between my promotors and myself. However, it is thanks to these discussions that the overall PhD. work has reached a level of balance between theory and practice which I'm very happy with.

Newton already noted "*If I have seen a little further it is by standing on the shoulders of Giants*". This applies also in the case of this dissertation. The proposed methodologies depend heavily on previous research work, most notably the dense optical flow estimation algorithms developed by Dr. Lixin Yang and Dr. Valentin Enescu. Without the technology they developed, I could not have been able to achieve the presented results.

During the years, a lot of colleagues crossed my path, both at the ETRO department at the VUB and at the Mechanics department at the RMA. I want to grasp this occasion to express my gratitude to you all. I've always been someone who enjoyed going to work and this is mainly because of the fine working atmosphere and spirit you all provided. In this context, a special thank you must be extended to Dr. Eric Colon, who made it possible for me to work on my PhD. project, even though this meant that I couldn't focus on other tasks and projects as may be required. Furthermore, I want to specifically thank all researchers who had the tough luck of sharing an office with me: Dr. Joeri Barbarien, Dr. Fabio Verdicchio, Dr. Thomas Geerinck and, the unluckiest of them all, Sid Ahmed Berrabah.

As with most Ph.D. projects, the time and effort put into this on has often exceeded normal office hours by quite a lot actually. This has naturally drawn on the people closest to me for their patience and understanding. In this regard a special thanks goes to my parents. As in all of my endeavors, my parents have been a constant source of support and encouragement.

But most of all, I have to express my infinite gratitude to my beloved Daniela, who suffered the most from this PhD., due to the time we couldn't spend together. For years, we organized our lives in function of this PhD., which asked a tremendous effort from her part, but she stood by me and supported me until the end. A final thank you must be extended to little Alessandra, whose smile gave me the energy to continue and whose eager fingers were always ready to help me typing on the laptop, although that wasn't always very productive.

Brussels, March 2010
Geert De Cubber

Contents

List of Figures

List of Tables

List of Algorithms

List of notations

In this text, vectors are denoted with bold lower-case letters. Matrices are denoted with bold capital letters. Images represent a spatial distribution of intensity in a two-dimensional plane. Mathematically this can be denoted as function of two spatial variables. To keep equations clear, sometimes indices and dependencies are not included. The most important symbols are listed below.

Operators

- $*$: Convolution.
- $^+$: Pseudo-Inverse.
- $\times$: Vector product.
- $[\mathbf{v}]_\times$: Skew-symmetric matrix of vector $\mathbf{v}$.
- $Det()$: Determinant of a Matrix.
- $Trace()$: Trace of a Matrix.
- d_E: Euclidian distance measure.
- d_M: Mahalanobis distance measure.

Indices

- i: spatial index.
- j: spatial index.
- k: temporal index.

Scalars

- d: Depth.
- e_d: Epipolar distance.
- f: Focal length, eventually separately expressed in x and y direction as (f_x, f_y).

- r: Residual
- s: Pixel size, eventually separately expressed in x and y direction as (s_x, s_y).
- w: Weight.
- x: 2D x-coordinate.
- X: 3D x-coordinate.
- y: 2D y-coordinate.
- Y: 3D y-coordinate.
- Z: 3D z-coordinate.
- η: number of motion model parameters.
- γ: scalar parameter.
- κ: structural dimension.
- λ_n: n^{th} eigenvalue.
- ν_n: Parameter in parametric equation.
- σ: Standard deviation or scale factor.
- υ: Data dimension.
- ζ: Homogeneous parameter.

Vectors

- $\mathbf{c}$: Camera centre point in 2D (Homogeneous 3-vector).
- $\mathbf{c}$: Camera centre point in 3D (Homogeneous 4-vector).
- $\mathbf{e}$: Epipole (homogeneous 3-vector).
- $\mathbf{f}$: Vectorized form of the Fundamental matrix (9-vector).
- $\mathbf{l}$: Line vector (3-vector).
- $\mathbf{m}$: Inhomogeneous image coordinates (2-vector).
- $\mathbf{M}$: Inhomogeneous 3D coordinates (3-vector).
- $\mathbf{t}$: Translation 3-vector.
- $\mathbf{u}$: Optical Flow.
- $\mathbf{v}$: Unspecified vector.
- $\mathbf{x}$: Homogeneous image coordinates (3-vector).

- $\mathbf{X}$: Homogeneous 3D coordinates (4-vector).
- ω: Rotation 3-vector.
- π: Plane 3-vector.

Matrices

- $\mathbf{0}_{m\times n}$: $m \times n$ Zero matrix.
- $\mathbf{H_C}$: 2×2 Harris image descriptor matrix.
- $\mathbf{F}$: 3×3 Fundamental matrix.
- $\mathbf{H}$: 2×2 Hessian matrix.
- $\mathbf{H}_\pi$: 3×3 2D Homography matrix.
- $\mathbf{I}_{m\times n}$: $m \times n$ Identity matrix.
- $\mathbf{K}$: 3×3 Camera Calibration Matrix.
- $\mathbf{M}_\pi$: 3×3 Matrix of planes.
- $\mathbf{P}$: 3×4 Camera matrix.
- $\mathbf{Q_t}$: 2×3 Matrix relating the translation to the translational part of the optical flow.
- $\mathbf{Q}_\omega$: 2×3 Matrix relating the rotation to the rotational part of the optical flow.
- $\mathbf{R}$: 3×3 Rotation matrix.
- $\mathbf{T}$: 4×4 Transformation matrix.
- $\mathbf{\Sigma}$: $m \times n$ Covariance matrix.

Functions

- A: Amplitude function.
- c: Harris auto-correlation function.
- C: Cross-correlation function.
- D: Lowe's Difference-of-Gaussian function.
- E: Energy Function.
- $G(x, y, \sigma)$: Variable-scale Gaussian.
- I: Image.
- I_x: Partial derivative of the Image in x.

- I_y: Partial derivative of the Image in y.
- L: Scale-space function.
- m_g: Gradient magnitude function for the SIFT-descriptor.
- PC: Phase Congruency measure.
- s: Fourier expansion of a series of waveforms.
- ρ: Error function.
- θ_g: Gradient orientation function for the SIFT-descriptor.

Spaces

- $\mathbb{P}^n$: Projective space.
- $\mathbb{R}^n$: Euclidean space.

Other symbols

- $\mathcal{T}$: Trifocal tensor.

Abstract

This research work tackles the problem of dense three-dimensional reconstruction from monocular and binocular image sequences. Recovering 3D-information has been in the focus of attention of the computer vision community for a few decades now, yet no all-satisfying method has been found so far. The main problem with vision, is that the perceived computer image is a two-dimensional projection of the 3D world. Three-dimensional reconstruction can thus be regarded as the process of re-projecting the 2D image(s) back to a 3D model, as such recovering the depth dimension which was lost during projection.

In this work, we focus on *dense* reconstruction, meaning that a depth estimate is sought for each pixel of the input image. Most attention in the 3D-reconstruction area has been on *stereo*-vision based methods, which use the displacement of objects in two (or more) images. Where stereo vision must be seen as a spatial integration of multiple viewpoints to recover depth, it is also possible to perform a temporal integration. The problem arising in this situation is known as the *Structure from Motion* problem and deals with extracting 3-dimensional information about the environment from the motion of its projection onto a two-dimensional surface. Based upon the observation that the human visual system uses both stereo *and* structure from motion for 3D reconstruction, this research work also targets the combination of stereo information in a structure from motion-based 3D-reconstruction scheme. The data fusion problem arising in this case is solved by casting it as an energy minimization problem in a *variational framework*.

To the women of my life, Daniela & Alessandra

Part I

Introduction to the state of the art

Chapter 1

Introduction to structure from motion and its applications

1.1 Visual 3D Perception

The physical world can be regarded as a three-dimensional geometric space. The dimensions constituting this space are generally called the length, width and height. This three-dimensional world is observed/perceived by humans by means of their five senses: hearing, touch, smell, taste and vision. Of all these sensing modalities, vision is the most powerful, which is shown by the fact that it occupies the most cortical space. The physics of vision is relatively well understood: properties of light and lenses and photo-receptors in the retina have been studied intensively in the last century. Even early stages of image representation in terms of neural signals leaving the retina have plausible theories which allow some insight into the information collected by the eyes [88]. However, once higher level representations are considered, the situation becomes less clear. How is visual information coded? How is attention focussed? How does contextual (a priori) information affect the raw information coming to the brain from the retina? How do we recognize objects? These are all active areas of vision research.

One important area of vision research concerns our ability to understand the three-dimensional nature of our environment. Indeed, the human eye can be regarded as a two-dimensional imaging device, meaning that the resulting image is (only) a 2D representation/projection of the 3D world. To recover the depth dimension which was lost during projection on the retina, the human visual system fuses multiple depth cues. These depth cues are grouped into two categories: monocular cues (cues available from the input of just one eye), binocular cues (cues that require input from both eyes) and motion cues. The monocular cues include:

- *Relative size*: Retinal image size allow us to judge distance based on our past and present experience and familiarity with similar objects. Remark on Figure 1.1 that background figures (the nudes) are pictured smaller than the foreground figures (the holy family), suggesting they are positioned further away.

- *Interposition* or *occlusion*: Interposition cues occur when there is overlapping of objects. The overlapped object is considered further away. Remark on Figure 1.1 that the foreground figures overlap the background figures, suggesting they are nearer to the viewer.

- *Perspective*: When objects of known distance subtend a smaller and smaller angle, they are interpreted as being further away. Parallel lines converge with increasing distance such as roads, railway lines, electric wires, etc. Remark on Figure 1.1 that the background figures are sitting on a bench-like structure. The convergence pattern of this bench gives the impression that it is further away from the viewer in the middle of the image and closer to the viewer at the extreme left and right.

- *Focus*: The lens of the eye can change its shape to bring objects at different distances into focus. Knowing at what distance the lens is focused when viewing an object means knowing the approximate distance to that object. Remark on Figure 1.1 that the foreground figures have very sharp details, while the background is slightly out of focus.

- *Light* and *shading*: Highlights and shadows can provide information about an object's dimensions and depth. Because our visual system assumes the light comes from above, a totally different perception is obtained if the image is viewed upside down. Remark on Figure 1.1 that the painter used lively shadows, notably on the clothes, to enhance the depth perception.

- *Color*: Relative color of objects also gives some clues to their distance. Due to the scattering of blue light in the atmosphere, distant objects appear more blue. Remark on Figure 1.1 that the mountains in the background are bluish.

- *Motion parallax*: The apparent relative motion of several stationary objects against a background when the observer moves gives hints about their relative distance.

- *Kinetic depth*: Movement of the observer causes objects that are close to the observer to move rapidly across the retina. However, objects that are far away move very little. In this way, the brain can tell roughly how far away these objects are.

The major binocular cue for depth perception is *stereopsis*. Because the eyes are about 5 cm apart, each eye sees a slightly different image of the world. By fusing both images, the brain is capable of inferring depth information.

Figure 1.1: Depth perception in art: The Holy Family with the infant St. John the Baptist (the Doni tondo) by Michelangelo [21]. The artist uses multiple monocular depth cues to re-create the illusion of depth in this painting.

The computer vision community has been researching for a few decades now analogous methodologies mimicking the depth perception abilities of the human visual system. Most attention has hereby been focussed on the stereopsis-based depth cue. The result is that depth-from-stereo has now evolved to a mature research subject. We refer the reader to the seminal paper [124] of Scharstein and Szeliski for a broad taxonomy on stereo algorithms, but in general, it can be noted that the current state of the art research in stereo vision is focussed on two topics: one seeking to maximize the quality of the resulting reconstruction and one seeking to minimize the algorithm execution speed. The first research direction has lead to algorithms showing high-quality dense stereo reconstructions [117], [167], with processing times of typically about 30 seconds per frame, whereas the latter research direction has resulted in (near) real-time algorithms [177], providing quasi-dense reconstruction results.

The transposition of human monocular depth vision skills to computer vision has achieved less attention than the stereopsis case. This can be partly explained by the fact that most of these approaches require some higher level processing and reasoning, which is not straightforward and, generally, not well understood in the human visual system as well. Nevertheless, considerable research work has been done in the areas of *depth from shading*, *depth from (de)focus* and *depth from interposition*. Despite their merits, it is unlikely that, except in some limited domains, these approaches will ever seriously rival stereo or motion-based sources of depth information [181]. Motion-based depth reconstruction approaches like *motion parallax* and *kinetic depth* present a more promising research direction. Where stereo vision must be seen as a spatial integration of multiple viewpoints to recover depth, motion-based depth reconstruction can be seen as performing a temporal integration. The problem arising in this situation is known as the *Structure from Motion* problem and deals with extracting three-dimensional information about the environment from the motion of its projection onto a two-dimensional surface. The recovery of depth through structure from motion is one of the main topics of this research work. Therefore, a short introduction to structure from motion is given in the following section.

1.2 Structure from Motion - based Reconstruction approaches

In general, there are two approaches to structure from motion. The first, feature based method is closely related to stereo vision. It uses corresponding features in multiple images of the same scene, taken from different viewpoints. The basis for feature-based approaches lies in the early work of Longuet-Higgins [82], describing how to use the epipolar geometry for the estimation of relative motion. In this article, the 8-points algorithm was introduced. It features a way of estimating the relative camera motion, using the essential matrix, which constrains feature points in two images. The first problem with these feature based techniques is of course the retrieval of correspondences, a problem which cannot be reliably solved in image areas with low texture. From these correspondences, estimates for the motion vectors can be calculated, which are then used to recover the depth. An advantage of feature based techniques is that it is relatively easy to integrate results over time, using bundle adjustment [155] or Kalman filtering [3]. Bundle adjustment is a maximum likelihood estimator that consist in minimizing the re-projection error. It requires a first estimate of the structure and then adjusts the bundle of rays between each camera and the set of 3D points.

The second approach for structure from motion uses the optical flow field as an input instead of feature correspondences. Optical flow is the distribution of apparent velocities of movement of brightness patterns in an image. Optical flow can arise from relative motion of objects and the viewer. Consequently, optical flow can give important information about the spatial arrangement of the objects viewed and the rate of change of this arrangement. The applicability of the optical flow field for structure from motion calculation originates from the epipolar constraint equation which relates the optical flow to the relative camera motion and 3D structure in a non-linear fashion.

In [47], Hanna proposed a method to solve the motion and structure reconstruction problem by parameterizing the optical flow and inserting it in the image brightness constancy equation. More popular methods try to eliminate the depth information first from the epipolar constraint and regard the problem as an egomotion estimation problem. Bruss & Horn already showed this technique in the early eighties using substitution of the depth equation [19], while Jepson & Heeger later used algebraic manipulation to come to a similar formulation [55].

The current state-of-the art in structure from motion systems mainly considers the construction of sparse feature-based scene representations, e.g. from points and lines. The main drawback of such systems is the lack of surface information, which restricts their usefulness, as the number of features is limited. In the past, optical flow - based structure from motion methods such as the famous Bruss & Horn [19] and Jepson & Heeger [55] algorithms were also mainly aimed at motion and structure recovery using very low resolution optical flows. With the increase in available processing power, however, the structure from motion community is now trying to address the dense reconstruction problem. The optical flow based structure from motion approaches are more suited to address the dense reconstruction problem, as they use the optical flow over the whole image field.

When reviewing the individual depth cues, as done in section 1.1, one must not forget that the human visual system uses all the depth cues discussed there *concurrently.* This indicates that robust and accurate depth perception requires a combination of methods rather than a sole one. This constatation inspired researchers to work on integrated three-dimensional reconstruction approaches and this is also one of the main focus points of this dissertation: *How to fuse a monocular depth cue like structure from motion with a binocular depth cue like stereo?*

1.3 Applications of 3D Reconstruction

The potential applications of the presented three-dimensional reconstruction approaches are widespread.

The movie and entertainment industry more and more relies on 3D computer graphics, requiring 3D models of real-world objects, scenes and actors to be placed in a computer generated world. Automated 3D reconstruction of natural scenes could greatly speed up this modeling process. Another interesting application in this context is presented by the advent of 3D television. Three dimensional television presents the viewer with a realistic 3D image of the television show, by using a 3D display technique. 3D television is in fact not a recent evolution, already in the beginning of the past century, 3D movies were recorded. The main problem with all efforts towards 3D television in the past century was that specialized glasses were required to achieve depth perception. These glasses see to it that both eyes receive a different image, which tricks the human brain into recreating a sense of depth according to the *stereopsis* depth cue explained before. As only one of the multiple human depth cues was triggered by this display technique, this often led to *motion sickness* for the viewers, as the different depth sensors receive contradicting signals. However, due to the improvement in digital display technology, it is now possible to develop 3D screens which do not require specialized glasses. This technology advancement makes it possible to deliver 3D content to a mass public and it can be envisaged that in the future all televisions will have standard 3D functionality. This creates a new problem of course for the content creation. For new content, specialized 3D equipment using stereoscopic cameras is probably the best solution. However, for all the existing film material, a solution will be required too. The automated calculation of dense depth maps from monocular input data, as discussed in this work, can here be used to convert legacy film and video material directly into 3D.

Another application domain is the field of augmented reality and human computer interaction, where automated 3D reconstruction could play a crucial role to enhance the human computer interfacing. Commercial applications here range from real object modeling e.g. for architectural purposes, to novel view generation algorithms, e.g. for enhancing the viewing experience of sporting events.

Currently, there is a wide range of research and applications for 3D reconstruction in the field of medical imaging and biometrics, with the purpose of accurately modeling the internal body organs or external body state. The premise for these medical applications is almost completely different than the one in our work: in general there is an a priori model present, the object to be modeled is relatively small and under control and often special imaging tools are used. As a result, the methodologies and results presented in this dissertation are not the best suited to be used in the field of medical imaging.

Another application field for 3D reconstruction is robotics. Indeed, in order to understand and reason about its environment, an intelligent robot needs to be aware of the three-dimensional status of this environment. Contemporary autonomous robots are therefore generally equipped with an abundance of sensors like for example Laser, ultrasound sensors, etc to be able to navigate in an environment. However, this stands in contrast to the ultimate biological example for these robots: us humans. Indeed, humans seem perfectly capable to navigate in a complex, dynamic environment using primarily vision as a sensing modality. With the advance in computer processing power, 3D reconstruction becomes increasingly available to the field robotics, where not only the quality of the end result is important, but also the required processing time.

1.4 Research Objectives

In light of the previous work done in the field of 3D reconstruction, the main objective of this research work is to develop a dense structure from motion recovery algorithm. This algorithm should operate on monocular image sequences, using the camera movement as a main depth cue. The term *dense* indicates that a depth estimate for each pixel of the input images is required. We will show that an iterative variational technique is able to solve this 3D reconstruction problem. However, to converge to a solution, the iterative technique requires proper initialization. For this initialization process, standard sparse structure from motion techniques are employed. These classical methods are not capable of estimating a dense reconstruction, but they do suffice to estimate the camera motion parameters and the 3D positions of some feature points, which is used as an initial value for the iterative solver.

Dense structure from motion is a promising technology, but suffers from one major disadvantage: the processing time is in general very long. This is the case for most of the existing approaches and it is no different in our work. This drawback makes it less suited for a number of applications where the timely delivery of results is an issue (e.g. robotics). To deal with these issues, a second research objective was to integrate the developed dense structure from motion algorithm into a stereo reconstruction context. In this way, the processing time could be drastically reduced (although this methodology is algorithmically more complex), as stereo adds a valuable constraint, which limits the search domain for solutions dramatically.

1.5 Main Contributions

As indicated in the previous section, the focus of this research work is dual:

1. *The development of a novel dense structure from motion approach.*

   We propose an approach which fuses sparse and dense information in an integrated variational framework. The aim of this approach is to combine the robustness of traditional sparse structure from motion methods with the completeness of optical flow based dense reconstruction approaches.

   The base constraint of the variational approach is the traditional image brightness constraint, but parameterized for the depth using the 2-view geometry. This estimation of the geometry, as expressed by the fundamental matrix, is automatically updated at each iteration of the solver. A regularization term is added to ensure good reconstruction results in image regions where the data term lacks information. An automatically updated regularization term ensures an optimal balance between the data term and the regularization term at each iteration step.

   A semi-implicit numerical scheme was set up to solve the dense reconstruction problem. The solver uses an initialization process which fuses optical flow data and sparse feature point matches.

2. *The development of a novel dense reconstruction method, combining stereo and structure from motion in an integrated framework.*

   We propose a stereo - motion reconstruction technique which combines stereo and motion data in an integrated framework. This technique uses the theorem of the Augmented Lagrangian to integrate stereo and motion constraints. The presented methodology is compared to a more classical global optimization technique.

An important aspect of our work and a differentiating factor with respect to similar research work is that we specifically target the reconstruction of outdoor scenes. This poses extra difficulties due to the unstructured, unbounded and complex nature of the terrain, the difficult lighting conditions, ... Other research on 3D reconstruction is mostly focused on indoor (lab) environments, which are small and nicely controlled.

1.6 Structure of this Dissertation

In order to enhance the readability, this dissertation is subdivided into three main parts.

The first part is essentially an introduction to the used methodologies and an extended state of the art, aiming at familiarizing the reader with the developed algorithms. The first chapter introduces the problem of dense reconstruction through structure from motion and points out the main contributions and applications of this research work. In the second chapter, an overview is given of the image formation process and some basic tools for image processing - required later in this dissertation - are introduced. The third and fourth chapter present an extended state of the art in the research fields of respectively sparse and dense structure from motion.

The second part of this document presents the monocular dense structure from motion algorithms which have been developed in the course of this PhD. work. The fifth chapter presents and discusses the methodologies and algorithms for monocular dense structure from motion and chapter six analyzes the results offered by these algorithms.

The third part of this document is devoted to the combination of stereo and structure from motion in an integrated framework. Chapter seven reviews the current state of the art techniques for the integration of multiple depth cues. Chapter eight presents the integrated framework for binocular dense structure from motion developed in the course of this PhD. work, while the results of this methodology are analyzed and discussed in chapter nine. Conclusions and closing remarks can be found in the final chapter ten.

Chapter 2

Basic Image Formation and Processing

2.1 Introduction

In this chapter, some basic computer vision concepts and tools are introduced. These concepts and tools are required to understand the algorithms introduced in the following chapters.

Section 2.2 explains how an idealized camera arrives at generating a 2D image of our 3D world. The perspective projection model introduced here defines the relationship between 3D world and 2D image data. Chapters 3 and 4 rely on this model to express sparse and dense 3D structure as a function of the information contained in the 2D image.

The problem of structure from motion can be re-stated as the problem of reconstructing a description of the geometry between multiple camera views. Once the geometry description is known, 3D structure and motion can be extracted accordingly. This is explained in section 2.3, where the projective geometry for multiple views is discussed.

Many image processing algorithms, notably sparse structure from motion algorithms as the ones presented in chapter 3, are based upon the analysis of the movement of salient image regions, called features. A basic requirement for this analysis is that these features can be detected, described and matched across different images. Section 2.4 introduces the selected algorithms for each of these processing steps.

Section 2.5 introduces the concept of optical flow, how it is related to the 3D structure and how it can be estimated. The dense reconstruction algorithms which are presented in parts 2 and 3 are all optical-flow based. Basically, they rely on the evaluation of the relationship between the 3D scene structure and 2D optical flow.

2.2 Image Formation

An image is created by projecting the 3D scene on a 2D image plane, which is a discrete and possibly highly discontinuous function (see appendix C). The drop from three-dimensional world to a two-dimensional image is a projection process in which one dimension is lost. The usual way for modeling this process is by central projection in which a ray from a point in space is drawn from a 3D world point through a fixed point in space, the center of projection. This ray will intersect a specific plane in space chosen as the image plane. The image plane is located at the distance of the focal length from the origin of the 3D axis along the Z-direction, and it is perpendicular to it. The complete scene is located at positive Z-ordinates and we view the image with viewing direction on negative Z-direction.

Let $I(x, y)$ be the image intensity at time t at the image point (x, y). The intersection of the ray with the image plane represents the image of the point. This model is in accordance with a simple model of a camera, in which a ray of light from a point in the world passes through the lens of a camera and impinges on a film or digital device, producing an image of the point. Ignoring such effects as focus and lens thickness, a reasonable approximation is that all the rays pass through a single point, the center of the lens.

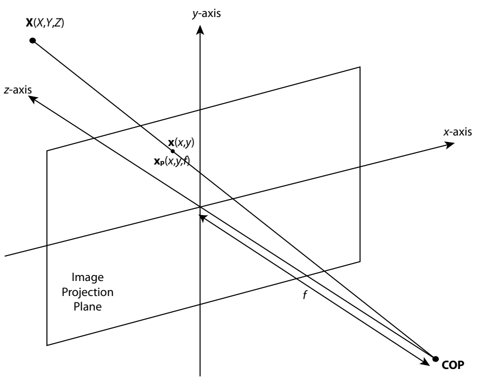


Figure 2.1: The Perspective Projection

In order to analyze the mapping process, it is advisable to first define the projective space $\mathbb{P}^n$. The Euclidean space $\mathbb{R}^n$ can be extended to a projective space $\mathbb{P}^n$ by representing points as homogeneous vectors. In this text, we denote the homogeneous counterpart of vector $\mathbf{x}$ as $\tilde{\mathbf{x}}$. A linear transformation of Euclidean space $\mathbb{R}^n$ is represented by matrix multiplication applied to the coordinates of the point. In just the same way a projective transformation of projective space $\mathbb{P}^n$ is a mapping of the homogeneous coordinates representing a point, in which the coordinate vector is multiplied by a non-singular matrix.

Central projection is then simply a mapping from $\mathbb{P}^3$ to $\mathbb{P}^2$. To describe this mapping, three coordinate systems need to be taken into account: the camera, image and world coordinate system.

Consider a point in $\mathbb{P}^3$ in the camera coordinate system, written in terms of homogeneous coordinates $\tilde{\mathbf{X}}_c(X_c, Y_c, Z_c, T)^T$, where T is the homogeneous parameter, introduced due to the switch to homogeneous coordinates. We can now see that the set of all points $\tilde{\mathbf{X}}(X, Y, Z, T)^T$ for fixed X, Y and Z, but varying T, form a single ray passing through the point center of projection. As a result, all these points map onto the same point, thus the final coordinate of $\tilde{\mathbf{X}}(X, Y, Z, T)^T$ is irrelevant to where the point is imaged. In fact, the image point is the point in $\mathbb{P}^2$ with homogeneous coordinates $\tilde{\mathbf{x}}_c(x_c, y_c, f)^T$, as defined by the projection equation

$$\begin{bmatrix} x_c \\ y_c \\ f \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}, \tag{2.1}$$

with f the focal length of the camera lens.

The mapping may thus be represented in its most simple form by a mapping of 3D homogeneous coordinates, represented by a 3×4 matrix $\mathbf{P}_0$ with the block structure $\mathbf{P}_0 = [\mathbf{I}_{3\times3} \,|\mathbf{0}_{3\times1}]$, where $\mathbf{I}_{3\times3}$ is the identity matrix and $\mathbf{0}_{3\times1}$ is a zero 3-vector.

In the image coordinate system, the mapping from $\tilde{\mathbf{x}}_c(x_c, y_c, f)^T$ to image coordinates is described. This mapping takes into account different centers of projection (x_0, y_0), non-square pixels and skewed coordinate axes. As such, it encompasses all the internal camera parameters. This mapping can be expressed in terms of matrix multiplication as:

$$\tilde{\mathbf{x}}_i = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & \alpha_x \cot(\theta) & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ f \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_c \\ y_c \\ f \end{bmatrix}, \tag{2.2}$$

where (x_0, y_0) is the coordinate of the principle point or image center, α_x and α_y denote the scaling in the x and y direction and θ is the angle between the axes, which is in general equal to $\pi/2$. The matrix $\mathbf{K}$ is an upper triangular matrix which provides the transformation between an image point and a ray in Euclidean 3-space. It encompasses all internal camera parameters and is called the *camera calibration matrix*. Throughout this work, we will assume that the cameras are calibrated, which means that $\mathbf{K}$ is known.

As a last step of projection, the description of the transformation between the camera and the world coordinate system is required. Changing coordinates in space is equivalent to multiplication by a 4×4 matrix:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}, \tag{2.3}$$

with $\mathbf{R}$ the rotation matrix and $\mathbf{t}$ the translation vector.

Concatenating the expressions 2.3, 2.2 and 2.1, it is clear that the most general image projection can be represented by an arbitrary 3×4 matrix of rank 3, acting on the homogeneous coordinates of the point in $\mathbb{P}^3$ mapping it to the imaged point in $\mathbb{P}^2$:

$$\tilde{\mathbf{x}}_i = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \mathbf{K}\,[\mathbf{R}|\,\mathbf{t}] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}. \tag{2.4}$$

It thus turns out that the most general imaging projection is represented by an arbitrary 3×4 matrix of rank 3, acting on the homogeneous coordinates of the point in $\mathbb{P}^3$ mapping it to the imaged point in $\mathbb{P}^2$:

$$\tilde{\mathbf{x}}_i = \mathbf{P}\tilde{\mathbf{X}}, \tag{2.5}$$

with:

$$\mathbf{P} = \mathbf{K}\,[\mathbf{R}|\,\mathbf{t}] \tag{2.6}$$

This matrix $\mathbf{P}$ is known as the camera matrix. It expresses the action of a projective camera on a point in space in terms of a linear mapping of homogeneous coordinates.

When returning to non-homogeneous coordinates for a camera based in the origin and ignoring non-square pixel aspect ratios (this means $\mathbf{P} = [\mathbf{I}_{3\times3}\,|\mathbf{0}_3\,]$), it can be observed that to map a 3D point $X = (X, Y, Z)$ to the image coordinates $\mathbf{x} = (x, y, f)$, the following perspective projection equations can be written:

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} = \frac{f}{Z} \begin{pmatrix} X \\ Y \end{pmatrix}, \tag{2.7}$$

in which x and y are the image coordinates. In order to reduce the complexity of some equations and for numerical stability, we can parameterize the depth by a proximity factor $d = \frac{1}{Z}$.

2.3 Multi-View Image Geometry

2.3.1 Two-View Geometry described by the Fundamental Matrix

The geometry between two views is called the *epipolar geometry.* This geometry depends on the internal parameters and relative position of the two cameras. The fundamental matrix $\mathbf{F}$ encapsulates this intrinsic geometry and was introduced by Faugeras in [38] and Hartley in [52]. It is a 3×3 matrix of rank 2. The fundamental matrix describes the relationship between matching points: if a point $\tilde{\mathbf{X}}$ is imaged as $\tilde{\mathbf{x}}$ in the first view, and $\tilde{\mathbf{x}}'$ in the second, then the image points must satisfy the relation $\tilde{\mathbf{x}}'^T\mathbf{F}\tilde{\mathbf{x}} = 0$. In this section, the epipolar geometry is described and the fundamental matrix is derived. The fundamental matrix is independent of scene structure. However, it can be computed from correspondences of imaged scene points alone, without requiring knowledge of the cameras internal parameters or relative pose.

To describe this mapping, first the geometric entities involved in epipolar geometry are introduced in figure 2.2. Here, the epipole $\tilde{\mathbf{e}}$ is the point of intersection of the line joining the camera centers (the baseline) with the image plane. Equivalently, the epipole is the image in one view of the camera center of the other view. It is also the vanishing point of the baseline (translation) direction. An epipolar plane is a plane containing the baseline. There is a one-parameter family of epipolar planes. An epipolar line is the intersection of an epipolar plane with the image plane. The epipolar line corresponding to $\tilde{\mathbf{x}}$ is the image in the second view of the ray back-projected from $\tilde{\mathbf{x}}$. Any point $\tilde{\mathbf{x}}'$ in the second image matching the point $\tilde{\mathbf{x}}$ must lie on the epipolar line $\tilde{\mathbf{l}}'$. All epipolar lines intersect at the epipole. An epipolar plane intersects the left and right image planes in epipolar lines, and defines the correspondence between the lines.

The mapping from a point in one image to a corresponding epipolar line in the other image may be decomposed into two steps. In the first step, the point $\tilde{\mathbf{x}}$ is mapped to some point $\tilde{\mathbf{x}}'$ in the other image lying on the epipolar line $\tilde{\mathbf{l}}'$. This point $\tilde{\mathbf{x}}'$ is a potential match for the point $\tilde{\mathbf{x}}$. In the second step, the epipolar line $\tilde{\mathbf{l}}'$ is obtained as the line joining $\tilde{\mathbf{x}}'$ to the epipole $\tilde{\mathbf{e}}'$. Figure 2.2 illustrates this mapping process.

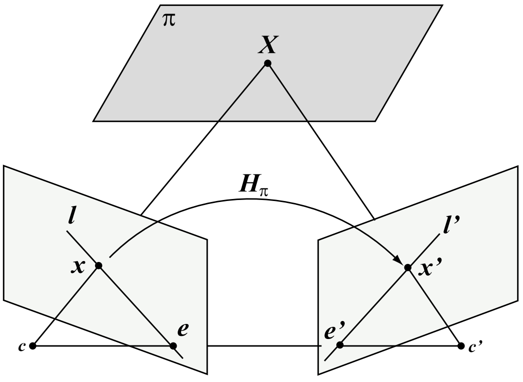


Figure 2.2: The mapping process from one camera according to the epipolar geometry

Consider a random plane π in space not passing through either of the two camera centers. The ray through the first camera center corresponding to the point $\tilde{\mathbf{x}}$ meets the plane $\boldsymbol{\pi}$ in a point $\tilde{\mathbf{X}}$. This point $\tilde{\mathbf{X}}$ is then projected to a point $\tilde{\mathbf{x}}'$ in the second image. This procedure is known as transfer via the plane $\boldsymbol{\pi}$. Since $\tilde{\mathbf{X}}$ lies on the ray corresponding to $\tilde{\mathbf{x}}$, the projected point $\tilde{\mathbf{x}}'$ must lie on the epipolar line $\tilde{\mathbf{l}}'$ corresponding to the image of this ray, as illustrated in 2.2. The points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ are both images of the 3D point $\tilde{\mathbf{X}}$ lying on a plane. The set of all such points $\tilde{\mathbf{x}}_i$ in the first image and the corresponding points $\tilde{\mathbf{x}}'_i$ in the second image are projectively equivalent, since they are each projectively equivalent to the planar point set $\tilde{\mathbf{X}}'$. Thus, there is a 2D homography $\mathbf{H}_{\boldsymbol{\pi}}$ mapping each $\tilde{\mathbf{x}}_i$ to $\tilde{\mathbf{x}}'_i$.

Given the point $\tilde{\mathbf{x}}'$, the epipolar line $\tilde{\mathbf{l}}'$ passing through $\tilde{\mathbf{x}}'$ and the epipole $\tilde{\mathbf{e}}'$ can be written as $\tilde{\mathbf{l}}' = \tilde{\mathbf{e}}' \times \tilde{\mathbf{x}}' = [\tilde{\mathbf{e}}']_\times \tilde{\mathbf{x}}'$ ($[\tilde{\mathbf{e}}']_\times$ being the skew-symmetric matrix form of $\tilde{\mathbf{e}}'$). Since $\tilde{\mathbf{x}}'$ may be written as $\tilde{\mathbf{x}}' = \mathbf{H}_{\boldsymbol{\pi}}\tilde{\mathbf{x}}$, we have:

$$\tilde{\mathbf{l}}' = [\tilde{\mathbf{e}}']_\times \mathbf{H}_{\boldsymbol{\pi}}\tilde{\mathbf{x}} = \mathbf{F}\tilde{\mathbf{x}}, \tag{2.8}$$

where we define $\mathbf{F} = [\tilde{\mathbf{e}}']_\times \mathbf{H}_{\boldsymbol{\pi}}$ as the fundamental matrix. Since $[\tilde{\mathbf{e}}']_\times$ has rank 2 and $\mathbf{H}_{\boldsymbol{\pi}}$ rank 3, $\mathbf{F}$ is a matrix of rank 2, which is logic as $\mathbf{F}$ represents a mapping from a 2-dimensional onto a 1-dimensional projective space.

The fundamental matrix satisfies the condition that for any pair of corresponding points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ in the two images

$$\tilde{\mathbf{x}}'^T\mathbf{F}\tilde{\mathbf{x}} = 0 \tag{2.9}$$

This is true, because if points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ correspond, then $\tilde{\mathbf{x}}'$ lies on the epipolar line $\tilde{\mathbf{l}}' = \mathbf{F}\tilde{\mathbf{x}}$ corresponding to the point $\tilde{\mathbf{x}}$. In other words $0 = \tilde{\mathbf{x}}'^T\tilde{\mathbf{l}}' = \tilde{\mathbf{x}}'^T\mathbf{F}\tilde{\mathbf{x}}$. If image points satisfy the relation $\tilde{\mathbf{x}}'^T\mathbf{F}\tilde{\mathbf{x}} = 0$ then the rays defined by these points are coplanar. This is a necessary condition for points to correspond. The importance of the relation 2.9 is that it gives a way of characterizing the fundamental matrix without reference to the camera matrices, i.e. only in terms of corresponding image points. This enables $\mathbf{F}$ to be computed from image correspondences alone.

2.3.2 Two-View Geometry described by the Essential Matrix

The essential matrix is the specialization of the fundamental matrix to the case of normalized image coordinates. Historically, the essential matrix was introduced by Longuet-Higgins in [82] before the fundamental matrix, and the fundamental matrix may be thought of as the generalization of the essential matrix in which the inessential assumption of calibrated cameras is removed. The essential matrix has fewer degrees of freedom, and additional properties, compared to the fundamental matrix.

Consider a camera matrix decomposed as $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$, and let $\tilde{\mathbf{x}} = \mathbf{P}\tilde{\mathbf{X}}$ be a point in the image. If the calibration matrix $\mathbf{K}$ is known, then we may apply its inverse to the point $\tilde{\mathbf{x}}$ to obtain the point $\widehat{\mathbf{x}} = \mathbf{K}^{-1}\tilde{\mathbf{x}}$. Then $\widehat{\mathbf{x}} = [\mathbf{R}|\mathbf{t}]\tilde{\mathbf{X}}$, where $\widehat{\mathbf{x}}$ is the image point expressed in normalized coordinates. It may be thought of as the image of the point $\tilde{\mathbf{X}}$ with respect to a camera $[\mathbf{R}|\mathbf{t}]$ having the identity matrix $\mathbf{I}$ as calibration matrix. The camera matrix $\mathbf{K}^{-1}\mathbf{P} = [\mathbf{R}\,|\,\mathbf{t}\,]$ is called a normalized camera matrix, the effect of the known calibration matrix having been removed. Now, consider a pair of normalized camera matrices $\mathbf{P} = [\mathbf{I}|\mathbf{0}]$

and $\mathbf{P}' = [\mathbf{R}|\mathbf{t}]$. The fundamental matrix corresponding to the pair of normalized cameras is customarily called the essential matrix and has the form:

$$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R} = \mathbf{R}\left[\mathbf{R}^T\mathbf{t}\right]_\times \tag{2.10}$$

The essential matrix can then be defined as:

$$\widehat{\mathbf{x}}'^T\mathbf{E}\widehat{\mathbf{x}} = 0 \tag{2.11}$$

in terms of the normalized image coordinates for corresponding points $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$. Substituting for $\widehat{\mathbf{x}}$ and $\widehat{\mathbf{x}}'$ gives $\tilde{\mathbf{x}}'^T\mathbf{K}'^{-1T}\mathbf{E}\mathbf{K}^{-1}\tilde{\mathbf{x}} = 0$. Comparing this with the relation $\tilde{\mathbf{x}}'^T\mathbf{F}\tilde{\mathbf{x}} = 0$ for the fundamental matrix, it follows that the relationship between the fundamental and essential matrices is:

$$\mathbf{E} = \mathbf{K}'^T\mathbf{F}\mathbf{K} \tag{2.12}$$

This relationship shows that once the camera calibration matrix $\mathbf{K}$ is known, the essential matrix can be calculated from the fundamental matrix.

The essential matrix holds all the information about the *external* calibration parameters: rotation and translation between the two camera frames.

$$\mathbf{E} = \mathbf{R}\,[\mathbf{t}]_\times\,, \tag{2.13}$$

with $[\mathbf{t}]_\times$ the skew-symmetric matrix form of the translation vector.

$$[\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{2.14}$$

Hartley introduced in [52] a method to decompose the essential matrix to find back the rotation matrix and translation vector using singular value decomposition and writing

$$\mathbf{E} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T, \tag{2.15}$$

where $\mathbf{\Lambda} = diag(\lambda, \lambda, 0)$. By defining the following two matrices:

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix},\ \mathbf{Z} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.16}$$

it is possible to write out the translation and rotation matrices:

$$[\mathbf{t}]_\times \approx \mathbf{U}\mathbf{Z}\mathbf{U}^T; \qquad \mathbf{R_1} \approx \mathbf{U}\mathbf{W}\mathbf{V}^T; \qquad \mathbf{R_2} \approx \mathbf{U}\mathbf{W}^T\mathbf{V}^T. \tag{2.17}$$

As can be noted, the solution to this problem is not unique. There are four possible rotation/translation pairs that must be considered based on the two possible choices of the rotation matrix, $\mathbf{R_1}$ and $\mathbf{R_2}$, and two possible signs of $\mathbf{t}$. Longuet-Higgins remarked in [82] that the correct solution to the camera placement problem may be chosen based on the requirement that the visible points be in front of both cameras. Therefore, the transformation matrices for the two camera frames are calculated. For the first camera frame, we have $\mathbf{P} = (\mathbf{I}|\mathbf{0})$ and for the second $\mathbf{P}^{'}$ is equal to one of the four following matrices:

$$\begin{aligned} \mathbf{P}_1^{'} &= \left(\mathbf{UWV}^T|\mathbf{U}\,(0,0,1)^T\right) \\ \mathbf{P}_2^{'} &= \left(\mathbf{UWV}^T|-\mathbf{U}\,(0,0,1)^T\right) \\ \mathbf{P}_3^{'} &= \left(\mathbf{UW}^T\mathbf{V}^T|\mathbf{U}\,(0,0,1)^T\right) \\ \mathbf{P}_4^{'} &= \left(\mathbf{UW}^T\mathbf{V}^T|-\mathbf{U}\,(0,0,1)^T\right) \end{aligned} \tag{2.18}$$

The choice between the four transformations for $\mathbf{P}^{'}$ is determined by the requirement that the point locations (which may be computed once the cameras are known) must lie in front of both cameras. Geometrically, the camera rotations represented by $\mathbf{UWV}^T$ and $\mathbf{UW}^T\mathbf{V}^T$ differ from each other by a rotation through 180 degrees about the line joining the two cameras. Given this fact, it may be verified geometrically that a single pixel-to-pixel correspondence is enough to eliminate all but one of the four alternative camera placements.

Once the rotation matrix is found, it can be written in the form of a rotation vector $\omega = \omega\mathbf{1}_\omega$, using the Rodrigues formula:

$$\mathbf{R} = \cos(\omega)\,\mathbf{I} + \sin(\omega)\,[\mathbf{1}_\omega]_\times + (1-\cos(\omega))\,\mathbf{1}_\omega\mathbf{1}_\omega^T \tag{2.19}$$

with the axis of rotation:

$$\mathbf{1}_\omega = \begin{bmatrix} \mathbf{R}_{32} - \mathbf{R}_{23} \\ \mathbf{R}_{13} - \mathbf{R}_{31} \\ \mathbf{R}_{21} - \mathbf{R}_{12} \end{bmatrix} \tag{2.20}$$

and the magnitude (angle) of rotation:

$$\omega = \arccos\left(\frac{Trace\,(\mathbf{R}) - 1}{2}\right) \tag{2.21}$$

2.3.3 Three-View Geometry described by the Trifocal Tensor

The trifocal tensor approach is an extension to the case of three views of the two-view geometry description. This approach maintains a similar projective geometry spirit and has been proposed and developed by Sashua [133], Hartley [49] and Faugeras [37]. The trifocal tensor is a $3 \times 3 \times 3$ array of numbers that relate the coordinates of corresponding points or lines in three views. Just as the fundamental matrix is determined by the two camera matrices, and determines them up to projective transformation, so in three views, the trifocal tensor is determined by the three camera matrices, and in turn determines them, again up to projective transformation.

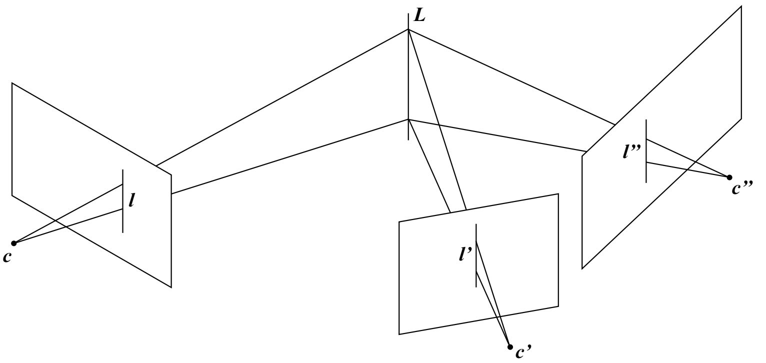


Figure 2.3: A line in 3-space is imaged as the corresponding triplet $\mathbf{l}$,$\mathbf{l}'$,$\mathbf{l}''$ in three views indicated by their centers, $\mathbf{c}$,$\mathbf{c}'$,$\mathbf{c}''$, and image planes.

There are several ways that the trifocal tensor may be approached. Here, the method followed by Hartley in [51] is taken over. Consider a line $\mathbf{L}$ in 3D space, which is projected onto three cameras, resulting in 3 lines $\mathbf{l}$, $\mathbf{l}'$ and $\mathbf{l}''$ in image space, as illustrated by Figure 2.3. These lines are obviously inter-related. The trifocal tensor expresses this relation by mapping lines in 2 images to a line in the remaining image. According to the three-view geometry model, the incidence relation for the i^{th} coordinate l_i of $\mathbf{l}$ can be written as:

$$l_i = \mathbf{l}'^T \mathcal{T}_i \mathbf{l}'' \tag{2.22}$$

By definition, the set of three matrices $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$ constitute the trifocal tensor in matrix notation. In tensor notation, the basic incidence relation 2.22 becomes:

$$l_i = l'_j l''_k \mathcal{T}_i^{jk} \tag{2.23}$$

By defining the vectors $\mathbf{a}_i$ and $\mathbf{b}_i$ as the i^{th} columns of the camera matrices for the three views, the three-view trifocal tensor formulation can also be written as:

$$\mathcal{T}_i^{jk} = \mathbf{a}_i^j \mathbf{b}_4^k - \mathbf{a}_4^j \mathbf{b}_i^k, \tag{2.24}$$

where $\mathbf{a}_4$ and $\mathbf{b}_4$ are the epipoles in views two and three respectively, arising from the first camera.

As with the fundamental matrix, once the trifocal tensor is known, it is possible to extract the three camera matrices from it, and thereby obtain a reconstruction of the scene points and lines. As ever, this reconstruction is unique only up to a 3D projective transformation; it is a projective reconstruction.

It is straightforward to compute the fundamental matrices $\mathbf{F}_{21}$ and $\mathbf{F}_{31}$ between the first and the other views from the trifocal tensor:

$$\mathbf{F}_{21} = [\mathbf{e}']_\times [\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3] \mathbf{e}''; \qquad \mathbf{F}_{31} = [\mathbf{e}'']_\times \left[\mathcal{T}_1^T, \mathcal{T}_2^T, \mathcal{T}_3^T\right] \mathbf{e}' \tag{2.25}$$

To retrieve the camera matrices, the first camera may be chosen as $\mathbf{P} = [\mathbf{I}|\mathbf{0}]$. Since $\mathbf{F}_{21}$ is known from equation 2.25, it is possible to derive the form of the second camera as:

$$\mathbf{P}' = [[\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3]\, \mathbf{e}''|\mathbf{e}'] \tag{2.26}$$

The third camera cannot be chosen independently of the projective frame of the first two. It turns out that $\mathbf{P}''$ can be written as:

$$\mathbf{P}'' = \left[\left(\mathbf{e}''\mathbf{e}''^T - I\right)\left[\mathcal{T}_1^T, \mathcal{T}_2^T, \mathcal{T}_3^T\right] \mathbf{e}'|\mathbf{e}''\right] \tag{2.27}$$

This decomposition shows that the trifocal tensor may be computed from the three camera matrices, and that conversely the three camera matrices may be computed, up to projective equivalence, from the trifocal tensor. Thus, the trifocal tensor completely captures the three cameras up to projective equivalence and we are able to generalize the method for two views to three views. There are several advantages to using such a three-view method for reconstruction.

- It is possible to use a mixture of line and point correspondences to compute the projective reconstruction. With two views, only point correspondences can be used.
- Using three views gives greater stability to the reconstruction, and avoids unstable configurations that may occur using only two views for the reconstruction.

2.3.4 Extension to Multiple Viewpoints

Like the trifocal tensor is an extension of the fundamental matrix in the case of 3 views, a similar extension can be made to the case of 4 views. This leads to the definition of a *quadrifocal tensor*, which relates coordinates measured in four views. The quadrifocal tensor was introduced by Triggs [154] and an algorithm for using it for reconstruction was given by Heyden [58] and Hartley [53]. Even though this seems a logical extension of the already presented two-view and three-view methods, the quadrifocal tensor suffers from some disadvantages, which impede its practical use. One of the main problems is the fact that the quadrifocal tensor is greatly overparametrized, using $3^4 = 81$ components to describe a geometric configuration that depends only on 29 parameters. The number of degrees of freedom can be calculated as follows. Each of the 4 camera matrices has 11 degrees of freedom (5 internal and 6 external), which makes 44 in total. However, the quadrifocal tensor is unchanged by a projective transformation of space, since its value is determined only by image coordinates. Hence, we may subtract $9+6 = 15$ for the degrees of freedom of a general 3D projective transformation. This means that no less than $81-29 = 52$ extra constraints must be fulfilled, which makes the quadrifocal tensor estimation process very difficult. Therefore, a more popular approach is to progressively reconstruct the scene, using two-view [113] or three-view [51] techniques and merge these results over time.

The task of reconstruction becomes easier if we are able to apply a simpler camera model, known as the affine camera. This camera model is a fair approximation to perspective projection whenever the distance to the scene is large compared with the difference in depth between the back and front of the scene. If a set of points are visible in all of a set of n views involving an affine camera, then a well-known algorithm, the factorization algorithm, as introduced by Tomasi and Kanade in [149], can be used to compute both the structure of the scene and the specific camera models in one step using the Singular Value Decomposition. This algorithm is very reliable and simple to implement. Its main difficulties are the use of the affine camera model, rather than a full projective model, and the requirement that all the points be visible in all views. This method has been extended to projective cameras in a method known as projective factorization [146]. Although this method is generally satisfactory, it can not be proven to converge to the correct solution in all cases. Besides, it also requires all points to be visible in all images.

Other methods for n-view reconstruction involve various assumptions, such as knowledge of four coplanar points in the world visible in all views [121], or six or seven points that are visible in all images in the sequence. Methods that apply to specific motion sequences, such as linear motion, planar motion or single axis (turntable) motion have also been developed.

The dominant methodology for the general reconstruction problem is bundle adjustment [155]. This is an iterative method, in which one attempts to fit a non-linear model to the measured data (the point correspondences). The advantage of bundle-adjustment is that it is a very general method that may be applied to a wide range of reconstruction and optimization problems. It may be implemented in such a way that the discovered solution is the Maximum Likelihood solution to the problem, that is a solution that is in some sense optimal in terms of a model for the inaccuracies of image measurements. Unfortunately, bundle adjustment is an iterative process, which can not be guaranteed to converge to the optimal solution from an arbitrary starting point. Much research in reconstruction methods seeks easily computable non-optimal solutions that can be used as a starting point for bundle adjustment. An excellent survey of these methods is given in [155]. A common impression is that bundle-adjustment is necessarily a slow technique, but when implemented carefully, it can be quite efficient.

2.4 Feature Detection, Description and Matching

Sparse image processing algorithms, as the ones presented in the following chapter, rely on the analysis of the movement of distinctive image features like step edges, line features or points in the image where the Fourier components are maximally in phase [101]. This analysis requires three steps. First, a feature detector identifies a set of image locations presenting rich visual information and whose spatial location is well defined. The second step is description: a vector characterizing local texture is computed from the image near the nominal location of the feature. Finally, the set of features needs to be correlated over the different images during the *Feature Matching* step.

The ideal system will be able to detect a large number of meaningful features in the typical image, and will match them reliably across different views of the same scene / object. Critical issues in detection, description and matching are robustness with respect to viewpoint and lighting changes, the number of features detected in a typical image, the frequency of false alarms and mismatches, and the computational cost of each step. Different applications weigh these requirements differently. For example, viewpoint changes more significantly in object recognition, SLAM and wide-baseline stereo than in image mosaicking, while the frequency of false matches may be more critical in object recognition, where thousands of potentially matching images are considered, rather than in wide-baseline stereo and mosaicing where only few images are present.

A couple of studies are available to choose the best combination of feature detector, descriptor and matcher for a given application. Schmid [125] characterized and compared the performance of several features detectors. Mikolajczik and Schmid [94] focused primarily on the descriptor stage. For a chosen detector, the performance of a number of descriptors was assessed. These evaluations of interest point operators and feature descriptors, have relied on the use of flat images, or in some cases synthetic images. The reason is that the transformation between pairs of images can be computed easily, which is convenient to establish ground truth. However, the relative performance of various detectors can change when switching from planar scenes to 3D images [42].

Different studies [99], [70], [125], [94], [42] have evaluated the performance of feature detectors and descriptors for images of 3D objects viewed under different viewpoint, lighting and scale conditions. These studies in general agree that the SIFT-approach delivers the most robust detection and description results. Based on these results, it was decided to use the SIFT-detector and descriptor. For feature matching, a k-D tree matching approach was used.

In recent years, several research groups have proposed improvements to the original SIFT detector, mainly for making it faster. Grabner et al. present in [46] an approach which approximates the original SIFT method, but is considerably faster through the use of efficient data structures. Sinha et al. have adopted in [135] a different approach for accelerating the SIFT detector, by implementing it on a Graphical Processing Unit (GPU). The idea here is to offload as much of the calculation work as possible to the GPU. In this research work, we still use the original SIFT method, as the speed of the SIFT process is not the main computational bottleneck in our processing pipeline.

These different approaches towards feature detection, description and matching are described more in detail in appendix A .

2.5 The Optical Flow

2.5.1 Definition of the Optical Flow

Optical flow is defined [61] as the apparent motion of brightness patterns observed when a camera is moving relative to the objects being imaged. It can be represented with a two-dimensional velocity vector **u** associated with each point on the image plane. $u(x, y)$ and $v(x, y)$ are the two components of the optical flow vector **u**. Figure 2.4 gives an example for illustration. The optical flow contains important information about cues for region and boundary segmentation, shape recovery, and so on.

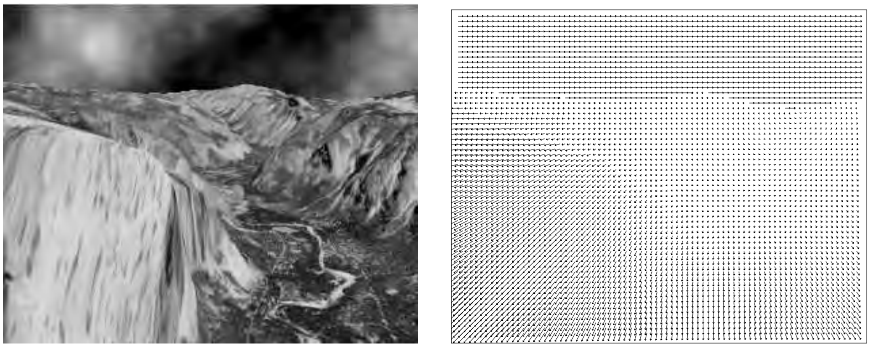

(a) (b)

Figure 2.4: Yosemite sequence with (a) an image, (b) optical flow.

The optical flow calculation starts from the assumption that at each image point (x, y), we expect that the intensity will be the same at time $t + \Delta t$ at the point $(x + \Delta x, y + \Delta y)$, where $\Delta x = u\Delta t$ and $\Delta y = v\Delta t$. The consistency intensity hypothesis of a point during its movement states that the intensity of a point keeps constant along its trajectory through the conservation of image intensity. This hypothesis is reasonable for small displacements or short range motion for which changes of light source are small. That is

$$I(x + u\Delta t, y + v\Delta t, t + \Delta t) = I(x, y, t) \tag{2.28}$$

for a small time interval Δt. If intensity varies smoothly with x, y and t, we can expand the left-hand side of the equation in a Taylor series.

$$I(x, y, t) + \Delta x\frac{\partial I}{\partial x} + \Delta y\frac{\partial I}{\partial y} + \Delta t\frac{\partial I}{\partial t} + e = I(x, y, t) \tag{2.29}$$

where e contains second and higher-order terms in $\Delta x, \Delta y$ and Δt, which is assumed negligible. After ignoring e, we get

$$\Delta x\frac{\partial I}{\partial x} + \Delta y\frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} = 0 \tag{2.30}$$

Observing the notations $u = \Delta x, v = \Delta y$ and with I_x, I_y, and I_t defined as the first order partial derivatives of I ($I_x = \frac{\partial I}{\partial x}, I_y = \frac{\partial I}{\partial y}, I_t = \frac{\partial I}{\partial t}$) and ∇I defined as the spatial intensity gradient ($\nabla I = (I_x, I_y)$), equation 2.30 can be written as:

$$I_x u + I_y v + I_t = 0 \ or \ \nabla I \cdot \mathbf{u} + I_t = 0 \tag{2.31}$$

In the later discussion, Δt is normalized to 1. From this expression, a normal velocity $u_\perp$ can be defined as the vector perpendicular to the constraint line, that is, the velocity with the smallest magnitude on the optical flow constraint line.

In the above linearized optical flow constraint, we assume that the object displacements are small or the image varies slowly in the spatial space. For large displacement fields and images, this linearization is no longer valid. Frequently, instead of the expression in equation 2.29, an alternative equality is used as follows, with the optical flow centered in the first image I_1.

$$I_1(x, y) = I_2(x + u, y + v) \tag{2.32}$$

This equation avoids the linearization. If the optical flow is centered in the second image I_2, the alternative equation states:

$$I_1(x - u, y - v) = I_2(x, y) \tag{2.33}$$

The equation 2.31 is known as the optical flow constraint or brightness constancy assumption. It defines a single local constraint on image motion.

The optical flow constraint expressed in equation 2.31 or equation 2.32 is not sufficient to compute both components of $\mathbf{u}$ as the optical flow constraint is ill-posed. Indeed, it is clear that one equation can not determine the two components of the optical flow, it requires to be supplemented with additional assumptions. Otherwise, only $u_\perp$, the motion component in the direction of the local gradient of the image intensity function, may be estimated. This phenomenon is known as the aperture problem [159] and only at image locations where there is sufficient intensity structure can the motion be fully estimated with the use of the optical flow constraint equation. How the optical flow can be calculated in an efficient way is discussed more in detail in appendix B.

2.5.2 The Relation of the Optical Flow to 3D Structure

Normally what we need for interpreting the 3D structure and motion of the scene is the image flow, which is the 2D projection of the instantaneous 3D velocity of the corresponding point in the scene. However, a sequence of intensity images of the scene (not the scene itself) is typically available. For motion, what is available to people is the optical flow, which is the best we can hope to recover starting from the intensity images alone. As the optical flow only describes 2D projected motion, in order to use it for structure from motion recovery, we have to depend on the assumption that, except for special situations, the optical flow is not too different from the motion field. This will allow us to estimate relative motion by means of the changing image intensities. [61].

The optical flow is related to the structure and motion parameters through the rigid motion equation:

$$\mathbf{v} = \boldsymbol{\omega} \times \mathbf{X} + \mathbf{t} \tag{2.34}$$

To understand this relation between the optical flow field and the structure and motion parameters, it is necessary to expand the formulation of the optical flow

$\mathbf{u} = (u, v)^T$. As stated in section 2.5.1, the optical flow is defined as the apparent image motion, meaning it can be written as the derivative of the projected image points (in 3D coordinates) to time:

$$\mathbf{u} = \frac{d\mathbf{x_p}}{dt} \tag{2.35}$$

This formulation contains no information on the 3D structure. However, by applying the chain rule, the 3D point coordinates $\mathbf{X}$ can be introduced:

$$\mathbf{u} = \frac{d\mathbf{x_p}}{d\mathbf{X}} \frac{d\mathbf{X}}{dt}, \tag{2.36}$$

where $\frac{d\mathbf{X}}{dt}$ is of course the 3D velocity of equation 2.34 and $\mathbf{x_p}$ and $\mathbf{X}$ can be written as:

$$\mathbf{u} = \frac{d(x, y, f)^T}{d(X, Y, Z)^T}\mathbf{v}. \tag{2.37}$$

Observing the relationship between 2D and 3D point coordinates in the perspective projection model expressed by equation 2.7, it is possible to rewrite equation 2.37 to:

$$\mathbf{u} = \frac{d(\frac{f}{Z}X, \frac{f}{Z}Y, f)^T}{d(X, Y, Z)^T}\mathbf{v}, \tag{2.38}$$

which can be expanded to:

$$\mathbf{u} = \begin{pmatrix} \frac{f}{Z}\frac{dX}{dX} & \frac{fX}{Z}\frac{d(1)}{dY} & fX\frac{d(1/Z)}{dZ} \\ \frac{fY}{Z}\frac{d(1)}{dX} & \frac{f}{Z}\frac{dY}{dY} & fY\frac{d(1/Z)}{dZ} \\ \frac{d(f)}{dX} & \frac{d(f)}{dY} & \frac{d(f)}{dZ} \end{pmatrix} . (\boldsymbol{\omega} \times \mathbf{X} + \mathbf{t}) . \tag{2.39}$$

Expanding the derivatives in equation 2.39 yields:

$$\mathbf{u} = \begin{pmatrix} \frac{f}{Z} & 0 & fX\frac{-1}{Z^2} \\ 0 & \frac{f}{Z} & fY\frac{-1}{Z^2} \\ 0 & 0 & 0 \end{pmatrix} . (\boldsymbol{\omega} \times \mathbf{X} + \mathbf{t}) , \tag{2.40}$$

or:

$$\mathbf{u} = \frac{f}{Z}\begin{pmatrix} 1 & 0 & \frac{-X}{Z} \\ 0 & 1 & \frac{-Y}{Z} \\ 0 & 0 & 0 \end{pmatrix} . \left(\boldsymbol{\omega} \times (X, Y, Z)^T + \mathbf{t}\right) . \tag{2.41}$$

The problem with this formulation is that it contains only 3D coordinate points, which cannot be measured directly. Therefore, the perspective projection equation 2.7 is used again, this time to convert from 3D coordinates $\mathbf{X}$ to image coordinates $\mathbf{x}$:

$$\mathbf{u} = \frac{f}{Z}\begin{pmatrix} 1 & 0 & \frac{-\frac{Z}{f}x}{Z} \\ 0 & 1 & \frac{-\frac{Z}{f}y}{Z} \\ 0 & 0 & 0 \end{pmatrix} . \left(\boldsymbol{\omega} \times (\frac{Z}{f}x, \frac{Z}{f}y, \frac{Z}{f}f)^T + \mathbf{t}\right) , \tag{2.42}$$

or shorter:

$$\mathbf{u} = \begin{pmatrix} 1 & 0 & -\frac{x}{f} \\ 0 & 1 & -\frac{y}{f} \\ 0 & 0 & 0 \end{pmatrix} \cdot \left(\boldsymbol{\omega} \times (x, y, f)^T + \frac{f}{Z}\mathbf{t} \right). \tag{2.43}$$

Equation 2.43 formulates a (rank 2) matrix - vector multiplication. Expanding this vector yields:

$$\mathbf{u} = \begin{pmatrix} 1 & 0 & -\frac{x}{f} \\ 0 & 1 & -\frac{y}{f} \\ 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} -\omega_Z y + \omega_Y f + \frac{f}{Z} t_X \\ \omega_Z x - \omega_X f + \frac{f}{Z} t_Y \\ -\omega_Y x + \omega_X y + \frac{f}{Z} t_Z \end{pmatrix}. \tag{2.44}$$

The product can now written out completely, giving:

$$\mathbf{u} = \begin{pmatrix} -\omega_Z y + \omega_Y f + \frac{f}{Z} t_X - \frac{x}{f}(-\omega_Y x + \omega_X y + \frac{f}{Z} t_Z) \\ \omega_Z x - \omega_X f + \frac{f}{Z} t_Y - \frac{y}{f}(-\omega_Y x + \omega_X y + \frac{f}{Z} t_Z) \end{pmatrix}. \tag{2.45}$$

This equation expresses the optical flow $\mathbf{u}$ as a function of the motion parameters (translation $\mathbf{t}$ and rotation $\boldsymbol{\omega}$), image coordinates $\mathbf{x}(x, y)$, focal length f and structural information contained in the Z depth coordinate.

A more compact formulation can be obtained by rewriting 2.45 as:

$$\mathbf{u} = \mathbf{Q}_{\boldsymbol{\omega}}\boldsymbol{\omega} + d\mathbf{Q}_{\mathbf{t}}\mathbf{t}, \tag{2.46}$$

by defining the proximity d as $d = \frac{1}{Z}$ and the matrices $\boldsymbol{Q}_{\boldsymbol{\omega}}$ and $\boldsymbol{Q}_{\boldsymbol{t}}$:

$$\begin{aligned} \mathbf{Q}_{\boldsymbol{\omega}} &= \begin{bmatrix} \frac{xy}{f} & -f - \frac{x^2}{f} & y \\ f + \frac{y^2}{f} & -\frac{xy}{f} & -x \end{bmatrix} \\ \mathbf{Q}_{\mathbf{t}} &= \begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix} \end{aligned} \tag{2.47}$$

The above equation can be expressed as:

$$\mathbf{u} = \mathbf{u}\left(x, y, d, \mathbf{t}, \boldsymbol{\omega}\right), \tag{2.48}$$

or, which is more interesting for us:

$$d = d\left(\mathbf{u}, x, y, \mathbf{t}, \boldsymbol{\omega}\right) \tag{2.49}$$

This defines the relation between the optical flow and the structure and motion parameters and leads to think that once optical flow and motion are known, structural information can be readily retrieved. Indeed, when we look at equation 2.46, it is clear that we can calculate the depth information given by the d depth parameter in two ways: one for each of the components of the optical flow:

$$\begin{cases} u = \sum\limits_{j=1}^{3} \mathbf{Q}_{\boldsymbol{\omega}_{1,j}} \boldsymbol{\omega}_j + d \sum\limits_{j=1}^{3} \mathbf{Q}_{\mathbf{t}_{\mathbf{1,j}}} \mathbf{t}_j \\ v = \sum\limits_{j=1}^{3} \mathbf{Q}_{\boldsymbol{\omega}_{2,j}} \boldsymbol{\omega}_j + d \sum\limits_{j=1}^{3} \mathbf{Q}_{\mathbf{t}_{\mathbf{2,j}}} \mathbf{t}_j \end{cases} \tag{2.50}$$

from where proximity estimates, d_1^* and d_2^*, can be extracted:

$$\begin{cases} d_1^* = \dfrac{u - \sum\limits_{j=1}^{3} \mathbf{Q}_{\boldsymbol{\omega}_{1,j}} \boldsymbol{\omega}_j}{\sum\limits_{j=1}^{3} \mathbf{Q}_{\mathbf{t}_{\mathbf{1,j}}} \mathbf{t}_j} \\ d_2^* = \dfrac{v - \sum\limits_{j=1}^{3} \mathbf{Q}_{\boldsymbol{\omega}_{2,j}} \boldsymbol{\omega}_j}{\sum\limits_{j=1}^{3} \mathbf{Q}_{\mathbf{t}_{\mathbf{2,j}}} \mathbf{t}_j} \end{cases} \tag{2.51}$$

The problem is that this process, which is called back-projection, is very sensitive to noise in the optical flow estimates as well as in the motion vector estimates. In practice, more advanced processing techniques are required to obtain a useful reconstruction result.

2.5.3 The Relation of the Optical Flow to 3D Scene Flow

While the optical flow is the two-dimensional motion field of points in an image, the 3D scene flow is the three-dimensional motion field of points in the world [161]. In the same way that optical flow describes an instantaneous motion field in an image, we can think of scene flow as a three-dimensional flow field $\frac{d\mathbf{X}}{dt}$ describing the motion at every 3D point $\mathbf{X}$ in the scene. Suppose there is a point $\mathbf{X} = \mathbf{X}(t)$ moving in the scene. The image of this point in camera i is $\mathbf{x_p} = \mathbf{x_p}(t)$. If the camera is not moving, the rate of change of $\mathbf{x_p}$ is uniquely determined as:

$$\frac{d\mathbf{x_p}}{dt} = \frac{\partial \mathbf{x_p}}{\partial \mathbf{X}} \frac{d\mathbf{X}}{dt} \tag{2.52}$$

Inverting this relationship is impossible without knowledge of the surface of the 3D model. To invert it, note that $\mathbf{X}$ depends not only on $\mathbf{x_p}$, but also on the time, indirectly through the surface, that is $\mathbf{X} = \mathbf{X}(\mathbf{x_p}(t), t)$.

Differentiating this expression with respect to time gives:

$$\frac{d\mathbf{X}}{dt} = \frac{\partial \mathbf{X}}{\partial \mathbf{x_p}} \frac{d\mathbf{x_p}}{dt} + \left.\frac{\partial \mathbf{X}}{\partial t}\right|_{\mathbf{x_p}} \tag{2.53}$$

This equation says that the motion of a point in the world is made up of two components. The first is the projection of the scene flow on the plane tangent to the surface and passing through $\mathbf{X}$. This is obtained by taking the instantaneous motion on the image plane (the optical flow $\mathbf{u} = \frac{d\mathbf{x_p}}{dt}$ as defined by equation 2.35), and projecting it out into the scene using the inverse Jacobian $\frac{\partial \mathbf{X}}{\partial \mathbf{x_p}}$.

The second term is the contribution to scene flow arising from the three-dimensional motion of the point in the scene imaged by a fixed pixel. It is the instantaneous motion of $\mathbf{X}$ along the ray corresponding to $\mathbf{x_p}$. The magnitude of $\left.\frac{\partial \mathbf{X}}{\partial t}\right|_{\mathbf{x_p}}$ is proportional to the rate of change of the depth of the surface along this ray.

Combining both terms, Equation 2.53 shows how the 3D scene flow is an extension of the 2D optical flow towards the three-dimensional case. The 3D scene flows is able to hold much more structural information than the 2D optical flow, but, as a drawback, it is more difficult to estimate, as will be discussed in section 4.5.

Chapter 3

Sparse Structure from Motion: Overview

3.1 Problem Statement and Global Methodology

Sparse structure and motion estimation poses the following problem: From a set of matched feature points across multiple images, how can we

- estimate the camera motion between all camera views, and
- estimate for each feature point its location in 3D for all camera views?

This problem is solved by setting up a relationship between the movement of features from one image to another and to the camera motion and the scene structure. The estimation of this relationship requires the knowledge of the geometry between the different camera views. This geometry can be described by the models for two-view, three-view and n-view image geometry presented in section 2.3. This chapter explains how these tools have been integrated to form a coherent framework for sparse structure and motion estimation.

The presented framework employs essentially a three-view sliding window approach where consecutive three-view matches are used to compute a trifocal tensor. To extract useful information, the trifocal tensor is then decomposed into camera matrices and fundamental matrices, which contain information on the structure and the camera motion. All three-view reconstructions are then merged together. The global methodology is sketched on figure 3.1. Three main parts can be discriminated: two-view pre-evaluation of the data, three-view reconstruction and multi-view integration. In the pre-evaluation phase, two-view reconstruction techniques are employed to select the best feature data from the total set of feature matches. Based on these matches, three-view reconstruction is performed in a following phase for all sets of 3 consecutive camera views, through the estimation of the three-view geometry as expressed by the trifocal tensor. This yields a set of reconstruction results which are temporally and spatially unrelated. Therefore, in a final multi-view integration phase, all three-view reconstruction results are aligned and merged, such that a consistent estimation for the structure and motion parameters can be obtained in a common reference frame. The remaining sections of this chapter describe each of the aspects of the reconstruction procedure illustrated by figure 3.1 more in detail.

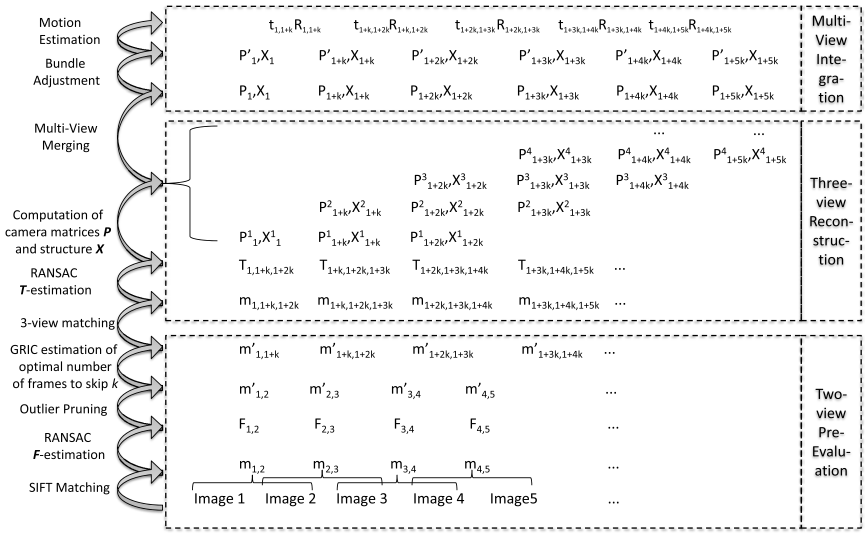

Motion Estimation
Bundle Adjustment
Multi-View Merging
Computation of camera matrices P and structure X
RANSAC T-estimation
3-view matching
GRIC estimation of optimal number of frames to skip k
Outlier Pruning
RANSAC F-estimation
SIFT Matching
$t_{1,1+k}R_{1,1+k}$ $t_{1+k,1+2k}R_{1+k,1+2k}$ $t_{1+2k,1+3k}R_{1+2k,1+3k}$ $t_{1+3k,1+4k}R_{1+3k,1+4k}$ $t_{1+4k,1+5k}R_{1+4k,1+5k}$
P'_1,X_1 P'_{1+k},X_{1+k} P'_{1+2k},X_{1+2k} P'_{1+3k},X_{1+3k} P'_{1+4k},X_{1+4k} P'_{1+5k},X_{1+5k}
P_1,X_1 P_{1+k},X_{1+k} P_{1+2k},X_{1+2k} P_{1+3k},X_{1+3k} P_{1+4k},X_{1+4k} P_{1+5k},X_{1+5k}
Multi-View Integration
P^4_{1+3k},X^4_{1+3k} P^4_{1+4k},X^4_{1+4k} P^4_{1+5k},X^4_{1+5k}
P^3_{1+2k},X^3_{1+2k} P^3_{1+3k},X^3_{1+3k} P^3_{1+4k},X^3_{1+4k}
P^2_{1+k},X^2_{1+k} P^2_{1+2k},X^2_{1+2k} P^2_{1+3k},X^2_{1+3k}
P^1_1,X^1_1 P^1_{1+k},X^1_{1+k} P^1_{1+2k},X^1_{1+2k}
$T_{1,1+k,1+2k}$ $T_{1+k,1+2k,1+3k}$ $T_{1+2k,1+3k,1+4k}$ $T_{1+3k,1+4k,1+5k}$ ...
$m_{1,1+k,1+2k}$ $m_{1+k,1+2k,1+3k}$ $m_{1+2k,1+3k,1+4k}$ $m_{1+3k,1+4k,1+5k}$ ...
Three-view Reconstruction
$m'_{1,1+k}$ $m'_{1+k,1+2k}$ $m'_{1+2k,1+3k}$ $m'_{1+3k,1+4k}$ ...
$m'_{1,2}$ $m'_{2,3}$ $m'_{3,4}$ $m'_{4,5}$ ...
$F_{1,2}$ $F_{2,3}$ $F_{3,4}$ $F_{4,5}$ ...
$m_{1,2}$ $m_{2,3}$ $m_{3,4}$ $m_{4,5}$ ...
Image 1 Image 2 Image 3 Image 4 Image5 ...
Two-view Pre-Evaluation

Figure 3.1: Framework for sparse structure and motion estimation

The choice of this three-view reconstruction technique over the more traditional two-view reconstruction method returns multiple advantages. First, three-view matches are inherently more robust than two-view matches as they contain data over a wider time-span. Secondly, the trifocal tensor estimation is more robust than direct fundamental matrix estimation. While the direct fundamental matrix estimation approach leads often to highly fluctuating estimates for the camera motion parameters, these parameters are already more correct and more smoothed out by the trifocal tensor reconstruction method, as more data has been taken into account for the estimation process. As a last advantage, the three-view reconstruction method is far better suited for multi-view merging, as it is possible for multiple views and camera motions to overlap between individual reconstructions.

The methodology adopted here is strongly related to the approach as proposed by Hartley and Zisserman in [51]. The technique proposed in this work on computer vision has become the de facto standard for sparse structure from motion estimation and is adopted by numerous researchers. The approach proposed here extends the work of Hartley and Zisserman by introducing the GRIC scoring scheme introduced by Torr in [151] to optimize the automatic feature matching process by assessing the validity of the epipolar geometry model and estimating an optimal framerate. Other extensions based on the work of Torr in [152] which were adopted in this approach include optimized robust estimators for the fundamental matrix.

```
Input: A sequence of Images $I_i$
Output: Camera Motion $(R_i, t_i)$, Fundamental Matrices $F_i$ and Sparse
        Structure $(X_i)$

1. Two-view pre-evaluation of the input data:
1.1  Compute point matches using SIFT between all view pairs, as
     described in Appendix A.
1.2  Compute the fundamental matrix using RANSAC and prune the
     matches, as described in [150].
1.3  Compute the optimal framerate using the GRIC criterion from the
     pruned matches, following equations 3.1 to 3.3.

2. Three-view reconstruction: For all consecutive image triplets:
2.1  Compute 3-view matches from 2-view matches, by keypoint
     matching, as described in Appendix A.3.
2.2  Compute the trifocal tensor using RANSAC and prune the 3-view
     matches, by computing equation 3.4.
2.3  Re-estimate the trifocal tensor using only inlier correspondences.
2.4  Compute the camera matrices and fundamental matrices from the
     trifocal tensor, by calculating equations 2.26 and 2.27.
2.5  Compute the structure information from the camera matrices, by
     expressing equation 3.6.

3. Multi-view integration:
3.1  Merge all views into a common projective structure, by solving
     equation 3.15.
3.2  Refine the reconstruction result using bundle adjustment, by
     solving equation 3.16.
3.3  Extract the camera motion from the camera matrices, following
     equation 2.6.
```

Algorithm 1: Overview of the Sparse Part of the Proposed Reconstruction Algorithm

The presented approach aims to combine the benefits of the existing two-view, three-view and n-view reconstruction approaches. The global methodology for sparse structure and motion estimation is sketched by Algorithm 1.

As such, the presented sparse structure from motion approach does not present any novel ideas; it's aim is to integrate the best performing state of the art algorithms to come to a 3D motion and structure estimation which can be used by the dense estimation process for initialization purposes.

3.2 Two-view pre-evaluation of the input data

In this phase, two-view reconstruction is performed to enhance the quality of the feature data the subsequent reconstruction algorithms need to operate on. Indeed, as pointed out before, all sparse analysis uses as a basis feature point detection, description and matching over a sequence of images. In chapter 2, section 2.4 and appendix A, a number of powerful tools for feature detection, description and matching were introduced, most notably the SIFT-features. However, as performant as these matching algorithms may be, using this raw match data directly for reconstruction operations, would lead to some serious problems:

- There will always be some outlier matches. As a result, estimating the scene geometry using for example a classical least-squares minimization method will fail, as the presence of outliers will prevent a correct solution to be found.
- The framerate at which images are taken, and thus at which features are matched, will not always allow reliable reconstruction results. This is the case because some minimal movement between the feature points in different images is needed to have a substantially high signal-to-noise ratio to be able to correctly assess the scene projection model.

To address the first issue, robust estimation of the fundamental matrix between pairs of views $m_{i,i+1}$ is performed using RANSAC [150]. The matches which do not obey the epipolar model, described by the as such estimated fundamental matrix $\mathbf{F}$, are considered as outliers and disregarded for further evaluation, leading to a gset of matches $m'_{i,i+1}$ which is pruned in space.

The second issue is addressed by the estimation of an optimal framerate using the Geometric Robust Information Criterion (GRIC)[151], which proposes an optimal framerate for the image sequence. The basic idea behind the framerate calculation is to estimate both the fundamental matrix and a plain homography and compare how well both models fit the data using the GRIC information criterion. The GRIC or Geometric Robust Information Criterion is a robust model selection criterion that is completely general. It is a scoring function for each model comprising two parts, one for the goodness of fit and one for the parsimony of the model. The first term is the minimum log likelihood of the data, the second is a penalty term, loosely proportional to a product of the number of parameters and the precision in those parameters. Assuming a Gaussian error model for the inliers, GRIC calculates a score function for each motion model taking into account the number n of inlier plus outlier correspondences, the residuals r , the standard deviation of the measurement error σ, the dimension of the data υ (4 for two views) the number η of motion model parameters ($\eta = 7$ for F, $\eta = 8$ for H), and the dimension κ of the structure ($\kappa = 3$ for F, $\kappa = 2$ for H):

$$GRIC = \rho\left(\frac{r^2}{\sigma^2}\right) + \lambda_1 \kappa n + \lambda_2 \eta, \tag{3.1}$$

where $\rho\left(\frac{r^2}{\sigma^2}\right)$ is a robust function of the residuals:

$$\rho\left(\frac{r^2}{\sigma^2}\right) = \min\left(\frac{r^2}{\sigma^2}, \lambda_3(\upsilon - \kappa)\right). \tag{3.2}$$

It can be seen that this is composed of a the standard sum of squares error term, a term to compensate for the dimension of the manifold and a differently weighted term proportional to the number of motion model parameters. These parameters have values $\lambda_1 = \ln 4$, $\lambda_2 = \ln(4n)$ and $\lambda_3 = 2$. For each model, the GRIC criterion is calculated and the model with the lowest score is indicated as most likely. Note that unlike other model selection criteria the GRIC criterion does not assert that the model with lowest score is the correct one, rather it provides the (negative) log of the posterior probability that the model is correct. Given two models under consideration the ratio of the exponentiated GRIC scores provides the relative odds of one being correct over the other.

Taking into consideration the constraints and parameters for a homography and the fundamental matrix, equation 3.1 can be written out to form the following GRIC-scoring equations for the homography and the fundamental matrix model:

$$\begin{aligned} GRIC_H &= \min\left(\frac{\|\mathbf{x}' - \mathbf{H}\mathbf{x}\|^2}{\sigma^2}, 4\right) + 2n\ln(4) + 8\ln(4n) \\ GRIC_F &= \min\left(\frac{\|\mathbf{x}'\mathbf{F}\mathbf{x}\|^2}{\sigma^2}, 2\right) + 3n\ln(4) + 7\ln(4n) \end{aligned} \tag{3.3}$$

Robust estimation of the fundamental matrix will only be possible when the epipolar geometry model is better suited to describe the current feature set than the homography model, i.e. $GRIC_F \leq GRIC_H$. In general, it will be noticed that for high framerates, the homography model will provide a better fit to the data as the disparity for the feature points is too small to reliably estimate $\mathbf{F}$. Therefore, it is advisable to skip some frames in this situation until $GRIC_F \leq GRIC_H$ before making a final robust estimate of $\mathbf{F}$. The number of frames for which $GRIC_F$ becomes smaller than $GRIC_H$ is the optimal inter-frame skip k.

To come to one consistent framerate for the whole image sequence to be reconstructed, a globally optimal framerate is calculated by taking an average of the individual optimal time steps for each frame.

3.3 Three-view reconstruction

In a second stage of reconstruction, the trifocal tensors are estimated for each consecutive set of three views. For this, first a set of matches over three views $m_{k(i-1)+1,ki+1,k(i+1)+1}$ is calculated by feature matching the 2 sets of two-view matches $m'_{k(i-1)+1,ki+1}$ and $m'_{ki+1,k(i+1)+1}$. Then, the three-view geometry is estimated by calculating the trifocal tensor $\mathcal{T}_{k(i-1)+1,ki+1,k(i+1)+1}$. Here, we take over the robust trifocal tensor estimation process as introduced by Hartley in [51].

The main problem for the estimation of the trifocal tensor is that only 18 of the 27 parameters are independent, as geometry only has 18 degrees of freedom. The fundamental matrix also satisfies an internal constraint, but in this case it is only a relatively simple one: the elements obey $\det(\mathbf{F}) = 0$. Here, $\mathcal{T}$ must thus satisfy several constraints to be a geometrically valid trifocal tensor. To get good results, one must take account of these constraints. For this we need the formulation of equation 2.24 which generates only valid trifocal tensors. The parameters of this formulation are the entries $\mathbf{a}_i^j$ and $\mathbf{b}_i^k$. This is still over-parametrized as there are 24 parameters in all.

For the algebraic estimation of $\mathcal{T}$ the algebraic error $\|\mathbf{Q}\mathbf{t}_a\|$ needs to be minimized, subject to the constraint $\|\mathbf{t}_a\| = 1$ and that $\mathcal{T}$ is of the form $\mathcal{T}_i^{jk} = \mathbf{a}_i^j \mathbf{b}_4^k - \mathbf{a}_4^j \mathbf{b}_i^k$.

The difficulty is that this constraint is a quadratic constraint in terms of the parameters. We must thus search the epipoles $\mathbf{a}_4^j$ and $\mathbf{b}_4^k$ first. If $\mathbf{a}_4^j$ and $\mathbf{b}_4^k$ are known, then $\mathcal{T}$ is linear in terms of the other parameters. We may write $\mathbf{t}_a = \mathbf{G}\mathbf{t}_b$, where $\mathbf{t}_b$ is the matrix of 18 remaining entries of camera matrices $\mathbf{A}$ and $\mathbf{B}$, $\mathbf{t}_a$ is the 27-vector of entries of $\mathcal{T}$, and $\mathbf{G}$ is a 27×18 matrix.

The function to minimize then becomes

$$\|\mathbf{Q}\mathbf{G}\mathbf{t}_b\| = 1, \tag{3.4}$$

subject to $\|\mathbf{G}\mathbf{t}_b\| = 1$.

The strategy, as proposed by Hartley in [51] for iterative algebraic $\mathcal{T}$ estimation is to vary the epipoles $\mathbf{a}_4^j$ and $\mathbf{b}_4^k$ to minimize the algebraic error $\|\mathbf{Q}\mathbf{t}_a\| = \|\mathbf{Q}\mathbf{G}\mathbf{t}_b\|$ and after that to use the Levenberg-Marquardt method to minimize this error. This is a 6×27 minimization problem with 6 inputs (the entries of the two epipoles) and 27 outputs (the algebraic error vector $\|\mathbf{Q}\mathbf{t}_a\| = \|\mathbf{Q}\mathbf{G}\mathbf{t}_b\|$), where each step requires estimation of $\mathbf{t}_b$ using the algebraic method.
In practice, this parameter estimation process is embedded into a RANSAC - scheme to enhance the robustness of the solution. The algorithm for the robust estimation of the trifocal tensor $\mathcal{T}$ can thus be summarized as:

- Select a random sample of 6 correspondences;
- Compute $\mathcal{T}$ (as explained above) from these correspondences;
- Measure support (number of inliers) for the solution;
- Choose the $\mathcal{T}$ with the largest number of inliers;
- Re-estimate $\mathcal{T}$ from inlier correspondences.

After the trifocal tensors are estimated, the camera matrices can be calculated by applying equations 2.26 and 2.27. Three camera matrices $\mathbf{P}^i_{k(i-1)+1}$, $\mathbf{P}^i_{ki+1}$ and $\mathbf{P}^i_{k(i+1)+1}$ (one for each view) are estimated as such. From the camera matrices, the structure in each camera view, $\mathbf{X}^i_{k(i-1)+1}$, $\mathbf{X}^i_{ki+1}$ and $\mathbf{X}^i_{k(i+1)+1}$, can be estimated by observing the general perspective projection equations from 3D to 2D, as explained in section 2.2:

$$\mathbf{x} = \mathbf{P}\mathbf{X}; \qquad \mathbf{x}' = \mathbf{P}'\mathbf{X}. \tag{3.5}$$

As such, once the $\mathbf{P}$ matrices are recovered the structure $\mathbf{X}$ may be recovered by triangulation [50]. However, obtaining an optimal solution can be costly. This is because the optimal estimate would minimize the reprojection error of the 3D points i.e. minimize the sum of squares of Euclidean distance between the observed point in each image and the reprojection using the projection matrices and putative 3D structure i.e:

$$\underset{\mathbf{X}}{\operatorname{argmin}}\, d_E\left(\mathbf{x}, \mathbf{P}\mathbf{X}\right)^2 + d_E\left(\mathbf{x}', \mathbf{P}'\mathbf{X}\right)^2 \tag{3.6}$$

where $d_E(\mathbf{a}, \mathbf{b})$ is the Euclidean distance between $\mathbf{a}$ and $\mathbf{b}$. This is equivalent to finding $(\hat{x}, \hat{y}, \hat{x}', \hat{y}')$ such that

$$\sum d_E = (x - \hat{x})^2 + (y - \hat{y})^2 + (\hat{x}' - x')^2 + (\hat{y}' - y')^2 \tag{3.7}$$

is a minimum and $(\hat{x}, \hat{y}, \hat{x}', \hat{y}')$ satisfies $\hat{\mathbf{x}}'\mathbf{F}\hat{\mathbf{x}} = 0$.

3.4 Multi-View integration

Figure 3.1 illustrates that for each consecutive set of three views, a new trifocal tensor is estimated. As each trifocal tensor returns 3 camera matrices, there is an overlap of camera matrices. In general, for each camera view, 3 different camera matrices and associated structure representations are calculated, as illustrated by figure 3.1. Up until this point in the structure estimation procedure, these 3D reconstructions between image triplets were treated independently of one another. Therefore, the individual correspondences of all pairs and triplets, must now be merged into a whole sequence to create a consistent N-view geometry encoded by the projective structure in the multi-view reconstruction phase.

We present here the hierarchical merging strategy introduced by Fitzgibbon and Zisserman in [40]. This approach uses image triplets as the basic building blocks. In such a basic unit, the structure of the scene observed by three cameras can be computed by calculating the associated trifocal tensor from point-to-point correspondences across the three views. These triplets are then registered into sub-groups, followed by merging these subsets and thus building the entire group. This situation is illustrated in figure 3.2.

The view-merging approach starts from the assumption that some 3D points are common in both sets. The homogeneous representation of these points may be denoted as $\mathbf{X}_i$ in the first frame and $\mathbf{X}'_i$ in the second frame. The point representations in the different metric frames are related by a 3-space homography $\mathbf{H}$ according to:

$$\mathbf{X}_i = \mathbf{H}\mathbf{X}'_i \tag{3.8}$$

Equivalently,

$$\mathbf{P}_n^j = \mathbf{P}'^j_n \mathbf{H}^{-1} \tag{3.9}$$

holds for the corresponding normalized projection matrices of the cameras common to both triplets. Since all measurements in real images are noisy, equations 3.8 and 3.9 will not be satisfied exactly. Therefore an error minimizing estimate for $\mathbf{H}$ has to be determined in order to register one triplet in the metric frame of the other triplet. The procedure proceeds in two steps: first, a homography of 3-space is computed which approximately registers one triplet in the other frame. Next, an optimal registration is obtained by bundle adjusting the

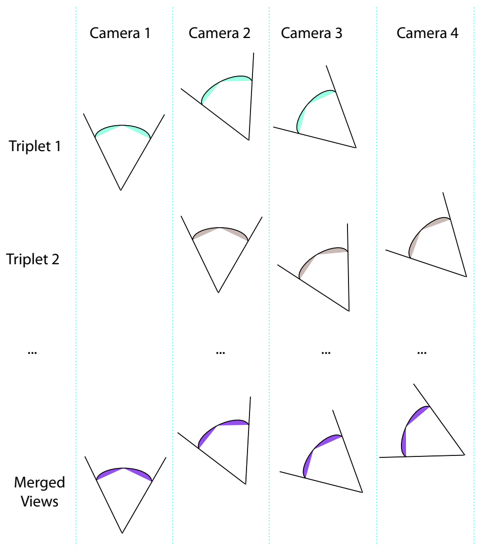


Figure 3.2: Hierarchical merging of 3-view reconstructions for multi-view reconstruction

entire sub-group of cameras and all observed 3D scene points. The approximations found in the first step provide initial guesses for this non-linear optimization procedure.

The result of this operation is a globally optimal and unique structure and motion description per camera view. The motion data is contained in the camera matrices $\mathbf{P}'_i$. However, as we consider only calibrated cameras, it is straightforward to compute the rotation matrices $\mathbf{R}_{i,j}$ and translation vectors $\mathbf{t}_{i,j}$ from the camera matrices, by solving the perspective projection equation 2.6.

The multi view merging process can thus be summarized as: [51]

- Merge the two correspondences between two sub-sequences using the two overlapping images.
- Estimate the space homography $\mathbf{H}_s$ between two common cameras using linear least squares.
- Apply the space homography for one of the two sub-sequences to bring both of them into the same projective basis.
- Optimize the sequence $[i...j]$ with all merged corresponding points using Bundle Adjustment.
- Calculate the motion parameters from the camera matrices

The final result of this step is a set of 3D points and camera poses, projectively consistent with all camera projection matrices on a common space projective basis.

The following two paragraphs explain the calculation of the space homography and the bundle adjustment processes more in detail.

Homography computation The homography between two metric frames can be described by a metric transformation:

$$\mathbf{H} = \begin{pmatrix} \sigma r_{11} & \sigma r_{12} & \sigma r_{13} & t_x \\ \sigma r_{21} & \sigma r_{22} & \sigma r_{23} & t_y \\ \sigma r_{31} & \sigma r_{32} & \sigma r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3.10}$$

with r_{ij} the coefficients of the rotation matrix $\mathbf{R}$, t_i the coefficients of the translation vector $\mathbf{t}$ and σ the relative scale between the structures. Therefore, the transformation between the two different metric frames counts 7 unknowns. Two stages are used to derive accurate estimates for those parameters: first a closed-form solution is obtained, which is then further refined in a non-linear stage. In order to compute a direct solution for the 7 parameters, the first step is to estimate the relative scale σ. Therefore the centroid of the each structure (consisting of the common 3D points $\mathbf{X}_i$, $\mathbf{X'}_i$ respectively) denoted with $\mathbf{C}$, $\mathbf{C}'$ is computed, then the distance of each point in the structure to its centroid is calculated. The relative scale between the two structures is then determined by the quotient of the mean distances:

$$\sigma = \frac{\frac{1}{n}\sum_{i=1}^{n} \|\mathbf{X}_i - \mathbf{C}\|}{\frac{1}{n}\sum_{i=1}^{n} \|\mathbf{X}'_i - \mathbf{C}'\|} \tag{3.11}$$

where $\|.\|$ denotes the $L2$-norm and n is the number of common points in both triplets. The second structure may then be re-scaled according to:

$$\mathbf{X'}_{s_i} = \sigma \mathbf{X}'_i, \tag{3.12}$$

such that equation 3.8 becomes:

$$\mathbf{X}_i = \mathbf{H}_s \mathbf{X'}_{s_i}, \tag{3.13}$$

with $\mathbf{H}_s = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix}$.

In order to obtain an initial estimate for the coefficients of $\mathbf{R}$ and $\mathbf{t}$ the squared distance between these two structures is minimized with respect to the coefficients of $\mathbf{H}_s$ using linear algebraic methods:

$$\min_{\mathbf{R},\mathbf{t}} \sum_i d_E \left(\mathbf{X}_i, \mathbf{H}_s \mathbf{X'}_{s_i}\right)^2, \tag{3.14}$$

where $d_E(\mathbf{v}_1, \mathbf{v}_2)$ denotes the Euclidean distance between the points $\mathbf{v}_1$ and $\mathbf{v}_2$. Finally, this is followed by a nonlinear minimization stage in order to refine the above derived initial values. This nonlinear estimation minimizes the reprojection error to the original measured and normalized image points with respect to all parameters of $\mathbf{H}$.

$$\min_{\sigma,\mathbf{R},\mathbf{t}} \sum_{ij} d_E\left(\mathbf{P}_n^j \mathbf{H}\mathbf{X}_i', x_{n_i}^j\right)^2 + d_E\left(\mathbf{P'}_n^j \mathbf{H}^{-1}\mathbf{X}_i, x_{n_i}^j\right)^2 \tag{3.15}$$

This non-linear minimization may be solved using standard techniques as for instance the Levenberg-Marquardt algorithm.

Hierarchies of larger sub-groups can be built, after all subsets, defined by their camera matrices and 3D points, are computed. By registering those subsets in one common coordinate frame an initial guess for the observed 3D structure (represented by 3D points) and all normalized camera matrices in the entire set of cameras is obtained. Again the registration of sub-groups is achieved by computing homographies of 3-space, as defined by equation 3.8 and 3.9, between the different subgroups.

Bundle Adjustment Bundle adjustment is the problem of refining a visual reconstruction to produce jointly optimal 3D structure and viewing parameter (camera pose and/or calibration) estimates [155]. It is used as the last step of sparse reconstruction in order to optimize the estimated structure and camera pose. The name *bundle adjustment* refers to the bundles of light rays originating from each 3D feature and converging on each camera center, which are adjusted optimally with respect to both structure and viewing parameters. It is often used as a last step of the structure and motion estimation process to produce globally optimal 3D structure and camera motion estimates.

This is achieved by minimizing a cost function that quantifies the model fitting error:

$$\min_{\mathbf{P}^j\mathbf{X}_i} \sum_{i\in\text{points}} \sum_{j\in\text{frames}} d_E\left(\mathbf{x}_i^j, \mathbf{P}^j\mathbf{X}_i\right)^2, \tag{3.16}$$

Solutions of this minimization problem are simultaneously optimal with respect to both structure and camera variations. Bundle adjustment thus amounts to minimizing the reprojection error between the observed and predicted image points, which is expressed as the sum of squares of a number of non-linear real-valued functions.

The minimization is achieved using non-linear least squares algorithms, of which the Levenberg-Marquardt algorithm has proven to be the most successful due to its use of an effective damping strategy that lends it the ability to converge promptly from a wide range of initial guesses.

The Levenberg-Marquardt algorithm involves the solution of linear systems known as the normal equations. Considering that the normal equations are solved repeatedly in the course of the Levenberg-Marquardt algorithm and that each computation of the solution to a dense linear system has complexity $O(N^3)$ in the number of parameters, it is clear that general purpose implementations of the Levenberg-Marquardt algorithm are computationally very demanding when employed to minimize functions depending on many parameters. Fortunately, when solving minimization problems arising in BA, the normal equations matrix has a sparse block structure owing to the lack of interaction among parameters for different 3D points and cameras. Therefore, considerable computational benefits can be gained by developing a tailored sparse variant of the Levenberg-Marquardt algorithm which explicitly takes advantage of the normal equations zeroes pattern.

One of those sparse bundle adjustment algorithms is the Sparse Euclidian Bundle Adjustment (SBA) algorithm presented by Lourakis in [83], which was used throughout this work. SBA is a generic sparse bundle adjustment package which is usable from C++ and is generic in the sense that it grants the user full control over the choice of parameters and functional relations describing cameras, 3D structure and image projections. Therefore, it can support a wide range of manifestations/parameterizations of the multiple view reconstruction problem.

Chapter 4

Dense Structure from Motion: Overview

4.1 Problem Statement

This chapter addresses the problem of dense recovery of structure from motion, more precisely the problem of estimating dense maps of depth and 3D motion from a temporal sequence of monocular images.

As mentioned in the previous chapter, sparse recovery, where depth is computed at a sparse set of points of the image, has been the subject of numerous well-documented studies [51, 151] and [152]. For dense recovery, however, one seeks to compute a depth and 3D motion estimate over the whole image. This subject has been significantly less researched in spite of the many studies on dense estimation of image motion [108, 54, 104] and [9]. This is mainly due to a lack of computing power in the past, as it is only thanks to the dramatic increase in processing power over the last decade, that dense reconstruction becomes possible on commodity computer hardware.

The problem of dense reconstruction from motion is different from the one posed in stereoscopy, as discussed in chapter 3. The two problems are conceptually similar, because one can be considered as a discrete version of the other, but their input and the processing of this input are dissimilar. As indicated in [131], one can see a difference from an abstract point of view, because stereoscopy implies a geometric motion or displacement between views, whereas structure from motion deals with the kinematic notion of motion of the viewing system and viewed objects in temporal sequences. A displacement is defined by an initial position and a final position, intermediate positions being immaterial. Consequently, the notions of time and velocity are irrelevant in stereoscopy. With structure from motion, in contrast, time and velocity are fundamental dimensions. One can also see a difference from a more practical point of view, because both the viewing system and viewed objects can move when acquiring temporal image sequences and all these motions have to be taken into account for 3D reconstruction, which is not the case in stereoscopy.

At the least, this leads to computational complications. For instance, assuming that environmental objects are rigid and images are sized 640×480, there can be over 2 million variables to evaluate at each instant of interpretation (six parameters for the motion and one for depth, and this for all image pixels). The number of these variables can be reduced if some form of motion-based segmentation [12] is applied first, but it is obvious that such a process is computationally quite intensive itself.

The main purpose of this chapter is to give the reader a quick overview of existing dense structure from motion approaches. A more extensive review of some commonly used 3-D reconstruction techniques can be found in [86].

The taxonomy presented here roughly divides the modern approaches towards dense reconstruction into one of 2 classes:

On one hand, there are *model-based* approaches [145], using as a basis mostly a *volume-based* description of the 3D structure to be reconstructed. Examples are voxel coloring [75], photo hulls [137], and level sets [39]. Volume-based methods start from a volumetric description (model) of a surface and fit this model to the measurements, which are the images. These approaches are generally not referred to as dense reconstruction from motion methods as they often do not directly exploit the motion between the different camera images. Instead, they use a large amount of images integrated into a single scheme, often without even trying to explicitly reconstruct the 3D camera motions. These approaches use a discretized volume and restrict possible depth values to a predefined accuracy. The 3D points can be represented by 3D pixels or *voxels* or the 3D model can be described directly by a mesh [44, 182]. Volume-based methods often use an energy minimization approach and are known to reconstruct high-quality models of small objects. Large, unbounded data-sets typically pose problems for these approaches as this leads to a combinatorial explosion. Multiple researchers [140, 158] have aimed to solve this problem, by using more simple 2D depth maps as a representation. Strecha uses in [140] the EM-algorithm to jointly estimate depths and visibility, which are modeled as a hidden Markov Random Field to model the inlier and outlier processes generating the images. Tylecek formulates in [158] the reconstruction problem in a Bayesian framework and minimizes the geometric error between the measured 3D points and the depth estimates, also taking into account the visibility constraints. Both techniques achieve a very high reconstruction precision, but have difficulties when scaling up to large sequences. Therefore, it is questionable whether volume based algorithms can combine reasonable speed and memory requirements for large image sequences with high accuracy for high-resolution images with fine details. For applications where visualization is the primary goal and accuracy is not the main issue or for applications where the model size is bounded, these algorithms perform very well, which explains their success in 3D imaging [67].

A second series of methods are *pixel-based* approaches [2, 141], which do not need 3D discretization and can compute depth for every pixel.

The algorithmically most simple pixel-based approaches aim to extend the feature-based sparse reconstruction methods discussed in chapter 3, by introducing some form of *segmentation* to estimate a depth value for all image pixels. Jancosek presents in [66] a method which oversegments the input images into segments of low variation in color and intensity. For each segment, a candidate 3D planar patch is generated. By combining the planar patches from multiple images, the algorithm can reduce the overall reconstruction error by filtering out the erroneous patches. Lee et al. present a similar method in [77], employing belief propagation as a method to combine, filter and refine the different segments. However, as the application of their work is geared towards 3D TV, they use a depth map as a representation.

A second series of pixel-based approaches are based upon the iterative solution of a partial differential equation (PDE), minimizing an energy functional. In many cases [131, 138, 178, 73, 175] the main data term of this energy functional expresses the *optical flow* constraint, as expressed by equation 2.31. Optical Flow based methods use the apparent motion of brightness patterns to relate estimate structure by relating the optical flow field, the structure and the camera motion. A basic problem with this approach is that the optical flow only provides projected 2D motion information. Moreover, due to noisy measurements, the presence of occlusions and some linearizing assumptions, advanced processing methods are required to obtain robust structural data. Pollefeys presented in [113] an approach to solve the dense reconstruction problem by combining state-of-the-art algorithms for uncalibrated projective reconstruction, self- calibration and dense correspondence matching. Faugeras presented a version of this method by adapting the level set technique to the scene reconstruction problem [39]. With the initial surface encompassing the scene, level sets was used to evolve the surface towards the objects in the scene. The advantage of level sets, although it has a tendency to be trapped in local minimas, is that it is fast and well-developed through the work of Osher and Sethian [111]. This method can also model occlusions through the computation of the visibility of each zero level set at each evolution step. Thus only cameras with an unoccluded viewpoint of the zero level set surface contribute to the computation of the speed function for the next time step. Unlike voxel coloring, a continuous surface and an analytic framework of the surface propagation can be modeled. However, many of the existing PDE-based techniques that aim to reconstruct the 3D scene through energy minimization, have applied the method to pair-wise feature matching of the available images. The limitation is that pair-wise matching techniques can at best only reproduce a 2.5D sketch of the scene (e.g. a depth map) and cannot produce a true 3D reconstruction.

To be able to avoid ambiguities when describing dynamic 3D objects, a third series of pixel-based approaches uses the counterpart of optical flow in 3D space, the *3D scene flow*. 3D scene flow was introduced in [162] and [183] and later used as a basis for dense 3D reconstruction in [114] and [184]. Scene flow based methods try to avoid the projection problems of the optical flow. Their main problem is that upgrading the optical flow to the scene flow drastically increases the number of unknowns, such that the reconstruction problem becomes even harder.

In the following sections, the methodologies which are briefly introduced above, are explained. As the approach applied in this work is a *variational pixel-based optical flow-based* method, this methodology is discussed more in detail.

4.2 Volume-based Methods

There exists a large body of work on building volumetric models from multiple calibrated images. A survey and performance evaluation of recent approaches are presented in [129].

The *Voxel Coloring* method [128] is one of the first deterministic methods that combine image silhouette and color information to build 3D volumetric models. Since any voxel on a Lambertian surface corresponds to consistent image patterns, a photo consistency test is applied to every voxel: if the color variance is larger than a threshold, the voxel is labeled as empty; otherwise it is labeled as part of the object surface. A number of subsequent approaches, such as *Space Carving* [75] and *Generalized Voxel Coloring* [136], extend the original approach with more general camera placements and more efficient labeling methods. These deterministic methods, however, face the difficulty of finding an appropriate threshold for carving the inconsistent voxels, which is inherently varying in different image regions. Furthermore, these methods do not impose global smoothness and may make conflicting decisions in the sequential carving process.

In contrast, energy minimization based methods do not have such problems. Instead of making a harsh decision for each voxel, a global energy function is defined to accumulate the variance of all voxels. This energy function is minimized by discrete optimization techniques, such as *Graph Cuts* [17]. Snow et al. first introduced the graph cuts method in [139] to find the optimal visual hull. This approach is extended in [164] and [112] by utilizing the visual hull as a topological constraints over the scene and searching for an optimal 3D volumetric cut in a voxel based graph. Further constraints are imposed with known surface patches.

Many modern reconstruction methods employ a surface model $S \subset \mathbb{R}^3$ to produce high-quality 3D models. Such a model can consist of an evolutive mesh surface, as presented by Zaharescu in [182]. Based on this, Zaharescu defines a similarity measure $\mathcal{M}_{ij}$ as the similarity between image I_i and the reprojection of image I_j into the other camera i via the surface S. The surface evolution equation can then be written as:

$$\frac{\partial S}{\partial t} = \left[-\lambda E_{regularization} + E_{data}\right] N, \tag{4.1}$$

where $E_{regularization}$ depends on the curvature, N represents the surface normal and E_{data} is a photoconsistency term that is a summation across pairs of cameras which depends upon derivatives of the similarity measure $\mathcal{M}$. Zaharescu solves this energy minimization problem using a level set method, using a mesh-based approach. Tylecek proposes in [158] a different approach, modeling the surface as a continuous Poisson-surface and incorporates also visibility estimation, next to the surface similarity measure. This leads to an over-determined linear system, which is solved using the quasi-minimal residual method. Other researchers like Jancosek [66] and Furukawa [44] model the surface by using a number of 3D planar patches. Jancosek builds these patches by oversegmenting the input images, while Furukawa directly uses the sparse matching data. An important aspect for the performance of these techniques is the approach towards expanding the surface model and filtering false matches, while enforcing photometric consistency and visibility constraints. In [140], Strecha goes a step further by jointly modeling the depth and visibility using a hidden Markov Random Field. The maximum likelihood estimates of the statistics of the inlier and outlier processes are obtained by an Expectation-Maximisation algorithm [32]. This algorithm keeps track of which points of the scene are visible in which images, and accounts for all likely visibility configurations.

Most volumetric approaches are limited to static scenes, but in [180] and [161], space carving approaches are extended to handle dynamic scenes with moving objects.

4.3 Segmentation-based Methods

Segmentation-based approaches use a feature-based technique as a starting point, and apply some form of image segmentation to divide the image in a number of regions. They then assume that all pixels within a given region move uniformly, and hence have the same depth. In [35], Ernst et al. propose a method to create segments corresponding to the underlying objects, or more accurate, to segment the image in regions containing only a single depth. This leads to a chicken-and-egg problem: for segmentation, the depth is needed, but for depth estimation, the segmentation is needed. For foreground-background segmentation or scenes containing a small number of well-defined objects, one could iteratively solve scene structure and segmentation [179]. For general video sequences, these methods always make the same key assumption that discontinuities in depth coincide with discontinuities in color (which is often true, but not always). The matching of segments S in the first image I_1 to the next image I_2, can be written as a match penalty [31], as a function of the disparity δ:

$$E(d) = \sum |I_2(\mathbf{x} + \Delta\mathbf{x}(\delta)) - I_1(\mathbf{x})| \tag{4.2}$$

i.e. for all pixels $\mathbf{x}$ in a segment S, their location in I_2 is estimated, based upon the proposed disparity δ by computing the absolute difference between the image intensities between their positions in I_1. The camera transformation relates the disparity value δ to a motion vector $\Delta\mathbf{x}$.

Solving equation 4.2 results in a depth per segment. Segmentation then groups pixels into a number of 4-connected segments, such that every pixel is part of exactly one segment. A multitude of feature-based and region-based methods have been presented to tackle the segmentation problem.

Ping Li et al. propose in [79] to use Delaunay triangulation to obtain a dense depth map from feature-based SfM. The triangulation technique assumes that the complete scene consists of piece-wise planar surfaces described by the triangles. Generally, this assumption works well if the three vertices of the triangle are close to each other and lie in the same object. However, problems may arise in certain

cases. The dense depth estimate is not accurate for those triangles covering the edges of two objects and the transition area between foreground and background. Another problem of triangulation is that in some image areas where too few feature points can be detected and tracked, triangulation is not applicable at all.
The layered approach [6], [166],[153] aims to segment the image into a set of regions, such that pixels within each region move in a manner consistent with a 2D parametric transformation (e.g. affine). Then, the edges of the region correspond to occlusion boundaries and the motion of all the pixels within the region (textureless or not) is determined by the (six in the case of affine) parameters of the motion model. Typically, an EM (expectation maximization) algorithm [32] is used to effect a segmentation. The use of the 2D affine motion model corresponds to an assumption that the layer maps to a plane in 3D viewed under orthographic conditions. This assumption is valid for a wide variety of scenes, even for non-planar objects for which the distance to the camera is sufficiently large relative to the depth variation across the object. The layered method tends to fail when non-planar objects are viewed in close up as the representation is not adequate in this case. In fact, any object that contains significant parallax effects will cause the breakdown of a layered representation with a global motion model. In order to overcome this problem, Torr et al propose in [153] an approach in which each pixel within a layer can have an associated disparity. The 2D layers, are usually not intended to capture 3D scene structure. In contrast, it was proposed in [7] that the scene should be decomposed into a collection of 3D layers (or sprites), each of which consists of a plane equation, a color image that captures the appearance of the sprite, a per-pixel opacity map, and a per-pixel depth-offset relative to the nominal plane of the layer. The advantage of this approach is that roughness or parallax effects on the layer can be modeled without the over-fragmentation and instability inherent in a purely 2D parametric approach. This approach to layered representation can be viewed as an extension of the plane plus parallax decomposition of image disparities across multiple views. The class of scenes the method works well for includes those that the 2D layered method works well for, but the method tends to fail for scenes for which there is a small amount of per pixel parallax on each plane.

4.4 Optical Flow-based Methods

Direct Methods Structure and motion can be computed either starting from the estimated flow or by inserting a suitable parameterization of the optical flow $\mathbf{u}$ in equation 2.46 or equation 2.31 and extracting structure and motion from the resulting equations. These *Direct Methods* were introduced by Horn and Weldon in [63]. One problem with these approaches is that for a single image pair, one has N equations and $N+6$ unknowns, where N is the number of points in the image, so some added constraint is needed. Negahdaripour and Horn [107] presented a closed form solution assuming a planar or quadratic surface. McQuirk [92] showed that, assuming a pure translation model, the subset of the image points with a non-zero spatial derivative but a zero time derivative gives the direction of motion.

The *Focus Of Expansion* is on a line perpendicular to the gradient at these points. The drawback of this approach is that using only this subset of points, a large part of the image information is ignored. Heel presented in [56] an approach to avoid this problem, by using multiple images of an image sequence. He employs Kalman Filters to build up a structure model from more than one image pair. However, due to its noisy nature, dense optical flow is not extremely suitable for tracking purposes. Moreover, dense flow fields are estimated at pixel locations, so some kind of re-sampling is required during the tracking between consecutive frames. The error introduced by this dense tracking is difficult to evaluate [174]. As in the discrete case, the Kalman filter has proved to be an efficient tool for multi-view integration, especially for keeping the complexity constant over time. However, dense flow poses serious problems for these algorithms along occlusion boundaries or in regions of low texturing and as a result, these problems are in general not addressed. Generally speaking, the Kalman filter is an optimal estimator under the hypothesis of linearity and Gaussian noise. This is rarely the case when real images are used.

Energy Formulation of Variational Methods Another category of dense structure from motion methods are variational approaches, where the reconstruction problem is formulated in an energy minimization framework, under the assumption of global smoothness of the solution. Variational methods thus seek to find a 3D interpretation of the image sequence that minimizes an energy functional:

$$E\left(d, \mathbf{t}, \omega\right) = \int\limits_{\Omega} \left(E_{data} + \mu . E_{regularization}\right) dxdy \tag{4.3}$$

In equation 4.3, the first term in the integrand is the term of conformity to data according to a fitting function. The other term is a regularization term which aims to smooth the solution over areas with continuous structure, trying to fill in the missing information in the data. Note that the energy functional expressed by equation 4.3 is implicitly also a function of the time ($E_{data} = E_{data}(t)$), as this term necessarily expresses a motion constraint.

For solving equation 4.3, often the Euler-Lagrange equation is written out, which leads to a partial differential equation problem:

$$\frac{\partial E}{\partial \mathbf{q}} - \frac{d}{dt}\left(\frac{\partial E}{\partial \dot{\mathbf{q}}}\right) = 0, \tag{4.4}$$

where $\mathbf{q}$ is a vector containing the independent variables which are defined in the energy definition (e.g. $\mathbf{t}, \omega, d$) and $\dot{\mathbf{q}} = \frac{\partial \mathbf{q}}{\partial t}$. The problem is that a discretization of equation 4.4 yields a large system of nonlinear equations that is very difficult to solve in general.

Several alternatives have been proposed for defining the energy term as well as for solving the energy minimization problem. We will now give an overview of some of these approaches:

- The variational method presented by Mitiche and Hadjres in [96] follows the minimum description length principle [102], [186]. This method assumes that environmental objects are rigid and seeks to partition the image domain into regions of constant depth and 3D motion by minimizing an objective function that measures the cost in bits to code the length of the boundaries of the partition and the conformity of the 3D interpretation to the image sequence spatio-temporal variations within each region of the partition. Mitiche and Hadjres ignore rotation, which allows them to obtain a simple constraint equation by substituting equations 2.46 in equation 2.31:

$$\alpha_1 t_1 + \alpha_2 t_2 + \alpha_3 t_3 + I_t Z = 0, \tag{4.5}$$

with $\alpha_1 = fI_x$, $\alpha_2 = fI_y$ and $\alpha_3 = -xI_x - yI_y$. They consider the translation vector $\mathbf{t}$ to be an independent variable at each image point and define an objective function over $\mathbf{t}$ and Z as:

$$\begin{aligned} E_{data} &= \tfrac{1}{2\log 2} \sum_{i\in\Omega} \tfrac{(\alpha_1 t_{1i} + \alpha_2 t_{2i} + \alpha_3 t_{3i} + I_t Z_i)^2}{\sigma^2} \\ &+ \tfrac{b}{2} \sum_{i\in\Omega} \sum_{j\in\mathcal{N}_i} \left(1 - \delta(Z_i - Z_j) \prod_{l=1,2,3} \delta(t_{li} - t_{lj})\right), \end{aligned} \tag{4.6}$$

with b the sum of the number of bits to code the direction of a boundary line segment, $\mathcal{N}_i$ is the set of the 4-neighbors of i, and

$$\delta(x) = \begin{cases} 1 & \text{for} \quad x = 0 \\ 0 & \text{for} \quad x \neq 0 \end{cases} \tag{4.7}$$

As this method is in essence a region filling approach, it does not have a smoothing term.

Like many other researchers [178][36], Mitiche and Hadjres make use of the gradient descent algorithm to minimize equation 4.3. Following this approach, the gradient of the objective function is calculated w.r.t. the independent variable(s) in the problem stated and the solution is evolved in the direction of the negative gradient. In this case, Mitiche and Hadjres differentiate the equation 4.6 with respect to t_{1i}, t_{2i}, t_{3i} and Z_i and write out the necessary conditions for a minimum.

- In another publication [131] of the same research group, Sekkati and Mitiche take into consideration general 3D rigid motion without ignoring rotation. By substituting equations 2.46 in equation 2.31, they then obtain a more general constraint from the optical flow:

$$I_t + \mathbf{r}^T \rho = 0, \tag{4.8}$$

where ρ is a 6-dimensional per-pixel motion vector: $\rho = \left(\frac{\mathbf{t}}{Z}, \frac{\omega}{Z}\right)$ and:

$$\mathbf{r} = \begin{pmatrix} fI_x \\ fI_y \\ -xI_x - yI_y \\ -fI_y - \frac{y}{f}(xI_x + yI_y) \\ fI_x + \frac{x}{f}(xI_x + yI_y) \\ -yI_x - xI_y \end{pmatrix}, \tag{4.9}$$

which allows them to formulate the data term as:

$$E_{data} = \int\limits_{\Omega} (I_t + \mathbf{r}\rho)^2 d\Omega, \tag{4.10}$$

As a regularization term, Sekkati and Mitiche opted for the anisotropic diffusion function defined in [5]:

$$E_{regularization} = \int\limits_{\Omega} \sum_{i=1}^{6} \Phi\left(\|\nabla_{\rho_i}\|\right) d\Omega \tag{4.11}$$

with $\Phi(s) = 2\sqrt{1+s^2} - 2$.

For solving the energy minimization problem, Sekkati and Mitiche use the half - quadratic minimization algorithm proposed in [5]. With the half - quadratic algorithm, equation 4.3 is minimized via the minimization of another functional which $E^*(\rho, \mathbf{b})$, with $\mathbf{b}$ a field of auxiliary variables. $E^*(\rho, \mathbf{b})$ is then minimized using an iterated two-step greedy minimization algorithm. The first step consists of computing the minimum of $E^*(\rho, \mathbf{b})$ with respect to $\mathbf{b}$ with ρ fixed, followed by finding the minimum of $E^*(\rho, \mathbf{b})$ with respect to ρ with $\mathbf{b}$ fixed. These two steps are repeated until convergence.

- Slesareva et al. chose in [138] for the data term a gradient constancy assumption $\left(\nabla I^{i+1}(\mathbf{x} + \boldsymbol{\delta}) = \nabla I^i(\mathbf{x})\right)$ between corresponding structures within consecutive frames, depending on a disparity $\boldsymbol{\delta}$. This leads to a data term of the following form:

$$E_{data} = \int\limits_{\Omega} \frac{1}{N} \sum_{i=0}^{N-1} \psi\left(\nabla I^{i+1}(\mathbf{x} + \boldsymbol{\delta}) - \nabla I^i(\mathbf{x})\right)^2 dx, \tag{4.12}$$

where N the number of frames, $\Omega \subset \mathbb{R}^2$ denotes the rectangular image domain, and $\psi(s^2) = \sqrt{s^2 + \varepsilon^2}$ is a L^1 penalizer with a small regularizing constant $\varepsilon > 0$ ensuring differentiability.

This approach has as an advantage that it is not affected by perturbations on the image intensities caused by illumination changes. For the regularization term, Slesareva uses the anisotropic image-driven regulariser of Nagel and Enkelmann [105], leading to a regularization term of the form:

$$E_{regularization} = \int\limits_{\Omega} \nabla \boldsymbol{\delta}^T D\,(\nabla I) \nabla \boldsymbol{\delta} dx. \tag{4.13}$$

Where **D** is the regularized projection matrix given by equation B.6.

For minimizing the combined energy functional, Slesareva et al. write out the Euler-Lagrange equation of formulation 4.3, introducing the data term of equation 4.12 and regularization term of equation 4.13 with the appropriate boundary conditions. They solve the resulting nonlinear partial differential equation with the help of two nested fixed point iterations. The outer loop fixes nonlinearities with previously computed values of $\boldsymbol{\delta}$, while the inner loop solves the resulting linear problem with the successive overrelaxation (SOR) method.

- Yezzi and Soatto presented in [178] an approach specifically targeted at being able to handle scenes with smooth surfaces and constant isotropic radiance. These scenes challenge most Structure from Motion algorithms because of the lack of photometrically distinct features. They seek to infer the shape of the scene, represented by a description of the surface $\mathcal{S}$, and set up a cost functional that aims at matching regions, and integrate the irradiance over the entire domain of each image. Their data fidelity term quantifies the discrepancy between measured images and the images predicted by the model. Therefore, they define a foreground radiance function $\mathbf{f} : \mathcal{S} \rightarrow \mathbb{R}$ and a background radiance function $\mathbf{h} : \Theta \rightarrow \mathbb{R}$ which projects the surface $\mathcal{S}$ and background to the image domain and a back-projection function $\pi^{-1} : \Omega \rightarrow \mathcal{S}$ which ray-traces an image point x_i to a surface $\mathcal{S}$.

$$E_{data}(\mathbf{f}, \mathbf{h}, \mathcal{S}) = \sum_{i=1}^{N} \int\limits_{\Omega} \left(f\left(\pi_i^{-1}(x_i)\right) - I_i(x_i) \right) dx_i \tag{4.14}$$

Yezzi and Soatto define two regularization terms, one which enforces the smoothness of the surface $E_{regularization_1} = \int\limits_{S} dA$, and one which enforces the smoothness of the foreground and background radiance functions **f** and **h**: $E_{regularization_2} = \int\limits_{S} \|\nabla \mathbf{f}\|^2 dA + \int\limits_{Background} \|\nabla \mathbf{h}\|^2 d\Theta$. Using this objective function, the shape of the model surface and the radiances are adjusted to match the measured images.

Like Mitiche and Hadjres in [96] and Faugeras and Keriven in and [36], Yezzi and Soatto use the gradient descent algorithm to solve the energy minimization problem.

- Kolmogorov & Zabih present in [73] an approach which is able to handle occlusions. Occlusions are a major challenge for the accurate computation of visual correspondence. Occluded pixels are visible in only one image, so there is no corresponding pixel in the other image. Ideally, a pixel in one image should correspond to at most one pixel in the other image, and a pixel that corresponds to no pixel in the other image should be labeled as occluded. This requirement is referred to as *uniqueness*. They address the correspondence problem by constructing a problem representation and an energy function, such that a solution which violates uniqueness will have infinite energy. Therefore, they define an energy function composed of 3 components. A data term E_{data} measures from the differences in intensity between corresponding pixels $\sum\limits_{p,q\in A}(I_1(p)-I_2(q))^2$. An occlusion term $E_{occlusion}$ imposes a penalty C_p for making a pixel occluded:

$$E_{occlusion}=\sum_{i=1}^{N}C_p \quad \forall \quad \text{occluded pixels} \tag{4.15}$$

A smoothness term $E_{smoothness}$ makes neighboring pixels in the same image tend to have similar disparities. The smoothness term requires the notion of a neighborhood of pixels $\mathcal{N}$ and is based on an interaction potential V:

$$E_{smoothness}=\sum_{\substack{\{a1,a2\}\in\mathcal{N}\\ \{a1,a2\}\in\mathcal{A}}} V_{a1,a2} \tag{4.16}$$

where $\mathcal{A}$ is the set of (unordered) pairs of pixels that may potentially correspond. In their experiments, Kolmogorov and Zabih used a simple Potts model for the smoothness term, consisting of an empirically selected decreasing function of $\Delta I(a_1,a_2)$:

$$V_{a_1,a_2}=\begin{cases} 3\lambda \text{ if } \Delta I(a_1,a_2)<5 \\ \lambda \text{ otherwise}\end{cases} \tag{4.17}$$

where $\Delta I(a_1,a_2)$ is the average of intensity values and λ is an empirically chosen parameter.

In [73], Kolmogorov & Zabih present an approach to solve the energy minimization problem through graph cuts. Although graph cut methods have been popular for minimizing energy functions, only a few researchers have attempted to apply graph cuts to reconstruct 3D scenes [73], [134], [139], [122]]. This approach avoids being trapped by early hard decisions and is able to resolve projection ambiguities that are spatially coherent. Of the different graph cuts algorithms applied to scene reconstruction, [122] did not consider visibility and had a spatial smoothness term that was not discontinuity preserving and thus have a tendency to produce over-smoothed results. Snow et al. compute in [139] the global minimum of an energy function that was an alternative to silhouette intersection and thus consequently does not consider photo-consistency. Kang et al. [134] do not treat input images symmetrically and subsequently compute a disparity map with respect to a single camera; while [73] considers a selection of pairs of interacting cameras and can compute a disparity map for each camera. While graph cuts are extremely powerful and can be formulated to compute the optimal solution avoiding being stuck in local minima, not all energy functions can be embedded into a graph for minimization. Kolmogorov has presented a paper [74] that discusses which energy functions can be minimized using graph cuts. The approach followed in [73] is called the expansion move algorithm. Simplified, it can be described as follows: a disparity δ is selected (in a fixed order or at random), the unique configuration within a single δ-expansion move (the local improvement step) is calculated. If this decreases the energy, then this configuration is selected; if there is no δ that decreases the energy, then the iteration is done. The critical step in this method is to efficiently compute the δ-expansion with the smallest energy, which is done using graph cuts.

- Strecha and Van Gool presented in [141] a PDE-based method for dense reconstruction which is based on an extensive evaluation of the *confidence* that the system has in the data coming from the different views. This confidence estimation ensures that only reliable data is spread. Their work is heavily based upon the work of Proesmans in [116], but adds a methodology to cope with variable lighting conditions. To this end, they expand the traditional optical flow constraint, serving as a basis for their data term, with an additional term, accounting for illumination changes:

$$I_2(\mathbf{x}+\delta_0) - I_1 + \frac{\partial I_2(\mathbf{x}+\delta_0)}{\partial \mathbf{x}}(\delta - \delta_0) + (1 - k(\mathbf{x}))I_1(\mathbf{x}) = 0, \tag{4.18}$$

where $k(\mathbf{x})$ takes care of a local intensity scaling in the image I_1.

As a regularization term, Strecha and Van Gool adopt an anisotropic diffusion method, blocking diffusion from places with a lower confidence in their correspondences.

The problem is solved through the iteration of a system of coupled, nonlinear diffusion equations. It is inspired by the work of Proesmans et al. [116]. Proesmans et al. propose a system of 6 coupled, nonlinear diffusion equations, yielding both disparity and depth discontinuity maps containing information on the occluded parts. The high number of equations is due to the symmetric exploitation of the two images: the system evaluates simultaneously a forwards and backwards correspondence search.

- Faugeras presented in [39] a method which originally expresses the simple image brightness brightness assumption between corresponding points $m_1(x, y)$ and $m_2(x, y)$:

$$E_{data} = \int_{\Omega} \left(I_1(m_1(x,y) - I_2(m_2(x,y))\right)^2 dxdy \tag{4.19}$$

He then extends this functional in two ways. The first improvement considers that the scene is made of fronto-parallel planes and replaces the difference of intensities by a measure of correlation:

$$E_{data} = -\int_{\Omega} \frac{\langle I_1, I_2 \rangle}{|I_1|\,|I_2|} dxdy \tag{4.20}$$

with $\langle I_1, I_2 \rangle$ the cross-correlation measure and $|I|^2 = \langle I, I \rangle$. A second improvement to this functional is to relax the hypothesis and to take into account the orientation of the tangent plane to the surface of the object. This improvement takes into account the fact that the rectangular window centered at m_2 is in fact not rectangular, but is the image in the second retina of the backprojection on the tangent plane to the object at the point (x, y) of the rectangular window centered at m_1. As, such Faugeras approximates the object and its depth $Z = f(x, y)$ in a neighborhood of each pixel by its tangent plane, but without assuming that this plane in fronto parallel. This leads to a third version of the data term, this time explicitly including the surface function f.

$$E_{data} = \int_{\Omega} \frac{\langle I_1, I_2 \rangle}{|I_1|\,|I_2|} (f, \nabla f, x, y)\, dxdy \tag{4.21}$$

For solving this problem, Faugeras employs the level set approach. With the initial surface encompassing the scene, level sets are used to evolve the surface towards the objects in the scene. The advantage of level sets, although it has a tendency to be trapped in local minima, is that it is fast and well-developed through the work of Osher and Sethian [111]. This method can also model occlusions through the computation of the visibility of each zero level set at each evolution step. Thus only cameras with an unoccluded viewpoint of the zero level set surface contribute to the computation of the speed function for the next time step. Using this method, a continuous surface and an analytic framework of the surface propagation can be modeled.

More dense reconstruction methods are discussed in the following chapter, where they are directly compared with the approach presented by this research work.

Advantages and Disadvantages of Variational Methods Compared to other methods variational techniques offer a number of specific advantages: They allow transparent modeling without hidden assumptions or post-processing steps. Moreover, their continuous formulation enables rotationally invariant modeling in a natural way. The filling-in effect creates dense depth maps with sub-pixel precision by propagating information over the entire image domain.
Most of these variational methods, however, do not allow for object motion, assuming that the viewing system alone is in movement. As indicated in [130], when the viewing system moves but environmental objects do not, the problem is simpler and 3D interpretation can be inferred by an indirect method which recovers depth following least-squares estimation of 3D motion. The interpretation of scenes where object movement can occur was investigated in [130],[96],[91],[100].

Many of the existing PDE-based techniques that aim to reconstruct the 3D scene through energy minimization, have applied the method to pair-wise feature matching of the available images. The limitation is that pair-wise matching techniques can at best only reproduce a 2.5D sketch of the scene and cannot produce a true 3D reconstruction.

4.5 Scene Flow-based Methods

A drawback of using optical flow as a basis for dense reconstruction is that optical flow only provides projected 2D motion information. It is clear that ambiguities exist when dynamic 3D objects/scenes are explained by using 2D optical flow. This is why the counterpart of optical flow in 3D space, 3D scene flow $\mathbf{u_s} = \mathbf{u_s}(u, v, w)$, is introduced [162], [183]. Like optical flow, 3D scene flow is defined at every point in a reference image. The difference is that the velocity vector in scene flow field contains not only x, y, but also z velocities. This also means that a multi camera setup is often required to compute reliable 3D scene flow.

Two types of methods prevail in the scene flow literature. In the first family of methods [162],[184], scene flow is constructed from previously computed optical flows in all the input images. Then, the scene structure is estimated from scene flow. It is obvious that once optical flow $\mathbf{u}(u, v)$ and disparity δ_t are estimated separately in the reference image sequence, the 3D scene flow is as simple as $(u, v, \delta_{t+1} - \delta_t)$. Ideally, if the optical flow and the disparity are accurate enough, this is correct. However, in practice both optical flow and disparity may be noisy and/or physically inconsistent through cameras. The heuristic spatial smoothness constraints applied to optical flow may also alter the recovered scene flow.

The second family of methods [184], [23] relies on spatio-temporal image derivatives. These variational methods are closely related to the optical flow based reconstruction techniques presented in section 4.4, as they also aim to solve an energy minimization problem as expressed by equation 4.3.

Zhang et al. formulate in [184] the problem as computing a four dimensional vector (u, v, w, d) at every point on the reference image, where the initial disparity is used as an initial guess. However, with serious occlusion and limited number of cameras, this formulation is very difficult because it implies solving four unknowns at every point. At least four independent constraints are needed to make the algorithm stable. Therefore, Zhang et al. formulate constraints on motion, disparity, smoothness and optical flow, and they add a confidence measurement on the disparity estimation. A problem with this method is that it is often limited to slowly-moving Lambertian scenes under constant illumination.

The method advocated by Pons et al. in [114] handles projective distortion without any approximation of shape and motion and can be made robust to appearance changes. The metric used in their framework is the ability to predict the other input views from one input view and the estimated shape or motion. Their method consists in maximizing, with respect to shape and motion, the similarity between each input view and the predicted images coming from the other views. They warp the input images to compute the predicted images, which simultaneously removes projective distortion. Their method allows for generic similarity measures M, but in [114], the opposite of the mutual between the images is used:

$$E_{data} = \sum_{i=1}^{N} \sum_{j \neq i} f\left(\delta_2 M\right) \tag{4.22}$$

The variation of the matching term involves the derivative of the similarity measure with respect to the second image $\delta_2 M$, as documented in [114]. The regularization term is typically the area of the surface, and the associated minimizing flow is a mean curvature motion:

$$E_{regularization} = \int_{\Omega} H dx, \tag{4.23}$$

where H denotes the mean curvature of S.

Part II

Dense reconstruction from monocular sequences

Chapter 5

Dense 3D Structure Estimation

5.1 Introduction and Problem Formulation

Dense structure from motion algorithms aim at estimating a 3D location for all image pixels. One could say that they seek to transform a normal camera from a 2D imaging device into a 3D imaging sensor. As discussed in chapter 4, there are multiple approaches towards dense structure from motion. The most modern dense structure from motion algorithms minimize the optical flow constraint and enforce smoothness in the depth field in a variational framework. However, due to the noisiness of the optical flow and due to projection ambiguities (leading a.o. to occlusions), these algorithms are still not very robust when confronted with unconstrained 3D camera motion and changing illumination conditions. One could argue that these problems are due to the fact that dense structure from motion is a relatively new field of research that emerged recently thanks to the rise in computing power.

5.2 The Proposed Methodology

5.2.1 Global Methodology

To address the classical dense structure from motion shortcomings, we adopt a dual approach for dense structure estimation, trying to combine the robustness of sparse reconstruction techniques with the completeness of the end result, as provided by dense reconstruction algorithms. This is achieved by first solving the sparse reconstruction problem, as explained in chapter 3. These results then serve as initial guesses for the dense reconstruction process, which fuses the sparse data with dense information coming from a densely estimated optical flow field. The optical flow is a projection of the 3D motion field and is related to the structure and motion properties, as explained in section 2.5.2. This relation between optical flow and structure and motion on one hand and the available sparsely reconstructed structure and motion parameters allow for integrated sparse-dense reconstruction, as sketched by figure 5.1.

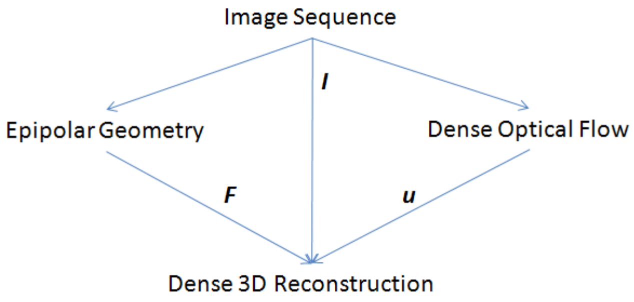


Figure 5.1: Dual structure from motion approach: Sparse Analysis of an image sequence returns a description of the multi-view geometry in the form of a linked series of Fundamental Matrices; Dense Analysis of the same image sequence returns a dense optical flow field. Both data streams are fused in a variational framework.

Here, a variational approach is presented to tackle this high-dimensional data fusion problem. This methodology formulates the problem of fusing dense image data - in the form of the image brightness constraint from the optical flow - with sparse data - in the form of the epipolar constraint of the sparse reconstruction - as an optimization problem. The basic problem of the calculus of variations is to determine the function $q(x, y)$ which minimizes or maximizes a functional in the domain $[x_1, x_2]$

$$J = \int_{x_1}^{x_2} F\left(q(x, y), q_x(x, y), q_y(x, y), x, y\right) dxdy, \tag{5.1}$$

with $q_x(x, y) = \frac{\partial q(x,y)}{\partial x}$ and $q_y(x, y) = \frac{\partial q(x,y)}{\partial y}$.

The integrand $F\left(q(x, y), q_x(x, y), q_y(x, y), x, y\right)$ is in our case composed of two constraint equations. A first constraint, $\phi_{model}(x, y)$, expresses the conformity of the current depth estimate at each pixel to the dense and sparse constraint models. A second constraint, $\phi_{regularization}(x, y)$, introduces anisotropic regularization to preserve the structure smoothness, while preserving depth discontinuities at boundary locations. The functional can thus be written as:

$$J = \int_{\Omega} \phi_{model}(x, y) + \mu\phi_{regularization}(x, y)dxdy \tag{5.2}$$

μ is a positive constant, denoting the relative impact of the regularization term. We will later show how this parameter can be estimated automatically.

Note that J depends not only on x and y but also on the motion vectors. This formulation is similar to the one presented by Alvarez in [2], but the difference is that Alvarez considers a pair of stereo images, whereas we consider time-shifted images from one and the same camera.

The extremal functions of equation 5.1 can be obtained by expressing the Euler-Lagrange equations:

$$\frac{\partial F}{\partial q} - \frac{d}{dx}\left(\frac{\partial F}{\partial q_x}\right) - \frac{d}{dy}\left(\frac{\partial F}{\partial q_y}\right) = 0. \tag{5.3}$$

The Euler-Lagrange equations are obtained by setting the first variational derivatives of the functional J with respect to each function equal to zero. Solving this partial differential equation (PDE) in an analytical way is in general not possible. Hence, iterative approaches are required. The result of these iterative solvers depends strongly on the initialization method, which is why we also introduce a dense depth map initialization method which fuses sparse and dense data. The numerical method for solving the partial differential equation problem is further discussed in section 5.3.

5.2.2 Image Brightness and Epipolar Constraint

As sketched by figure 5.1, there are two main input paths to the dense reconstruction process: the sparse epipolar reconstruction and the dense optical flow estimation. Each of these paths needs to be present in the formulation of the energy minimization problem. Therefore, sparse information in the form of an estimate of the fundamental matrix **F** is introduced in the dense optical flow.

As expressed by equations 2.33 and 2.31, there are 2 - equivalent - ways of expressing the optical flow constraint:

1. Directly from the constant image brightness assumption, which follows from the Lambertian assumption that corresponding pixels have equal grey values:
$$I_1(\mathbf{x}) - I_2(\mathbf{x}+\mathbf{u}) = 0, \tag{5.4}$$
where $I_{1,2}$ are the image intensities of the first and second image.

2. As a function of the image derivatives:
$$I_{1,x}u + I_{1,y}v + I_{1,t} = 0 \textit{ or } \nabla I_1 \cdot \mathbf{u} + I_{1,t} = 0 \tag{5.5}$$

The latter formulation is to be preferred, as it relies on more robust derivatives of the image, instead of directly on the image brightness values. This may seem counter-intuitive, as applying a gradient operation is a classical source for errors in computer vision. However, in this work, we used the spatial gradient estimator proposed by Vieville and Faugeras in [163], which provides a robust intensity gradient estimate.

To express the optical flow as a function of depth, we adopt the approach introduced by Alvarez in [2]. This formulation uses the fundamental matrix, which has as an advantage that, as the fundamental matrix is one of the most basic descriptors of the two-view geometry, it is more robust. The disadvantage is that it is more difficult to work with, as no direct depth parameterization is obtained. This formulation uses the definition of the optical flow as the image flow between corresponding pixels, $\mathbf{x}' = \mathbf{x} + \boldsymbol{u}$, or:

$$\mathbf{x}' = \mathbf{x} + \|\mathbf{u}\| \mathbf{1}_{\boldsymbol{u}}, \tag{5.6}$$

with $\|\mathbf{u}\|$ the magnitude of the optical flow $\|\mathbf{u}\| = \sqrt{u^2 + v^2}$ and $\mathbf{1}_{\boldsymbol{u}} = \begin{bmatrix} \frac{u}{\sqrt{u^2+v^2}} \\ \frac{v}{\sqrt{u^2+v^2}} \end{bmatrix}$ the unitary flow vector associated to the points $\mathbf{x}$ and $\mathbf{x}'$.

The position of the matching coordinates can thus be considered a function the current pixel location $\mathbf{x}$, the pixel depth Z, and the 3D camera motion, which can be expressed by the motion parameters $\boldsymbol{t}$ and $\boldsymbol{\omega}$, or by the fundamental matrix $\mathbf{F}$:

$$I_1(\mathbf{x}) = I_2(\varphi(\mathbf{x}, Z, \mathbf{F})) \tag{5.7}$$

To know the function φ, we first introduce the notations:

$$\begin{aligned} a(x,y) &= f_{11}x + f_{12}y + f_{13} \\ b(x,y) &= f_{21}x + f_{22}y + f_{23}\,, \\ c(x,y) &= f_{31}x + f_{32}y + f_{33} \end{aligned} \tag{5.8}$$

with $f_{i,j}$ the components of the fundamental matrix $\mathbf{F}$, as defined by equation 2.9. Using this notation, the epipolar line $\mathbf{l}_e$ can be written as:

$$a(x,y)x' + b(x,y)y' + c(x,y) = 0 \tag{5.9}$$

We can now express the unitary normal vector $\mathbf{1}_N$ and unitary tangential vector $\mathbf{1}_T$ of the epipolar line $\mathbf{l}_e$ given by equation 5.9:

$$\mathbf{1}_N = \begin{bmatrix} \frac{a}{\sqrt{a^2+b^2}} \\ \frac{b}{\sqrt{a^2+b^2}} \end{bmatrix}; \mathbf{1}_T = \begin{bmatrix} \frac{b}{\sqrt{a^2+b^2}} \\ \frac{-a}{\sqrt{a^2+b^2}} \end{bmatrix}. \tag{5.10}$$

This yields a new formulation for expressing the relationship between 2 corresponding pixels:

$$\mathbf{x}' = \mathbf{x} - \gamma \mathbf{1}_N - \zeta \mathbf{1}_T, \tag{5.11}$$

with:

- $\gamma = \frac{\tilde{\mathbf{x}}_1^t \mathbf{F} \tilde{\mathbf{x}}_1}{\sqrt{a^2+b^2}}$: the distance (modulus a sign) of the point $\mathbf{x}$ to its epipolar line $\mathbf{l}_e$ in the second image. γ can be calculated easily for each point thanks to the knowledge of the fundamental matrix.

- ζ: the distance (modulus a sign) of $\mathbf{x_0}$, the projection of the point $\mathbf{x}$ on the epipolar line $\mathbf{l}_e$, to the point $\mathbf{x}'$ that lies along the epipolar line $\mathbf{l}_e$. ζ can be regarded as a parameter related to the depth and remains to be estimated.

Substituting equation 5.11 in equation 5.6 allows to deduce a relationship between the optical flow and the depth parameter ζ:

$$\mathbf{u}\mathbf{1}_{\boldsymbol{u}} = -\gamma \mathbf{1}_N - \zeta \mathbf{1}_T. \tag{5.12}$$

Using this relationship, Alvarez derived in [2] the following expression for the disparity components:

$$\begin{aligned} u(x,y) &= \frac{-\zeta(x,y)b(x,y)}{\sqrt{a^2(x,y)+b^2(x,y)}} - \frac{a(x,y)x+b(x,y)y+c(x,y)}{a^2(x,y)+b^2(x,y)}a(x,y) \\ v(x,y) &= \frac{\zeta(x,y)a(x,y)}{\sqrt{a^2(x,y)+b^2(x,y)}} - \frac{a(x,y)x+b(x,y)y+c(x,y)}{a^2(x,y)+b^2(x,y)}b(x,y) \end{aligned} \tag{5.13}$$

Equation 5.13 can be written more simply as:

$$\begin{array}{l} u\left(\zeta(x,y),\mathbf{F}\right) = a_1(x,y)\zeta(x,y) + b_1(x,y) \\ v\left(\zeta(x,y),\mathbf{F}\right) = a_2(x,y)\zeta(x,y) + b_2(x,y) \end{array}, \tag{5.14}$$

by introducing the following definitions:

$$\begin{aligned} a_1(x,y) &= \tfrac{-b}{\sqrt{a^2(x,y)+b^2(x,y)}} \\ a_2(x,y) &= \tfrac{a}{\sqrt{a^2(x,y)+b^2(x,y)}} \\ b_1(x,y) &= -\tfrac{a(x,y)x+b(x,y)y+c(x,y)}{\sqrt{a^2(x,y)+b^2(x,y)}}a(x,y) \\ b_2(x,y) &= -\tfrac{a(x,y)x+b(x,y)y+c(x,y)}{\sqrt{a^2(x,y)+b^2(x,y)}}b(x,y) \end{aligned} \tag{5.15}$$

As the distance $\gamma(x,y)$ of a point to its epipolar line is known for each point (x,y), equation 5.14 indicates that the computation of the magnitude of the optical flow $\mathbf{u}(x,y)$, also called the disparity, is equivalent to the computation of the depth function $\zeta(x,y)$.

$$\|\mathbf{u}(x,y)\|^2 = \zeta(x,y)^2 + \gamma(x,y)^2 \tag{5.16}$$

This depth - optical flow relation of equation 5.14 can now be introduced in the optical flow equations to find an expression for the term $\phi_{model}(x,y)$ of equation 5.2. Using the constant brightness based optical flow constraint, we get:

$$\phi_{data} = \left(I_1(x,y) - I_2(x + a_1\zeta + b_1, y + a_2\zeta + b_2)\right)^2 \tag{5.17}$$

For the image derivatives based optical flow constraint of equation 5.5, we get:

$$\phi_{data} = \left(I_{1,x}\left[a_1\zeta + b_1\right] + I_{1,y}\left[a_2\zeta + b_2\right] + I_{1,t}\right)^2 \tag{5.18}$$

The parameters a_1, a_2, b_1, b_2 in equations 5.17 and 5.18 depend on the two-view geometry expressed by the fundamental matrix $\mathbf{F}$. As discussed in chapter 3, we assume the fundamental matrix to be known through a preliminary sparse reconstruction procedure, such that the depth field $\zeta(x,y)$ can be estimated through minimization. In practice, the estimate for $\mathbf{F}$ is often not very accurate and it can be seen immediately from equation 5.14 that this would seriously corrupt the disparity estimates. To address this issue, the estimate for the fundamental matrix **F** is iteratively improved over time, as will be discussed in section 5.2.5.

When working with color images, the color information can be used as well, instead of solely the image intensity. Practically, this means that we have to sum the intensity over of the color bands:

$$\phi_{data} = \sum_{m=1}^{N} \left(I_1^{c_m}(x,y) - I_2^{c_m}(x + a_1\zeta + b_1, y + a_2\zeta + b_2\right)^2, \tag{5.19}$$

or

$$\phi_{data} = \sum_{m=1}^{N} \left(I_{1,x}^{c_m}\left[a_1\zeta + b_1\right] + I_{1,y}^{c_m}\left[a_2\zeta + b_2\right] + I_{1,t}^{c_m}\right)^2, \tag{5.20}$$

where N is the number of color channels (in general $N = 3$) and c_i denotes the i^{th} color channel. To be able to cope with changing illumination conditions, the images can be preprocessed with a color constancy algorithm or the image can be converted to a color space which is more invariant to illumination changes.

5.2.3 Depth Regularization

Using only the constraint equations described in the previous section would lead to serious problems due to (image) and temporal (movement) ambiguities. For example, matching of image intensities typically fails on monochrome surfaces, because due to the fact that there are multiple solutions, the numerical stability cannot be assured and the solver converges to random solutions or does not converge at all. What is needed to solve this problem is a regularization term which extrapolates and smooths the structural data over pixels which belong to the same physical object at the same distance. Multiple smoothing function have been proposed and used in a structure from motion framework, as discussed in section 4.4. The main problem these smoothing terms face is the preservation of discontinuities. Indeed, regularization should not over-smooth the solution such that depth discontinuities are no longer visible. Nagel and Enkelmann took into account this consideration and proposed in [105] an anisotropic smoothing term which preserves the depth discontinuities. The Nagel and Enkelmann regularization model has already been proven successful in a range of independent experiments [138][2] and formulates a regularization term of the following form:

$$\phi_{regularization} = \left(\nabla\zeta\right)^T \mathbf{D}\left(\nabla I_1\right)\left(\nabla\zeta\right) \tag{5.21}$$

Where **D** is a regularized projection matrix given by:

$$D\left(\nabla I_1\right) = \frac{1}{\left|\nabla I_1\right|^2 + 2\upsilon^2} \begin{pmatrix} \left(\frac{\partial I_1}{\partial y}\right)^2 + \upsilon^2 & -\frac{\partial I_1}{\partial x}\frac{\partial I_1}{\partial y} \\ -\frac{\partial I_1}{\partial x}\frac{\partial I_1}{\partial y} & \left(\frac{\partial I_1}{\partial x}\right)^2 + \upsilon^2 \end{pmatrix}, \tag{5.22}$$

with υ a regularization parameter.

$D(\nabla I_1)$ has the eigenvectors ∇I_1 and $\nabla I_1^{\perp}$. The corresponding eigenvalues are given by:

$$\lambda_1 = \frac{v^2}{\|\nabla I_1\|^2 + 2v^2}, \quad \lambda_2 = \frac{\|\nabla I_1\|^2 + v^2}{\|\nabla I_1\|^2 + 2v^2} \tag{5.23}$$

$D(\nabla I_1)$ has the following properties:

- In the interior of objects $\|\nabla I_1\|$ becomes zero, such that $\lambda_1 \rightarrow 1/2$ and $\lambda_2 \rightarrow 1/2$. This results in isotropic behavior within regions and the smoothing of the $\zeta(x, y)$ function in the interior of regions.

- At boundaries $\|\nabla I_1\|$ goes to ∞, such that $\lambda_1 \rightarrow 0$ and $\lambda_2 \rightarrow 1$. As a result the contribution of the regularization term of equation 5.21 will be minor compared to the image brightness constraint term. This assures that the smoothing takes place only along the direction of the boundaries and not across boundaries.

Using this approach, discontinuities can be preserved while the energy functional is minimized. The linear isotropic model can be regarded as a special case with $D(\nabla I_1) = Id$.

5.2.4 Derivation of the Euler-Lagrange Equation

Following the definition of equation 5.3, the Euler-Lagrange equation for this problem can be written as:

$$\frac{\partial F}{\partial \zeta} - \frac{d}{dx}\left(\frac{\partial F}{\frac{\partial \zeta}{dx}}\right) - \frac{d}{dy}\left(\frac{\partial F}{\frac{\partial \zeta}{dy}}\right) = 0, \tag{5.24}$$

with $F = \phi_{data}(x, y) + \mu\phi_{regularization}(x, y)$. $\phi_{data}(x, y)$ is defined by the constraint equation 5.17 for the constant brightness based optical flow constraint, or by equation 5.17 when the image-derivatives based optical flow constraint is considered. $\phi_{regularization}(x, y)$ defined by the regularization constraint of equation 5.21.

Following the constant image brightness based optical flow constraint, the first term in equation 5.24 only concerns the constraint equation 5.17, such that:

$$\begin{aligned} \frac{\partial F}{\partial \zeta} = \frac{\partial \phi_{data}}{\partial \zeta} &= \frac{\partial \left(I_1(x, y) - I_2(x + a_1\zeta + b_1, y + a_2\zeta + b_2)\right)^2}{\partial \zeta} \\ &= 2\left(I_1(x, y) - I_2(x + a_1\zeta + b_1, y + a_2\zeta + b_2)\right)\partial_\zeta I_2, \end{aligned} \tag{5.25}$$

where $\partial_\zeta I_2 = \frac{\partial I_2(x+\delta_x(\zeta(x,y),\mathbf{F}),y+\delta_y(\zeta(x,y),\mathbf{F}))}{\partial \zeta}$ is a spatial derivative of the image intensity in the direction of the epipolar line.

Following the image derivatives based optical flow constraint, the first term in equation 5.24 yields:

$$\begin{aligned}\frac{\partial F}{\partial \zeta} = \frac{\partial \phi_{data}}{\partial \zeta} &= \frac{\partial \left(I_{1,x}\left[a_1\zeta + b_1\right] + I_{1,y}\left[a_2\zeta + b_2\right] + I_{1,t}\right)^2}{\partial \zeta} \\ &= 2\left(I_{1,x}\left[a_1\zeta + b_1\right] + I_{1,y}\left[a_2\zeta + b_2\right] + I_{1,t}\right)\left(a_1 I_{1,x} + a_2 I_{1,y}\right)\end{aligned} \tag{5.26}$$

Only the regularization term in the function F contains derivatives of the form $\frac{\partial\zeta}{dx}$ and $\frac{\partial\zeta}{dy}$, such that:

$$\begin{aligned}\frac{d}{dx}\left(\frac{\partial F}{\frac{\partial\zeta}{dx}}\right) + \frac{d}{dy}\left(\frac{\partial F}{\frac{\partial\zeta}{dy}}\right) &= \frac{d}{dx}\left(\frac{\partial\big(\mu(\nabla\zeta)^T D(\nabla I_1)\nabla\zeta\big)}{\frac{\partial\zeta}{\partial x}}\right) + \frac{d}{dy}\left(\frac{\partial\big(\mu(\nabla\zeta)^T D(\nabla I_1)\nabla\zeta\big)}{\frac{\partial\zeta}{\partial y}}\right) \\ &= 2\mu D_{1,1}\frac{\partial^2\zeta}{\partial x^2} + 4\mu D_{1,2}\frac{\partial^2\zeta}{\partial x\partial y} + 2\mu D_{2,2}\frac{\partial^2\zeta}{\partial y^2} \\ &= 2\mu div\left(D\left(\nabla I_1\right)\nabla\zeta\right)\end{aligned} \tag{5.27}$$

In summary, the proposed method for dense reconstruction poses the problem as a variational problem, expressed by the combination of a data-driven term and a regularization term, as posed by equation 5.2. The data-driven term expresses the conformity of the depth estimate at each pixel to the dense and sparse constraint models. The regularization term introduces anisotropic regularization to preserve the structure smoothness, while preserving depth discontinuities at boundary locations.

The solution for this problem is obtained by writing the Euler-Lagrange equations for this problem, following equation 5.24. For the regularization term, the expression given by equation 5.27 is obtained in this way. For the data-driven term, two formulations are proposed.

The first formulation uses the image brightness based optical flow constraint of equation 5.4, leading to the Euler-Lagrange term given by equation 5.25. Substituting the equation 5.25 and 5.27 in the Euler-Lagrange equation (equation 5.24), a first formulation of the PDE-problem is obtained:

Formulation 1:

$$2\left(I_1(x,y) - I_2(x + a_1\zeta + b_1, y + a_2\zeta + b_2)\right)\partial_\zeta I_2 - 2\mu\nabla\left(D\left(\nabla I_1\right)\nabla\zeta\right) = 0 \tag{5.28}$$

The second formulation uses the image derivatives based optical flow constraint of equation 5.5, leading to the Euler-Lagrange term given by equation 5.26. Substituting the equation 5.26 and 5.27 back in the Euler-Lagrange equation (equation 5.24), a second formulation of the PDE-problem is obtained:

Formulation 2:

$$2\left(I_{1,x}\left[a_1\zeta + b_1\right] + I_{1,y}\left[a_2\zeta + b_2\right] + I_{1,t}\right)\left(a_1 I_{1,x} + a_2 I_{1,y}\right) - 2\mu\nabla\left(D\left(\nabla I_1\right)\nabla\zeta\right) = 0 \tag{5.29}$$

Solving these partial differential equations for $\zeta(x,y)$ returns a solution for the depth field. How this is effectively done is explained in section 5.3. However, before one can begin solving equation 5.28 or 5.29, it is necessary to formulate a methodology to update the estimated description of the two-view geometry, as expressed by the fundamental matrix, and to integrate these results in the variational framework.

5.2.5 Iterative Update of the 2-view Geometry

The methodology for estimating the per-pixel depth field, as described above, starts from a fixed estimate of the two-view geometry, as described by the fundamental matrix $\mathbf{F}$. $\mathbf{F}$ can be estimated through traditional sparse structure from motion algorithms as described in chapter 3. However, this estimate is subject to noise, which will bias the results. Therefore, the estimate of the 2-view geometry should be improved, taking into account the structural data. This can be done iteratively, as the depth estimation process is an iterative one. Indeed, the depth parameter $\zeta_i(x,y)$ is in fact a function of the iteration number, expressed by the time parameter t: $\zeta_i(x,y)=\zeta_i^t(x,y)$, as with each iteration, a new value for $\zeta_i^t(x,y)$ is estimated. It is therefore possible to re-insert these structure results at each iteration to improve the estimation of the two-view geometry.

Considering that the depth parameter for each pixel $\zeta_i^t(x,y)$ is known and the components of the fundamental matrix are the unknown parameters, the problem can be expressed as:

Formulation 1:

$$\underset{f_{1..9}}{\operatorname{argmin}} \sum_{i=1}^{N} \left(I_1\left(\mathbf{x}_i\right) - I_2\left(\mathbf{x}_i + \boldsymbol{u}\left(\mathbf{x}_i, \zeta_i^t, \mathbf{f}\right)\right)\right)^2, \tag{5.30}$$

Formulation 2:

$$\underset{f_{1..9}}{\operatorname{argmin}} \sum_{i=1}^{N} \left(a_1\left(\mathbf{f}\right)\zeta_i^t I_{1,x} + b_1\left(\mathbf{f}\right) I_{1,x} + a_2\left(\mathbf{f}\right)\zeta_i^t I_{1,y} + b_2\left(\mathbf{f}\right) I_{1,y} + I_{1,t}\right)^2, \tag{5.31}$$

with $\mathbf{f} = f_{1..9}$ a vector consisting of the components of the fundamental matrix.
Equations 5.30 and 5.31 both express a non-linear least squares problem, aiming at minimizing the sum of squared residuals $r_i = I_1\left(\mathbf{x}_i\right) - I_2\left(\mathbf{x}_i + \boldsymbol{u}\left(\mathbf{x}_i, \zeta_i^t, \mathbf{f}\right)\right)$ or $r_i = a_1\left(\mathbf{f}\right)\zeta_i^t I_{1,x} + b_1\left(\mathbf{f}\right) I_{1,x} + a_2\left(\mathbf{f}\right)\zeta_i^t I_{1,y} + b_2\left(\mathbf{f}\right) I_{1,y} + I_{1,t}$. This least squares problem can be solved iteratively using the Gauss-Newton algorithm:

$$\mathbf{f}^{k+1} = \mathbf{f}^k - \left(\mathbf{J}^T\mathbf{J}\right)^{-1}\mathbf{J}^T\mathbf{r}, \tag{5.32}$$

with $\mathbf{r}$ the vector of residuals r_i and $\mathbf{J}$ is the Jacobian matrix consisting of partial derivatives of r_i to the different components of $\mathbf{f}$:

$$J_{ij} = \frac{\partial r_i}{\partial f_j}. \tag{5.33}$$

In some cases (degenerative motion), the condition number of the Jacobian matrix can become very high, making the problem ill-conditioned. This means that the application of equation 5.32 will lead to problems. To this end, we introduced a Kalman-filtering post-processing step to the motion update process, to make sure that erroneous results are disregarded.

Following the fundamental matrix update equation 5.32, the two view geometry description (expressed by the fundamental matrix $\mathbf{F}$) is iteratively updated by using the current estimate of the structure description (expressed by the depth parameter ζ^t). Consecutive structure reconstruction results are used to improve the estimate of $\mathbf{F}$, which are on their turn used to improve the structure estimation, as explained in section 5.2.2.

5.3 Numerical Implementation

5.3.1 Model Discretization

A first numerical scheme for solving the Euler-Lagrange problem stated by equations 5.28 and 5.29 which was implemented consisted of an explicit scheme. In the explicit scheme, forward differencing is used to discretize the depth field ζ. This way, the numerical solution at the current iteration level is computed from a discrete approximation of the original PDE, in which the data and regularization terms are evaluated at the previous time level. This numerical scheme provides a simple solution to the discretization problem, as it consists of a straightforward loop over N iterations. However, although the cost of each iteration step is quite low, the total computational cost is rather large, as the time parameter $\Delta\sigma$ must be very small to prevent oscillations. Experiments using this numerical scheme showed that, in order to achieve stable results with this scheme, the time parameter $\Delta\sigma$ should be chosen extremely low, making the number of iteration steps prohibitively large. As a result, a more stable scheme was adopted and a semi-implicit scheme was chosen. The semi-implicit scheme can be seen as a compromise between the simplicity of the explicit scheme and the stability of the implicit scheme. Like the implicit scheme, the semi-implicit scheme is unconditionally stable, thus $\Delta\sigma$ can be relatively large without producing any unstable nonphysical oscillations.

The discretization of the Euler-Lagrange equation based on the constant image differences based optical flow constraint (equation 5.28) is discussed first. For this problem, the Euler-Lagrange equation can be solved, provided that an initial condition is given, by introducing a time parameter σ and by calculating the asymptotic state for $\sigma \rightarrow \infty$:

$$\Rightarrow \begin{cases} \frac{\partial\zeta(\mathbf{x})}{\partial\sigma} = -\left(I_1\left(\mathbf{x}\right) - I_2\left(\mathbf{x} + \boldsymbol{u}\left(\zeta(\mathbf{x}), \mathbf{f}\right)\right)\right) \partial_\zeta I_2(\mathbf{x}) + \mu\nabla\left(D\left(\nabla I_1(\mathbf{x})\right)\nabla\zeta(\mathbf{x})\right) \\ \zeta(\mathbf{x}, \sigma = 0) = \zeta_0 \end{cases} \tag{5.34}$$

The method for obtaining the initial condition ζ_0 is discussed in section 5.3.3.

A problem for the implementation of this semi-implicit scheme is the calculation of the derivative term $\partial_\zeta I_2(\mathbf{x}) = \frac{\partial I_2(\mathbf{x}+\boldsymbol{u}(\zeta(\mathbf{x}),\mathbf{f}))}{\partial\zeta}$. Therefore, we develop this derivative term further using the chain rule:

$$\begin{aligned} \frac{\partial I_2\left(\mathbf{x} + \boldsymbol{u}\left(\zeta\left(\mathbf{x}\right), \mathbf{f}\right)\right)}{\partial\zeta} &= \frac{\partial I_2\left(\mathbf{x} + \boldsymbol{u}\left(\zeta\left(\mathbf{x}\right), \mathbf{f}\right)\right)}{\partial\left(\mathbf{x} + \boldsymbol{u}\right)} \frac{\partial\left(\mathbf{x} + \boldsymbol{u}\right)}{\partial\zeta} \\ &= \frac{\partial I_2\left(\mathbf{x} + \boldsymbol{u}\left(\zeta\left(\mathbf{x}\right), \mathbf{f}\right)\right)}{\partial x} \frac{-b\left(\mathbf{x}\right)}{\sqrt{a\left(\mathbf{x}\right)^2 + b\left(\mathbf{x}\right)^2}} \\ &\quad + \frac{\partial I_2\left(\mathbf{x} + \boldsymbol{u}\left(\zeta\left(\mathbf{x}\right), \mathbf{f}\right)\right)}{\partial y} \frac{a\left(\mathbf{x}\right)}{\sqrt{a\left(\mathbf{x}\right)^2 + b\left(\mathbf{x}\right)^2}} \\ &= \frac{a\left(\mathbf{x}\right) I_{2,y}\left(\mathbf{x} + \boldsymbol{u}\left(\zeta\left(\mathbf{x}\right), \mathbf{f}\right)\right) - b\left(\mathbf{x}\right) I_{2,x}\left(\mathbf{x} + \boldsymbol{u}\left(\zeta\left(\mathbf{x}\right), \mathbf{f}\right)\right)}{\sqrt{a\left(\mathbf{x}\right)^2 + b\left(\mathbf{x}\right)^2}} \end{aligned} \tag{5.35}$$

The derivative $u_{i,j}^{\zeta^k}$ can then be calculated as:

$$u_{i,j}^{\zeta^k} = \frac{a_{i,j} I_{2,y}\left(\mathbf{x}_{i,j} + \boldsymbol{u}\left(\zeta_{i,j}^k\left(\mathbf{x}_{i,j}\right), \mathbf{f}^k\right)\right) - b_{i,j} I_{2,x}\left(\mathbf{x} + \boldsymbol{u}\left(\zeta_{i,j}^k\left(\mathbf{x}\right), \mathbf{f}^k\right)\right)}{\sqrt{a_{i,j}^2 + b_{i,j}^2}} \tag{5.36}$$

The notation $u_{i,j}^{\zeta^k}$ is used to stress that the depth parameter $\zeta_{i,j}^k$ at time index k is involved in the calculation of $u_{i,j}^{\zeta^k}$. This means that $u_{i,j}^{\zeta^k}$ needs to be re-calculated after every iteration.

Combining this information a function $\phi_{data_{i,j}}^k$ can be defined, which represents the data term for the constant image differences based optical flow constraint:

$$\phi_{data_{i,j}}^k = \left(I_1(\mathbf{x}_{i,j}) - I_2\left(x_i + a_{1_{i,j}}(\mathbf{f}^k)\zeta_{i,j}^k + b_{1_{i,j}}(\mathbf{f}^k), y_j + a_{2_{i,j}}(\mathbf{f}^k)\zeta_{i,j}^k + b_{2_{i,j}}(\mathbf{f}^k)\right)\right) u_{i,j}^{\zeta^k}, \tag{5.37}$$

with $\zeta_{i,j}^k$ the estimate of the depth parameter ζ at gridpoint (i, j) at time instance k and $\mathbf{f}^k$ the fundamental matrix components at time instance k, auto-updated as discussed in section 5.2.5.

The discretization of the Euler-Lagrange equation based on the image derivatives based optical flow constraint (equation 5.29) follows a similar, yet somewhat more simple, approach:

$$\Rightarrow \begin{cases} \frac{\partial\zeta(\mathbf{x})}{\partial\sigma} = -\left(I_{1,x}\left[a_1\zeta + b_1\right] + I_{1,y}\left[a_2\zeta + b_2\right] + I_{1,t}\right)\left(a_1 I_{1,x} + a_2 I_{1,y}\right) + \mu\nabla\left(D\left(\nabla I_1\right)\nabla\zeta\right) \\ \zeta(\mathbf{x}, \sigma = 0) = \zeta_0 \end{cases} \tag{5.38}$$

Equivalently to equation 5.37, a function $\phi^k_{data_{i,j}}$ can be defined, which represents the data term for the image derivatives based optical flow constraint:

$$\begin{aligned}\phi^k_{data_{i,j}} = &\big(I_{1,x}(\mathbf{x}_{i,j})\left[a_{1_{i,j}}(\mathbf{f}^k)\zeta^k_{i,j} + b_{1_{i,j}}(\mathbf{f}^k)\right] \\ &+ I_{1,y}(\mathbf{x}_{i,j})\left[a_{2_{i,j}}(\mathbf{f}^k)\zeta^k_{i,j} + b_{2_{i,j}}(\mathbf{f}^k)\right] \\ &+ I_{1,t}(\mathbf{x}_{i,j})\big)\left(a_{1_{i,j}}(\mathbf{f}^k)I_{1,x}(\mathbf{x}_{i,j}) + a_{2_{i,j}}(\mathbf{f}^k)I_{1,y}(\mathbf{x}_{i,j})\right)\end{aligned} \tag{5.39}$$

If $D(\nabla I) = \begin{pmatrix} p & q \\ q & r \end{pmatrix}$ with the parameters p, q, r defined by equation B.6, then the diffusion term $\phi^k_{regularization_{i,j}}$ can be written as:

$$\begin{aligned}\phi^k_{regularization_{i,j}} =& \mu^k\left[\frac{p_{i+1,j}+p_{i,j}}{2}\frac{\zeta^{k+1}_{i+1,j}-\zeta^{k+1}_{i,j}}{h_1^2} + \frac{p_{i-1,j}+p_{i,j}}{2}\frac{\zeta^{k+1}_{i-1,j}-\zeta^{k+1}_{i,j}}{h_1^2}\right] \\ &+ \mu^k\left[\frac{r_{i,j+1}+r_{i,j}}{2}\frac{\zeta^{k+1}_{i,j+1}-\zeta^{k+1}_{i,j}}{h_2^2} + \frac{r_{i,j-1}+r_{i,j}}{2}\frac{\zeta^{k+1}_{i,j-1}-\zeta^{k+1}_{i,j}}{h_2^2}\right] \\ &+ \mu^k\left[\frac{q_{i+1,j+1}+q_{i,j}}{2}\frac{\zeta^{k+1}_{i+1,j+1}-\zeta^{k+1}_{i,j}}{2h_1h_2}\right] \\ &+ \mu^k\left[\frac{q_{i-1,j-1}+q_{i,j}}{2}\frac{\zeta^{k+1}_{i-1,j-1}-\zeta^{k+1}_{i,j}}{2h_1h_2}\right] \\ &+ \mu^k\left[\frac{q_{i-1,j+1}+q_{i,j}}{2}\frac{\zeta^{k+1}_{i-1,j+1}-\zeta^{k+1}_{i,j}}{2h_1h_2}\right] \\ &+ \mu^k\left[\frac{q_{i+1,j-1}+q_{i,j}}{2}\frac{\zeta^{k+1}_{i+1,j-1}-\zeta^{k+1}_{i,j}}{2h_1h_2}\right],\end{aligned} \tag{5.40}$$

with

h_1, h_2 the pixel size in the x and y direction;

μ^k the diffusion parameter regulating the influence of the regularization term, automatically estimated as explained in section 5.3.2;

p, q, r the elements of the projection matrix D.

Combining the results of equations 5.37, 5.39 and 5.40, the linear semi-implicit scheme becomes:

$$\frac{\zeta^{k+1}_{i,j}-\zeta^k_{i,j}}{\Delta\sigma} = -\phi^k_{data_{i,j}} + \phi^k_{regularization_{i,j}}, \tag{5.41}$$

with $\Delta\sigma$ the time step size.

Separating the terms with time indices k and $k+1$ yields:

$$\begin{aligned}
\frac{\zeta_{i,j}^{k}}{\Delta\sigma} - \phi_{data_{i,j}}^{k} = \zeta_{i,j}^{k+1} & \left[\frac{1}{\Delta\sigma} + \mu^{k}\frac{p_{i+1,j} + 2p_{i,j} + p_{i-1,j}}{2h_1^2} + \mu^{k}\frac{r_{i,j+1} + 2r_{i,j} + r_{i,j-1}}{2h_2^2} \right. \\
& \left. + \mu^{k}\frac{q_{i+1,j+1} + q_{i-1,j-1} + q_{i-1,j+1} + q_{i+1,j-1}}{4h_1h_2} \right] \\
& - \mu^{k}\frac{p_{i+1,j} + p_{i,j}}{2h_1^2}\zeta_{i+1,j}^{k+1} - \mu^{k}\frac{p_{i-1,j} + p_{i,j}}{2h_1^2}\zeta_{i-1,j}^{k+1} \\
& - \mu^{k}\frac{r_{i,j+1} + r_{i,j}}{2h_2^2}\zeta_{i,j+1}^{k+1} - \mu^{k}\frac{r_{i,j-1} + r_{i,j}}{2h_2^2}\zeta_{i,j-1}^{k+1} \\
& - \mu^{k}\frac{q_{i+1,j+1} + q_{i,j}}{4h_1h_2}\zeta_{i+1,j+1}^{k+1} - \mu^{k}\frac{q_{i-1,j-1} + q_{i,j}}{4h_1h_2}\zeta_{i-1,j-1}^{k+1} \\
& - \mu^{k}\frac{q_{i-1,j+1} + q_{i,j}}{4h_1h_2}\zeta_{i-1,j+1}^{k+1} - \mu^{k}\frac{q_{i+1,j-1} + q_{i,j}}{4h_1h_2}\zeta_{i+1,j-1}^{k+1}
\end{aligned} \tag{5.42}$$

As can be noted from equation 5.42, the system matrix **S** has the following structure:

$$\mathbf{S} = \left[\begin{array}{cccccccccccc}
\ddots & \ddots & & & \ddots & \ddots & & & & & & \\
\ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & & & & \\
& \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & & & \\
& & \ddots & \ddots & & & \ddots & & & & & \\
\ddots & \ddots & & & \ddots & \ddots & & & & \ddots & & \\
\ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \\
& \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots \\
& & \ddots & \ddots & & & \ddots & \ddots & & & \ddots & \ddots \\
& & & & \ddots & \ddots & & & \ddots & \ddots & & \\
& & & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \\
& & & & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots \\
& & & & & & \ddots & \ddots & & & \ddots & \ddots
\end{array} \right] \tag{5.43}$$

The system matrix, represented by equation 5.43 is a huge, but sparse matrix. When processing an image sequence with a resolution of $M \times N$, the size of the matrix is $(MN)^2$. To put this into perspective: for a low-resolution camera sequence consisting of images of resolution 640×480, this leads to a system matrix with 94.371.840.000 elements. Luckily, the system matrix is also very sparse: it is a 3-diagonal matrix with *only* 2.7 million non-zero entries, meaning that only 0.003% of the system matrix is actually filled.

Using the system matrix formulation, the system equation to be solved can be written as:

$$\boldsymbol{S}\boldsymbol{\zeta} = \mathbf{v}, \tag{5.44}$$

with $\boldsymbol{\zeta}$ the vector of depth parameters and $\mathbf{v}$ the vector of known values, composed from the left hand side of equation 5.42. The structure matrix $\mathbf{S}$ is a large, but sparse matrix. Therefore, the system equation 5.44 can be efficiently solved using sparse matrix algebra.
The system is solved by a least squares subspace trust region method based on the interior-reflective Newton method, as described in [27]. Each iteration involves the approximate solution of a large linear system using the method of preconditioned conjugate gradients. The iteration ends when the difference between 2 consecutive depth estimates falls below a chosen threshold.

5.3.2 Estimation of the Degree of Regularization

The diffusion parameter μ is used to control the diffusivity during the diffusion process, i.e. to decide to which extent the disparity field is diffused at different locations of the disparity field δ. In the literature, the diffusion parameter is generally estimated empirically. There are several drawbacks related to this approach. First, the use of problem-dependent parameters with values which have to be empirically re-estimated for each set of input data, is a tedious process which hinders deterministic benchmarking of reconstruction algorithms. Second, considering that the solution of equation 5.28 must be sought using an iterative updating technique, there is no reason to assume that one and the same value for the regularization parameter μ would be the optimal value throughout the iterative process. Therefore, it is beneficial to dynamically select the diffusion parameter.

Here, we use a method based upon the work of Yang who proposes in [176] a method to dynamically estimate μ. This estimation is based upon the assumption that pixels for which the gradient of the disparity field is above a certain percentage, for example 90% of the maximum of the histogram, are discontinuous locations. Here, the diffusion should stop or at least be largely reduced. This threshold is defined by Yang as:

$$T_g^k = \frac{(\mu^k)^2}{\tilde{K}^k + 2(\mu^k)^2}. \tag{5.45}$$

In equation 5.45, $\tilde{K}^k$ is the threshold of $\|\nabla \boldsymbol{\delta}^k\|^2$ at time instance k associated with the location in the histogram of the disparity gradient where the percentage of the histogram reaches 90% of the maximum of the histogram. T_g^k is the corresponding value of the diffusion function at time instance k. It is a small (e.g. 0.01), but known value, allowing us to determine from equation 5.45 the diffusion parameter μ^k by:

$$\mu^k = \sqrt{\frac{\tilde{K}^k T_g^k}{1 - 2T_g^k}}, \tag{5.46}$$

where μ^k ensures that the diffusion function would be less than T_g^k when the pixel locations belong to discontinuities.

Using this strategy, μ^k can be automatically adjusted according to the disparity gradient distribution at each step of the diffusion process. The value of μ^k is based upon the estimated disparity field during the diffusion process to better control the diffusion. This gives a more reasonable estimate of μ compared to a fixed value for the whole diffusion process.

5.3.3 Initialization

To obtain an initial value for the depth field ζ_0, it is sufficient to calculate an initial value for the flow field. Using equation 5.16, ζ_0 can be written as:

$$\zeta_0 = \sqrt{u_0^2 - \gamma^2} \tag{5.47}$$

The calculation of u_0^2 is based upon the fusion of dense information from the optical flow and reconstructed sparse features. The reason for fusing both data streams is that both cues have their advantages and disadvantages:

- From a set of sparsely matched feature points $\mathbf{x}^F(x^F, y^F)$ and $\mathbf{x}'^F(x'^F, y'^F)$, it is straightforward to calculate a feature flow magnitude map $\boldsymbol{\delta}^F$:

  $$u^F(\mathbf{x}^F) = \sqrt{(x'^F - x^F)^2 + (y'^F - y^F)^2} \tag{5.48}$$

  These sparse feature matches and the associated depth field can be estimated accurately, but they only contain sparse information.

- A densely estimated optical flow field $\mathbf{u}$ has as a problem that, in general, this optical flow field is scaled, such that it is only possible to retrieve a relative estimate for the depth. To obtain an absolute measure, it is necessary to estimate a scale factor, σ_s, as:

  $$\sigma_s = \underset{\sigma_s}{\operatorname{argmin}} \sum_{\mathbf{x}^F} \left(\boldsymbol{u}^F\left(\mathbf{x}^F\right) - \sigma_s \mathbf{u}\left(\mathbf{x}^F\right)\right)^2 \tag{5.49}$$

  The knowledge of the scale factor σ_s between the flow fields as estimated by the dense optical flow and by the feature matching, allows to define a new flow field magnitude map, which is correctly scaled:

  $$u^{flow}(\mathbf{x}) = \sigma_s \sqrt{u(\mathbf{x})^2 + v(\mathbf{x})^2} \tag{5.50}$$

  The remaining problem is that the estimated optical flow is in general less robust than the feature flow as calculated by point correspondences. It would thus be beneficial to combine the two types of data.

The problem when trying to fuse the estimated flow field magnitude map $u^{flow}(\mathbf{x})$ and the feature flow field magnitude map $u^F(\mathbf{x}^F)$ is that the latter function is only defined at the feature points, whereas the former is a dense function defined at each image pixel. Therefore a region growing algorithm is applied to the sparse disparity map $u^F(\mathbf{x}^F)$.

For each pixel $\mathbf{x}$, we estimate a weight function Ξ, expressing the possibility that this pixel belongs to the same region $(\Re\left(\mathbf{x}^F\right))$ of a feature point $\mathbf{x}^F$, taking into account the distance $\left\|\mathbf{x}-\mathbf{x}^F\right\|^2$ between the two points, and the difference between the flow magnitude values $u^{flow}(\mathbf{x})$ and $u^F(\mathbf{x}^F)$:

$$\Xi\left(\mathbf{x} \in \Re\left(\mathbf{x}^F\right)\right) = \frac{1}{1+\|\mathbf{x}-\mathbf{x}^F\|^2} \frac{1}{1+\left(u^{flow}(\mathbf{x}) - u^F\left(\mathbf{x}^F\right)\right)^2} \tag{5.51}$$

At each pixel location $\mathbf{x}$, the region with the highest weight is selected, and the corresponding flow magnitude value $u^F\left(\mathbf{x}^F\right)$ is associated to the *dense feature-based flow field* $u^{features}\left(\mathbf{x}\right)$.

The initial flow magnitude map $u_0(\mathbf{x})$ can then be calculated by combining $u^{features}\left(\mathbf{x}\right)$ and $u^{flow}\left(\mathbf{x}\right)$ as follows:

$$u_0(\mathbf{x}) = u^{features}(\mathbf{x}) + \frac{u^{flow}(\mathbf{x}) - \frac{\sum_{i=1}^{n} u^{flow}(\mathbf{x})}{n}}{\max(u^{flow}(\mathbf{x})) - \min(u^{flow}(\mathbf{x}))} \tag{5.52}$$

By substituting the initial flow magnitude map of equation 5.52 into equation 5.47, an initial value for the depth field ζ_0 is obtained.

5.4 Overview and comparison of the proposed method

5.4.1 Summary of the Dense Reconstruction Algorithm

Traditional structure from motion approaches limit themselves to the sparse reconstruction problem, which is well described in the literature [51] and has been handled in chapter 3. The first dense structure from motion approaches were extensions of their sparse counterparts, e.g. feature-based or sparse-flow based [174] methods using advanced postprocessing techniques to fill up the gaps between the reconstruction points. The same approach is initially followed in this work. The difference is that the obtained reconstruction result is *only* used as initialization for an energy minimization process. The methodology for solving this problem is illustrated by Algorithm 2, which sketches the dense reconstruction algorithm in simplified way.

```
Input: A sequence of Images $I_i$
Output: A dense map of the depth parameter $\zeta$ for each image

1. Initialization:
1.1  Perform sparse reconstruction, following Algorithm 1.
1.2  Compute the optical flow, as described in Appendix B.
1.3  Estimate an initial value for the depth field by computing
     equation 5.47, as described in section 5.3.3.
1.4  Compute the spatial and temporal gradients of the image.
1.5  Compute elements of the regularization matrix, following
     equation B.6.

2. Iterative optimization of the depth field:
2.1  Update the spatial gradient of the depth field.
2.2  Update the estimate of the diffusion parameter $\mu$, according to
     equation 5.46.
2.3  Construct the Structure Matrix $\mathbf{S}$ of equation 5.44.
2.4  Construct the $\mathbf{v}$-vector of equation 5.44.
2.5  Solve the system equation 5.44.
2.6  Update the motion parameter estimate by expressing
     equation 5.32.
2.7  Repeat step 2 if no convergence is reached.
```

Algorithm 2: Overview of the Dense Part of the Proposed Reconstruction Algorithm

The initialization of the proposed algorithm is based upon the solution of the sparse reconstruction problem according to the previously presented Algorithm 1 and the calculation of a dense optical flow field. These two types of information are combined to estimate an initial value for the dense depth field, as described in section 5.3.3.

The iterative optimization of the depth field uses the numerical scheme, described in section 5.3.1. This scheme combines a data term and a regularization term into a linear system, expressed by equation 5.44. The sparse structure matrix $\mathbf{S}$ of the linear system is constructed using the elements of the regularization matrix, following equation B.6. The $\mathbf{v}$-vector of the linear system is constructed using the data term of equation 5.17 or equation 5.18, depending on the chosen formulation. The system is solved by a least squares subspace trust region method based on the interior-reflective Newton method, as described in [27]. Additionally, the diffusion parameter, balancing the degree of regularization, is re-estimated at each iteration, by updating equation 5.46. Also the estimate of the fundamental matrix and the motion parameters is re-estimated at each iteration, by updating equation 5.32. The solver iterates until convergence, measured by calculating the difference between two successive depth field estimates. Finally, the output of this algorithm is a dense depth map for each of the input images.

5.4.2 Relation to Related Work

The proposed methodology towards dense reconstruction falls into the category of energy minimization based reconstruction techniques, as it poses the scene reconstruction problem as an energy minimization problem, as defined by equation 4.3. These approaches pose a modern way of integrating different data cues into a coherent framework and have been used before by a number of researchers [39, 2, 141].

Other modern 3D scene reconstruction techniques [182, 158, 66, 44] build up a surface *model* of the environment. These approaches have the added advantage over our approach that the explicit estimation of a (surface) model of the 3D structure implicitly handles spatial and temporal smoothness constraints, whereas these have to be enforced explicitly in our approach, which does not make any prior assumption on the model of the reconstructed 3D surface, nor does it aim to build up a model explicitly. Furthermore, these methods have the benefit of being capable of directly outputting a high-quality 3D model. Our approach only calculates depth maps, which need to be integrated later (for example, using a technique like ICP [185]). On the other hand, as these model-based approaches rely on the reconstruction of a high-resolution 3D model, they have difficulties when scaling up to large sequences. The envisaged application domain for the proposed algorithm mainly considers large unbounded natural sequences, and for this situation, a pixel-based approach is preferable above a model-based approach. The reason for this lies in the fact that model-based approaches require an a priori model of the environment and that it is quite impossible to produce such an initial 3D model for large unbounded natural sequences.

In the context of energy minimization based reconstruction methodologies, there are 2 main differentiating factors which can be identified to discriminate between the different algorithms:

- The definition of the energy and regularization functions for equation 4.3.
- The choice of the numerical scheme for solving the energy minimization problem.

We will now describe for each of these aspects how the presented algorithm compares itself to related work and why these choices where made.

Definition of the energy and regularization functions The methodology adopted here for dense reconstruction is strongly related to the approach as proposed by Alvarez in [2]. In this work, Alvarez proposes an image-based PDE solution for dense 3D reconstruction using stereo images. In this work, we extend this approach towards another type of input data: a sequence of images and its dense optical flow. This is not trivial, as upgrading the fixed stereo geometry to a dynamically moving camera drastically increases the uncertainty on the relative camera poses. The dynamic nature of moving camera sequences causes large displacements, resulting in difficulties for correspondence matching. We addressed these issues by introducing several new concepts to the algorithm:

- A robust sparse reconstruction approach (Algorithm 1) to provide a good initial estimate of the motion parameters and the sparse structure, in spite of large dynamic movements in the scene.
- An intelligent approach towards estimating an initial value for the depth field, by fusing dense optical flow and sparse reconstruction data, as described in section 5.3.3.
- The implementation of a semi-implicit numerical scheme, guaranteeing convergence to an unconditionally stable solution, as described in section 5.3.1.
- A dynamic approach for updating the estimate of the motion parameters, integrated in the iterative depth field estimation and as described in section 5.3.3.
- A dynamic approach towards automatically re-estimating at each iteration step the value of the diffusion parameter μ, thereby seeking an optimal balance between data-driven convergence and regularization, as described in section 5.3.2

Another related dense reconstruction approach is the method presented by Slesareva et al. in [138]. Slesareva et al. also choose the constant image brightness - based optical flow formulation (equation 4.12), which corresponds to equation 5.17. Moreover, they also use the Nagel-Enkelmann regularization model (equation 4.13). However, as will be proven in the following chapter, the constant image brightness - based optical flow constraint, corresponding to formulation 1 and equation 5.28 is in practice vastly inferior to the image differences based optical flow constraint, corresponding to formulation 2 and equation 5.18. In fact also the methods presented by Yezzi and Soatto in [178], Kolmogorov & Zabih in [73], Strecha and Van Gool in [141] and Faugeras in [39] all use some form of the constant image brightness - based optical flow constraint, but they have added some extra data processing to deal with the limitations of this technique. Yezzi and Soatto consider smooth surfaces and define a radiance function to which images are matched, Kolmogorov & Zabih add a term to deal with occlusions, Strecha and Van Gool try to reason with the reliability of the data and Faugeras includes a surface function for matching. The experimental validations we conducted and which are presented in the following chapter lead us to conclude that it was better to abandon the constant image brightness - based optical flow formulation and use only the image differences based optical flow constraint, which delivers far better results than the constant image brightness - based optical flow formulation. It could, however, be expected that extending our method with the added processing techniques, presented by the aforementioned researchers, could improve our approach even further, but examining this was out of the focus of this research work. The expression of formulation 2 is related to the methodology advocated by Sekkati and Mitiche in [131], and as defined by equation 4.10. However, the regularization term used by Sekkati and Mitiche, given by equation 4.11, is totally different, as it only considers the motion parameters and not the actual depth field. As such, there is no actual regularization of depth data in this approach. This is a serious shortcoming, as it cannot be expected that - in practical situations with natural sequences - a correct depth estimate is found at each pixel location, without any diffusion of the solution from nearby points. Due to these considerations, approaches which do not include any form of depth regularization, like the one of Mitiche and Hadjres in [96] or the one of Faugeras in [39], are prone to inconsistencies in the reconstructed depth field. It can be argued that the expression of the depth regularization is a key factor in the definition of the total energy functional for dense depth reconstruction. For defining a regularization term, Kolmogorov & Zabih use in [73] an interaction potential which relates points to their matching neighborhoods. A problem with this kind of approaches arises at depth boundaries, where the solution is smoothed out over the boundaries, due to isotropic diffusion over the depth boundaries. This effect was noted by researchers and anisotropic diffusion models were developed to counter this problem. Strecha and Van Gool presented in [141] a method towards anisotropic smoothing based upon the confidence of the depth data. This methodology provides an interesting idea as it introduces reasoning with data confidence information in the 3D reconstruction process. However, this method ignores the physical reality that the smoothing of the depth solution should be stopped at object borders, which coincide with depth boundaries, but not necessarily with data confidence regions. Our approach makes use of the Nagel-Enkelmann regularization term, defined by equation B.6, as this regularization model provides anisotropic smoothing. As the anisotropic diffusion directly uses the (estimated) depth field, it preserves depth boundaries. This is a particularly important aspect for the reconstruction of 3D scenes, as the physical world is made up of objects with discontinuous depth projections, making it essential for a depth reconstruction to recognize and deal with these boundaries. Due to its beneficial properties, the Nagel-Enkelmann regularization technique has been used before for 3D reconstruction by a number of researchers [138, 2].

Choice of the numerical scheme The numerical solution of the energy minimization problem posed by equation 4.3 is solved by many researchers [96, 178] using the traditional gradient descent method. The main disadvantage of the gradient descent approach is that convergence is only guaranteed locally, since the algorithm can easily get trapped in local minima. To counter this problem, other researchers [138] have written out the Euler-Lagrange equation (equation 4.4) of the energy functional, leading to a partial differential equation problem with better convergence properties, but which is more difficult to solve. Also in our approach, the Euler-Lagrange expression is used, but in contrast to Slesareva et al. who choose a numerical solving method using successive overrelaxation, we choose a semi-implicit numerical scheme as a method for the numerical solution of the energy minimization problem. The reason for this choice is that a semi-implicit scheme presents a balanced compromise between the simplicity of an explicit scheme as proposed in [1] and the stability of an implicit scheme, as used by Alvarez in [2].

The basis for each iterative solver is a good initialization of the depth field. Therefore, we developed an intelligent approach for estimating an initial value for the depth field, based upon the fusion of dense optical flow data and sparse reconstruction data, as described in section 5.3.3. Most other researchers initialize the depth field to zero or some other constant value. This causes difficulties for the numerical solver: as the solution is too far from the initial value, the solver gets trapped in local minima which do not necessarily correspond to the global minimum.

Due to the tight relation between structure and motion, it is not possible to calculate a good estimate for the depth field and the scene structure without having a good estimate for the (camera) motion. Depth and motion are deeply interrelated parameters and should, as a consequence, be estimated simultaneously. Interestingly, almost all researchers totally ignore this tight relation between depth and motion. Instead, they assume the camera motion to be known or they assume it was estimated a priori by a sparse reconstruction algorithm and they do not consider any update for these motion parameters during the depth estimation process. In [131], Sekkati and Mitiche present an iterated two-step algorithm for solving the motion parameters and the depth field in an integrated way. In our method, we follow a similar dynamic approach for updating the estimate of the motion parameters. In contrast to the method presented by Sekkati and Mitiche, our approach does not integrate the motion estimation into the energy functional to be optimized. The reason for this is that the scale of the 2 problems is too different to be considered in the same optimization scheme: for an image of resolution M by N, the depth estimation features MN unknowns, whereas there are only 6 unknowns (3 for translation and 3 for rotation) for the motion estimation. To avoid this problem, we adopted a methodology where the 2-view geometry is re-estimated after each iteration of the depth field, using the new depth estimate, as such improving the estimate of the motion parameters at each iteration.

All energy - minimization based depth reconstruction methodologies which are based on the combination of a data term and a regularization term, as presented by equation 4.3, are faced with the problem of choosing a value for the diffusion parameter μ. This diffusion parameter controls the balance between the data-driven convergence and regularization and is almost always chosen empirically using a process of trial-and-error. In our methodology, we have presented a dynamic approach for choosing the diffusion parameter μ and for automatically re-estimating it at each iteration step, as described in section 5.3.2. This is an important aspect of the algorithm, as it opens the door towards a more user-friendly application of the 3D reconstruction algorithm, without the need for a tedious parameter - setting process.

5.5 Conclusions

To reconstruct a dense depth field, it is necessary to maximize the information which can be retrieved from the data. In this chapter, we proposed an approach which fuses sparse and dense information in an integrated variational framework. The aim of this approach is to combine the robustness of traditional sparse structure from motion methods with the completeness of optical flow based dense reconstruction approaches.

The base constraint of the variational approach is the gradient based optical flow constraint, but parameterized for the depth using the 2-view geometry. This estimation of the geometry, as expressed by the fundamental matrix, is automatically updated at each iteration of the solver. A regularization term is added to ensure good reconstruction results in image regions where the data term lacks information. An automatically updated regularization term ensures an optimal balance between the data term and the regularization term at each iteration step.

A semi-implicit numerical scheme was set up to solve the dense reconstruction problem. The solver uses an initialization process which fuses optical flow data and sparse feature point matches.

Chapter 6

Results & Analysis for Monocular Reconstruction

6.1 Description of the Test Procedure

In this chapter, the results of the proposed dense structure from motion algorithm are presented. The algorithm can be roughly subdivided into three main parts:

1. Sparse structure and motion estimation as discussed in chapter 3.
2. Estimation of an initial value for the disparity map, based on the sparse structure data and the estimated optical flow, according to equations 5.47 - 5.52.
3. Estimation of a dense disparity field using the PDE-based approach discussed in chapter 5.

The analysis deals with each of these aspects separately.

The main problem in evaluating the performance of any 3D reconstruction algorithm, is the absence of quality ground truth data. Available data on the internet most often only consists of series of images (sequences) with some camera data (calibration parameters and/or in exceptional cases camera motion). In order to overcome this problem, we constructed an artificial 3D scene in a commercial CAD package and added a well defined camera which we set up to follow a predefined trajectory. Figure 6.1 shows the 3D model along with the camera trajectory.

Following this approach, we were able to control all variables - depth information, camera calibration data and camera motion - needing to be estimated by the structure estimation algorithms. We then made photo-realistic renderings of the scene as seen by the camera at different time-steps. These renderings serve as base data for the image processing algorithms (feature detection and matching, optical flow calculation, PDE - based reconstruction). Next to the photo-realistic renderings, also the depth information was exported at this stage by constructing depth maps at each time frame. Figure 6.2 shows some frames out of a 40-frame sequence which was constructed this way, together with the corresponding depth map rendering.

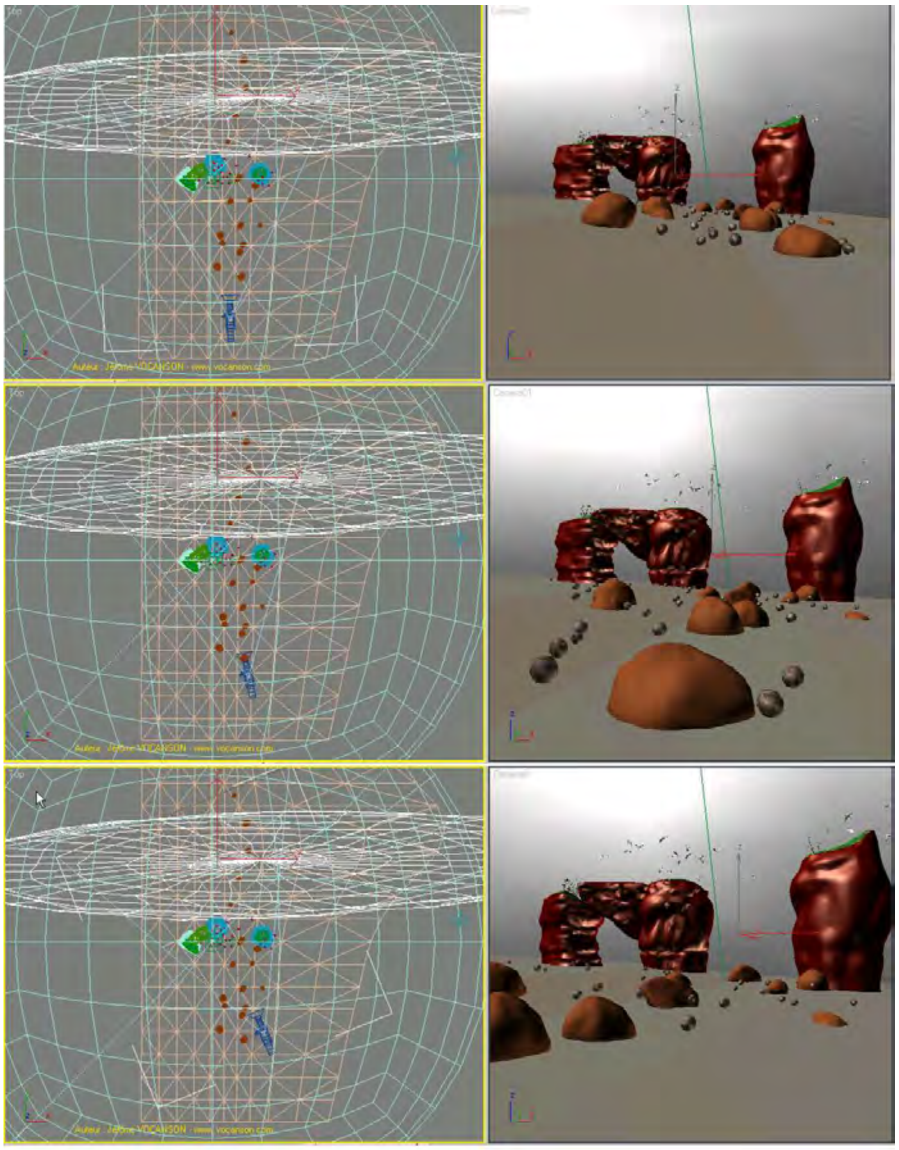

Auteur : Jérôme VOCANSON - www.vocanson.com
Auteur : Jérôme VOCANSON - www.vocanson.com
Auteur : Jérôme VOCANSON - www.vocanson.com

Figure 6.1: 3D model and camera trajectory

Based on this data, is is now possible to compute for any pixel of an image the ground truth corresponding pixel in all frames of the sequence, and - at the same time - the camera motion is completely known. As such, the ground truth depth is known (see bottom row of Figure 6.2) and also the ground truth optical flow (see Figure 6.3).

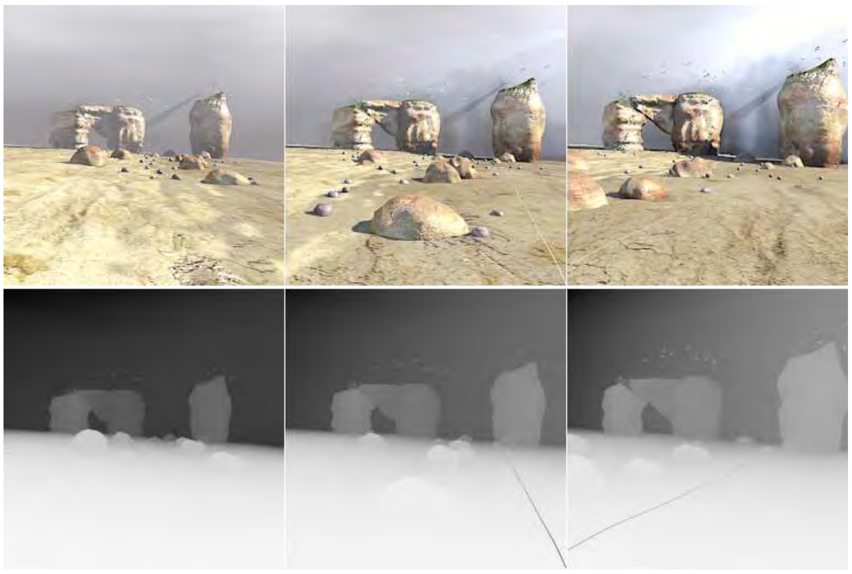


Figure 6.2: Some frames of the Seaside sequence, with the respective depth maps

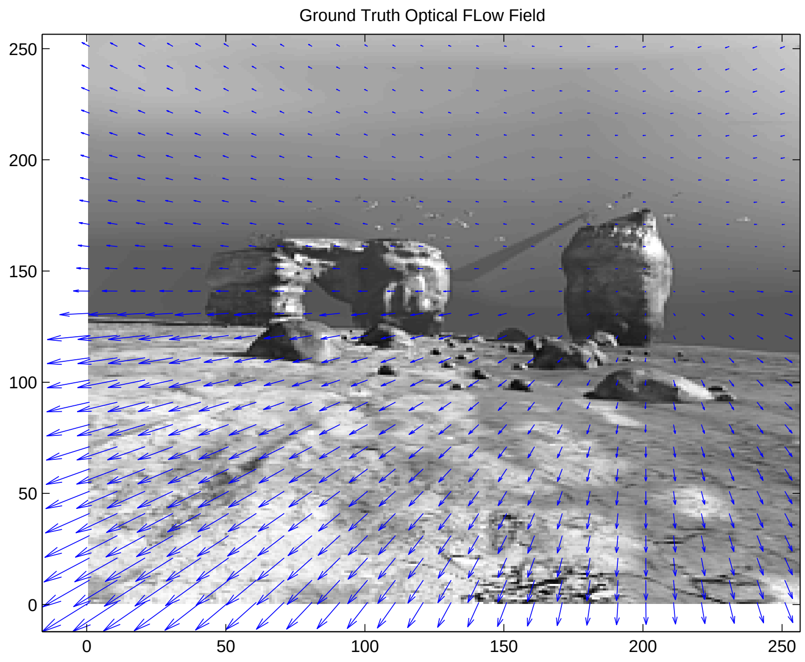
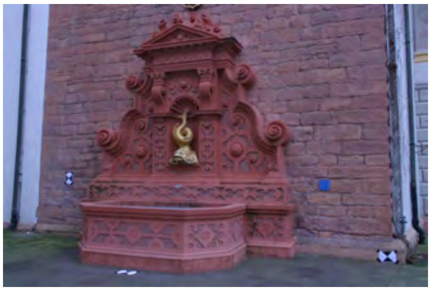


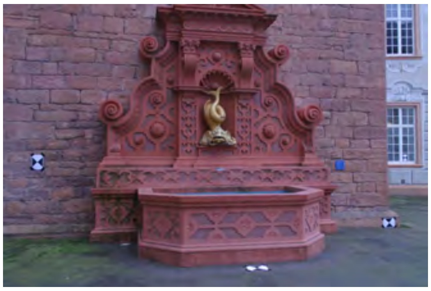
Figure 6.3: Ground Truth Optical Flow.

This allows us to extract useful data to analyze the whole calculation chain of the algorithm., leading to a comprehensive testing and analysis workflow for the evaluation of structure from motion algorithms. Current approaches mostly limit themselves in the evaluation to the reconstruction one specific scene. It is firstly hard to quantify the accuracy of a 3D reconstruction on paper and secondly, this leads the way for algorithms to be tuned towards certain scenes or motion patterns. In our workflow, it is straightforward to change the 3D scene and camera trajectories.

Although the advanced rendering techniques of state of the art 3D CAD programs output very realistic images, one must be cautious in judging image processing algorithms based on results based on synthetic input data. Therefore, in a second series of performance evaluations, the presented algorithms were tested on a traditional benchmarking sequence, provided by Strecha et al. [143]. The *fountain* sequence consists of a series of shots from a water fountain, as shown on figure 6.4, recorded by a high-resolution camera. The reconstruction results from this sequence serve to compare the performance of the proposed method to existing state-of-the-art approaches.

(a) Image 0


(b) Image 3


(c) Image 6

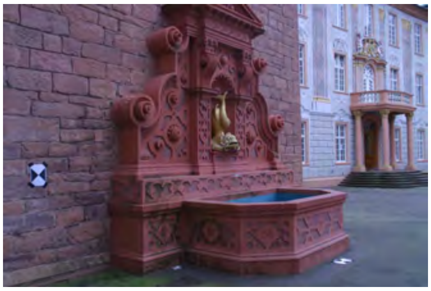

(d) Image 9

Figure 6.4: Some frames of the Fountain sequence [143]

To further validate our approach, we used in a third phase a natural outdoor sequence as input data. This sequence was recorded using a simple commercial low resolution hand-held camera in outdoor conditions. The sequence, which is shown on Figure 6.5, was filmed by a person walking around a statue featuring a mixed background of trees, buildings and traffic. Due to the nature of this sequence, no ground truth data is presentable, so the performance of the reconstruction algorithm needs to be visually judged from the final 3D reconstruction.

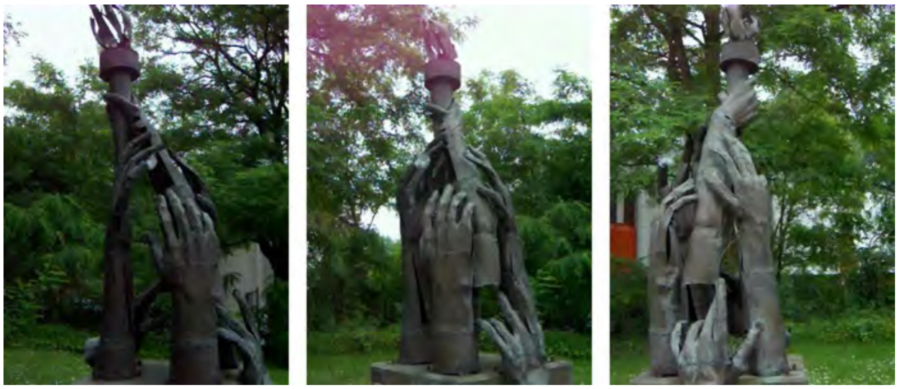


Figure 6.5: Some frames of the Hands sequence, shot with a commercial in-hand camera

A fourth image sequence used for experimental validation consists of a video, filmed with a low-quality commercial camera from within a vehicle, while driving around in town. Figure 6.6 shows some frames of this sequence which has a total of 41 frames. For this sequence, only the dense 3D reconstruction results are shown.

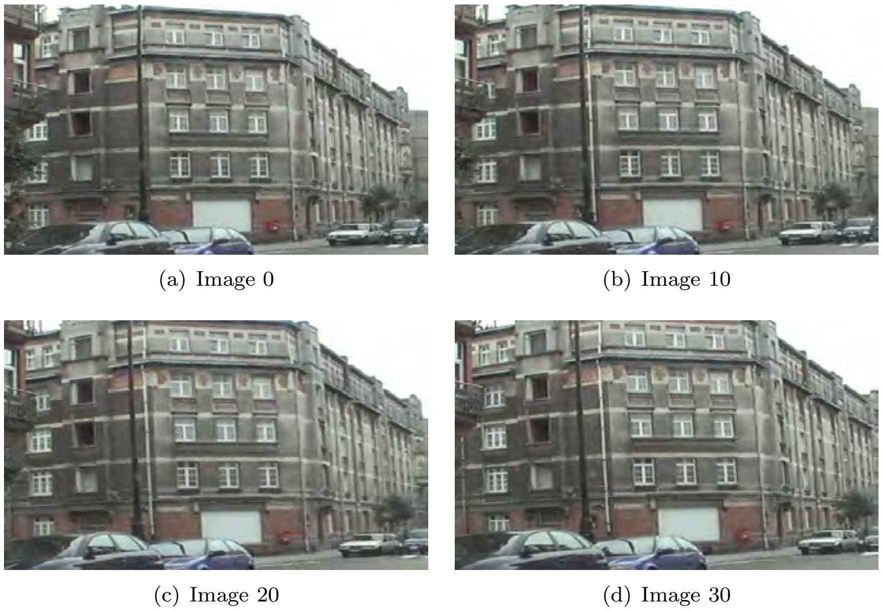


Figure 6.6: Some frames of the Street sequence

Choosing an evaluation sequence like the last 2 sequences which were presented above means that the reconstruction algorithm needs to deal with a number of critical problems:

- Erratic, non-deterministic movement which is hard to estimate
- Low resolution and low-fidelity input data (see for example the torch on the texture-mapped image which has turned from gray to violet in the second frame of Figure 6.5)
- Difficult outdoor variable lighting conditions.
- Relatively large distance from the camera to the subject
- Complex background

Most existing benchmark sequences for multi-view image reconstruction nicely avoid these problems. Typically small, simple objects are filmed with a camera following a well-described motion path.

6.2 Sparse Reconstruction

6.2.1 "Seaside" Synthetic Sequence

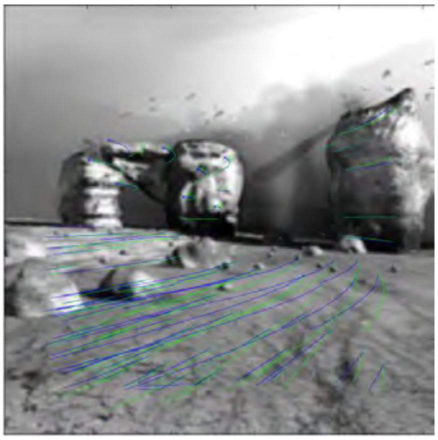
As described in section 2.4, we use SIFT features and kD-tree matching over multiple images to obtain a set of reliably matched features. Figure 6.7(a) shows (in blue) the trajectories of detected features in a sequence of 40 images. These trajectories can be compared to the ground truth match trajectories in green. It can be observed that, in general, both lines remain very close to one another.

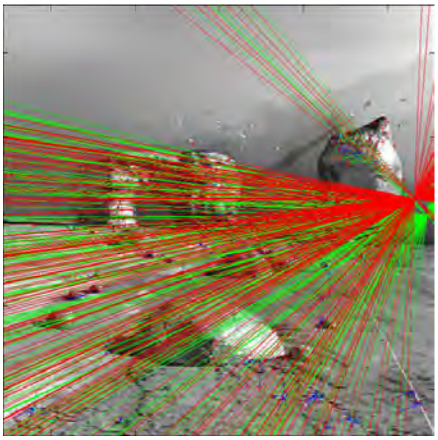
The first step in the epipolar reconstruction process is checking wether the epipolar geometry is the right model for the transform between the points $\mathbf{x}$ and $\mathbf{x}'$. When the distance between the two camera viewpoints is too small (framerate too low), the mapping can better be described by a homography and applying the epipolar model will likely lead to an unstable solution. Therefore, the optimal framerate must be calculated by evaluating the GRIC criterion as expressed by equation 3.1. The optimal framerate is first calculated for each camera frame individually. This is achieved by comparing the GRIC-scoring given by equations 3.3 for the homography model ($GRIC_H^{k\to\Delta k}$) and the GRIC-scoring for the epipolar model ($GRIC_F^{k\to\Delta k}$) for a frame k and a frame $k+\Delta k$, for increasing Δk. As soon as $GRIC_F^{k\to\Delta k} \geqq GRIC_H^{k\to\Delta k}$, Δk is said to be the optimal optimal amount of frames to skip starting from camera frame k. Figure 6.7(c) shows the optimal framerate according to each frame. To have a constant framerate over the whole sequence, we define the global optimal framerate to be the mean of the votes of all individual camera frames (in this case: 40 frames).

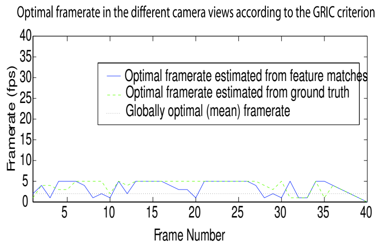
Figure 6.7(d) shows the residual of the RANSAC - algorithm for the estimation of the trifocal tensor T. This error is computed by calculating the re-projection error $\sum\limits_{features} \left([\mathbf{x}']_\times \left(\sum\limits_i x^i T_i \right) [\mathbf{x}'']_\times \right)^2$ with the matches and the calculated camera matrices from the trifocal tensor. It is thus a direct measure of how well the model (the trifocal tensor T) fits the data. The figure shows that the RANSAC algorithm converges.

The trifocal tensor itself, having 27 elements, is a concept which is hard to visualize. Therefore, we limit ourselves for the description of the multi-view geometry to an analysis of the two-view epipolar geometry derived from the trifocal tensor. Figure 6.7(b) compares for one image the ground truth epipolar lines (in green) with the estimated two-view geometry in red. The difference between both geometry representations is certainly not negligible. However, the estimate is sufficiently close to the correct result to allow it to be used in the further processing stages. Note also that this result is iteratively improved later during the PDE-based structure estimation, as explained in section 5.2.5.

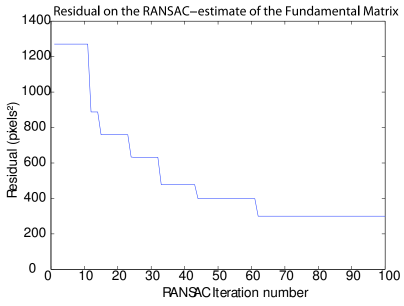
Figure 6.8 illustrates the results of the sparse structure from motion reconstruction process. For a sparse set of feature points, the 3D positions are estimated and the pose of each camera viewpoint is estimated. The camera pose estimated directly from three-view matches is shown in red, whereas the ground truth camera poses are represented by green cones. It can be noted that the estimated solution for the camera pose diverges from the ground truth solution. This is


(a) Trajectories of Matched Features: The estimated feature matches (in blue) compared to the ground truth matches


(b) Epipolar Geometry: The ground truth (green) and estimated geometry (red) compared



(c) Optimal Framerate Estimation for each frame according to the GRIC-criterion. The figure shows the GRIC score on the Y-axis for a number of frames (X-axis)



(d) Residual of the RANSAC Trifocal Tensor estimation process over a number of iterations. The figure shows the RANSAC residual on the Y-axis for a number of iterations (X-axis)

Figure 6.7: Sparse Reconstruction Results

normal, as no multi-view merging and integration is performed up until this point in the algorithm. The bundle-adjusted camera pose estimation, represented by blue cones, partly solves this divergence problem. This shows the strength of and the need for the bundle adjustment processing step.

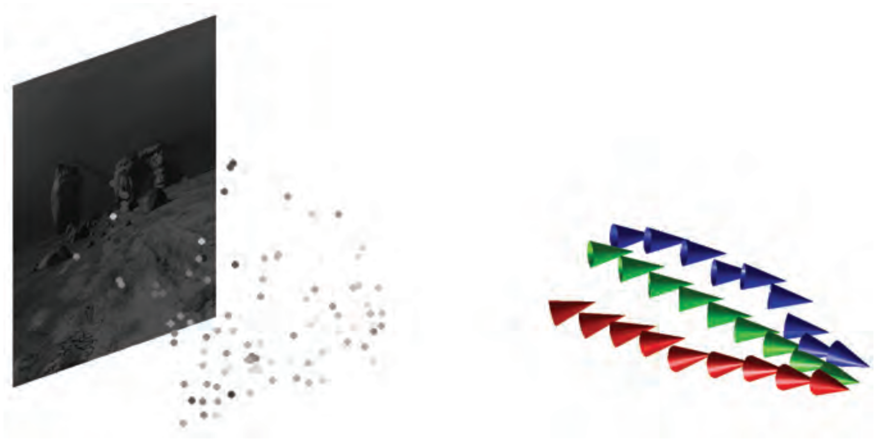
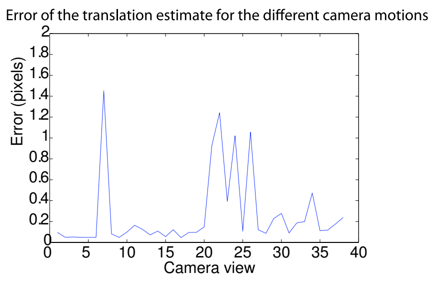
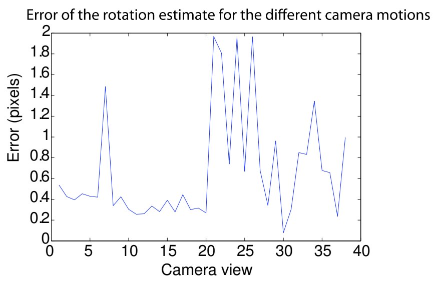


Figure 6.8: Estimated 3D sparse feature points and camera views for the synthetic sequence. (green = ground truth; red = three-view motion estimation; blue = after bundle adjustment)

Figure 6.8 shows a visual representation of the estimated structure and motion, but does not allow to judge the reconstruction fidelity. To better assess the accuracy of the motion and structure estimates, the square errors for the motion and structure parameters are plotted in figure 6.9. Figures 6.9(a) and 6.9(b) show the error on the estimation of the translation and rotation vector for the different camera views, measured as an Euclidian distance between the estimated and ground truth translation and rotation vector. The error on the sparse structure estimation is shown in figure 6.10. The errors on the estimation of the motion vectors are certainly significant, and they are more important for the rotation vector than for the translation vector, but the algorithm generally succeeds in providing an estimate which is close to the correct value and the error is close to zero. The remaining error peaks are caused by an erroneous choice of the motion parameters. Indeed, as indicated by equations 2.17 and 2.18, there are four possible rotation/translation pairs, and sometimes the algorithm selects the wrong one. It may be noted, from these error plots, that the camera views for which the estimation is worst, that is between frame 20 and frame 25, correspond to camera views for which the optimal framerate as calculated by the GRIC scoring function was much higher (5), than the eventually chosen globally optimal framerate of 2. It seems that for correctly estimating the motion vectors between these camera views, we should have skipped some frames. However, for reasons of consistency throughout the processing steps, it is necessary to choose 1 single



(a) Error on the estimation of the translation vector for the different camera views, measured as the Euclidean distance between the estimated and ground truth translation vector.



(b) Error on the estimation of the rotation vector for the different camera views, measured as the Euclidean distance between the estimated and ground truth rotation vector.

Figure 6.9: Errors on the Motion Estimation

globally optimal framerate, which we do by taking a mean of the framerates proposed by the GRIC criterion. This proves the usefulness of the GRIC criterion in predicting the optimal intra-frame step for correct motion estimation.

In theory, it should be possible to select for each frame the best framerate, which would lead to a non-uniform framerate over the whole sequence. However, this would mean that also in all subsequent processing steps, the framerate (step-size) between images should be treated as a variable and not as a constant. This would lead to major consistency problems in the numerical implementation, so we chose not to go this way.

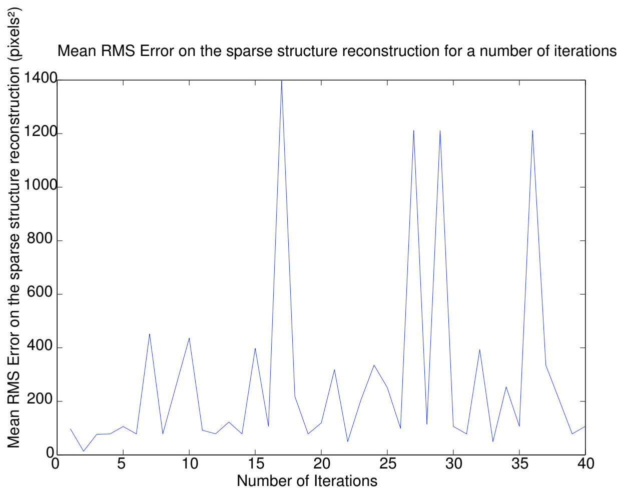


Figure 6.10: Errors on the Structure Estimation.

6.2.2 "Fountain" Benchmarking Sequence

For the natural *Fountain* and *Hands* sequences, only the end result of the sparse reconstruction is discussed. Figure 6.11 depicts the sparse reconstruction result for the *Fountain* sequence. The colored dots depict the reconstructed 3D point cloud and the blue line shows the camera trajectory.

The disperse nature of the point cloud makes it hard to see the structure of the 3D model. The estimated camera trajectory coincides with the ground truth camera trajectory within an error margin of 6.5%

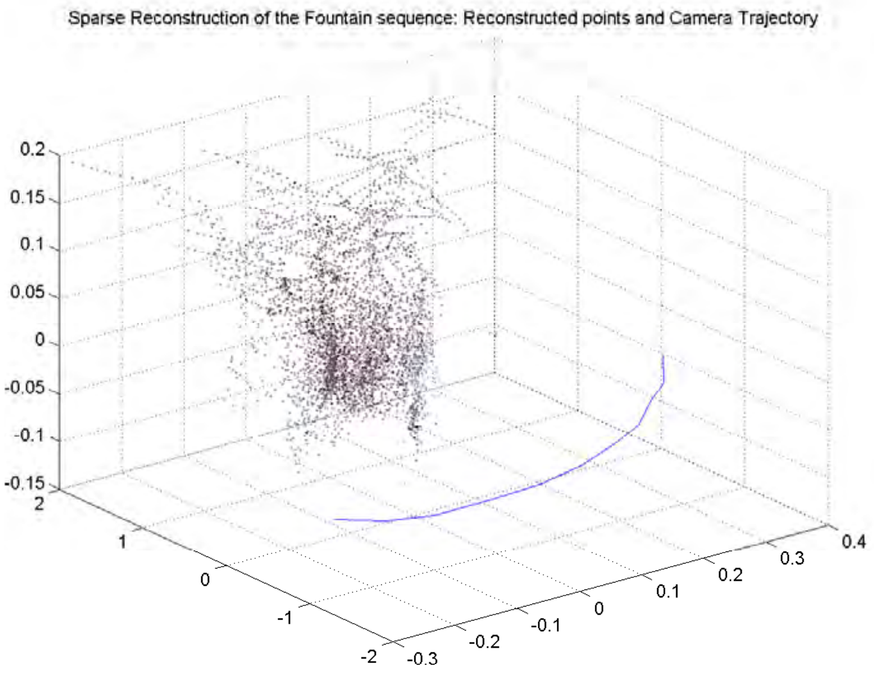


Figure 6.11: Estimated 3D sparse feature points and camera trajectory for the Fountain sequence.

6.2.3 "Hands" Natural Sequence

Figure 6.12 illustrates the results of the sparse structure from motion reconstruction process for the *Hands* sequence. The colored dots depict the reconstructed 3D points and the blue line shows the camera trajectory. As for this experiment the scene was filmed by an hand-held camera, the subsequent camera motions are certainly not totally the same, nor perfectly following some trajectory, although we tried to film in a smooth way.

As no ground truth data is present for this sequence, it is not possible to quantitatively assess the correctness of the presented result, but the general motion pattern and sparse feature reconstructions are consistent with respectively the camera movement and the real structure.

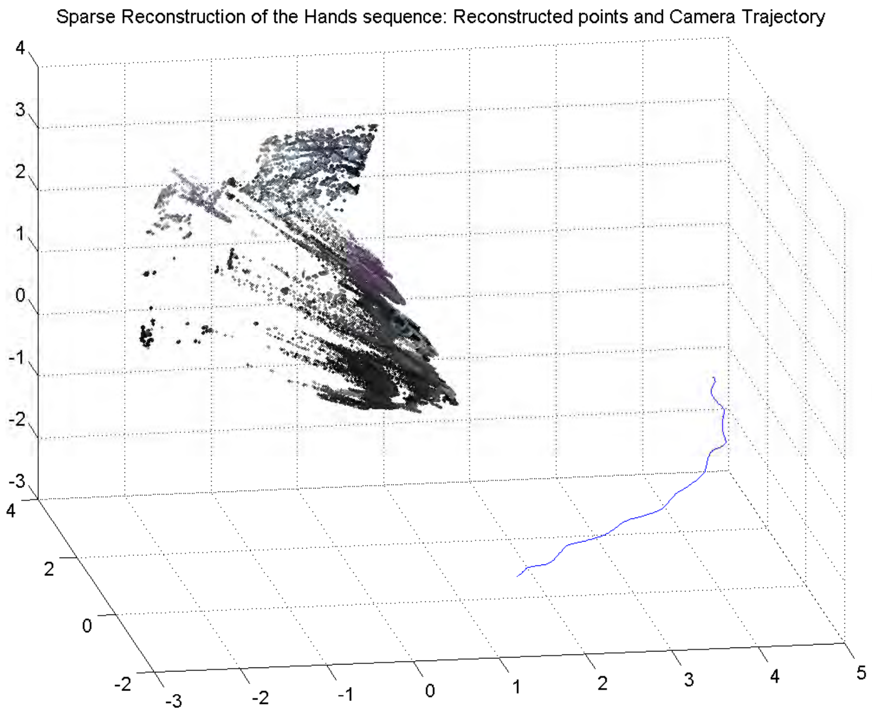


Figure 6.12: Estimated 3D sparse feature points and camera trajectory for the natural sequence.

6.3 Initialisation

6.3.1 "Seaside" Synthetic Sequence

Dense structure estimation relies on the calculation of an initial value for the depth map. As discussed in section 5.3.3, the estimation of the initial depth map is based on the fusion of dense data in the form of a densely estimated optical flow field and sparse data in the form of sparsely reconstructed feature points. Figure 6.13a shows the optical flow field. We used the optical flow estimation approach presented by Lucas and Kanade in [87]. The optical flow is represented as a vector field where each vector indicates a (scaled) image motion vector.

From the dense optical flow, a correctly scaled dense flow magnitude map u^{flow} is calculated by comparing sparse and dense data, as discussed in section 5.3.3. This flow magnitude map u^{flow} is shown on Figure 6.13b.

Sparse reconstruction, as discussed in chapter 3, yields a full 3D reconstruction for a limited set of feature points. From this information, a dense feature flow magnitude map $u^{features}$ is constructed by region growing, as discussed in section 5.3.3. This flow magnitude map $u^{features}$ is shown on Figure 6.13c.

The information contained in the two magnitude maps, u^{flow} and $u^{features}$, is combined to estimate an initial depth map map ζ_0, shown on Figure 6.13d, which can be used for initialization.

To measure the effect that errors on the translation or rotation vector and on the optical flow have on the initial value estimation, we measured the square root error between the estimated initial value and the ground truth depth for different amounts of Gaussian noise added to each of the algorithm parameters. The result of this analysis is shown in Figure 6.14. This figure plots the error on the initial value as a function of the percentage of additive Gaussian noise on the translation and rotation vector and on the optical flow.

From this analysis on figure 6.14, it is clear that the initialization result is very sensitive to the camera translation parameter. Normally distributed errors on the optical flow and the rotation vector tend to have far less influence on the final error of the initialization. As can be noticed on Figure 6.14a, the error values (on the Y-axis) are much higher than for Figures 6.14b and 6.14c. It is apparent that the result of the initialization is most sensitive to errors on the translation vector.

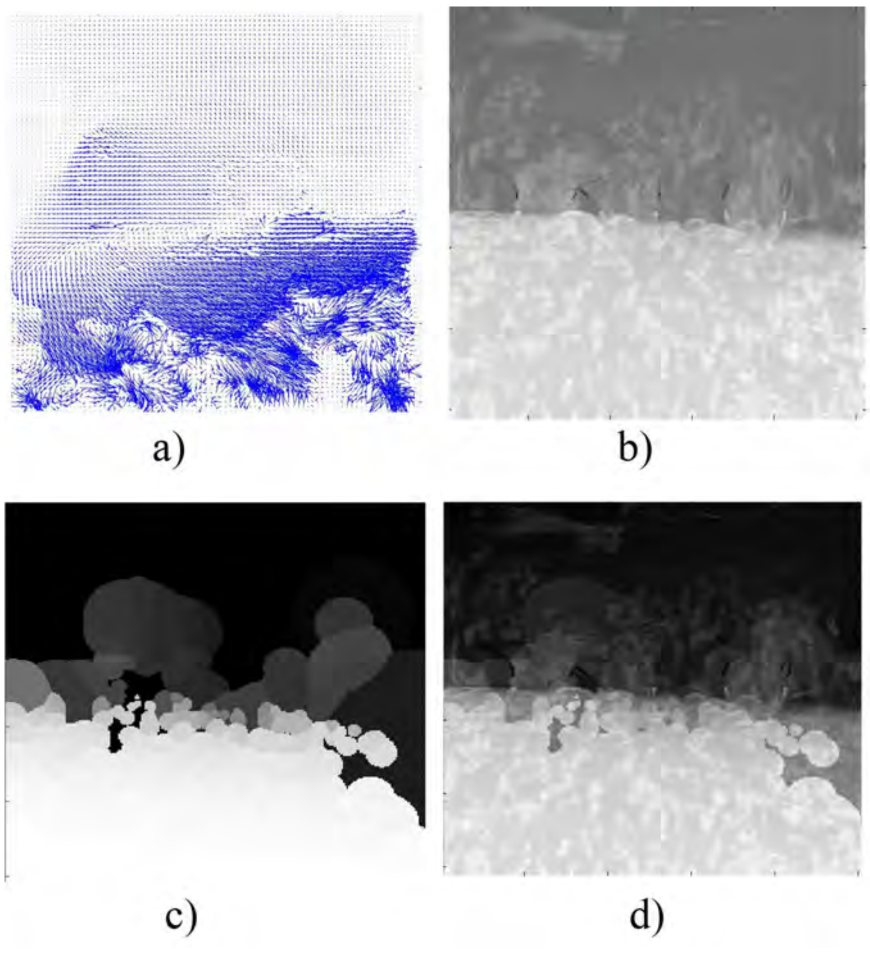


Figure 6.13: Initialization for a Synthetic Sequence: a) Optical Flow; b) Disparity Map reconstructed from the optical flow u^{flow}; c) Disparity Map estimated from sparsely reconstructed features $u^{features}$; d) Final initial value ζ_0

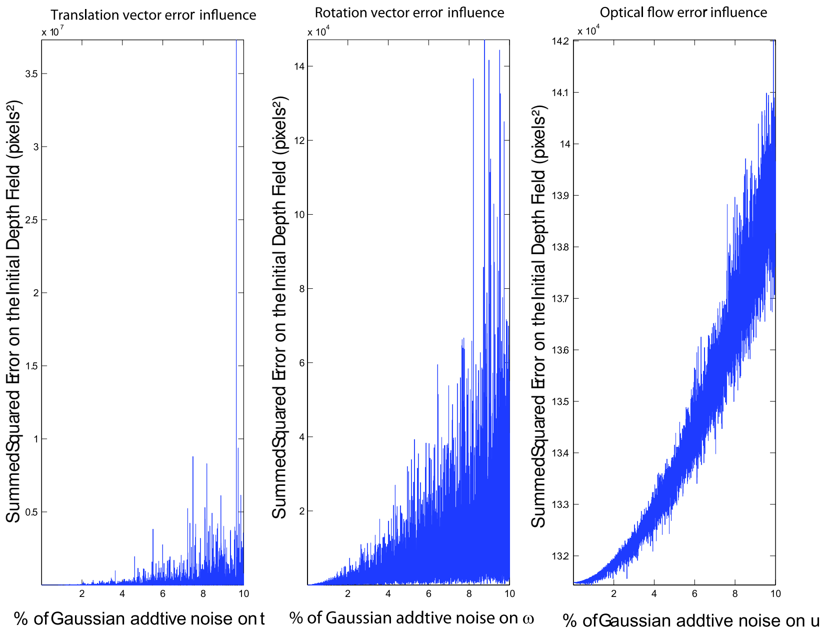


Figure 6.14: Total accumulated error $(\sum(\zeta_{Estimated} - \zeta_{GroundTruth})^2)$on the initialization as function of the error on a) the translation vector, b) the rotation vector, c) the optical flow. The X-axis indicates the percentage of additive Gaussian noise.

6.3.2 "Fountain" Benchmarking Sequence

Figure 6.15 shows the result of the initial estimation of the disparity map for the *Fountain* sequence, while Figure 6.16 shows the initial flow field. From this initialization of Figure 6.15, the form of the water fountain can already be distinguished. However, the initial estimate lacks detail, most notably on the back wall. This is probably due to the fact that too few features were chosen in this zone. One of the disadvantages of the feature detection and matching approach adopted in this work, is that it does not enforce spatial spreading of the features. This means that if there are feature-rich zones in the image, such as the fountain itself and the side walls, then no features are chosen in the other areas of the image, which will cause the initialization to fail in these zones.

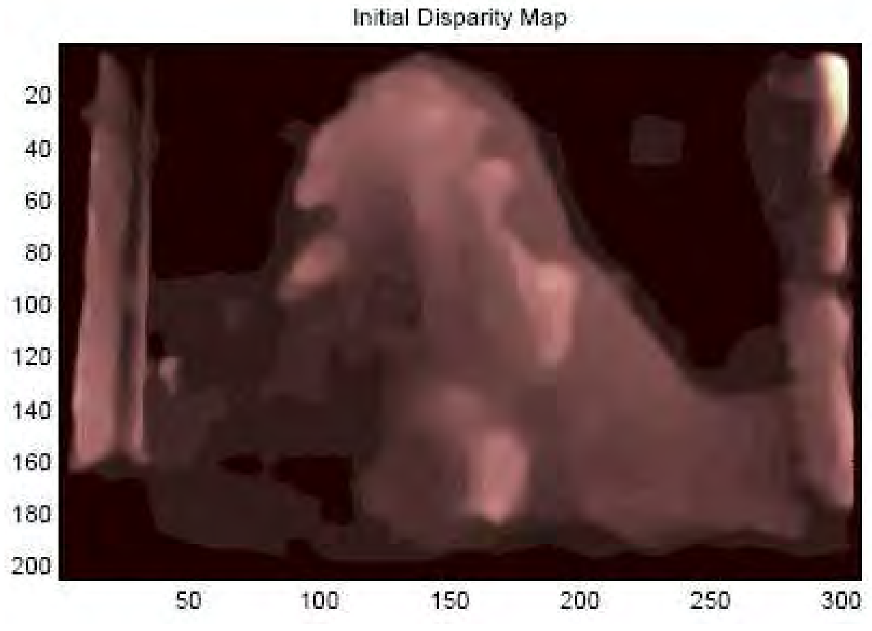


Figure 6.15: Initial Estimation of the Disparity Map for the Fountain Sequence.

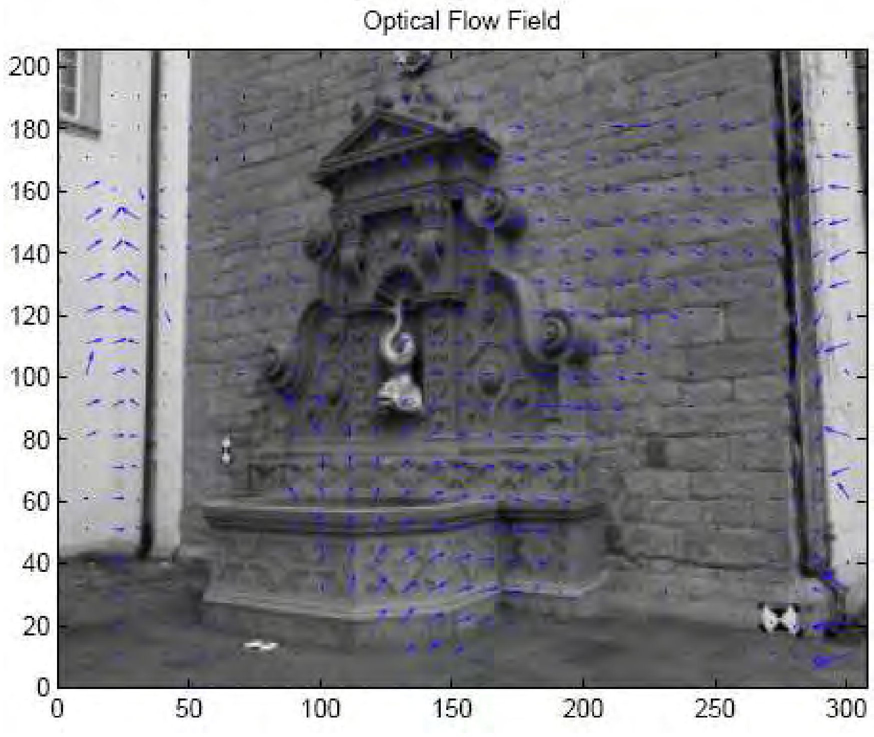


Figure 6.16: Initial Estimation of the Optical Flow for the Fountain Sequence.

6.3.3 “Hands” Natural Sequence

Figure 6.17 shows the result of the initial disparity map estimation for the natural sequence. Figure 6.17(a) shows the densely estimated optical flow field from which a dense depth and disparity map can be estimated. The statue on the foreground is clearly distinguishable in the flow field. Figure 6.17(b) shows the disparity map $\delta^{features}$ from sparse feature reconstruction. The final disparity map ζ_0, which is used as an initial value for the optimization scheme is shown on Figure 6.17(c).

The reconstruction result of Figures 6.17(b) and 6.17(c) show some errors in the top left and right of the image. The movement of the weaving treetops has resulted in sparse reconstruction errors in these areas, leading to an erroneous result. Also, the sky area was estimated too close because of a lack of features.

A quantitative measure of the error on this initialization result is not possible, as no ground truth data is present for the natural sequence, but by comparing Figure 6.17(c) with the color images of Figure 6.5, one can conclude that in general, the foreground and background are well-separated and that the depth gradient over the structure is correctly estimated. Figure 6.17(c) as such proves that it is possible to obtain a reasonable initial estimate for the depth field with this method, even in outdoor scenes with difficult lighting conditions.

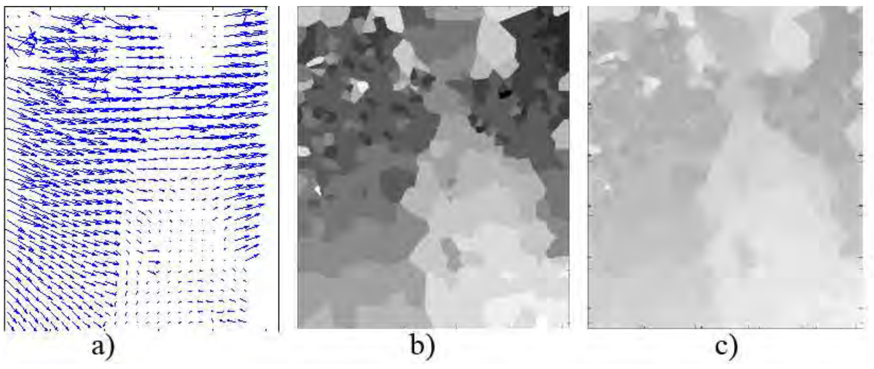


Figure 6.17: Initialization for a Natural Sequence: a) Optical Flow; b) Depth From Features; c) Estimated Initial Value ζ_0

6.4 Dense Reconstruction

In this section, the reconstruction results are discussed with different parameter settings. The reconstruction result consists of an estimate for the depth map and the optical flow field after the iteration. From the dense depth information, a 3D model is reconstructed. For scenes with ground truth data, the obtained results are compared to the ground truth data. Therefore, the following metrics are employed:

- $$Error_{Depth} = \sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(\zeta(i,j) - \zeta_{GroundTruth}(i,j))^2}{MN}} \tag{6.1}$$

  with ζ the estimated depth and $\zeta_{GroundTruth}$ the ground truth depth.

- $$Error_{Flow} = \sqrt{\frac{\sum_{k=1}^{2}\sum_{i=1}^{M}\sum_{j=1}^{N}(\mathbf{u}(i,j,k) - \mathbf{u}_{GroundTruth}(i,j,k))^2}{2MN}} \tag{6.2}$$

  with $\mathbf{u}$ the estimated optical flow field and $\mathbf{u}_{GroundTruth}$ the ground truth optical flow.

- $$Error_{Translation} = \sqrt{\frac{\sum_{i=1}^{3}(\mathbf{t}(i) - \mathbf{t}_{GroundTruth}(i))^2}{3}} \tag{6.3}$$

  with $\mathbf{t}$ the estimated translation vector and $\mathbf{t}_{GroundTruth}$ the ground truth translation vector.

- $$Error_{Rotation} = \sqrt{\frac{\sum_{i=1}^{3}(\omega(i) - \omega_{GroundTruth}(i))^2}{3}} \tag{6.4}$$

  with ω the estimated rotation vector and $\omega_{GroundTruth}$ the ground truth rotation vector.

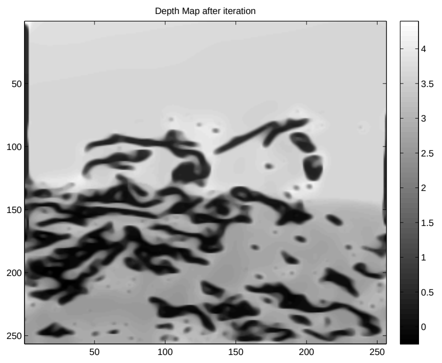
- *Accuracy*[129]: Measures the the similarity between the reconstructed and the ground truth model by calculating the distance d such that a given percentage of the reconstruction is within d from the ground truth model. Seitz [129] uses an accuracy threshold of 90%, i.e., an accuracy of $1.0mm$ means that 90% of the points are within one mm of the ground truth model.

- *Completeness*[129]: Measures the the similarity between the reconstructed and the ground truth model by calculating the percentage of the ground truth model that is within a given distance from the reconstruction. For completeness, Seitz uses an inlier threshold of $1.25mm$, i.e., a completeness of 95% means that 95% of the points are within $1.25mm$ of the ground truth model.

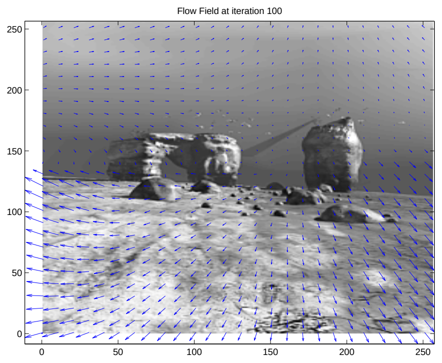
6.4.1 “Seaside” Synthetic Sequence

The dense reconstruction algorithm, discussed in chapter 5, introduces two distinct ways of formulating the energy minimization problem. The first approach uses the constant image brightness constraint of equation 5.4 to come to the formulation expressed by equation 5.28. The second method uses the image derivatives - based optical flow constraint of 5.5 to obtain the formulation given by equation 5.29. Moreover, chapter 5 presented some methodologies to enhance the reconstruction process, such as automatically estimating the degree of regularization (as discussed in section 5.3.2) and iteratively updating the estimate of the motion parameters (as discussed in section 5.2.5). Experimental validation must point out which of these formulations delivers the best results and whether the proposed enhancements do deliver an added value for the dense reconstruction result. Therefore, the analysis in this section is split up for each of the proposed formulations, such that a quantitative and qualitative comparison is possible and such that the effect of each of the enhancements can be measured.

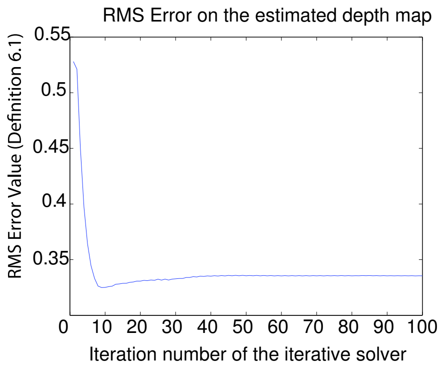
a) Formulation 1 given by Equation 5.28, not using the diffusion parameter update, not using the motion parameter update Figure 6.18 shows the results of dense reconstruction using the formulation of Equation 5.28. When analyzing the reconstructed depth map of Figure 6.18(a), it is clear that the resulting depth map features too much contrast, rendering the depth map unusable. This behavior is due to the nature of the image brightness constraint of equation 5.4, which relies on a direct per-pixel comparison of image brightness values. This approach is highly susceptible to mismatches, leading to badly reconstructed pixels and zones, showing up as the black areas on the depth map of Figure 6.18(a). The evolution of the reconstruction error during the iterative process, measured according to equation 6.1, is visualized on Figure 6.18(c). It can be noted that the error drops significantly, indicating that the algorithm does converge. However, the error doesn’t substantially evolve after about 10 iterations. This behavior is due to the choice of the constraint equation: the mismatched zones cannot be reconstructed correctly and lead to a steady-state error. For the optical flow estimation, whose end result is shown on Figure 6.18(b), the situation is better: the final optical flow field of Figure 6.18(b) is very close to the ground truth flow field of Figure 6.3. This can also be judged from the evolution of the error on the optical flow, measured according to equation 6.2, as shown on Figure 6.18(d). The error on the optical flow shows a descending behavior, indicating that the algorithm using the first constraint equation is better suited for optical flow estimation, than for dense depth estimation.



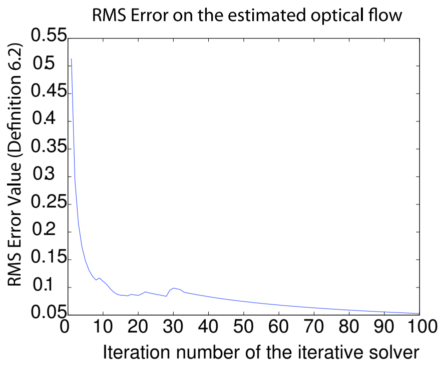
(a) Final Depth Map



(b) Final Optical Flow



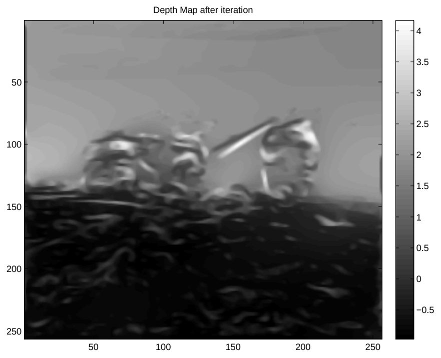
(c) Error on the Depth Map



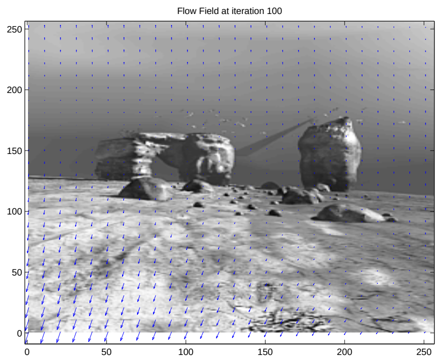
(d) Error on the Optical Flow

Figure 6.18: Performance Evaluation on the Seaside Synthetic Sequence. Parameters: First Optical Flow Constraint, not using the diffusion factor update; not using the motion parameter update.

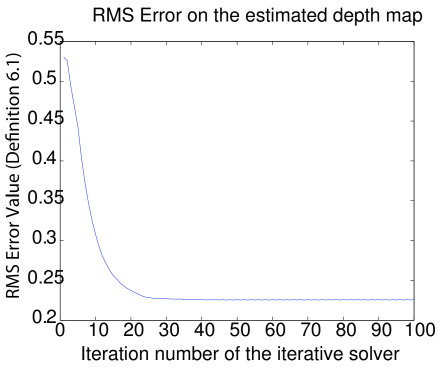
b) Formulation 2 given by Equation 5.29, not using the diffusion parameter update, not using the motion parameter update When comparing the results of Figure 6.18 with those of Figure 6.19, which shows the same experimental results, using the second formulation of Equation 5.29, based upon the image - derivatives based optical flow constraint, it becomes immediately apparent that the depth reconstruction of Figure 6.19(a) is much better. The relative depths of all objects in the scene are much better represented in Figure 6.19(a), using the second formulation, than in Figure 6.18(a), using the first formulation of Equation 5.28. This is because the second formulation of Equation 5.29 does not rely on per-pixel image brightness differences, but rather on image derivatives, which is a more spatially and temporally coherent measure. The remaining errors in the depth reconstruction of Figure 6.19(a) can mainly be found at the edges of objects. This is to be expected, because at these locations, the spatial derivatives are often not consistent and lead to erroneous depth estimations. The depth reconstruction result can also be judged from analyzing the error on the depth map during the iterative process, as visualized by Figure 6.19(c), which shows a better behavior than Figure 6.18(c) in the previous case. However, also here, there is a remaining steady-state error, as the errors due to the spatially incoherent image gradient calculation cannot be removed entirely. Also due to this fact, the optical flow estimation result using the second constraint equation is slightly less good than in the previous case, as evidenced by Figures 6.19(b) and 6.19(d). This is, however, not our main concern, as the algorithm is supposed to be in the first place a depth estimation algorithm, not an optical flow estimation algorithm. The conclusion from this comparison is thus that the second formulation is to be preferred when considering depth reconstruction and as a result, it is only this formulation which we'll discuss further on.



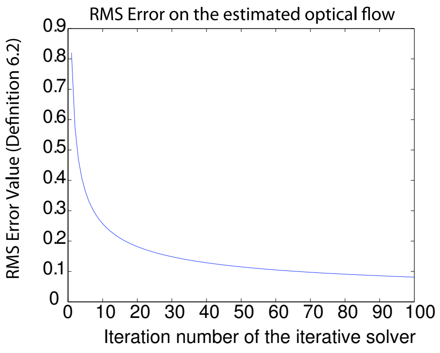
(a) Final Depth Map



(b) Final Optical Flow



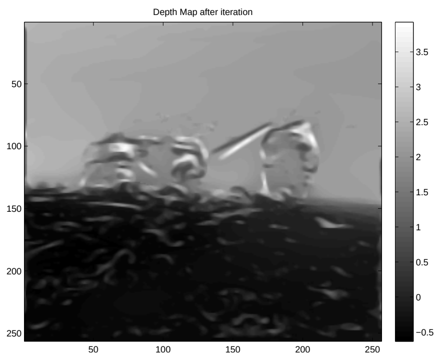
(c) Error on the Depth Map



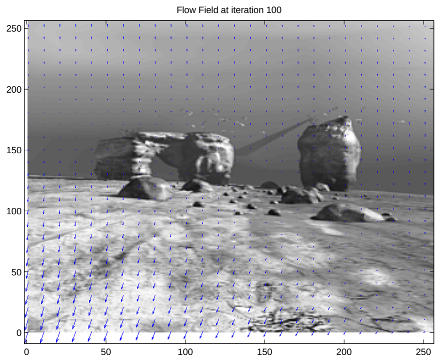
(d) Error on the Optical Flow

Figure 6.19: Performance Evaluation on the Seaside Synthetic Sequence. Parameters: Second Optical Flow Constraint, not using the diffusion factor update; not using the motion parameter update.

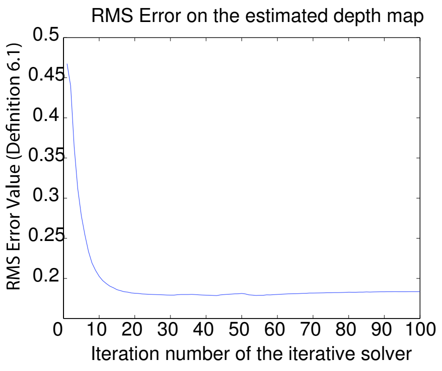
c) Formulation 2 given by Equation 5.29, using the diffusion parameter update, not using the motion parameter update The experimental results discussed above did not consider the use of the update of the diffusion parameter μ. To evaluate the importance of the introduction of this additional aspect of the presented algorithm, Figure 6.20 shows the reconstruction results when using the diffusion parameter update, as explained in section 5.3.2. By comparing the results of Figures 6.19(c) and 6.20(c), it can be noted that the final error on the depth reconstruction is substantially lower when using the diffusion parameter update, allowing us to conclude that the inclusion of the diffusion parameter update presents a valuable added value for the algorithm.



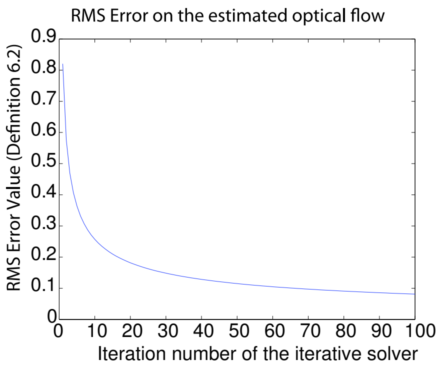
(a) Final Depth Map



(b) Final Optical Flow



(c) Error on the Depth Map



(d) Error on the Optical Flow

Figure 6.20: Performance Evaluation on the Seaside Synthetic Sequence. Parameters: Second Optical Flow Constraint, using the diffusion factor update; not using the motion parameter update.

The evolution of the diffusion parameter value during the iterative process is shown in Figure 6.21. It can be noted that the diffusion parameter increases gradually to boost regularization in the early phases of the iterative process. However, when the solver has converged to a static solution, the diffusion parameter also settles for a static value of - in this case - about 0.65.

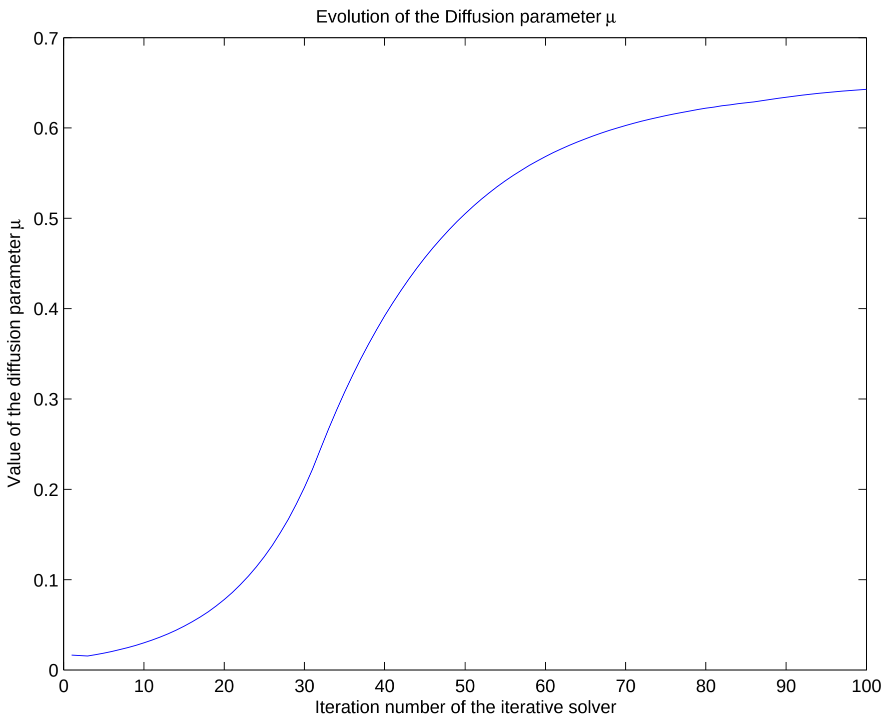
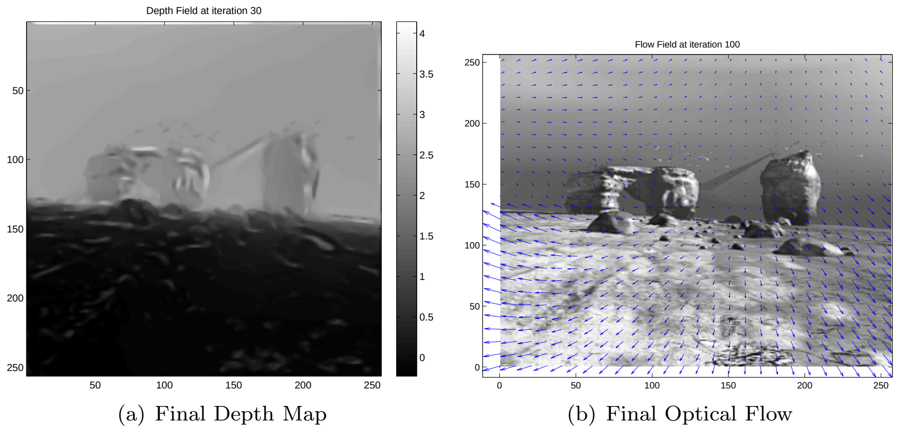
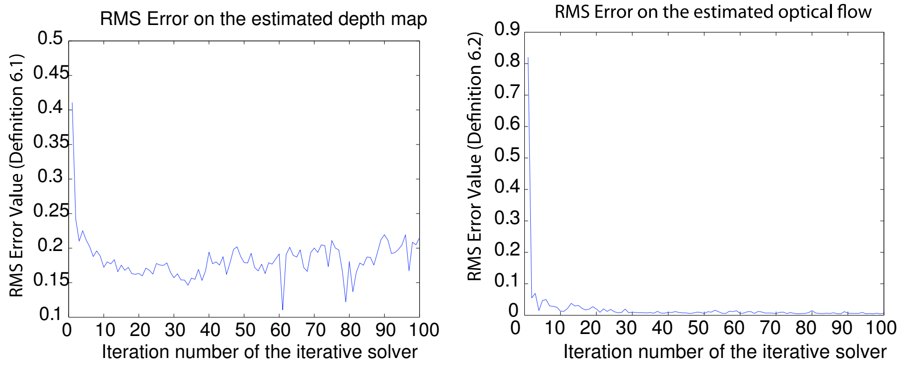


Figure 6.21: Evolution of the Diffusion Parameter μ

d) Formulation 2 given by Equation 5.29, using the diffusion parameter update, using the motion parameter update A second aspect of the computational work-flow which was not considered in the experimental results presented so far, is the update of the estimates for the motion parameters, as discussed in section 5.2.5. In order to evaluate the usefulness of the motion parameter update, Figure 6.22 shows the reconstruction results when using motion parameter updating.



(a) Final Depth Map (b) Final Optical Flow



(c) Error on the Depth Map (d) Error on the Optical Flow

Figure 6.22: Performance Evaluation on the Seaside Synthetic Sequence. Parameters: Second Optical Flow Constraint, using the diffusion factor update; using the motion parameter update.

Comparing the results of Figure 6.22 with those of Figure 6.20 shows that the final depth reconstruction result of Figure 6.22(a) has improved even further. Although the final error on the depth field is lower than in the previous cases, the error on the depth map, as shown on Figure 6.22(c), has a more fluctuating behavior. This is because the motion parameter estimation process gives at each iteration a new estimate for the motion parameters, which is a priori uncorrelated with the previous estimate. Since the depth estimation is highly susceptible to changes in the values of the motion parameters (see the analysis of Figure 6.14), this leads to a more fluctuating error value on the depth map. We did apply (Kalman) filtering to smooth these results, but did not want to go too far in this, in order not smooth out completely the results of the motion parameter update process.

The added value of using the motion parameter update is further evidenced by analyzing the resulting optical flow field, visualized in Figure 6.22(b) and the error on the flow field, shown on Figure 6.22(d). It can be noted that the error on the flow field decreases far better than in the previous cases. This is of course due to the fact that using this motion parameter updating technique, the final estimates for the motion parameters much better match the ground truth data, than the initial guesses for these motion parameters.

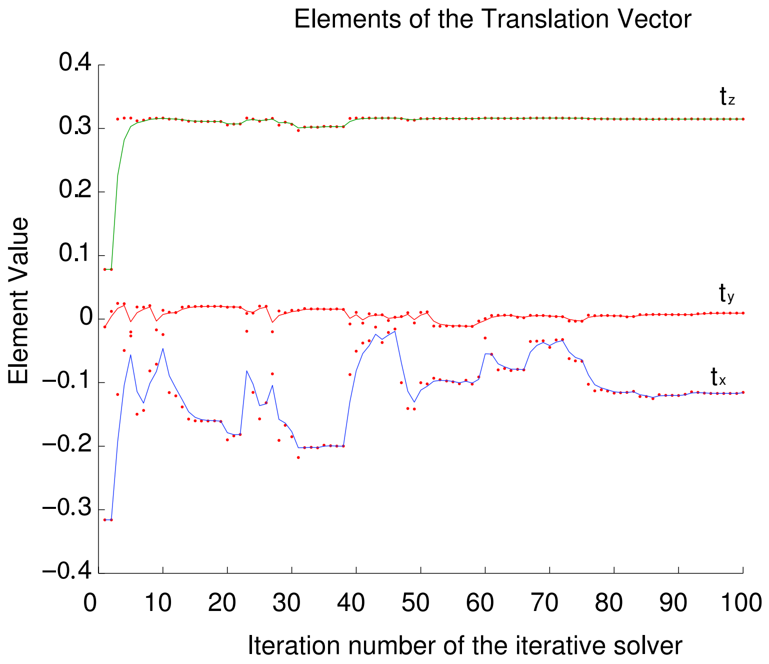


Figure 6.23: Translation Vector Elements

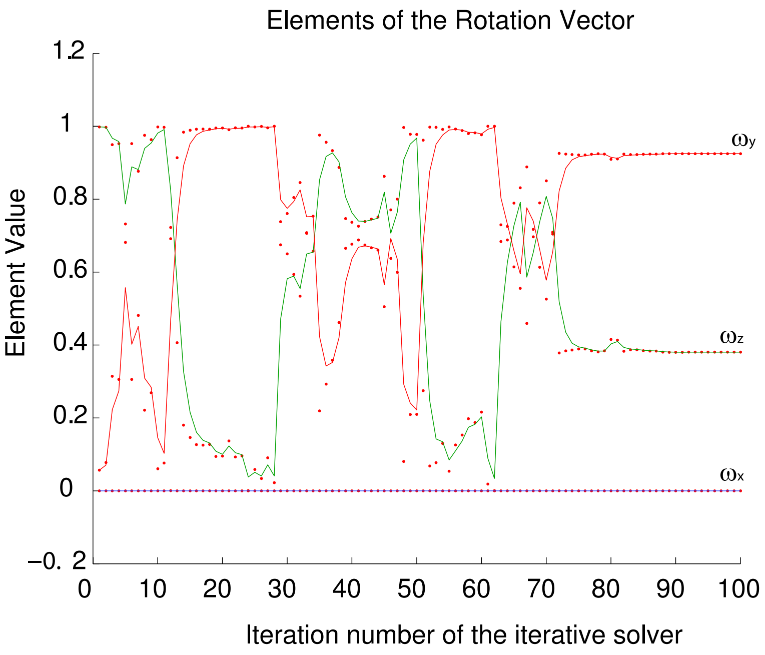


Figure 6.24: Rotation Vector Elements

The influence of the motion parameter updating process can be judged from Figures 6.23 and 6.24, showing the different components of, respectively the normalized translation and rotation vector. It can be seen that the solver often switches between two solutions. This is to be expected, as ego-motion estimation typically yields multiple solutions and it is hard for a computer algorithm to choose the correct one. This problem finds its cause in the singular value decomposition of the essential matrix, which yields 4 possible solutions for the camera projection matrix, as expressed by equations 2.18. The choice between these 4 transformations is not always obvious, as in some cases, there are multiple valid solutions.To deal with this issue, Kalman filtering of the translation and rotation vector estimates was used, and, in case of doubt, the solution which is closest to the Kalman filter prediction is chosen. However, because it is possible to have discontinuous motion patterns, the filter still accepts abrupt changes for each of the motion vectors. Due to this behavior, the estimation of the rotation vector still switches between different solutions, as shown by Figure 6.24. This could be remedied by restricting the step change of the rotation vector, but we chose not to do this in order not to miss discontinuities in the movement pattern. Another possible improvement of the motion parameter update process would be to use the LevenbergMarquardt algorithm for solving the least squares problem posed by equations 5.30 and 5.31, instead of the Gauss-Newton algorithm which was implemented here. The Gauss-Newton algorithm was chosen for its simplicity, but it does have problems, most notably when the matrix $\left(\mathbf{J}^T\mathbf{J}\right)$ is near singular.

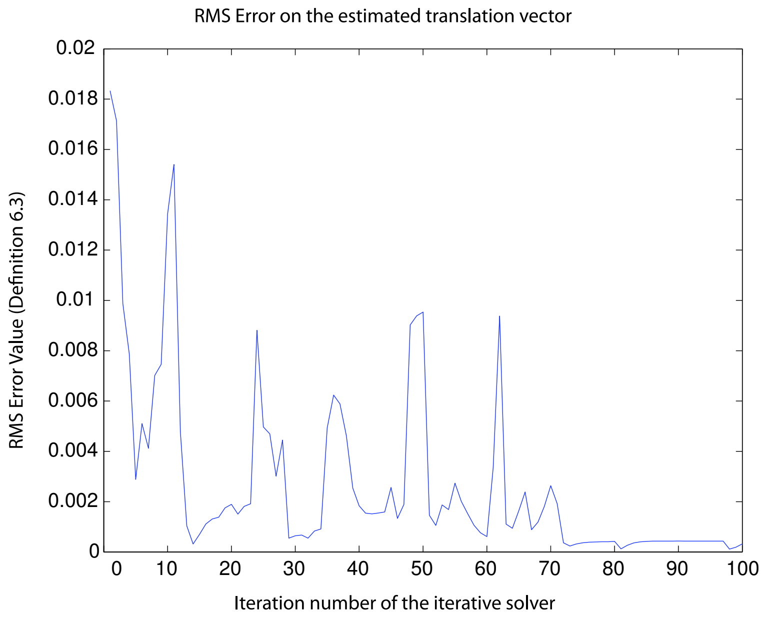


Figure 6.25: Error on Translation Vector

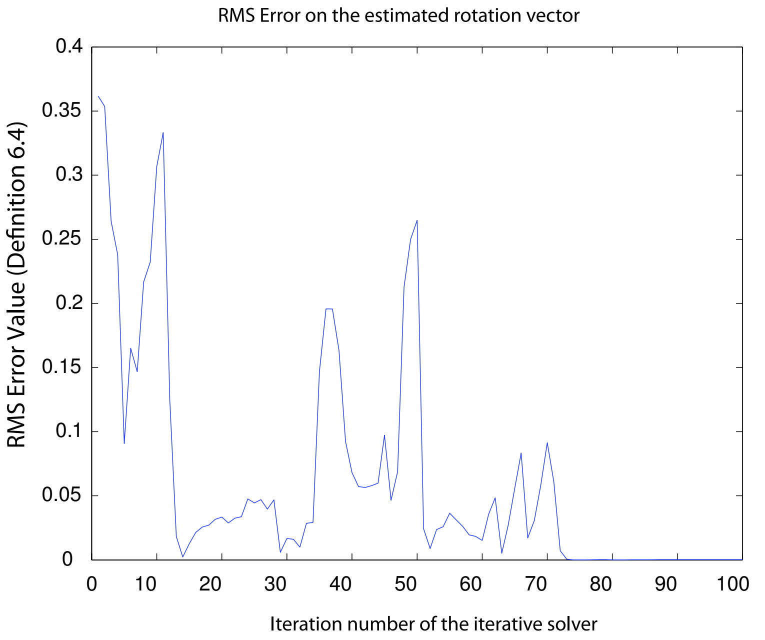


Figure 6.26: Error on Rotation Vector

Figures 6.25 and 6.26 show the errors on these parameters, according to metrics 6.3 and 6.4. Notice that the final error on translation as well as on rotation is substantially lower than the initial value, allowing us to conclude that the motion parameter update achieves in enhancing the end result of the structure and motion estimation algorithm.

Figure 6.27 shows the evolution of the diffusion parameter μ during the iterative process and shows a similar behavior as in the previous case of Figure 6.21. First, the diffusion parameter increases to speed up regularization and later, when the solver converges to a valid solution, the diffusion parameter also converges settles to a value which is about the same as in the previous case.

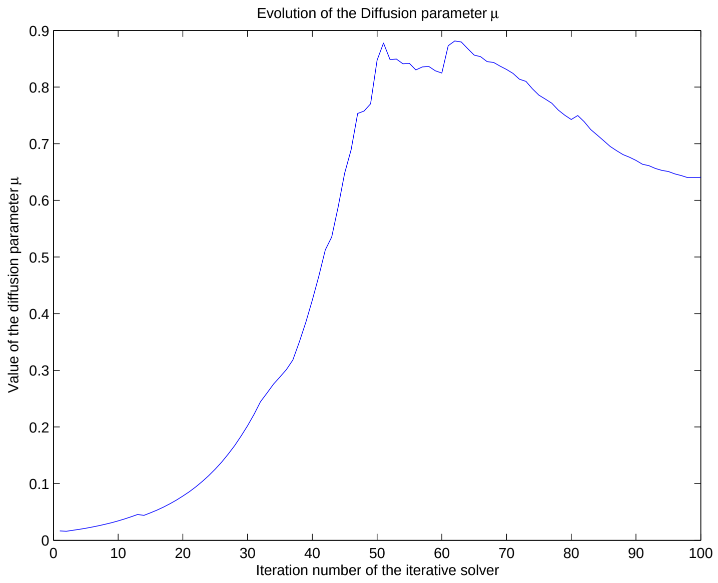


Figure 6.27: Evolution of the Diffusion Factor μ

Figure 6.28 shows the residual of the energy function, decreasing drastically during the iterative process, which shows that the algorithm converges.

Figure 6.29 shows a 3D view of the difference between the estimated depth map and the ground truth depth map. It can be noted from this map that the main errors occur at the horizon line and at the borders of the image, due to the spatially incoherent gradient estimates at these locations.

As a quantitative analysis, Figure 6.30(a) shows the model accuracy measure, which is represented as the evolution of the error in function of the percentage of the reconstruction which is within the given error bound from the ground truth model. As this is a synthetic data set and an absolute error measure is thus not that relevant, the reconstructed and ground truth model were normalized and the error was represented as a percentage. It can be noted from Figure 6.30(a) that the relative error is very low. Even when considering a 95%, percentage of the reconstruction, there is only a small percentage (0.3%) of outliers.

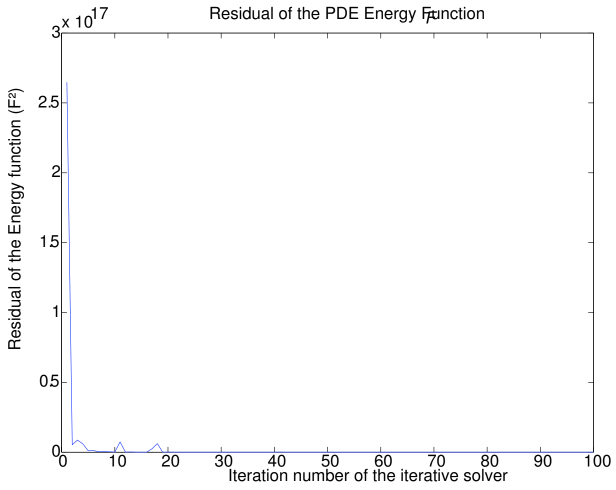


Figure 6.28: Evolution of the Residual

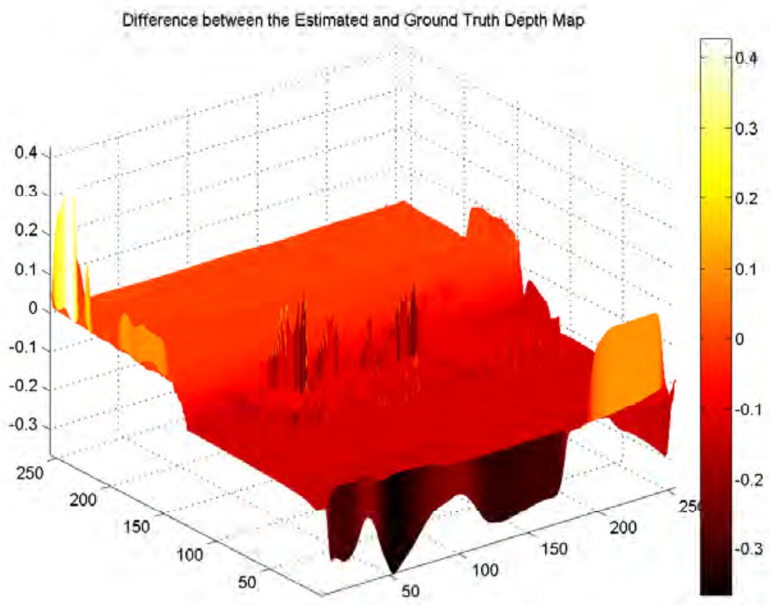
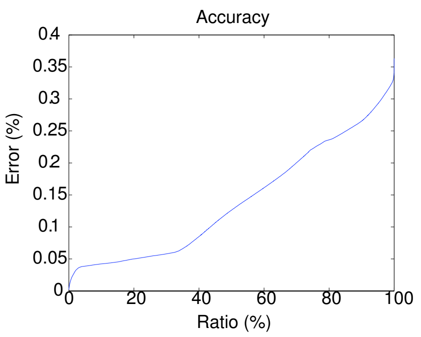
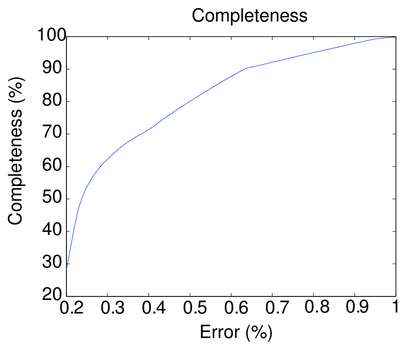


Figure 6.29: Difference Map between the Estimated and Ground Truth Depth Map



(a) Accuracy



(b) Completeness

Figure 6.30: Accuracy and Completeness measures (Definitions: see p. 100) of the Seaside Sequence.

Figure 6.30(b) shows the model completeness measure, which is represented as the evolution of the completeness in function of the error. The completeness is somewhat lower than anticipated, other reconstruction algorithms (e.g. [44]) are able to achieve a higher degree of completeness for lower error values. This is most likely due to the fact that we use standard ICP to integrate the individual reconstruction results. More advanced techniques for multi-view integration are now available, however, this is beyond the scope of this research work.

A qualitative analysis of the results of the structure and motion estimation algorithm over multiple frames is given by Figures 6.31 and 6.32. Figure 6.31 shows the reconstructed trajectory of the camera with the different frames. Figure 6.31 does not allow a direct comparison with the ground truth camera trajectory, because the estimated trajectory is too close to the real camera motion path, with respect to the scale of the figure, which can already be judged from Figures 6.25 and 6.26, showing the small final error on the translation and rotation estimation.

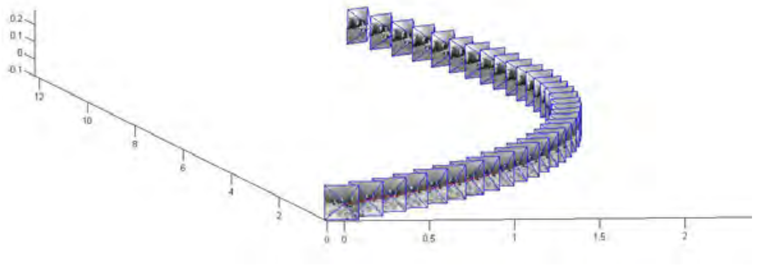


Figure 6.31: Reconstructed Camera Motion Path and Different Camera Views

The individual reconstruction results for all frames were integrated using the Iterative Closest Point (ICP) method [185] to form one consistent 3D representation of the imaged environment. Figure 6.32 shows the 3D model which was reconstructed as such. Comparing this model to the ground truth model of Figure 6.33 shows a strong resemblance between both 3D models.

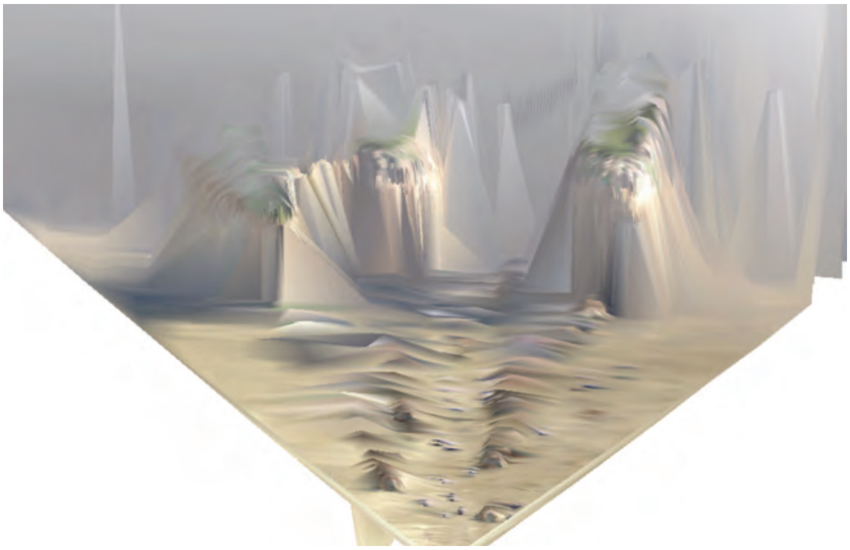


Figure 6.32: Reconstructed 3D Model of the environment

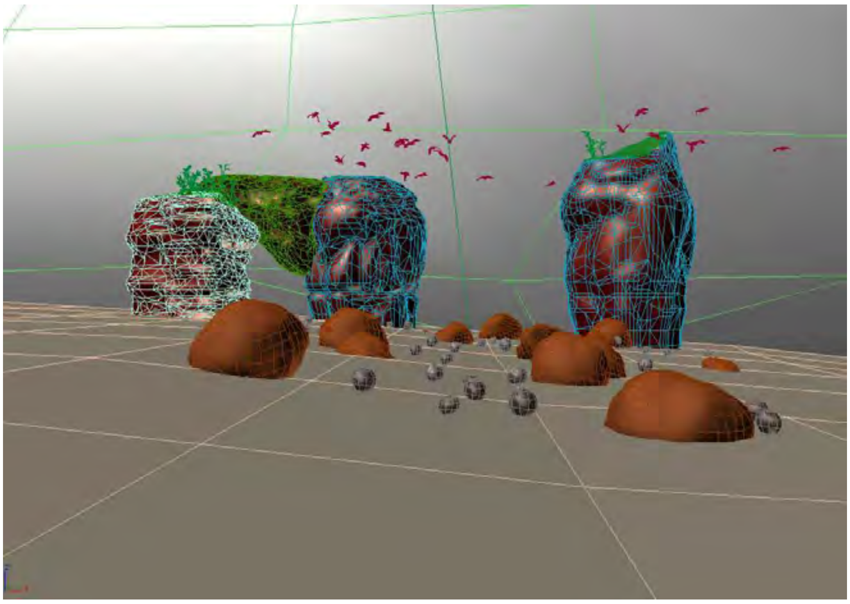
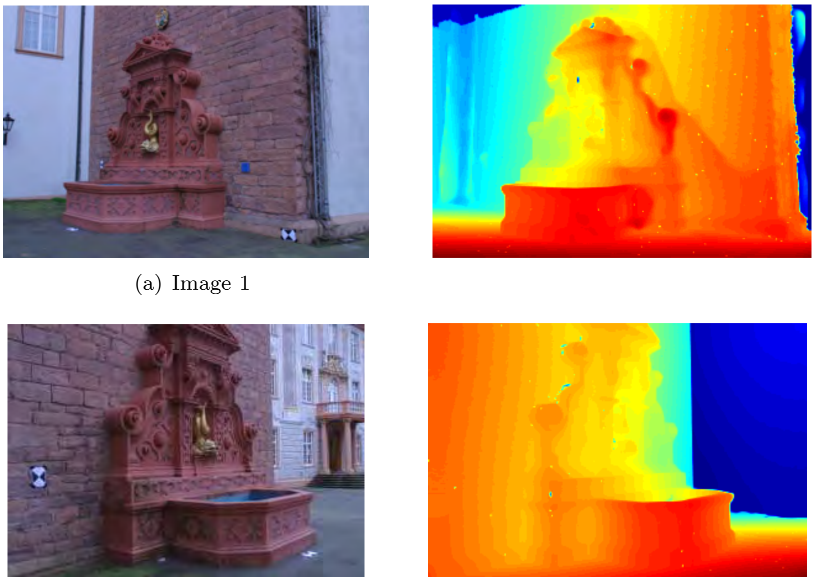


Figure 6.33: Original 3D Model of the environment

6.4.2 "Fountain" Benchmarking Sequence

Figure 6.34 shows the reconstructed depth maps for some frames of the *Fountain* sequence. It can be noted that the depth maps present a visually excellent reconstruction of the scene structure.



(a) Image 1

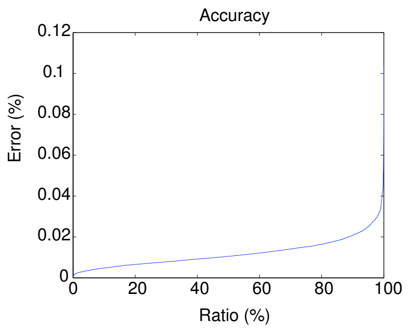
(c) Image 8

(d) Reconstructed Depth Map 8

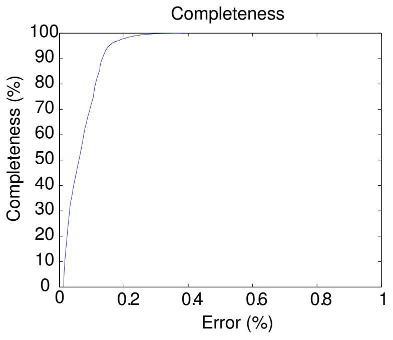
Figure 6.34: Dense Reconstruction Results for the Fountain Sequence: Depth Maps

As a quantitative measure, the accuracy and completeness measures for the fountain sequence are measured. These are shown in Figure 6.35. The accuracy and completeness measures follow a similar general behavior as with the *Seaside* sequence. The accuracy measure shows that the error stays low for a larger number of pixels (higher ratio), and the completeness rises more quickly than for the *Seaside* sequence.

The accuracy and completeness results can be compared to the results of other algorithms on the same sequence, as reported in [143]. Table 6.1 compares the numerical values of the 90% accuracy and 1.25mm completeness measures of the presented method with 3 other methods [143]. From this analysis, it is clear that the proposed algorithm is not the best one for the accuracy, as the Furukawa algorithm [44] scores better with a lower relative error of 2.04, compared to 2.08 for our algorithm. Also for the completeness, the presented algorithm is not the best performer, as the algorithm by Strecha achieves a higher degree of completeness of 87.06%, compared to 85.6% for our algorithm. However, the presented algorithm presents overall the best compromise between accuracy and completeness, coming close to the top performers in each category.



(a) Accuracy



(b) Completeness

Figure 6.35: Accuracy and Completeness measures of the Fountain Sequence.

Table 6.1: Comparison of Accuracy and Completeness Measures[143]

	Accuracy	Completeness
Furukawa [44]	2.04	73.02%
Strecha 2004 [144]	2.28	87.06%
Strecha 2006 [140]	4.09	85.57%
Presented Method	2.08	85.6%

The accuracy of the reconstruction D_s^{ij} can also be evaluated by building a histogram h_k over the relative errors, following [143]:

$$h_k \propto \sum_{ij} \delta_k \left(\left| D_l^{ij} - D_s^{ij} \right|, D_\sigma^{ij} \right) \tag{6.5}$$

D_l^{ij} is the expected depth value at pixel position i and camera j according to the ground truth model and D_σ^{ij} its corresponding variance. Furthermore, $\delta_k()$ is an indicator function which evaluates to 1 if the depth difference $\left| D_l^{ij} - D_s^{ij} \right|$ falls within the variance range $\left[kD_\sigma^{ij}, (k+1)D_\sigma^{ij} \right]$ and evaluates to 0 otherwise.

Figure 6.36 further compares the relative error histogram of the presented method with a number of state of the art techniques, listed in table 6.2. The last (11th) bin of these histograms collect the occurrence of relative errors larger than 30σ.

Compared to the other algorithms, the presented method shows relatively few hits in the first bin, corresponding to errors lower than 3σ. However, it also shows very few hits in the last bin, corresponding to large errors over 30σ. This means that the presented method is not the most accurate in absolute terms, but it achieves at estimating a dense reconstruction where the errors are well constrained. The relative absence of very large errors means that the reconstruction results are robust and trustable.

Of course, these results are also partly dependent on the parameter settings of the algorithm. Most notably, the diffusion parameter plays an important role

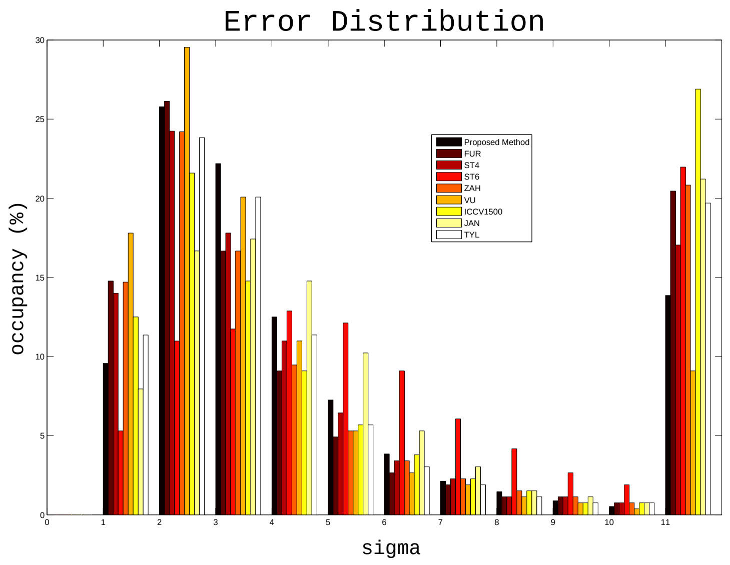


Figure 6.36: Comparison of the Histogram of the Relative Error

in balancing the amount of diffusion in the depth reconstruction. As explained in section 5.3.2, we applied an automated method for iteratively updating the diffusion factor μ. Judging from Figure 6.36, it can now be observed that the automated estimate for the diffusion factor μ is probably a bit too high, meaning that there is a bit too much diffusion. This over-relaxation smooths out large errors, but as a downside, it also results in fewer pixels which are estimated within the 3σ error range.

Figure 6.37 accumulates the histogram values of Figure 6.36 to build up a histogram of the cumulative relative error. In the figure, the desired behavior is a curve which rises very rapidly to 100%, meaning that the reconstructed model is close to the ground truth within a very small error range. From this comparison,

Table 6.2: References for State of the Art Dense Reconstruction Techniques

Abbreviation	Reference
FUR	[44]
ST4	[144]
ST6	[140]
ZAH	[182]
VU	[165]
JAN	[66]
TYL	[158]

it can be noted that only the recent algorithm of [165] shows a better (higher situated) cumulative occupancy behavior.

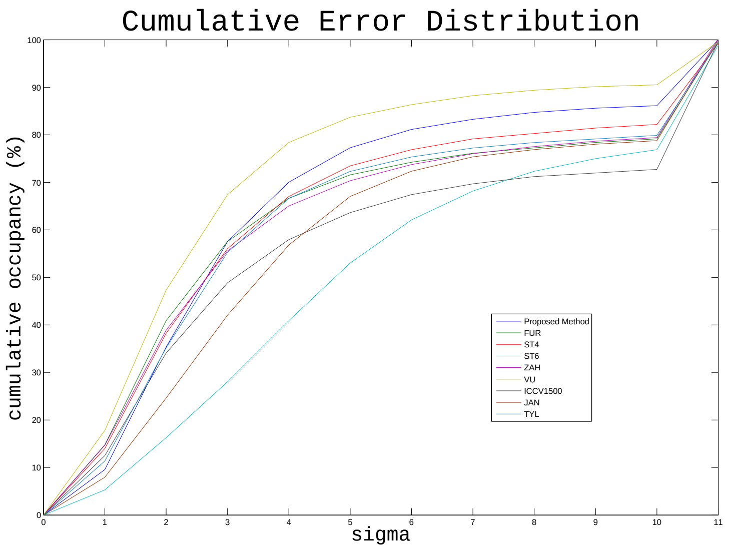
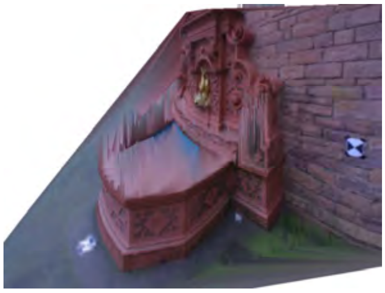
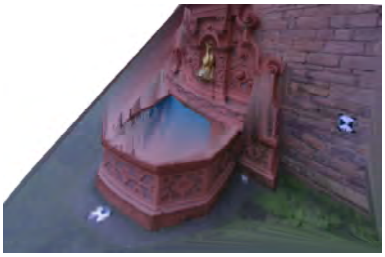
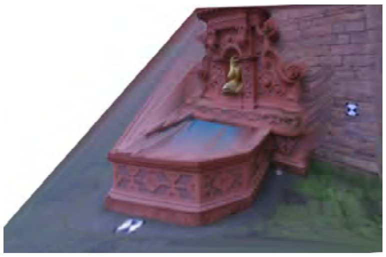


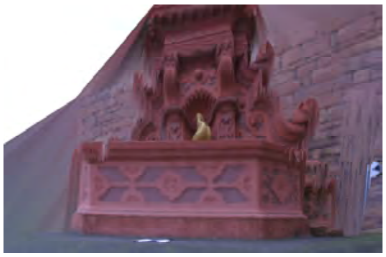
Figure 6.37: Comparison of the Histogram of the Cumulative Relative Error

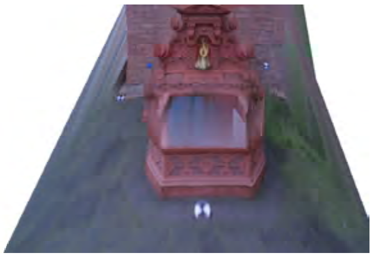
Finally, Figure 6.38 shows some views of the reconstructed 3D model of the water fountain, textured with the original color information. It can be observed that the original 3D structure is visually well represented, as indicated above by the quantitative measures.

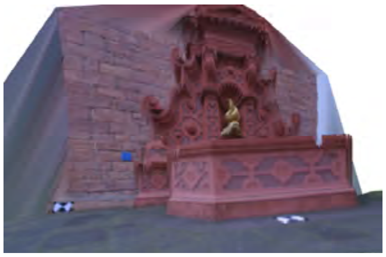

(a) View 1

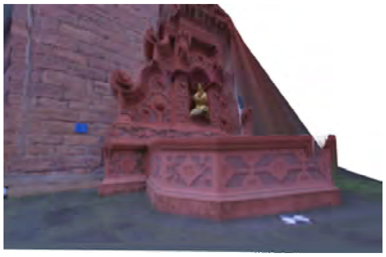

(b) View 2


(c) View 3


(d) View 4


(e) View 5


(f) View 6


(g) View 7

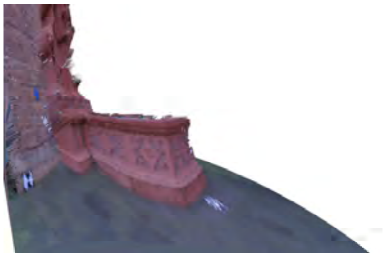

(h) View 8

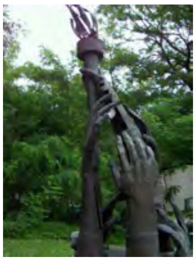
Figure 6.38: Dense Reconstruction Results for the Fountain Sequence: 3D Model

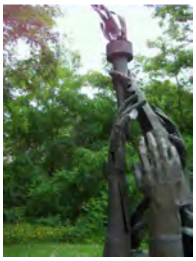
6.4.3 "Hands" Natural Sequence

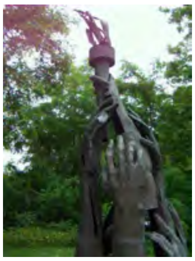
The *Hands* natural sequence presents a particularly difficult case for dense 3D reconstruction, as it consists of a large series of frames (400 frames in total), filmed with a low-quality commercial camera in challenging outside illumination conditions. Due to the nature of this sequence, no ground truth data is present, so only qualitative data is presented in this section to evaluate the presented reconstruction algorithm on this sequence.

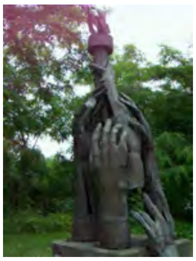
Figure 6.39 shows the depth reconstruction result for some frames of the *Hands* natural sequence. It is apparent that the dense reconstruction technique succeeds in obtaining a correct depth estimate for this quite complex natural sequence. Relative depths have been well estimated, continuous areas have continuous depths and discontinuities are well preserved.

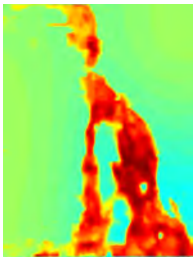
Figure 6.40 shows some views of the reconstructed 3D model of the hands statue. Due to the large amount of frames, the limited computer memory, and the fact that the ICP integration over multiple frames was not possible for this sequence, and only a textured model based on one frame is shown. It can be observed that the hands statue is clearly distinguishable from the background and that the reconstructed 3D structure of the statue corresponds to the real statue. However, the presented reconstruction result in Figure 6.40 also shows a weakness of the proposed algorithm, already indicated in the previous section: the amount of detail on the statue is not as high as achievable with some other reconstruction algorithms. This behavior is consistent with the reconstruction results of the Seaside and Fountain sequence, where it was shown to be due to a slight over-relaxation, smoothing out the reconstruction results. The result of this is that some fine details are lost, which was shown on Figure 6.36 by the fact that there are relatively few data points within the first bin, corresponding to the 3σ error range. The advantage of our approach, however, is that the overall reconstruction result has a high quality, without disturbing outliers.

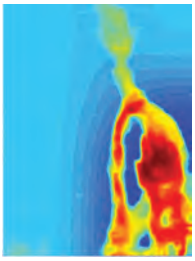

(a) Frame 25

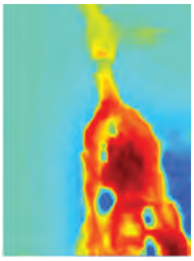

(b) Frame 75

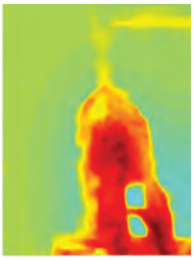

(c) Frame 125

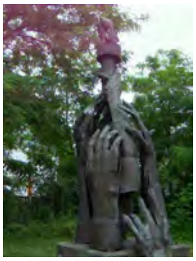

(d) Frame 175

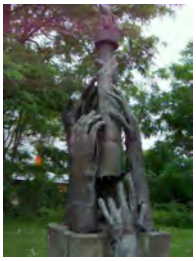

(e) Frame 25

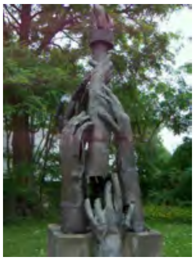

(f) Frame 75

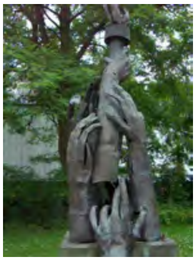

(g) Frame 125

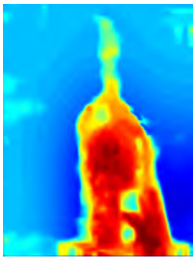

(h) Frame 175

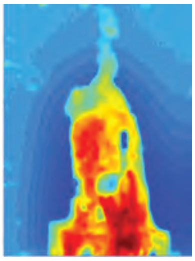

(i) Frame 225

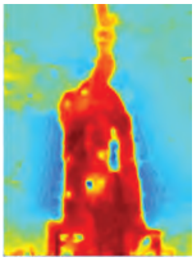

(j) Frame 275


(k) Frame 320


(l) Frame 375


(m) Frame 225


(n) Frame 275


(o) Frame 320

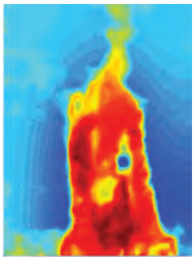
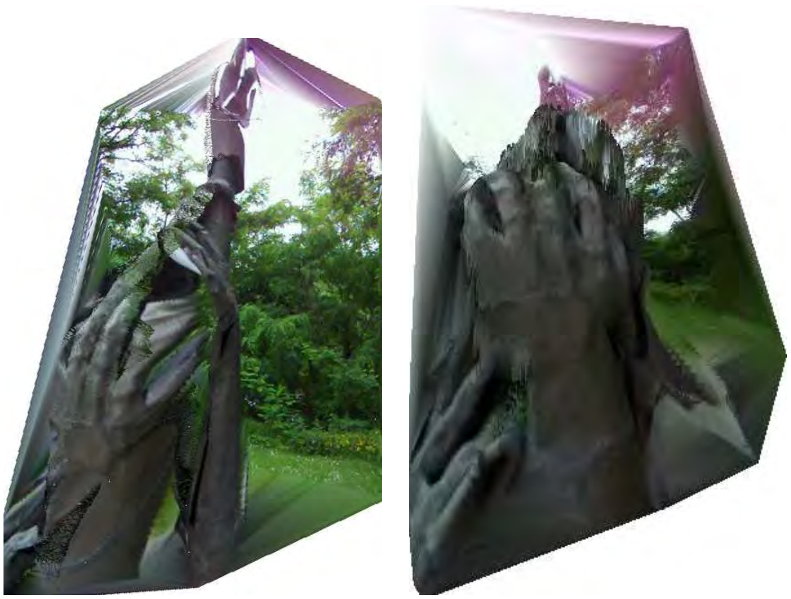

(p) Frame 375

Figure 6.39: Dense Reconstruction Results for the Natural Sequence: Some frames and their corresponding Depth Maps as estimated by the reconstruction algorithm


(a) View 1 (b) View 2

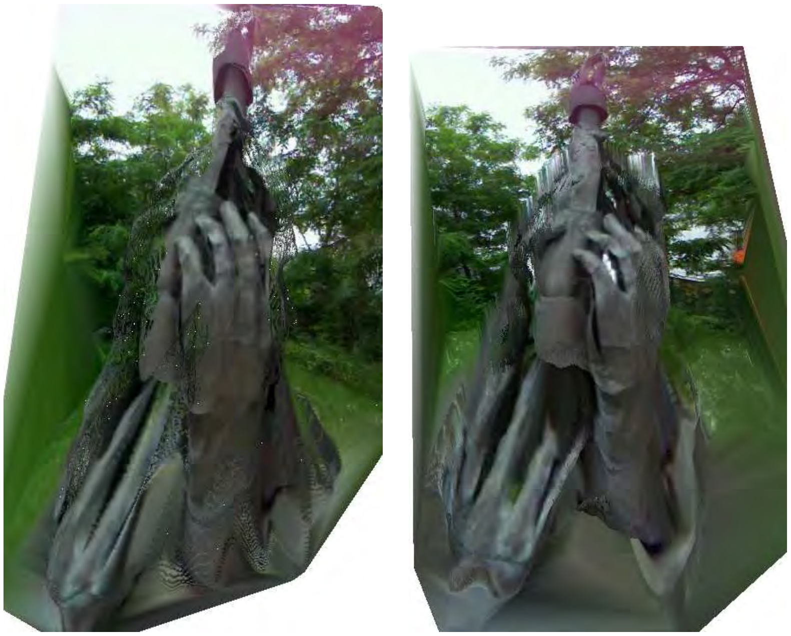

(c) View 3 (d) View 4

Figure 6.40: Dense Reconstruction Results for the Natural Sequence: Novel Views based on the Reconstructed 3D Model of the Statue

6.4.4 "Street" Architectural Sequence

To validate the presented dense structure from motion methodology on larger scale man-made structures than in the *Fountain* sequence, an additional sequence was processed. This *Street* sequence consists of a video, filmed with a low-quality commercial camera from within a vehicle, while driving around in town. Due to the nature of this sequence, no ground truth data is present, so only qualitative data in the form of reconstructed depth maps is presented in this section to evaluate the presented reconstruction algorithm.

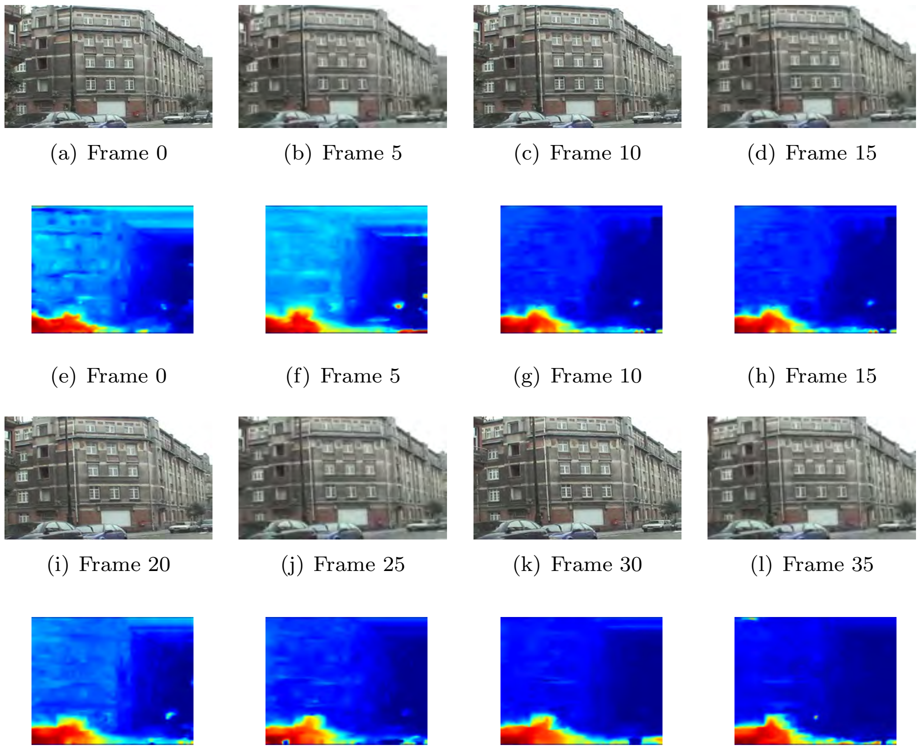
Figure 6.41 shows the depth reconstruction result for some frames of the *Street* sequence. It can be noted that, also for this case, the dense reconstruction technique succeeds in obtaining a good depth estimate, as the different planar regions (walls) have consistent depth values. The main problem for the reconstruction of this sequence is the quality of the input material. As indicated above, this sequence was shot with a low-cost commercial camera, which saved the image sequence as a compressed MPEG-2 movie file. The compression artefacts which were thus induced, cause the sparse feature matching and the dense reconstruction to fail in some areas.

6.5 Conclusions

In this chapter, the dense reconstruction technique presented in chapter 5 was tested on a synthetic and three natural sequences. The depth reconstruction results of Figures 6.22(a), 6.34, 6.39 and 6.41 show a very good overall depth representation. These results can be used to build a high-quality 3D model of the scene, as shown on Figures 6.32, 6.38 and 6.40. These models show detailed information about the imaged scene. As such, a simple in-hand commercial video-camera can be turned from a 2D into a 3D imaging device.

As proven by the quantitative analysis and comparison with multiple state of the art methods [44, 144, 140, 182, 165, 66, 158] on the *Fountain* sequence, the presented algorithm performs very well in comparison with other state of the art methods. Without being the actual top performer for one specific quality measure, it achieves at estimating a globally optimal reconstruction, which balances the accuracy and completeness measures. One of the disadvantages of the proposed algorithm is that it suffers from a slight over-relaxation, due to the automated process of estimating the diffusion parameter μ. Other state of the art methods which estimate the relaxation parameter manually (which is a long and tedious process of trial and error) can reduce the number of relative errors within the 3σ range and, as such, achieve a higher degree of fine detail. However, globally, our method delivers robust and reliable 3D reconstruction results, due to the relative absence of very large errors. As such, it proves to be a valuable candidate for the dense reconstruction of natural sequences.

One of the main disadvantages of the proposed approach, compared to more traditional reconstruction techniques, is without doubt the computation time. The presented method contains some processing steps which are time-consuming. The main problem is the optical flow estimation, which requires about 45 minutes



(a) Frame 0 (b) Frame 5 (c) Frame 10 (d) Frame 15

(e) Frame 0 (f) Frame 5 (g) Frame 10 (h) Frame 15

(i) Frame 20 (j) Frame 25 (k) Frame 30 (l) Frame 35

(m) Frame 20 (n) Frame 25 (o) Frame 30 (p) Frame 35

Figure 6.41: Dense Reconstruction Results for the Street Sequence: Some frames and their corresponding Depth Maps as estimated by the reconstruction algorithm

per frame for 640×480 pixel images. It was not the focus of this work to improve the optical flow estimation algorithm, so we did not really attack this problem. However, the optical flow estimation process is highly parallelizable, meaning that major speed gains could be obtained from using hyperthreading or a multi-core computer, or by switching to a GPU implementation. Recently, other researchers [4] have presented great results in the area of fast dense optical flow estimation.

The iterative estimator on itself is also quite slow. An initial implementation using the Gauss-Seidel algorithm for numerical calculation of the system equation 5.44 required 2 hours per iteration step. As a minimum of 20 iterations are necessary to achieve a steady solution, this would mean that it would take at least two days to process a single frame. As a result, some effort was spent to speed up the iterative solver by applying an optimized least-squares approach to the system equation 5.44. This led to a considerable speedup, as now only approximately 9 seconds are required for 1 iteration, when processing input images with a resolution of 256 by 256 pixels. However, for a standard 640×480 pixel image, this means that still about 1 minute is needed per iteration a PC with a 3GHz CPU and with 1GB of RAM. As the number of iterations required is generally about 20, this means a total execution time of about 65 minutes per frame. In fact when including all other processes into the account (feature detection, feature matching, RANSAC estimation of the fundamental matrix, motion estimation, bundle adjustment, initialization, ...) the total execution time per frame is about 75 minutes. This is obviously far away from real-time reconstruction techniques.

The main reason for this slowness is that the algorithms were not developed with execution speed in mind. As an example, most algorithms are written in Matlab to facilitate the development process, but this does have its repercussions on processing speed. As a result most of the presented algorithms could be vastly improved for speed by switching to a C++ implementation.

Another factor to be taken into account is the required RAM memory. As dense reconstruction algorithms need to reason with huge chunks of data which all need to be loaded into memory at the same time to be interrelated, the RAM memory is stressed. In order for the presented algorithms to run, it was already necessary to change some of the base functions of Matlab to more efficiently deal with the addition and multiplication of huge sparse matrices. Still, the programs will not run on a PC with less than 1GB of RAM memory, but will take advantage of the greater amount of memory of current PC's. The algorithms also benefit from switching to a 64-bit operating system, due to the larger amount of memory accessible to applications.

To clarify the source of the calculation cost better, table 6.3 identifies the calculation time of some of the sub-algorithms of the dense reconstruction approach. This data was recorded using the *Fountain* benchmarking sequence. Table 6.3 splits up the algorithm in a number of subtasks which can be related to Algorithm 2:

- *Sparse Reconstruction* refers to the whole sparse reconstruction approach described by Algorithm 1.
- *Optical Flow Estimation* refers to the image motion estimation process discussed in Appendix B.
- *Other Initialization* refers to the estimation of an initial value of the depth field and computation of spatio-temporal gradients and the regularization matrix.
- *Structure Matrix Calculation* refers to the computation of all elements of the sparse structure matrix
- *Iterative Optimization* refers to all iterative steps of the algorithm: updating the depth estimate, solving the system equation, updating motion and diffusion parameters, ...

When interpreting the end result of 72 minutes as a total calculation time one must not forget that this considers only 2 images. The full sequence consists of 10 images, meaning that processing the whole sequence takes 12 hours, and this is without any subsequent 3D modeling steps. Application of standard ICP on the end results will easily add another hour to the total calculation time.

Table 6.3: Computational Cost of Sub-Algorithms

Task (See Algorithm 2)	Execution Count	Time for 1 execution (min)	Total Time (min)
Sparse Reconstruction	1	1	1
Optical Flow Estimation	1	45	45
Other Initialization	1	1	1
Structure Matrix Calculation	1	5	5
Iterative Optimization	100	0.2	20
Total Time			72

The analysis of table 6.3 shows clearly that the optical flow estimation is the main contributing factor to the total calculation time. In a way, this is good news, because recent research has made it possible to speed up the optical flow estimation process by a factor of 100, by porting it to a GPU implementation. This shows the immense effect that the use of a scale-conscious and parallelized implementation can have on the calculation time. Indeed, conceptually, the 3D reconstruction algorithm is related to the optical flow estimation algorithm, such that it can be hoped that a similar speed-up could be gained by porting also the iterative optimization code of table 6.3 to a GPU implementation. Combined with the new optical flow estimator, this would bring the calculation time down to a few minutes per frame.

Part III

Dense reconstruction from binocular sequences

Chapter 7

Integration of Depth Cues: Overview of Methodologies

7.1 Problem Statement

Neuro-psychological experiments on humans have shown that there exist multiple visual cues that humans use for three-dimensional perception.

The most important cue for depth perception is probably *stereopsis*. Stereopsis results from the small distance between the two pupils, which causes humans to see two slightly different images of the world. This displacement between the horizontal positions of corresponding images is called the binocular disparity. The amount of the displacement depends directly on the relative distance of the objects from the eye. Recent studies [109], [156], [22] have examined the neural mechanisms that mediate the processing of the disparity depth cue. It turns out that our brain possesses neurons which are tuned for 3-D surface orientation estimation from disparity gradients. From this information, high-level signals about the 3-D surface structure are inferred. In computer vision, the depth-from-stereopsis reconstruction is based on the evaluation of the two-view geometry, as described in section 2.3.1.

A second depth cue results from the movement of the observer. For a moving observer, the relative distances of objects determine the amount and the direction of their relative movement in the retina image. This so-called *motion parallax* effect is the basis for all structure from motion algorithms, as the ones presented in part 2. The cortical processing of this monocular depth cue has been less studied when compared with the processing of binocular cues such as disparity.

The remaining visual depth cues are called *pictorial* depth cues [29], are based on the analysis of a single 2D image and were introduced in section 1.1.

To accomplish efficient three-dimensional perception in complex environments, the human brain combines all these different types of visual information about the depth structure. The neural mechanisms explaining precisely how the brain computes and fuses this information remains at present largely unknown, although some work [170] has been presented considering multiple correlated cues, providing insights into sites where information about individual depth cues may converge. Like in the human visual system, also in the computer vision community, a number of studies have been done addressing the depth cue combination problem.

The dense structure from motion 3D reconstruction technique presented in part 2 only uses one of these visual depth cues: depth from motion. As such, it is to be expected to fail in a number of situations, such as scenarios containing too much or not enough motion, degenerate motion (e.g., rotation-only) or degenerate structure (e.g., a coplanar scene). Moreover, applying structure from motion to non-static scenes with moving or deformable objects is still a difficult task. The experimental results of the dense structure from motion algorithm for large, uncontrolled sequences, reflect this statement: dense structure from motion is - in certain conditions - able to yield a very good 3D reconstruction result, but the accuracy of this solution cannot be guaranteed over a longer time sequence, when the camera movement and the scene geometry are totally unconstrained. In these situations, only a fusion of multiple cues could lead to a robust solution.

As multiple depth cues exist, an ideal 3D reconstruction architecture would integrate all these depth cues to form an integrated 3D model. The following section discusses some generic theories for depth cue fusion in the human visual system and derives some generic approaches for depth cue combination in computer vision. In order not to over-complicate the data fusion problem, we chose in this work to integrate the dense structure from motion reconstruction technique presented in part 2 with a depth from stereopsis approach. Therefore, section 7.3 of this chapter focusses more specifically on traditional methodologies for the integration of stereo and motion cues.

7.2 Depth Cue Integration in general

All of the cues to depth structure give a certain amount of information to the visual system regarding depth. The computations and neural mechanisms required to extract the structural information from the retinal images are still largely unknown, but they are likely to be radically different for each cue. Despite this, we perceive coherent 3D structures, which means that somehow, the information provided by different depth cues is combined by the brain in a coherent structure [76]. Individually, each cue gives some indicative information for determining depth, and some cues appear to give stronger, or more convincing, information than other cues. Additional cues increase ones ability to accurately determine depth information, even when the additional cue is not nearly as strong as the original cue [69]. No one cue is necessary for an accurate portrayal of depth nor does any one cue dominate our depth perception in all scenarios [30]. While this information amounts to a clearer understanding of how depth perception detection works, it does not create a convincing argument for how they are integrated in the visual system.

Four theories are commonly presented as possible explanations for understanding how the diverse cues interact with one another in the human visual system

[69], [20].

- *Accumulation* or *Weighted Linear Combination* or *Weak Fusion.* Each cue strengthens the estimate integrated by the visual system after each cue is processed separately. The following research indicated some form of linear combination as the explanation for various cue combinations:
  - stereo, perspective, and proximity luminance [34]
  - motion parallax, occlusion, height in the picture plane and size [18]
  - motion parallax and stereo [120]
- *Cooperation* or *Strong Fusion.* Cues cooperate with each other prior to obtaining depth estimates. The relative importance of each cue varies depending on the reliability of the cue in a given situation. Reliability can be altered when visual information is noisy or when there is incomplete information for that cue [89].
- *Disambiguation.* One cue is used to locally disambiguate a representation derived by another, all of which provide inherently ambiguous information (i.e., stereo can disambiguate shading). Information from separate depth cues indicates which of two explanations is more accurate based on weighting of each module [34], [14].
- *Veto.* Cues veto one another when there is conflicting information, but because of weighting, there is a prioritization of which cues are ignored when in conflict. One cue conveys depth information that will not be challenged by other cues when the information is continuous; when it is not, cues that are less weighted are ignored. As an example, Blthoff and Mallott report in [20] that when stereo indicates a flat surface but shading indicates an ellipsoid, no significant depth is perceived, indicating that stereo can veto shape from shading.

Various cues have been compared against one another in order to determine what the relationship between them might be. Evidence is available for all of the potential explanations, with multiple combinations of cues and in differing scenarios. Research also indicates that observers are able to learn cue combination strategies and adapt them depending on the stimuli and the type of training [65]. This raises the question of whether or not disentanglement is possible.

In the computer vision community, recent work on depth cue combination poses the problem of depth perception as a problem of statistical inference, where depth information should be inferred by the visual system from noisy measurements and prior information [22]. In this view, the visual systems estimates of depth implied by various cues are written as depth likelihood functions and prior depth information is represented by prior probabilities. Both are thus modeled by probability distributions over a metric space. Bayesian models allow optimal combination of such information to predict small, graded changes in depth perception that have been verified experimentally [60], [59], [72]. However, it is unclear how information from cues which do not yield a metric depth representation (e.g. occlusion), can be incorporated within this framework.

Bayesian parameter estimation can be used to generate statistically optimal solutions to the problem of cue integration. However, the complexity and dimensionality of these solutions is frequently prohibitive. Moreover, Schrater and Kersten showed in [127] that the complexity and performance of these Bayesian estimators depends strongly on the detailed formulation of the task, including the choice of representation for the scene variables. As the Bayesian inference models discussed above at present do not yet succeed at describing the combination of depth cues for all possible cues, some researchers have focussed on finding solutions to combine selected depth cues.

In [79], Li et al. propose a framework for creating per-pixel depth maps from monocular videos using the combination of structure from motion and pictorial cues such as focus, occlusion and texture. However, the proposed algorithm mainly uses structure from motion to create the depth map whenever it is applicable and only uses the pictorial cues as a fall back, so no real depth cue fusion is performed.

White and Forsyth present in [171] a method for reconstructing the shape of a deformed surface from a single view. For this, they combine normalized cues from shading and texture to produce a field of unambiguous normals. Using these normals, they reconstruct the 3D geometry. Their method yields high quality reconstructions from single views, but as a disadvantage, it requires a texture estimate of the surface, which must be obtained as prior knowledge.

7.3 Classical ways of Integrating Stereo and Motion

Stereopsis and (structure-from-)motion are considered powerful cues in isolation, meaning that for most observers both sources of information independently provide compelling sensations of depth. As such, the stereo and motion cues are likely candidates to be fused together to solve problems arising from the incompleteness of the individual cues. However, combining motion/stereo constraints from multi-view image sequences requires extra caution. This is because some points in the reference image may be invisible (occluded) in another view. If the integration algorithm is not aware of this and still combines the motion and stereo constraints from the occluded view, the results could be very wrong.

Both stereopsis and motion parallax can yield an absolute measure of depth at each point in the scene, given some additional information about the observers position. In the case of motion the additional information required is knowledge of ego-motion and eye rotation. For stereopsis to provide absolute depth values, the distance from the observer to the fixation point must be known as well as the observers inter-ocular separation. This means that stereopsis and structure from motion essentially yield the same data, and, in principle, combining the information from stereo and motion allows three-dimensional shape to be extracted without the need for additional information about viewing distance or egomotion.

A common approach towards the problem of matching 2 sets of 3D data is the Iterative Closest Point (ICP) algorithm presented in [185]. This method optimizes a set of freeform curves represented by a set of chained points to come to an optimal 3D representation. The data of the space curves are available in the form of a set of chained 3D points from the stereo or structure from motion algorithm. The key idea underlying the ICP approach is the following. Given that the motion between two successive frames is small, a curve in the first frame is close to the corresponding curve in the second frame. By matching points on the curves in the first frame to their closest points on the curves in the second, it is possible to find a motion that brings the curves in the two frames closer (i.e., the distance between the two curves becomes smaller). Iteratively applying this procedure, the algorithm yields successively better motion estimates.

ICP and other approaches which aim to combine depth information from individual reconstructions assume that each depth processing module is completely independent, such that their output can be combined at a higher level. These approaches base themselves on Marr's model [90] of the visual system. Following this model, the stereo and motion depth cue follow a different processing path and 3D information is combined in a $2\frac{1}{2}$ sketch stage. However, a number of psychophysical observations [68],[118] point to close links between the motion and stereopsis processing systems. Rogers and Graham documented in [119] extensive similarities between depth from motion parallax and stereopsis. They discovered that the shapes of the sensitivity functions for depth modulation as a function of spatial frequency are highly similar for stereopsis and motion parallax [119]. However, for some of their observers absolute sensitivity to binocular disparity was greater than that for relative motion. Rogers and Graham proceeded to demonstrate that not only were the two sources of information very similar but there were also interactions in their processing. An ambiguous perception of a surface specified by either stereopsis or motion parallax could be biased by prior exposure to an unambiguously perceived surface specified by the other source of information. Nawrot and Blake demonstrated in [106] the same biasing effect of stereograms upon ambiguous kinetic depth effect stimuli. Fulvio et al. investigated in [33] the integration of three-dimensional information provided over time by the stereo and motion depth cues and found that the perceptual derivation of surface orientation from motion was affected by the prior presentation of static stereo information in the same spatial location. They conclude that interactions exist among the stereo and motion cues of depth information, even when they are provided at different moments of time.

Algorithms relying on posteriori fusion of individual depth and motion reconstructions are therefore not in congruence with the processing of stereo and motion data in the human visual system, as such an a-posteriori fusion of data totally ignores the inherent correlations between the stereo and structure from motion theorems. Indeed, instead of calculating a stereo and structure from motion depth cue separately, one could obtain an advantage from combining motion information within the disparity-searching process of stereo vision [95].

Already in 1986, Waxman and Duncan proposed in [168] a stereo-motion fusion algorithm. They define a *binocular difference flow* as the difference between the left and right optical flow fields, where the right flow field is shifted by the current disparity field. Waxman and Duncan then show that the difference flow and the ratio of rate of change of disparity to disparity are equivalent for image regions containing planar surfaces. Whilst this does not provide a direct solution to the correspondence problem (disparity must be known in order to calculate the difference flow), they suggest two ways in which binocular difference flows can be used in stereo correspondence. Firstly, a vertical motion constraint can be derived as the y component of the difference flow for two corresponding points is the same. Therefore, this constraint can be used to supplement conventional stereo correspondence techniques. Potential matches can be identified by using the x component values of the difference flow for a rate of change of disparity and then searching over a range of disparity recording the ratio scores. A local support metric can then choose matches whose ratios are in agreement. Their method depends on recovering the full motion field from the optical flow field, but relaxes the rigid body constraint by segmenting the scene. Three problems still remain. Firstly, the segmentation process is problematic, being susceptible to noise and variations in the density of reliable optical flow points. Secondly, the technique will only work for scenes containing significant motions. Thirdly, independently moving objects must have significant variations between their motions for robust segmentation.

In 1993, Li and Duncan presented a method for recovering structure from stereo and motion [78]. They assume that the cameras undergo translation but no rotational motion. In the first step of their algorithm, the translational motion parameters are determined from linear equations. In the second step of the calculation, with the knowledge of the estimated translational motion parameters, the binocular flow information is used to find features in one image that correspond to given features in the other image. Tests on laboratory scenes present good results, however the constraint of having only translational motion is hard to fulfill for a natural sequence.

Sudhir et. al. [147] model the visual processes as a sequence of coupled Markov random fields. The Markov random field formulation allows to define appropriate interactions between the stereo and motion processes and outlines a solution in terms of an appropriate energy function. The Markov random field property allows to model the interactions between stereo and motion in terms of local probabilities, specified in terms of local energy functions. These local energy functions express constraints helping the stereo disambiguation by significantly reducing the search space. The integration algorithm as proposed by Sudhir et. al. makes the visual processes tightly constrained and reduced the possibility of an error. Moreover, it is able to detect stereo occlusions and sharp object boundaries in both the disparity and the motion field. However, as this is a local method, it has difficulties when there are many regions with homogeneous intensities. In these regions, any local method of computation of stereo and motion is unreliable.

Strecha and Van Gool present in [142] and [141] a PDE-based approach for 3D reconstruction from multi-view stereo. Their method builds upon the PDE-based approach for dense optical flow estimation by Proesmans et. al. in [116] and reasons on the occlusions between stereo and motion to estimate the quality or confidence of correspondences. The evolution of the confidence measures is driven by the difference between the forward and backward flow in the stereo and motion direction. Based on this estimated per-pixel and per-depth cue quality or confidence measure, their weighting scheme guides the relative influences of both depth cues at every iteration and at every pixel during the evolution towards the solution.

7.4 Conclusions

In this chapter, we presented some findings of neuro-psychological research, indicating how the human brain processes depth information. From this discussion, it is clear that - unlike the single-cue depth-from-motion technique presented in part 2 - the human brain combines a variety of depth cues to come to a consistent structural representation. In computer vision, it would therefore also be beneficial to combine multiple depth cues, in order to avoid the disadvantages of single-cue reconstruction. Generic methods for fusing multi-cue depth information have been presented before and some of these methods were discussed in this chapter. In this research work, we limit ourselves to the combination of depth from disparity (stereopsis) and depth from motion (structure from motion). Previous work regarding the fusion of stereo and motion cues was discussed, to serve as a basis for the development of a novel method towards stereo-motion combination, as presented in the following chapter.

Chapter 8

Integration of Dense Stereo and Dense Structure from Motion

8.1 Introduction and Problem Formulation

As discussed in the previous chapter, the integration of the stereo and motion depth cues offers the potential of a superior depth perception, as the combination of temporal and spatial information makes it possible to reduce the uncertainty in the depth reconstruction result and to augment the precision. However, this requires the development of a data fusion methodology which is able to combine the advantages of each method, without propagating any errors induced by one of the depth reconstruction cues. Therefore, the mathematical formulation of the problem of combining stereo and motion information must be carefully considered.

Like in the monocular case, described in chapter 5, the dense depth reconstruction problem can be casted as an energy minimization problem, as shown before by a number of researchers [142, 173]. However, in the binocular case, the problem is that the solving methodology depends on the simultaneous evaluation of multiple *constraints* which have to be balanced carefully.

This is sketched on Figure 8.1, which shows the different constraints needing to be imposed for a sequence shot with a moving binocular camera. Consider a pair of rectified stereo images (I_1^l, I_1^r) shot at time $t = t_0$ and a stereo pair (I_2^l, I_2^r) shot at time $t = t_0 + t_k$, with t_k being determined by the framerate of the camera. A point $\mathbf{x}_1^l$ in the reference frame I_1^l can now be related to a point $\mathbf{x}_1^r$ via the stereo constraint, as well as to a point $\mathbf{x}_2^l$ via the motion constraint. Using the stereo and motion constraint in combination, the point $\mathbf{x}_1^l$ can even be related to a point $\mathbf{x}_2^r$, and this via a stereo + motion or a motion + stereo path. It is evident that, ideally, all these interrelations should be taken into consideration, and this for all the pixels in all the frames in the sequence. However, when confronted with long sequences of high-resolution video, this leads to prohibitively large systems of equations, which cannot be solved in an efficient way.

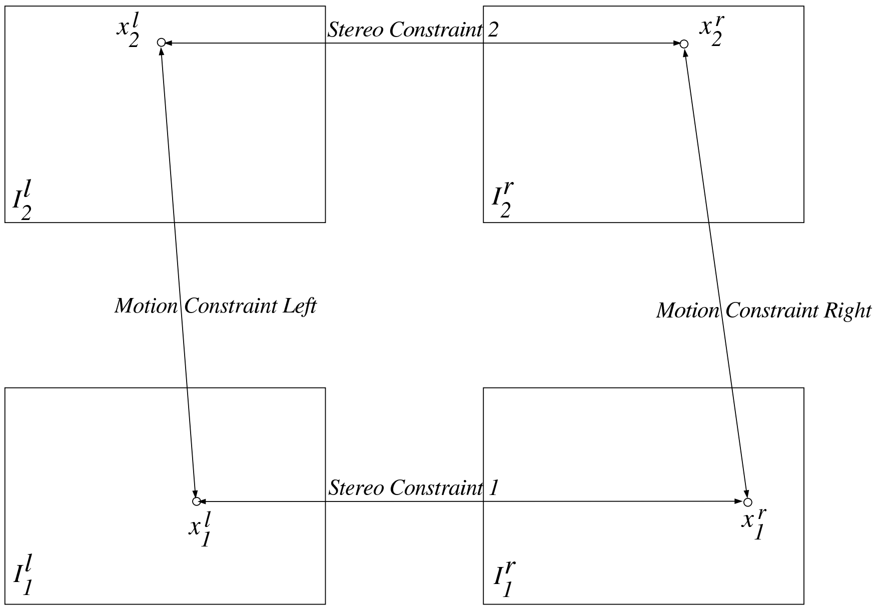


Figure 8.1: Motion and Stereo Constraints on a binocular sequence

In the following, we will present a methodology for addressing the stereo - motion integration problem for dense reconstruction.

8.2 The Proposed Methodology

The stereo-motion integration problem can be regarded as a high-dimensional data fusion problem. Like in the monocular case, our approach towards solving this problem starts by posing the problem as an optimization problem. As such, the main problem is finding a suitable functional which minimizes the error on the dense reconstruction.

As discussed in the previous section, it is in principle possible, using the stereo and motion constraints of a sequence of stereo images, to relate every pixel from a reference image to a pixel in each of the other images of the sequence and vice versa. However, when confronted with a long sequence of high-resolution images, following this approach would lead to a massive amount of constraint equations to be solved simultaneously, which would make the problem impossible to solve in practice. Therefore, in order to limit the dimensionality of the problem, we adopt a different approach, where the sequence of stereo images is processed sequentially. Following this methodology, a pair of stereo images is related to a successive pair, as sketched in figure 8.2.

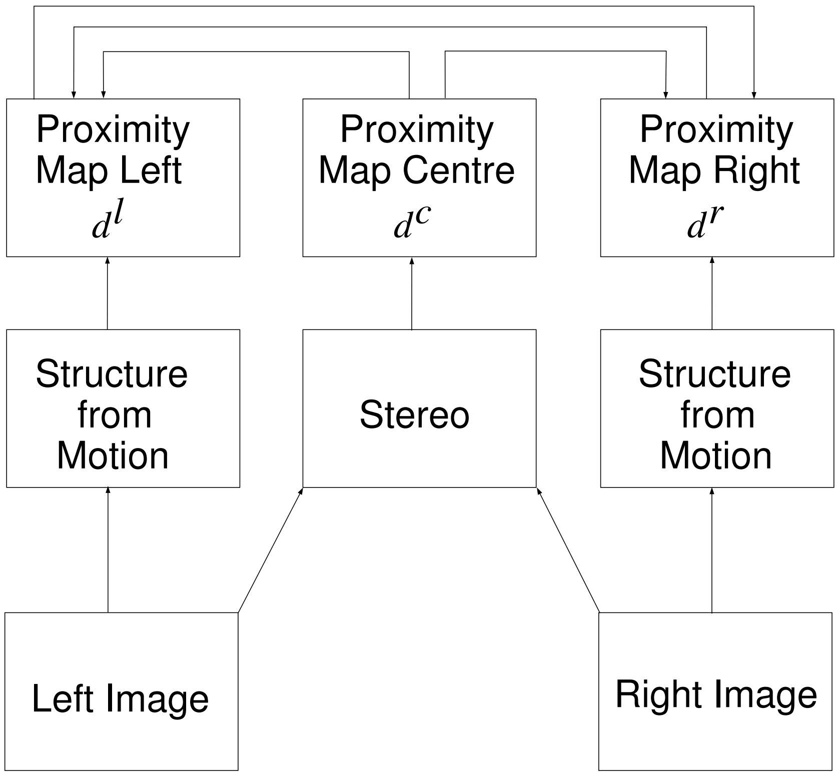


Figure 8.2: Processing strategy of a binocular sequence: From a left and right image sequence, proximity maps are calculated through stereo and dense structure from motion. These maps are iteratively improved by constrained optimization, using the augmented Lagrangian method.

Figure 8.2 considers a binocular image stream consisting of left and right images of a stereo camera system. The left and right streams are processed individually, using the dense structure from motion algorithm, presented in chapter 5, resulting in, respectively, a left and right proximity map d^l and d^r. In parallel, the left and right images are combined using the stereo algorithm [43, 103], embedded in the Bumblebee stereo camera. The stereo camera performs only the stereo computation and, as a result of this, a new proximity map from stereo d^c can be defined. The reason for calling this proximity map d^c lies in the fact that it is defined in the reference frame of a virtual central camera of the stereo vision system.

Of course, there exist strong interrelations between the different proximity maps d^l, d^c and d^r, which need to be expressed to ensure consistency and to improve the reconstruction result. Therefore, we adopt an approach where, the left proximity map d^l is optimized, subject to two constraints, relating it to d^c

and d^r respectively. In parallel, the right proximity map d^r is optimized, also subject to two constraints, relating it to d^c and d^l.

The dense stereo - motion reconstruction problem can thus be stated as a constrained optimization problem:

$$\text{Find} \quad \min_{\mathbf{x}\in\Omega} E(\mathbf{x}) \quad \text{subject to}: \quad \theta_i(\mathbf{x}) = 0 \quad \text{for} \quad i = 1, ..., n \tag{8.1}$$

where the function $E(\mathbf{x})$ is the objective functional and $\theta_i(\mathbf{x})$ expresses a number of constraint equations.

A traditional solving technique for constrained optimization problems as the one posed by equation 8.1 is the Lagrangian multiplier method, which converts a constrained minimization problem into an unconstrained minimization problem of a Lagrange function $\mathfrak{L}_o(\mathbf{x}, \lambda)$:

$$\mathfrak{L}_o(\mathbf{x}, \lambda) = E(\mathbf{x}) + \sum_{i=1}^{n} \lambda_i \theta_i(\mathbf{x}), \tag{8.2}$$

where λ_i are the Lagrange multipliers. Because the optimal Lagrange multipliers are unknown, they have to be estimated iteratively.

In theory, the Lagrangian methodology expressed by equation 8.2 can be used to solve the stereo - motion reconstruction problem, however, to improve the convergence characteristics of the optimization scheme, it is better to use the augmented Lagrangian $\mathfrak{L}(\mathbf{x}, \lambda)$. The augmented Lagrangian, which was presented by Powell and Hestenes in [115] and [57], adds a quadratic penalty term to the original Lagrangian:

$$\mathfrak{L}(\mathbf{x}, \lambda) = E(\mathbf{x}) + \sum_{i=1}^{n} (\lambda_i \theta_i(\mathbf{x})) + \frac{\rho}{2} \sum_{i=1}^{n} \theta_i(\mathbf{x})^2, \tag{8.3}$$

with a penalty parameter $\rho > 0$.

In the context of dense stereo - motion reconstruction, we seek to simultaneously minimize 2 energy functions: E^l for the left image and E^r for the right image. These objective functions are a function of the proximity maps d_1^l and d_1^r (index "1" indicates the first image), which we seek to optimize. This optimization problem is subject to 4 constraint equations:

1. $\theta_{lc}^l(d_1^l, d_1^c) = 0$ relates d_1^l to the proximity map obtained from stereo d_1^c.
2. $\theta_{lr}^l(d_1^l, d_1^r) = 0$ relates d_1^l to the proximity map of the right image d_1^r.
3. $\theta_{rc}^r(d_1^r, d_1^c) = 0$ relates d_1^r to the proximity map obtained from stereo d_1^c.
4. $\theta_{rl}^r(d_1^r, d_1^l) = 0$ relates d_1^r to the proximity map of the left image d_1^l.

According to the Augmented Lagrangian theorem and the definition given by equation 8.3, we can write the augmented Lagrangian for the left image thus as follows:

$$\begin{aligned} \mathfrak{L}_1^l(d_1^l, \lambda_{lc}^l, \lambda_{lr}^l) = E^l(d_1^l) + \lambda_{lc}^l.\theta_{lc}^l(d_1^l, d_1^c) + \frac{\rho}{2}\left[\theta_{lc}^l(d_1^l, d_1^c)\right]^2 \\ + \lambda_{lr}^l.\theta_{lr}^l(d_1^l, d_1^r) + \frac{\rho}{2}\left[\theta_{lr}^l(d_1^l, d_1^r)\right]^2 \end{aligned} \tag{8.4}$$

For the right image we have in a similar fashion:

$$\begin{aligned}\mathfrak{L}_1^r(d_1^r,\lambda_{rc}^r,\lambda_{rl}^r) = E^r(d_1^r) + \lambda_{rc}^r.\theta_{rc}^r(d_1^r,d_1^c) + \frac{\rho}{2}\left[\theta_{rc}^r(d_1^r,d_1^c)\right]^2 \\ + \lambda_{rl}^r.\theta_{rl}^r(d_1^r,d_1^l) + \frac{\rho}{2}\left[\theta_{rl}^r(d_1^r,d_1^l)\right]^2\end{aligned} \tag{8.5}$$

The energy function in equations 8.4 and 8.5 expresses the relationship between structure and motion between successive images and therefore uses a similar formulation to the one presented in chapter 5, writing the energy function as a combination of a data and a regularization term:

$$E = \phi_{data} + \mu\phi_{regularization} \tag{8.6}$$

For the monocular case, we presented 2 formulations for expressing the data term, following equation 5.17 or equation 5.18. As was proven in chapter 6, the latter formulation, using the image derivatives based optical flow constraint, delivers superior results, and as a result, it is this formulation which we'll also use in the binocular case. However, it is not possible to use exactly the same expression as in equation 5.18, as the depth parametrization is different in the monocular and binocular case. Indeed, in the monocular case, a depth parameter $\zeta = \sqrt{\mathbf{u}^2 - \gamma^2}$ is considered, whereas in the binocular case, the proximity $d = \frac{1}{Z}$ is used. However, in a similar fashion to the way equation 5.18 was constructed, it is possible to integrate the relation between flow and structure, given by equation 2.46 into the image derivatives based optical flow constraint of equation 5.5, giving:

$$\phi_{data} = \left(I_{1,x}\left[\mathfrak{a}_1 d + \mathfrak{b}_1\right] + I_{1,y}\left[\alpha d + \beta\right] + I_{1,t}\right)^2, \tag{8.7}$$

with:

$$\begin{aligned}\begin{bmatrix}\mathfrak{a}\\ \alpha\end{bmatrix} &= \mathbf{Q}_t\mathbf{t} = \begin{bmatrix}-ft_x + xt_z\\ -ft_y + yt_z\end{bmatrix}\\ \begin{bmatrix}\mathfrak{b}\\ \beta\end{bmatrix} &= \mathbf{Q}_\omega\omega = \begin{bmatrix}\frac{xy}{f}\omega_x - \left(f + \frac{x^2}{f}\right)\omega_y + y\omega_z\\ \left(f + \frac{y^2}{f}\right)\omega_x - \frac{xy}{f}\omega_y - x\omega_z\end{bmatrix}\end{aligned} \tag{8.8}$$

As expressed by equation 8.6, a regularization is used to filter out erroneous reconstruction results and to smooth and extrapolate the structural data over related pixels. A key aspect here is of course to find out which pixels are related (e.g. belonging to the same object on the same distance), such that proximity information can be propagated and which pixels aren't related. Like in the monocular case, we make use of the Nagel-Enkelmann anisotropic regularization model, as defined in [105]. This time, however, the regularization is applied on the proximity map d, such that the regularization term can be written as:

$$\phi_{regularization} = \left(\nabla d\right)^T \mathbf{D}\left(\nabla I_1\right)\left(\nabla d\right), \tag{8.9}$$

with $\mathbf{D}$ the regularized projection matrix given by equation 5.22.

The energy functions $E^l(d_1^l)$ and $E^r(d_1^r)$ can then be defined as:

$$E^l(d_1^l) = \phi_{data}^l(d_1^l) + \mu\phi_{regularization}^l(d_1^l) \tag{8.10}$$

and

$$E^r(d_1^r) = \phi_{data}^l(d_1^r) + \mu\phi_{regularization}^l(d_1^r) \tag{8.11}$$

with $\phi_{data}^l(d_1^l)$ and $\phi_{data}^l(d_1^l)$ the expression of the data term, as given by equation 8.7 for, respectively, the left and right image and $\phi_{regularization}^l(d_1^l)$ and $\phi_{regularization}^l(d_1^l)$ the expression of the regularization term, according to equation 8.9. The diffusion parameter μ regulates the balance between the data and regularization term. Like in the monocular case, it is estimated iteratively, as explained in section 5.3.2.

The first constraint $\theta_{lc}^l(d_1^l, d_1^c)$ expresses the similarity between the estimated left proximity map d_1^l and the proximity map from stereo d_1^c. In order to calculate this similarity measure, the proximity map from stereo must be warped to the left camera. This warping process can be performed using equation 2.46, which relates the optical flow to the structure and motion parameters, such that we can write:

$$\mathbf{u} = \mathbf{u}(\mathbf{x}, d, \omega, \mathbf{t}) = \begin{bmatrix} u(x, y, d, \omega, \mathbf{t}) \\ v(x, y, d, \omega, \mathbf{t}) \end{bmatrix} \tag{8.12}$$

When considering the 3D motion and structure parameters known, this relation allows to calculate the optical flow, which is the movement of pixels in image space.

$$\theta_{lc}^l(d_1^l, d_1^c) = \left(d_1^l(\mathbf{x}) - d_1^c\left(\mathbf{x} + \mathbf{u}\left(\mathbf{x}, d_1^l(\mathbf{x}), \omega^{cl}, \mathbf{t}^{cl}\right)\right)\right)^2 \tag{8.13}$$

Note that in this case, the relationship between optical flow and structure and motion, as expressed by equation 2.46, is used to estimate a disparity measure. However, the "motion" which is considered in this case is in fact the displacement between the left camera and the virtual central camera, which is known a priori. Since we consider rectified stereo images, the rotational movement between the cameras is zero ($\omega_{stereo} = \mathbf{0}$) and the translational movement is according to the X-axis over a distance of half the stereo baseline b, such that $\mathbf{t}^{cl} = \mathbf{t}_{stereo}$, with $\mathbf{t}_{stereo} = (b, 0, 0)^T$. Concerning the structural data, it suffices to fill in the current estimate of the proximity map d_1^l. As such, the warping process is integrated in the optimization scheme and will gradually improve over time. This allows us to write the first constraint $\theta_{lc}^l(d_1^l, d_1^c)$ as:

$$\theta_{lc}^l(d_1^l, d_1^c) = \left(d_1^l(\mathbf{x}) - d_1^c\left(\mathbf{x} + \mathbf{u}\left(\mathbf{x}, d_1^l(\mathbf{x}), \mathbf{0}, \frac{\mathbf{t}_{stereo}}{2}\right)\right)\right)^2 \tag{8.14}$$

For the second constraint on the left proximity map, we can write in a similar fashion:

$$\theta_{lr}^l(d_1^l, d_1^r) = \left(d_1^l(\mathbf{x}) - d_1^r\left(\mathbf{x} + \mathbf{u}\left(\mathbf{x}, d_1^l(\mathbf{x}), \mathbf{0}, \mathbf{t}_{stereo}\right)\right)\right)^2 \tag{8.15}$$

Note that, in this case, we use the translation over the whole baseline $\mathbf{t}_{stereo} = (b, 0, 0)^T$ for warping the right proximity map to the left proximity map.

The constraints on the right proximity map can be expressed accordingly:

$$\theta_{rc}^r(d_1^r, d_1^c) = \left(d_1^r(\mathbf{x}) - d_1^c\left(\mathbf{x} + \mathbf{u}\left(\mathbf{x}, d_1^r(\mathbf{x}), \mathbf{0}, \frac{-\mathbf{t}_{stereo}}{2}\right)\right)\right)^2 \tag{8.16}$$

and

$$\theta^r_{rl}(d^r_1, d^l_1) = \left(d^r_1(\mathbf{x}) - d^l_1\left(\mathbf{x} + \mathbf{u}\left(\mathbf{x}, d^r_1(\mathbf{x}), \mathbf{0}, -\mathbf{t}_{stereo}\right)\right)\right)^2 \tag{8.17}$$

By integrating the definitions of the energy functions of equations 8.10 and 8.11, and the constraint equations 8.14 - 8.17 into the formulation of the augmented Lagrangian functions, given by equations 8.4 and 8.5, the constrained minimization problem stated in equation 8.1 is now completely defined. How this problem is numerically solved is discussed in the following section.

8.3 Numerical Implementation

To solve the optimization problem, it is necessary to write the augmented Lagrangians of equations 8.4 and 8.5 into a numerical form, which allows iterative optimization. Here, we use the space indices i and j and iteration index k to denominate the different variables and functions. Using this formulation, equation 8.4 can be discretized to:

$$\begin{aligned}\left(\mathcal{L}^l_1\right)^k_{i,j} = \left(E^l\right)^k_{i,j} + \left(\lambda^l_{lc}\right)^k_{i,j} \cdot \left(\theta^l_{lc}\right)^k_{i,j} + \frac{\rho}{2}\left[\left(\theta^l_{lc}\right)^k_{i,j}\right]^2 \\ + \left(\lambda^l_{lr}\right)^k_{i,j} \cdot \left(\theta^l_{lr}\right)^k_{i,j} + \frac{\rho}{2}\left[\left(\theta^l_{lr}\right)^k_{i,j}\right]^2\end{aligned} \tag{8.18}$$

with the energy function:

$$\begin{aligned}\left(E^l_1\right)^k_{i,j} = &\left(\left(I^l_{1,x}\right)_{i,j}\left[a^l_1\left(d^l_1\right)^k_{i,j} + b^l_1\right] + \left(I^l_{1,y}\right)_{i,j}\left[\alpha^l_1\left(d^l_1\right)^k_{i,j} + \beta^l_1\right] + \left(I^l_{1,t}\right)_{i,j}\right)^2 \\ &+ \left(\mu^l_1\right)^k\left(p^l_1\right)_{i,j}\left(\frac{\left(d^l_1\right)^k_{i+1,j} - \left(d^l_1\right)^k_{i,j}}{h_1}\right)^2 \\ &+ \left(\mu^l_1\right)^k 2\left(q^l_1\right)_{i,j}\frac{\left(d^l_1\right)^k_{i+1,j} - \left(d^l_1\right)^k_{i,j}}{h_1}\frac{\left(d^l_1\right)^k_{i,j+1} - \left(d^l_1\right)^k_{i,j}}{h_2} \\ &+ \left(\mu^l_1\right)^k\left(r^l_1\right)_{i,j}\left(\frac{\left(d^l_1\right)^k_{i,j+1} - \left(d^l_1\right)^k_{i,j}}{h_2}\right)^2\end{aligned} \tag{8.19}$$

The constraints given by equations 8.14 and 8.15 measure the dissimilarity between the left proximity map and, respectively, the (warped) central and right proximity map. However, these proximity maps are discrete and possibly highly discontinuous functions, which makes them impractical to work with in an optimization scheme. Therefore, we define an interpolation function $f_I(d, x, y)$ which interpolates the discrete function d in the real-valued point (x, y). The interpolation function f_I which is used here is the bi-cubic spline interpolation function, defined by equation C.5 and further described in Appendix C. Using this bi-cubic interpolation function, the constraint ϕ^l_{lc}, given by equation 8.14, can be discretized as:

$$\left(\phi_{lc}^{l}\right)_{i,j}^{k}=\left(\left(d_{1}^{l}\right)_{i,j}^{k}-f_{I}\left(\left(d_{1}^{c}\right)^{k},i+u\left(i,j,\left(d_{1}^{l}\right)_{i,j}^{k},\mathbf{0},\tfrac{\mathbf{t_{stereo}}}{2}\right),j+v\left(i,j,\left(d_{1}^{l}\right)_{i,j}^{k},\mathbf{0},\tfrac{\mathbf{t_{stereo}}}{2}\right)\right)\right)^{2} \tag{8.20}$$

where u and v are, respectively, the horizontal and vertical functions of the optical flow, as calculated from known structure and motion, as defined by equation 8.12. Equation 8.20 can be further simplified by inserting the values for the motion parameters in the expressions $u\left(x,y,d,\omega,\mathbf{t}\right)$ and $v\left(x,y,d,\omega,\mathbf{t}\right)$, as already shown in equation 8.28, such that:

$$\left(\phi_{lc}^{l}\right)_{i,j}^{k}=\left(\left(d_{1}^{l}\right)_{i,j}^{k}-f_{I}\left(\left(d_{1}^{c}\right)^{k},i-f\frac{b}{2}\left(d_{1}^{l}\right)_{i,j}^{k},j\right)\right)^{2} \tag{8.21}$$

The second constraint on the left proximity map can be discretized similarly as:

$$\left(\phi_{lr}^{l}\right)_{i,j}^{k}=\left(\left(d_{1}^{l}\right)_{i,j}^{k}-f_{I}\left(\left(d_{1}^{r}\right)^{k},i-fb\left(d_{1}^{l}\right)_{i,j}^{k},j\right)\right)^{2} \tag{8.22}$$

An important aspect of the numerical scheme is the choice of the Lagrangian multipliers $\lambda_{i,j}^{k}$. These are initialized to a certain initial value and then iterated. The idea behind the updating of the multipliers is that when the solution for $\mathbf{x}^{k}$ converges to a local minimum $\mathbf{x}^{*}$, then the λ^{k} must converge to the corresponding Lagrange multiplier λ^{*}. This condition can be expressed by differentiating the Augmented Lagrangian of equation 8.3 with respect to $\mathbf{x}$:

$$\nabla_{x}\mathfrak{L}\left(\mathbf{x},\lambda\right)=\nabla E\left(\mathbf{x}\right)+\sum_{i=1}^{n}\left(\lambda_{i}\nabla\theta_{i}\left(\mathbf{x}\right)\right)+\rho\sum_{i=1}^{n}\left(\theta_{i}\left(\mathbf{x}\right)\nabla\theta_{i}\left(\mathbf{x}\right)\right) \tag{8.23}$$

In the local minimum, $\nabla E\left(\mathbf{x}^{*}\right)=0$ and the optimality conditions on the Augmented Lagrangian require that also $\nabla_{x}\mathfrak{L}\left(\mathbf{x}^{*},\lambda^{*}\right)=0$, such that we can deduce:

$$\lambda_{i}^{*}=\lambda_{i}+\rho\theta_{i}\left(\mathbf{x}\right) \tag{8.24}$$

Based upon the result of equation 8.24, it is immediately possible to derive an update scheme for the Lagrangian multipliers, such that they converge to λ_{i}^{*}:

$$\left(\lambda_{lc}^{l}\right)_{i,j}^{k+1}=\left(\lambda_{lc}^{l}\right)_{i,j}^{k}+\rho\left(\theta_{lc}^{l}\right)_{i,j}^{k} \tag{8.25}$$

and

$$\left(\lambda_{lr}^{l}\right)_{i,j}^{k+1}=\left(\lambda_{lr}^{l}\right)_{i,j}^{k}+\rho\left(\theta_{lr}^{l}\right)_{i,j}^{k} \tag{8.26}$$

The expression of the energy function in equation 8.19 and the constraint equations 8.21 and 8.22 completely defines the formulation of the augmented Lagrangian of equation 8.18, governing the iterative optimization of the left proximity map d^{l}. As such, the constrained optimization problem of equation 8.1 is transformed into an unconstrained optimization problem, which can be solved using a classical method.

As a numerical solving technique, we use the method presented by Brent in [16]. Brent's method switches between inverse parabolic interpolation and golden section search. Golden section search [41] is a methodology for finding the minimum of a bounded function by successively narrowing (bracketing) the range of values inside which the minimum is known to exist. This range is also updated using inverse parabolic interpolation, but only if the produced result is acceptable (in the current interval and representing a noticeable improvement over the previous guess.) If not, then the algorithm falls back to an ordinary golden section step. Algorithm 3 summarizes the different steps of Brent's method.

```
Input: A function $f(x)$ and two points $a$ and $c$, such that
       $a < \arg\min_x f(x) < c$
Output: The minimum of the function $f(x)$
repeat
    Choose a point $b$, such that $a < b < c$.
    Fit a parabola through $a$, $b$ and $c$ and calculate the point $x$
    where it reaches a minimum.
    if $f(b) < f(x)$ then
        The new bracketing triplet of points is $a < b < c$.
    else
        The new bracketing triplet of points is $b < x < c$.
    end
    if the proposed range interval is acceptable then
        Use the interval proposed by inverse parabolic interpolation
    else
        Choose a new point $x = a - b + c$ and evaluate $f(x)$.
        if $f(b) < f(x)$ then
            The new bracketing triplet of points is $a < b < c$.
        else
            The new bracketing triplet of points is $b < x < c$.
        end
    end
until the distance between the two outer points of the triplet is
sufficiently small ;
```

Algorithm 3: Overview of Brent's optimization method using inverse parabolic interpolation and golden section search

The optimization method described above converges to a minimum within the search interval. Therefore, it is crucial that a good initial value is available for all status variables, such that the local minimum coincides with the global minimum. In the monocular case, the methodology towards estimating an accurate initial value for the depth parameter was based on the integration of dense optical flow data and sparse structure from motion, as described in section 5.3.3. Both types of data are also present in the binocular case, so in theory we could adopt just the same mechanism. However, in practice, it is a far better approach to use the dense disparity map from stereo as an initial value. The reason for this is that the camera displacement between the left and right stereo frames is much better known and is moreover fixed over time, than the displacement between two frames of the same camera, for which the movement parameters must be estimated using sparse structure from motion. Therefore, the warping of stereo data towards the left and the right image can be performed with much higher reliability then in the monocular case. Indeed, warping the stereo disparities to the left and right proximity maps is extremely simple when using rectified stereo images, as this implies that the depth $Z = \frac{1}{d_{stereo}}$ calculated from stereo can be integrated in the optical flow constraint of equation 2.46. However, this optical flow constraint can be largely simplified in the stereo case, because the rotation between the left and right stereo cameras is zero ($\omega_{\mathbf{stereo}} = \mathbf{0}$) and the stereo translation is for rectified cameras necessarily along the x-axis with an amount specified by the half baseline b in the positive or negative direction, depending on wether the left or the right image is considered:

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \mathbf{Q}_\omega \omega_{\mathbf{stereo}} + \frac{1}{Z}\mathbf{Q_t t_{stereo}} = \mathbf{Q}_\omega \mathbf{0} + \frac{1}{Z}\begin{bmatrix} -f & 0 & x \\ 0 & -f & y \end{bmatrix}\begin{bmatrix} \pm\frac{b}{2} \\ 0 \\ 0 \end{bmatrix} \tag{8.27}$$

As such, equation 2.46 reduces to:

$$u = \pm f \frac{1}{Z}\frac{b}{2}. \tag{8.28}$$

Using the relationship between the stereo depth and the optical flow u, it is possible to define the equations providing an initial value for the left and right proximity maps d^l and d^r:

$$\begin{cases} d^l_{initial}(x,y) = d_{stereo}(x - f d_{stereo}(x,y)\frac{b}{2}, y) \\ d^r_{initial}(x,y) = d_{stereo}(x + f d_{stereo}(x,y)\frac{b}{2}, y) \end{cases} \tag{8.29}$$

As can be noted, equation 8.29 contains no real unknown data, next to the stereo proximity map d_{stereo}, which needs to be estimated of course, this in contrast to the monocular case, where the translation and rotation between cameras needs to be taken into account. It is for this reason that in the binocular case, the initial guess is significantly better than in the monocular case and that, because the initial value is closer to the final solution, the optimization can also be performed much faster.

Next to the estimation of a good initial value, the application of Brent's optimization method also requires that the minimum and maximum boundaries where the solution is to be found be known. In our case, it means that a minimum and maximum proximity value must be available for each pixel of the left and right images. These minimum and maximum proximity maps are calculated based on the 3σ error interval of the initial value of the proximity maps:

$$\begin{array}{l} d^l_{\min} = d^l_{initial} - 3\sigma\left(d^l_{initial}\right) \\ d^l_{\max} = d^l_{initial} + 3\sigma\left(d^l_{initial}\right) \\ d^r_{\min} = d^r_{initial} - 3\sigma\left(d^r_{initial}\right) \\ d^r_{\max} = d^r_{initial} + 3\sigma\left(d^r_{initial}\right) \end{array} \tag{8.30}$$

where $d^l_{initial}$ and $d^l_{initial}$ are calculated according to equation 8.29

Following Brent's method of the golden section search with parabolic interpolation, it is possible to solve the unconstrained optimization problem for the left proximity map, as posed by equation 8.18. For the right proximity map, a set of similar expressions can be found, starting from the Augmented Lagrangian:

$$\begin{aligned}(\mathcal{L}_1^r)^k_{i,j} = (E^r)^k_{i,j} + (\lambda^r_{rc})^k_{i,j} \cdot (\theta^r_{rc})^k_{i,j} + \frac{\rho}{2}\left[(\theta^r_{rc})^k_{i,j}\right]^2 \\ + \left(\lambda^l_{rl}\right)^k_{i,j} \cdot (\theta^r_{rl})^k_{i,j} + \frac{\rho}{2}\left[(\theta^r_{rl})^k_{i,j}\right]^2\end{aligned} \tag{8.31}$$

with the energy function:

$$\begin{aligned}(E_1^r)^k_{i,j} = &\left(\left(I^r_{1,x}\right)_{i,j}\left[a_1^r\,(d_1^r)^k_{i,j} + b_1^r\right] + \left(I^r_{1,y}\right)_{i,j}\left[\alpha_1^r\,(d_1^r)^k_{i,j} + \beta_1^r\right] + \left(I^r_{1,t}\right)_{i,j}\right)^2 \\ &+ (\mu_1^r)^k (p_1^r)_{i,j}\left(\frac{(d_1^r)^k_{i+1,j} - (d_1^r)^k_{i,j}}{h_1}\right)^2 \\ &+ (\mu_1^r)^k 2 (q_1^r)_{i,j}\frac{(d_1^r)^k_{i+1,j} - (d_1^r)^k_{i,j}}{h_1}\frac{(d_1^r)^k_{i,j+1} - (d_1^r)^k_{i,j}}{h_2} \\ &+ (\mu_1^r)^k (r_1^r)_{i,j}\left(\frac{(d_1^r)^k_{i,j+1} - (d_1^r)^k_{i,j}}{h_2}\right)^2\end{aligned} \tag{8.32}$$

and with the constraints:

$$\left(\phi^l_{rc}\right)^k_{i,j} = \left((d_1^r)^k_{i,j} - f_I\left((d_1^c)^k, i + f\frac{b}{2}\,(d_1^r)^k_{i,j}, j\right)\right)^2 \tag{8.33}$$

and

$$\left(\phi^l_{rl}\right)^k_{i,j} = \left((d_1^r)^k_{i,j} - f_I\left(\left(d_1^l\right)^k, i + fb\,(d_1^r)^k_{i,j}, j\right)\right)^2 \tag{8.34}$$

The Lagrangian multipliers λ^r are updated at each iteration, following:

$$(\lambda^r_{rc})^{k+1}_{i,j} = (\lambda^r_{rc})^k_{i,j} + \rho\,(\theta^r_{rc})^k_{i,j} \tag{8.35}$$

and

$$(\lambda^r_{rl})^{k+1}_{i,j} = (\lambda^r_{rl})^k_{i,j} + \rho\,(\theta^r_{rl})^k_{i,j} \tag{8.36}$$

As shown above, there are, in fact, two functions which are optimized at the same time: one using $\left(\mathcal{L}_1^l\right)^k_{i,j}$ which optimizes the left proximity map d^l and one using $(\mathcal{L}_1^r)^k_{i,j}$ which optimizes the right proximity map d^r. In the proposed algorithm, these functions are optimized alternatively, hereby always using the latest result for both proximity maps. This methodology is explained more clearly in Algorithm 4, which summarizes the Augmented Lagrangian - based stereo - motion reconstruction algorithm.

The Augmented Lagrangian - based stereo - motion reconstruction methodology, presented in Algorithm 4, differentiates itself from the current state of the art in stereo - motion reconstruction by a number of key factors:

```
Input: A sequence of Stereo Images $I^l, I^r$
Output: A dense proximity map $d^l(x, y)$ and $d^r(x, y)$ for each image $i$

1. Initialization:
1.1 Perform sparse reconstruction on the left and right images, using
      Algorithm 1.
1.2 Compute the stereo disparity between $I^l$ and $I^r$, using a stereo
      algorithm. Here, we use the embedded algorithm of the
      Bumblebee stereo vision camera, used for this research,
      as described in [43, 103].
1.3 Warp the stereo disparity image to obtain an initial value of the
      left and right proximity map $d^l$ and $d^r$, using equation 8.29.
1.4 Compute elements of the (left and right) regularization matrix,
      following equation B.6.
1.5 Set an initial value for the Lagrange multipliers $\left(\lambda^l_{lc}\right)^0_{i,j}$, $\left(\lambda^l_{lr}\right)^0_{i,j}$,
      $(\lambda^r_{rc})^0_{i,j}$ and $(\lambda^r_{rl})^0_{i,j}$.
1.6 Set an initial value for the penalty parameter $\rho > 0$
1.7 Set an initial value for the proximity maps according to
      equations 8.30

2. Iterative optimization of the proximity fields for each pixel:
2.1 Minimize $\left(\mathcal{L}^l_1\right)^k_{i,j}$, as expressed by equation 8.18, with the
      unconstrained method of Algorithm 3 from a starting point
      $\left(d^l_1\right)^k_{i,j}$. Let $\left(d^l_1\right)^{k+1}_{i,j}$ denote the approximation to the solution.
2.2 Minimize $(\mathcal{L}^r_1)^k_{i,j}$, as expressed by equation 8.31, with the
      unconstrained method of Algorithm 3 from a starting point
      $(d^r_1)^k_{i,j}$. Let $(d^r_1)^{k+1}_{i,j}$ denote the approximation to the solution.
2.3 Update the Lagrange multipliers for the left proximity map,
      using equations 8.25 and 8.26.
2.4 Update the Lagrange multipliers for the right proximity map,
      using equations 8.35 and 8.36.
2.5 Update the estimate of the diffusion parameter $\mu$, according to
      equation 5.46.
2.6 Increase the penalty parameter $\rho$, if the constraint violations at
      iteration step $k + 1$ are not sufficiently smaller than at
      iteration step $k$.
2.7 Set $k = k + 1$ and go to step 2.1 if no convergence is
      reached and $k < k_{max}$

3. Repeat the above for all images $i$.
```

Algorithm 4: Overview of the Augmented Lagrangian - based Stereo - Motion Reconstruction Algorithm

1. The processing strategy, presented in Figure 8.2, considers 3 sources of information for the structure estimation process: a left and a right proximity map from motion, and a proximity map from stereo. During optimization, information from the (central) proximity map from stereo is transferred to the left and right proximity maps, which are the ones actually being optimized simultaneously. During the optimization process, data is constantly being interchanged between both optimizers, as they are highly dependent. The advantage of this concurrent optimization methodology is that it provides a symmetric processing cue. This makes it easer to handle the uncertainties induced by the unknown displacements between the different cameras, in comparison to other approaches [147] who consider only 1 reference image and warp all other images to this reference image for matching and depth estimation. Other researchers have noted this too and have for this reason used even more depth or proximity maps. In [142], Strecha and Van Gool combine 4 proximity maps d_i^l, d_{i+1}^l, d_i^r and d_{i+1}^r, as displayed in Figure 8.1. The problem with using so many proximity maps, however, is that the problem size is increased drastically, and with it, also the computation time.

2. The expression of the energy functions, using equations 8.19 and 8.32, follows the methodology for dense structure estimation from monocular data, as set up in chapter 5. This means for example that the image derivatives - based optical flow constraint of equation 5.5 is used. Most other dense stereo - motion reconstruction algorithms, like the ones presented by Worby in [173] and Strecha and Van Gool in [142], use the constant image brightness - based optical flow constraint of equation 5.4. However, our analysis in chapter 6 learned that the image derivatives - based optical flow constraint delivers better results than the constant image brightness - based optical flow constraint. Another aspect of the expression of the energy functions is that it includes a methodology for automatically re-estimating at each iteration step the value of the diffusion parameter μ, as described in section 5.3.2.

3. The proposed methodology poses the dense stereo - motion reconstruction problem as a constrained optimization problem and uses the Augmented Lagrangian to turn this into an unconstrained optimization problem which can be solved with a classical method. Whereas other researchers go to great lengths to express the stereo - motion reconstruction problem as a Markov Random Field [147] or a Graph Cut [173] optimization problem, the approach we followed is very natural, as the stereo - motion reconstruction problem is by nature a highly constrained and tightly coupled optimization problem and the Augmented Lagrangian has been proven before [110] to be an excellent method for these kind of problems.

8.4 Conclusions

In this chapter, we have presented a method for dense stereo - motion based reconstruction. This approach casts the stereo - motion reconstruction problem as an optimization problem, where the left and right proximity map are estimated concurrently by combining information from the stereo and motion depth cues.

The proposed stereo - motion dense reconstruction methodology extends the monocular dense structure from motion approach, described in chapter 5, and uses the technique of the Augmented Lagrangian to integrate different constraints in a coherent framework. For this, we adopted an integrated processing strategy using dense structure from motion data and dense stereo data.

In the following chapter, we will evaluate the presented approach and compare its performance to a more classical method.

Chapter 9

Results & Analysis for Binocular Reconstruction

9.1 Description of the Test Procedure

The validation and evaluation of a dense stereo - motion reconstruction algorithm requires the use of an image sequence shot with a moving stereo camera. Unfortunately, for this case of dense stereo - motion reconstruction, there is no real set of standard sequences used for benchmarking, unlike in the monocular case, where such sequences do exist [143].

For this reason, we recorded a new sequence, using a Bumblebee stereo vision camera. This sequence, shown in Figure 9.1, was shot in an indoor office environment. Due to the nature of this sequence, it is impossible to retrieve ground truth data about the depth field. The translation of the camera is mainly along its optical axis (Z-axis) and along the positive X-axis. The rotation of the camera is almost only along the positive Y-axis.

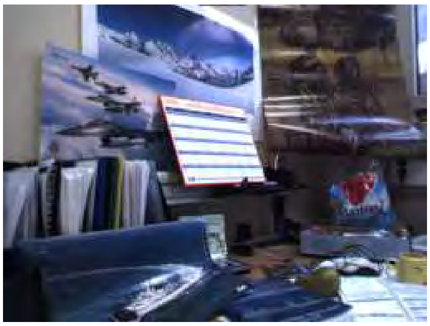
It can be noted from Figure 9.1, that the used sequence consists of a cluttered environment, presenting serious challenges for any reconstruction algorithm:

- Cluttered environment with many objects at different scales of depth.
- Relatively large totally untextured areas (e.g. the wall in the upper left) making correspondence matching very difficult.
- Areas with specular reflection (e.g. on the poster in the upper right of the image), violating the Lambertian assumption, traditionally made for stereo matching.
- Variable lighting and heavy reflections in the window on the upper right, causing saturation effects and incoherent pixel colors across different frames.

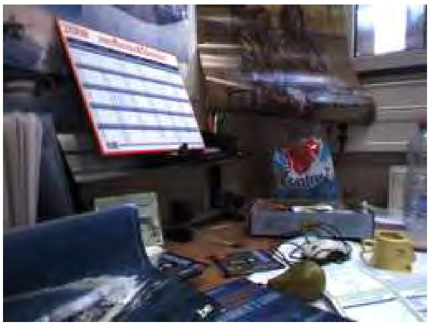
In the following sections, we will focus our evaluation on how the iterative optimization methodology presented in chapter 8 deals with these issues and how

(a) Frame 1, Left Image

(b) Frame 1, Right Image


(c) Frame 5, Left Image

(d) Frame 5, Right Image


(e) Frame 10, Left Image

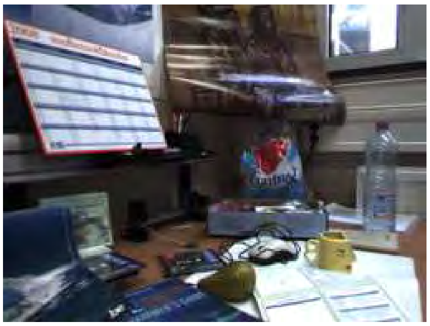
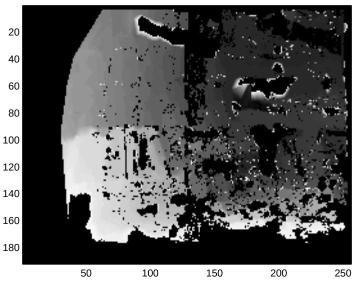
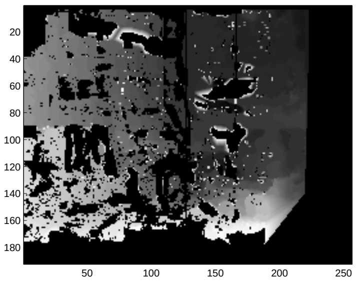

(f) Frame 10, Right Image

Figure 9.1: Some frames of the binocular Desk sequence

well it is able to reconstruct the structure of this scene. However, it must not be forgotten that this iterative optimizer is also dependent on an initialization procedure, which can influence the reconstruction result significantly. The initialization step of Algorithm 4 estimates an initial value for the left and right depth field. This method consists of warping a stereo proximity image to the left and right camera reference frame. The result of this process is shown in Figure 9.2.



(a) Left Initial Proximity Map



(b) Right Initial Proximity Map

Figure 9.2: Left and Right Initial Proximity Maps

It is clear from Figure 9.2 that the initial values for the left and right proximity map still contains a lot of "blind spots", or areas where no (reliable) proximity data is available. These areas are caused by a failing correspondence search in the stereo vision algorithm. The applied stereo algorithm performs an area based correlation with SAD (Sum of Absolute Differences) on band passed images [43, 103]. This algorithm is fairly robust and it has a number of validation steps that reduce the level of noise. However, the method requires texture and contrast to work correctly. Things like occlusions, repetitive features and specular reflections can cause problems and can lead to gaps in the proximity maps, as shown on Figure 9.2. In the following sections, we will also evaluate how well the presented dense stereo - motion algorithm is able to cope with these blind spots and see whether it is capable of filling in the areas where structural data is missing.

To compare our method to the state of the art, we implemented a more classical dense stereo - motion reconstruction approach, which is explained in the following section. This approach defines classical stereo and motion constraints, alongside the Nagel-Enkelmann regularization constraint. These constraints are integrated into one objective function, which is solved using a traditional trust-region method. As such, this approach presents a relatively simple and straightforward solution. This methodology is used to serve as a base benchmarking method for the Augmented Lagrangian based stereo - motion reconstruction technique, presented in chapter 8.

9.2 Global Optimization - based Integration Approach

9.2.1 The Proposed Methodology

The processing technique for the global optimization based reconstruction approach is slightly different from the method presented in chapter 8. Stereo pairs are processed 2-by-2, meaning that in every processing step, 4 images are considered, as presented in Figure 9.3.

The constraints acting on this image quadruplet express the stereo relationship between the left and the right image and the motion relationship between successive images. Moreover, regularization is also here required to propagate the solution in areas where insufficient data is present.

A crucial aspect of the design of the energy functionals ϕ is the choice of the status variables which are iteratively estimated during the optimization process. In the case of the global optimization approach, we have chosen for the motion parameters (translation vector $\mathbf{t}$ and rotation vector ω) and the left and right proximity map, d^l and d^r, with $d = \frac{1}{Z}$. The reason for using both the proximity map for the left and the right image is that, using this approach, the formulation of the functional ϕ_{motion} can be written in a symmetric way. Obviously, both proximity maps are not independent, which is expressed by imposing a (stereo) constraint ϕ_{stereo} between them.

The optimization problem can thus be generally defined as:

$$\left[t, \omega, d^l, d^r\right] = \underset{t,\omega,d^l,d^r}{\arg\min} \int F(\omega, \mathbf{t}, d^l, d^r) dxdy \tag{9.1}$$

with:

$$F \rightarrow \begin{cases} \phi_{motion} = 0 \\ \phi_{stereo} = 0 \\ \phi_{regularization} = 0 \end{cases} \tag{9.2}$$

In the following paragraphs, the expression of the different constraint equations, forming equation 9.2, is discussed.

The motion constraint ϕ_{motion} can be formulated by expressing equation 2.46, which relates the dense optical flow $\mathbf{u} = (u, v)$ to the structure and motion parameters. Like in the monocular case, we make use of a previously calculated dense optical flow field for the dense depth estimation. However, in this case we do not use the dense optical flow for the initialization, but as a parameter in the motion constraint. Structural data is present in equation 2.46 in the form of a proximity parameter d, whereas the camera motion is parameterized through a translation vector $\mathbf{t} = (t_x, t_y, t_z)$ and a rotation vector $\omega = (\omega_x, \omega_y, \omega_z)$. Equation 2.46 provides two equations for each pixel $\mathbf{x} = (x, y)$ of each image, one for the horizontal flow $u(\mathbf{x})$ and one for the vertical flow $v(\mathbf{x})$. As there are two images (left and right) to be considered, this means that, the expression of the motion constraint adds 4 constraint equations to the objective functional for each pixel, corresponding to the first 4 equations of equation 9.3.

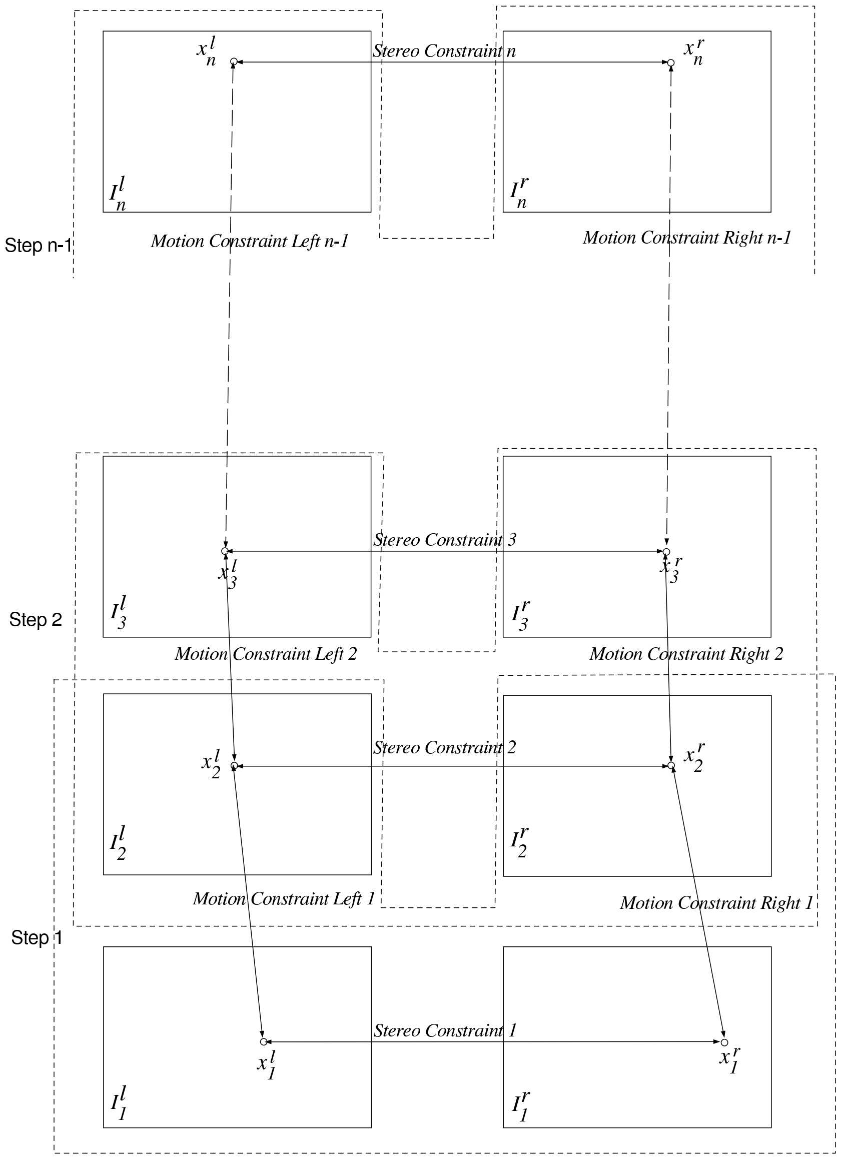


Figure 9.3: Sequential processing of a binocular sequence: Stereo pairs are processed 2-by-2, expressing 1 stereo and 2 motion constraints for each reconstruction

The stereo constraint ϕ_{stereo} expresses the consistency between the left and right proximity field. Therefore, the left and right proximity maps are compared and the stereo constraint is formulated as the dissimilarity between both fields, as expressed by the fifth equation of equation 9.3

As a third aspect of the optimization functional defined by equation 9.2 is the regularization. Following the Nagel-Enkelmann regularization model, two regularization constraints (one for the left proximity map and one for the right proximity map) are expressed. These form the last two equations of equation 9.3.

The objective functional formulation can thus be written as:

$$F \rightarrow \begin{cases} \frac{xy}{f}\omega_x - \left(f + \frac{x^2}{f}\right)\omega_y + y\omega_z - d^l(\mathbf{x}) f t_x + d^l(\mathbf{x}) x t_z - u^l(\mathbf{x}) = 0 \\ \left(f + \frac{y^2}{f}\right)\omega_x - \frac{xy}{f}\omega_y - x\omega_z - d^l(\mathbf{x}) f t_y + d^l(\mathbf{x}) y t_z - u^l(\mathbf{x}) = 0 \\ \frac{xy}{f}\omega_x - \left(f + \frac{x^2}{f}\right)\omega_y + y\omega_z - d^r(\mathbf{x}) f t_x + d^r(\mathbf{x}) x t_z - u^r(\mathbf{x}) = 0 \\ \left(f + \frac{y^2}{f}\right)\omega_x - \frac{xy}{f}\omega_y - x\omega_z - d^r(\mathbf{x}) f t_y + d^r(\mathbf{x}) y t_z - u^r(\mathbf{x}) = 0 \\ \left(d^l(\mathbf{x}) - d^r\left(\mathbf{x} + \mathbf{u}\left(\mathbf{x}, d^l(\mathbf{x}), \omega_{stereo}, \mathbf{t}_{stereo}\right)\right)\right)^2 = 0 \\ \left(\nabla d^l\right)^T D\left(\nabla I^l\right)\left(\nabla d^l\right) = 0 \\ \left(\nabla d^r\right)^T D\left(\nabla I^r\right)\left(\nabla d^r\right) = 0 \end{cases} \tag{9.3}$$

As can be noted, the objective functional of equation 9.3 consists of 7 equations per pixel. This means that for an image with N pixels, a system of equations with $7N$ equations in $2N + 6$ unknowns must be solved. It is obvious that for high resolution images, this poses an immense optimization problem, which is not easy to solve on commodity hardware and requires a carefully chosen numerical implementation.

The numerical implementation of the global-optimization - based stereo-motion reconstruction approach makes use of a large-scale trust-region method [26, 25]. Trust region methods are iterative optimization techniques which seek to optimize a function by building a model that approximates the objective function [28]. These kind of approaches are particularly useful for large-scale problems, like the one posed by dense stereo - motion reconstruction, as formulated by equation 9.1, because they limit drastically the required number of function evaluations, thereby reducing the calculation time, and because of their strong convergence properties. Despite these advantages, it must be noted that trust region methods are local minimizers. There is no guarantee that the local minimum where the optimizer converges to is also a global minimum. This stresses the need for a good initial guess of the status variables.

The basic idea behind the trust region method, presented in [26, 25] is very simple and is clarified by Algorithm 5, which summarizes the global optimization - based reconstruction method:

```
Input: A sequence of Stereo Images $I_i^l, I_i^r$
Output: A dense proximity map $d^l(x,y)$ and $d^r(x,y)$ for each image $i$
1. Initialization:
1.1 Perform sparse reconstruction on $I_i^l$ & $I_{i+1}^l$, using Algorithm 1.
1.2 Perform sparse reconstruction on $I_i^r$ & $I_{i+1}^r$, using Algorithm 1.
1.3 Compute the optical flow between $I_i^l$ & $I_{i+1}^l$ and $I_i^r$ & $I_{i+1}^r$,
      as described in Appendix B.
1.4 Compute the stereo disparity between $I_i^l$ and $I_i^r$, using a stereo
      algorithm. Here, we use the embedded algorithm of the
      Bumblebee stereo vision camera, as described in [43, 103].
1.5 Warp the stereo disparity image to obtain an initial value of the
      left and right proximity maps $d^l$ & $d^r$, using equation 2.46.
1.6 Compute elements of the (left and right) regularization matrix,
      following equation B.6.
2. Iterative optimization of the depth field for each pixel:
2.1 At each iteration $x_k$, build a model $\psi$ approximating the objective
      function $F$. Following the methodology described in [26],
      this model can be written as:

          $\psi_k = g_k^T s_k + \frac{1}{2} s_k^T \left(H_k + C_k\right) s_k$        (9.4)

      with $s_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, $g_k = \nabla F(\mathbf{x}_k)$ and $H_k = \nabla^2 F(\mathbf{x}_k)$.
      $C_k = D_k diag\left(g_k\right) J_k D_k$, with $D_k$ a scaling matrix and
      $J_k$ the Jacobian
2.2 Calculate a step $s_k$ to a trial point within the trust region
      at which point a sufficient model decrease is obtained, by solving
      the following subproblem:

          $\min_s \left(\psi_k\left(s\right) : \|D_k s\| \leq \Delta_k\right)$        (9.5)

      The condition $\|D_k s\| \leq \Delta_k$ assures that the solution leads to a
      new trial point within the current trust region $\Delta_k$.
2.3 Calculate the ratio of achieved reduction of the objective function
      to the predicted reduction according to the model:

          $\rho_k = \frac{F\left(x_k + s_k\right) - F(x_k) + \frac{1}{2} s_k^T C\left(x_k\right) s_k}{\psi_k\left(s_k\right)}$        (9.6)

2.4 If the ratio $\rho_k$ is sufficiently positive, then define the next guess as
      the trial point: $x_{k+1} = x_k + s_k$. Otherwise, $x_{k+1} = x_k$.
2.5 Update the trust region size $\Delta_k$ and the scaling matrix $D_k$ to
      reflect the evolution of the convergence (e.g. when the ratio $\rho_k$ is
      sufficiently large, $\Delta_k$ is increased, otherwise it is decreased).
3. Repeat the above for all images $i$.
```

Algorithm 5: Overview of the Global Optimization - based Stereo - Motion Reconstruction Algorithm

The global optimization methodology presents a stereo motion reconstruction technique which follows the traditional approach of defining stereo and motion constraints and optimizing them all together, using an optimization technique like a trust region method. One differentiating factor of this stereo - motion reconstruction approach, with respect to classical methods, lies in the combined use of both the left and the right proximity map for optimization. Most traditional methods, e.g. the one presented by Sudhir et al. in [147], consider a reference image (e.g. the first left image) and warp all other images to this reference image for matching and depth estimation. The problem with this approach is that it creates an asymmetric processing cue for left and right images, causing the model errors due to uncertainties in the motion parameters to spread unevenly, which makes it harder to solve the optimization problem. The global optimization methodology avoids this by incorporating both the left and right proximity map in the status variable. However, as we'll see later on when comparing its performance to the Augmented Lagrangian based method presented in chapter 8, its simple design will give rise to some reconstruction problems.

9.2.2 Results & Analysis

A disadvantage of working with natural sequences is that it is impossible to compare the calculated reconstruction data to ground truth information. Therefore, we have to validate our approach by analyzing the convergence parameters of the different algorithms.

For the global optimization - based stereo - motion reconstruction approach, a first optimization parameter which can be evaluated is the value of the objective function, given by equation 9.3 at the current estimate of the optimization variables (motion parameters, left and right proximity map) at each iteration step. As the objective function, defined by equation 9.3, consists of 7 equations per pixel, this means that $7N$ functions could be evaluated, with N the number of pixels. It is clear that, in practice, this is undesirable and also nearly impossible to visualize. Therefore, the function values are summed over all pixels:

$$\begin{aligned}
F^l_{motion,h} &= \sum_{\forall x,y} \left| \tfrac{xy}{f}\omega_x - \left(f + \tfrac{x^2}{f}\right)\omega_y + y\omega_z - d^l(\mathbf{x}) f t_x + d^l(\mathbf{x}) x t_z - u^l(\mathbf{x}) \right| \\
F^l_{motion,v} &= \sum_{\forall x,y} \left| \left(f + \tfrac{y^2}{f}\right)\omega_x - \tfrac{xy}{f}\omega_y - x\omega_z - d^l(\mathbf{x}) f t_y + d^l(\mathbf{x}) y t_z - u^l(\mathbf{x}) \right| \\
F^r_{motion,h} &= \sum_{\forall x,y} \left| \tfrac{xy}{f}\omega_x - \left(f + \tfrac{x^2}{f}\right)\omega_y + y\omega_z - d^r(\mathbf{x}) f t_x + d^r(\mathbf{x}) x t_z - u^r(\mathbf{x}) \right| \\
F^r_{motion,v} &= \sum_{\forall x,y} \left| \left(f + \tfrac{y^2}{f}\right)\omega_x - \tfrac{xy}{f}\omega_y - x\omega_z - d^r(\mathbf{x}) f t_y + d^r(\mathbf{x}) y t_z - u^r(\mathbf{x}) \right| \\
F_{stereo} &= \sum_{\forall x,y} \left(d^l(\mathbf{x}) - d^r\left(\mathbf{x} + \mathbf{u}\left(\mathbf{x}, d^l(\mathbf{x}), \omega_{stereo}, \mathbf{t}_{stereo}\right)\right)\right)^2 \\
F^l_{regularization} &= \sum_{\forall x,y} \left| \left(\nabla d^l\right)^T D\left(\nabla I^l\right)\left(\nabla d^l\right) \right| \\
F^l_{regularization} &= \sum_{\forall x,y} \left| \left(\nabla d^r\right)^T D\left(\nabla I^r\right)\left(\nabla d^r\right) \right|
\end{aligned} \tag{9.7}$$

To further reduce the number of functions to be shown, the horizontal and vertical component of the left and right optical flow constraints are also summed:

$$\begin{aligned} F^l_{motion} &= F^l_{motion,h} + F^l_{motion,v} \\ F^r_{motion} &= F^r_{motion,h} + F^r_{motion,v} \end{aligned} \tag{9.8}$$

In this way, it is possible to obtain a clearer overview of how the motion and stereo constraints are balanced. Figure 9.4 shows the evolution of the different constituents of the objective function F function value, as defined above.

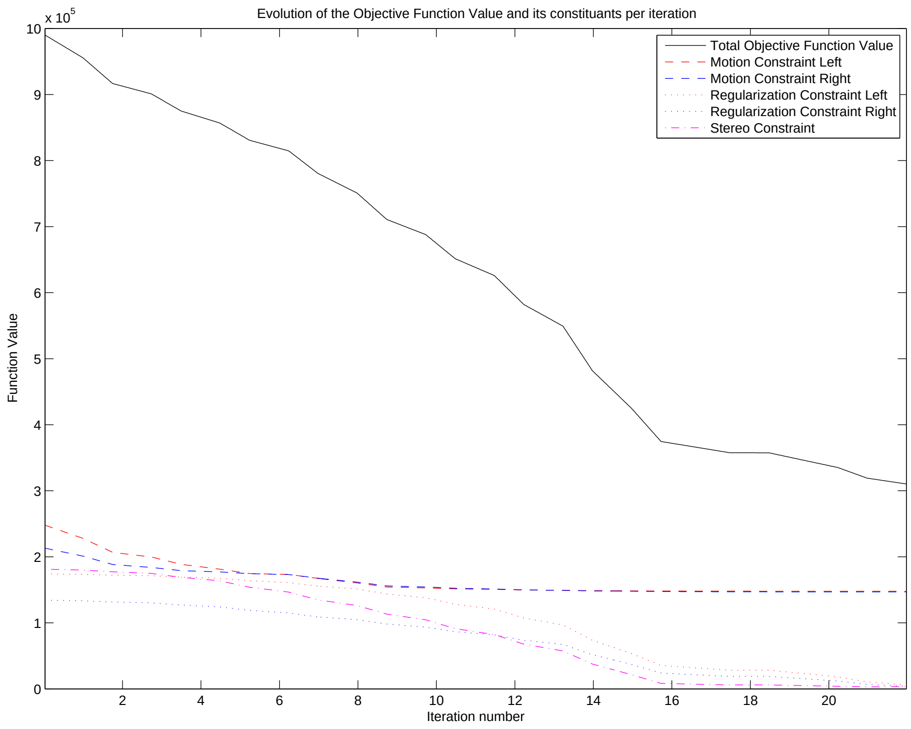


Figure 9.4: Evolution of the Objective Function using the Global Optimization Algorithm

Figure 9.4 shows that all constituents of the objective function show a monotonically decreasing behavior, indicating that the method converges. As a result, also the total objective function value $F_{total} = F^l_{motion} + F^r_{motion} + F_{stereo} + F^l_{regularization} + F^r_{regularization}$ converges.

It may be noted from Figure 9.4 that the largest contributions to the total objective function value F_{total} are the left and right motion constraints F^l_{motion} and F^l_{motion}. The reason for this lies in the fact that the motion constraint relies on the estimation of the camera motion parameters. This estimation process is inherently less robust than in the stereo case, where the camera displacement is fixed and known a priori. As such, the motion constraints will feature a larger residual error on their objective function, compared to the stereo constraint.

It may also be noted that the left motion constraint F^l_{motion} starts at a higher function value than F^r_{motion}. This is due to the nature of the movement in the sequence. As discussed in section 9.1 and as shown on Figure 9.1, the *Desk* sequence features next to the translation along the optical axis also a rotation around the positive Y-axis. This 3D motion causes a relatively larger displacement between two successive left images than between two successive right images. Due to this larger displacement, the error on the camera motion estimation between I^l_k and I^l_{k+1} will also be larger, leading to a higher initial function value for F^l_{motion}. However, by updating the estimate for the motion parameters, the optimization algorithm succeeds after some iterations at minimizing the difference between the two motion constraints. Despite this, Figure 9.4 shows that the function value for the motion constraints remains relatively high, indicating that the optimization performs not very well for this constraint.

Figure 9.4 also shows that the stereo constraint and regularization constraints, are well optimized, as indicated by the evolution of their function values, which decreasing drastically after some iterations. From Figure 9.4 can thus be concluded that the stereo and regularization constraints, as defined by equation 9.3 are well optimized and show a converging behavior, but also that the motion constraints are not optimized very well. In fact, Figure 9.4 shows us that the optimization is geared too much into the direction of the stereo and regularization constraints (because there is no adequate constraint balancing system in place), whereas the motion constraint is ignored. We will see further in the text how this affects the performance of the global optimization based integration approach with respect to the presented Augmented Lagrangian based approach.

As discussed in section 9.2.2, the global optimization method relies on a large-scale trust-region method [26, 25]. The evolution of this optimization process can be evaluated by considering the step size s_k, as shown on Figure 9.5. This optimization parameter decreases monotonically, showing the convergence of the optimization method.

Figure 9.6 and 9.7 show the evolution of the different components of, respectively, the translation vector and the rotation vector. In order to allow for a better comparison, all graphs were produced with the same limits for the y-axis.

In general, it can be concluded from Figures 9.6 that the reconstruction algorithms succeeds at retrieving the correct camera translational motion pattern. Indeed, the translation of the camera for the *Desk* sequence is mainly along the Z-axis, and with a small component in the X-direction, as estimated correctly by the reconstruction algorithm. As the translation features no translation in the Y-direction, we can have an estimate of the error on the translation estimation, by looking at the estimated value for t_y. Figure 9.6(b) shows that this error is really small in comparison to the values of t_z and t_x. It can also be noted that this error is further reduced over the course of the iterative process, leading us to conclude that the optimization works well for estimating the translation vector.

The imposed camera rotation was mainly around the positive Y-axis, which is also estimated correctly, as shown on Figure 9.7. However, in this case, it

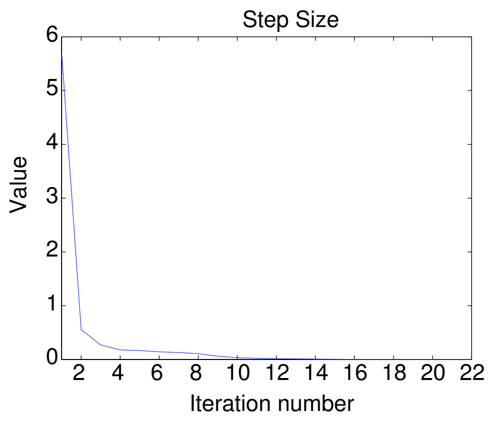
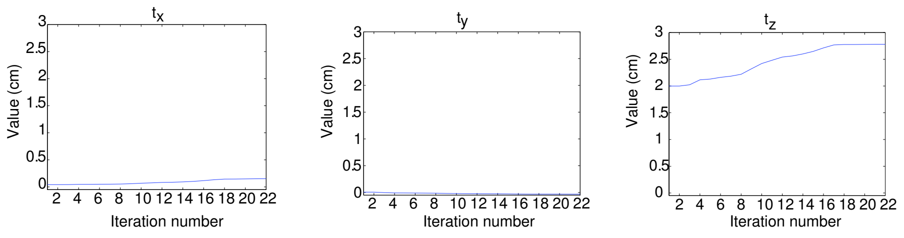


Figure 9.5: Evolution of the Step Size s_k using the Global Optimization Algorithm

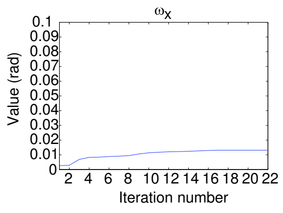


(a) t_x (b) t_y (c) t_z

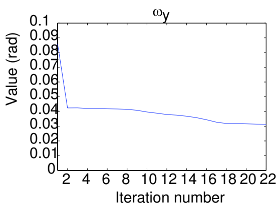
Figure 9.6: Evolution of the Translation Vector $\mathbf{t}$ using the Global Optimization Algorithm

is obvious that the values of ω_x and ω_z cannot be ignored next to the value of ω_y. In reality, a small rotation around the Z-axis was also present in the camera motion, but the estimated rotation around the X-axis must certainly be considered erroneous. It must be noted that in this case, the initial estimate of the rotation vector, as given by the sparse reconstruction algorithm (Algorithm 1), is a better approximation of the physical reality than the final estimate after optimization.

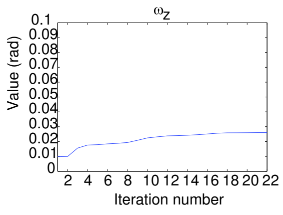
This leads us to conclude that the motion update process succeeds in improving the estimate for the translation vector, but fails when the rotation vector is concerned. The reason for this lies in the different scales of both problems. As rotational values are much smaller than translational values, they are more susceptible to noise. In order to solve this problem, it would be required to define different optimizers for translation, rotation and depth, which is not done in this global optimization methodology.



(a) ω_x



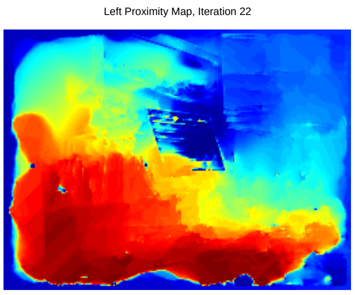
(b) ω_y



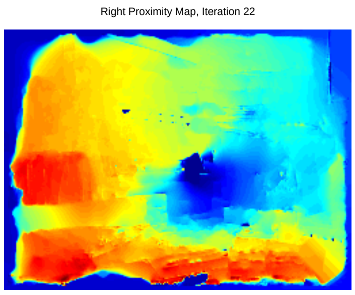
(c) ω_z

Figure 9.7: Evolution of the Rotation Vector ω using the Global Optimization Algorithm

Figure 9.8 shows the reconstructed left and right proximity maps using the global optimization based stereo - motion algorithm.



(a) Left proximity map, (d_1^l)

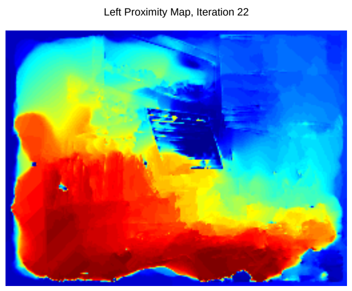


(b) Right proximity map, (d_1^r)

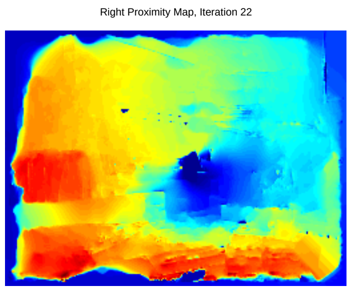
Figure 9.8: Left and Right Proximity Maps using the Global Optimization Algorithm

Overall, the reconstruction of the proximity field correlates with the physical reality, as imaged on Figures 9.1(a) and 9.1(b). The foreground items on the desk appear closer (red) than the background (blue) walls and the depth gradient of the walls can also be found back, most notably on the right proximity field. There are, however, some serious errors in the reconstructed proximity fields:

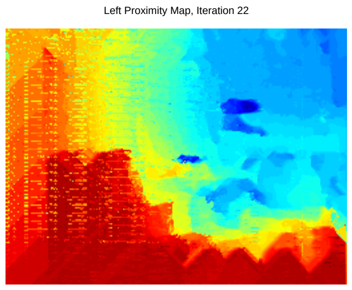
- In both images, the edges are wrongly estimated at a very high depth (proximity near zero). This is a normal effect at the borders of the image, because at these locations, there is less information to reason with (e.g. it is impossible to interpolate data points). This is a common problem and is generally solved by extending the image canvas before the calculations or by cutting the image canvas after the calculations. We chose not to do this, in order not to falsify the evaluation of the base reconstruction methodology with pre- or postprocessing techniques.
- In both images, there are some dots visible, corresponding to areas where



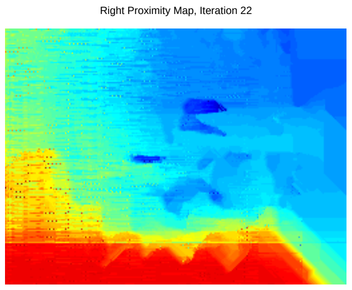
(a) Frame 1, Left proximity map d_1^l



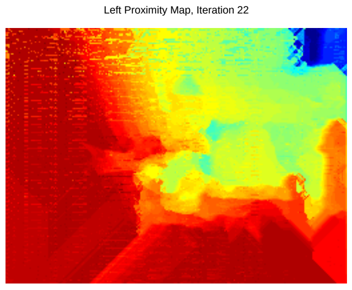
(b) Frame 1, Right proximity map d_1^r



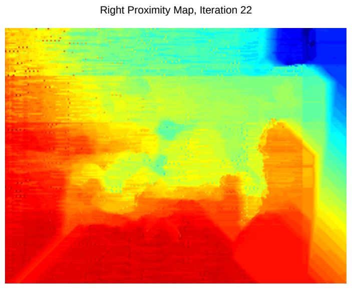
(c) Frame 5, Left proximity map d_5^l



(d) Frame 5, Right proximity map d_5^r



(e) Frame 10, Left proximity map d_{10}^l



(f) Frame 10, Right proximity map d_{10}^r

Figure 9.9: Proximity Maps for different frames of the Desk sequence using the Global Optimization Algorithm

the regularization has not fully succeeded and where the holes in the initial proximity fields, displayed in Figure 9.2, have not been filled in.

- In the left image, the depth gradient on the left wall is not represented well by the proximity map.
- In the left image, the board in the center is not well reconstructed. The problem here lies in a combination of a lack of stereo and motion correspondences and failing regularization. For the right image, this board is reconstructed much better, there is only a problem at its lower left, which is estimated much to far.
- In the lower left quadrant of the right image, a rectangular smooth proximity gradient can be observed, which does not correspond to any physical object and is only cause by an exaggerated regularization.

The reconstruction problems indicated above lead us to conclude that the global optimization based stereo - motion reconstruction technique is capable of outputting a global estimate of the proximity field, but fails at delivering a high-quality structural reconstruction, which can be used for 3D modeling.

The same problems occur for all frames of the image sequence, as shown on Figure 9.9, which displays the reconstructed proximity maps for different frames of the *Desk* sequence. The reconstructed proximity maps for the fifth and the tenth image of the sequence suffer heavily from over-relaxation, making the proximity field to be smeared out over large areas. The reason for this lies in the fact that in the global optimization method, there is no way of balancing the different constraints of the objective functional, making it possible for the regularization constraint to become much more apparent than the data-based constraints.

9.3 Augmented Lagrangian - based Integration Approach

In order to compare the performance of the Augmented Lagrangian based stereo - motion reconstruction technique, this algorithm was subjected to the same tests as the global optimization based approach.

Figure 9.10 shows the evolution of the objective function for the left and right proximity field, according to equations 8.18 and 8.31. Both functions show a monotonically decreasing behavior, indicating that the method converges. It is, however, evident that the function value for the left camera is always higher than for the right camera. As discussed in the previous section, this behavior is due to the fact that the left camera undergoes a larger inter-frame trajectory, which induces larger errors on the motion estimation. In contrary to the global optimization approach, the Augmented Lagrangian method does not feature an automated update procedure for the motion vectors. As a result, the difference between both function values does not diminish during the iterative process like it did for the global optimization approach, as shown in Figure 9.4. As the

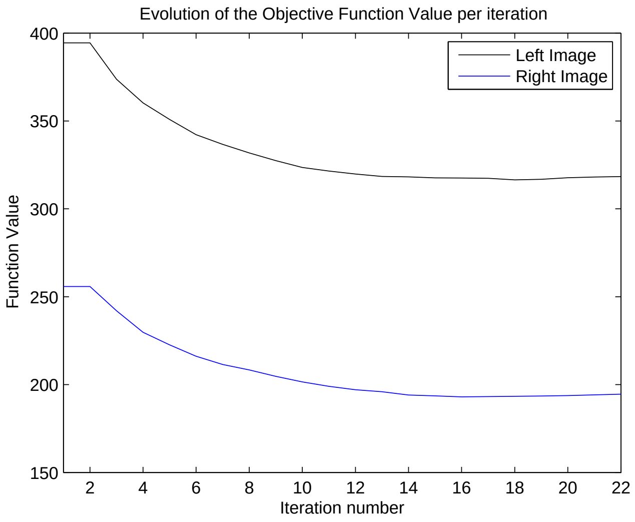


Figure 9.10: Evolution of the Objective Function using the Augmented Lagrangian Algorithm

objective function definitions are quite different, a direct comparison of function values to the ones featured for the global optimization approach has no use, but it can be noted that the Augmented Lagrangian method converges faster to a stable solution than the global optimization method. This indicates that fewer iterations will be required to converge to a solution very near the optimal value.

To evaluate the convergence properties further, a second convergence parameter which can be considered is the function gradient, here defined as the sum of squared differences between successive depth fields: $G^l = \sum_{i,j} \left[\left(d^l\right)_{i,j}^{k+1} - \left(d^l\right)_{i,j}^{k} \right]^2$.

Figure 9.11 shows the evolution of this parameter during the iterative process. The monotonically decreasing nature of G^l and G^r indicates convergence. Also in Figure 9.11, we can note the recurring behavior that G^l lies higher than G^r, for the reasons discussed above.

For the Augmented Lagrangian method, it is also useful to evaluate the evolution of the Lagrangian multipliers during the iterative optimization process. To this end, Figure 9.12 shows the magnitude of the 4 Lagrangian Multiplier maps $\left(\lambda_{lc}^l, \lambda_{lr}^l, \lambda_{rc}^r, \lambda_{rl}^r\right)$ at 3 different time steps of the iterative optimizer.

For the first iteration step state, all multipliers are initialized to a fixed value (in this case: 1). The left column of Figure 9.12 conveys with the state of the multipliers at the second time step, when all constraints have been evaluated once and the Lagrangian multipliers were updated, according to equations 8.25,8.26, 8.35 and 8.36. These Lagrangian multiplier maps show relatively large values for the Lagrangian multipliers, notably in the lower right corner. The reason for

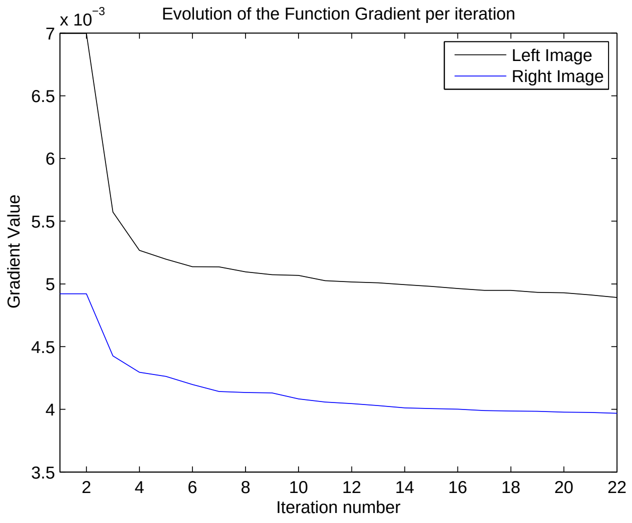
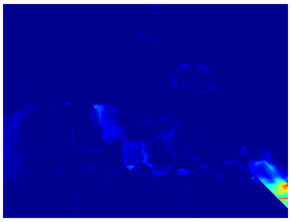
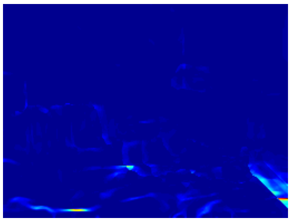


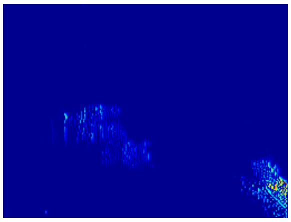
Figure 9.11: Evolution of the Gradient using the Augmented Lagrangian Algorithm

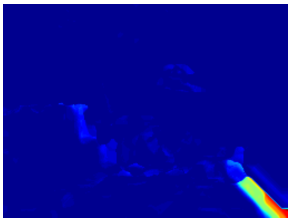
this behavior can be found by comparing the Lagrangian multiplier maps to the initial proximity maps, shown in image 9.2. The areas where high values for the Lagrangian multipliers are registered correspond to areas where depth information is missing. As such, the different constraints are violated in these regions, and the value for the Lagrangian multipliers increases drastically. The second and third column of Figure 9.12 show the state of the Lagrangian multipliers at respectively, the tenth and the final time step. As can be seen, the intensity of the Lagrangian multipliers has been reduced, because the constraints are no longer violated as much as in the initial stage of the optimization process.

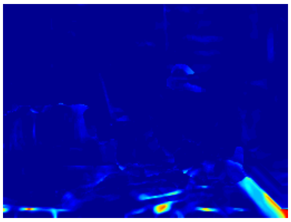
To better assess the effect of the combination of the different lagrangian multipliers during the iterative process, Figure 9.13 shows the mean value over all pixels for λ^l_{lc}, λ^r_{rc}, λ^l_{lr} and λ^r_{rl} for each iteration step. As can be noted from Figure 9.13, the mean value of the lagrangian multipliers first rises dramatically. The reason for this behavior lies in the fact that in this phase of optimization, the constraints are still heavily violated and the multipliers induce large step sizes, such that the algorithm can converge to a solution fast. It can also be seen that the different constraints give rise to significantly different maxima for the λ_k, with $\max \lambda^r_{rc}$ being for example 3 times as high as $\max \lambda^l_{lr}$, due to the different expression of the constraint equations. After reaching a maximum, already quite soon in the iterative procedure, the mean value of the lagrangian multipliers decreases monotonically. This behavior is to be expected. Indeed, λ_k can be regarded as the rate of change of the quantity being optimized as a function of the constraint variable. As such, it should ideally converge to zero. The mean value of the la-

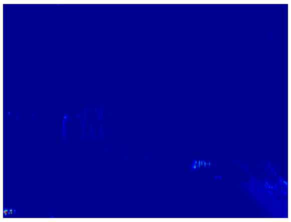

(a) $(\lambda^l_{lc})^2$

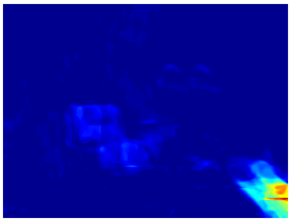

(b) $(\lambda^l_{lc})^{10}$

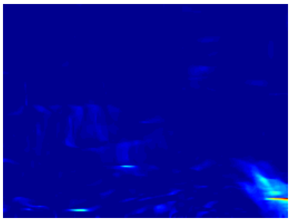

(c) $(\lambda^l_{lc})^{22}$

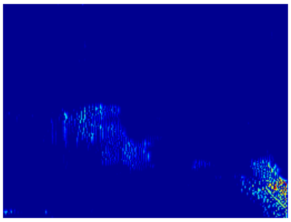

(d) $(\lambda^l_{lr})^2$

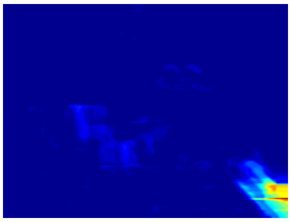

(e) $(\lambda^l_{lr})^{10}$

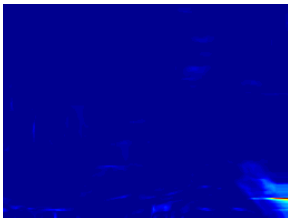

(f) $(\lambda^l_{lr})^{22}$


(g) $(\lambda^r_{rc})^2$


(h) $(\lambda^r_{rc})^{10}$


(i) $(\lambda^r_{rc})^{22}$


(j) $(\lambda^r_{rl})^2$


(k) $(\lambda^r_{rl})^{10}$

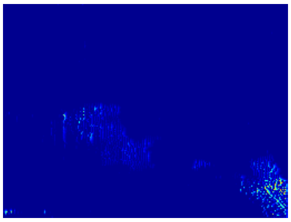

(l) $(\lambda^r_{rl})^{22}$

Figure 9.12: Evolution of the Lagrange Multipliers using the Augmented Lagrangian Algorithm

grangian multipliers associated to both of the "central" constraints, $\max \lambda^r_{rc}$ and $\max \lambda^l_{lc}$, decreases quite dramatically compared to the other two curves, associated to "left-to-right" constraints. Again, this is to be expected, as it is easier to relate the left image to a central image then to the right image, which requires a larger translation and rotation and with it, in general also a larger error. As a result of this behavior, $\max(\lambda^r_{rc})$ and $\max\left(\max \lambda^l_{lc}\right)$ are able to decrease to a value very near zero, indicating good convergence, whereas $\max \lambda^l_{lr}$ and $\max \lambda^r_{rl}$ suffer from a small residual error. This situation can be compared to the evolution of the different constituents of the objective function for the global optimization method, as shown by Figure 9.4. When comparing Figures 9.4 and 9.13, it is clear that the augmented lagrangian based approach achieves a much better balance in the convergence behavior of the different constraints, whereas there is a large discrepancy between stereo and motion constraints in the case of the global optimization based approach. This optimized balancing of constraints is an important benefit of the augmented lagrangian based approach, as it effects directly the end result.

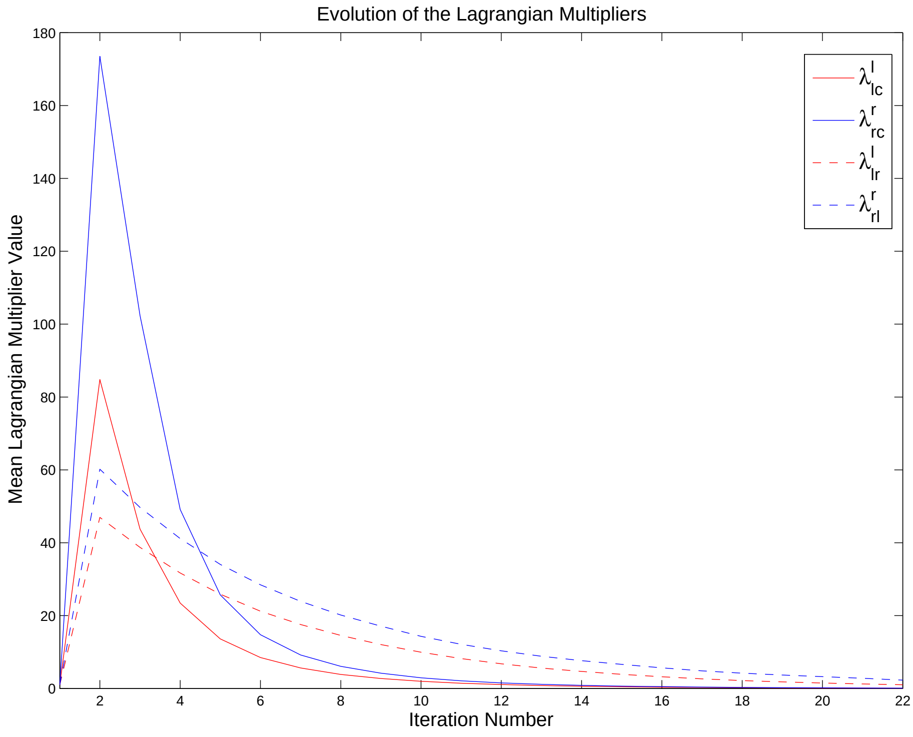


Figure 9.13: Evolution of the Mean Value of the Lagrange Multipliers

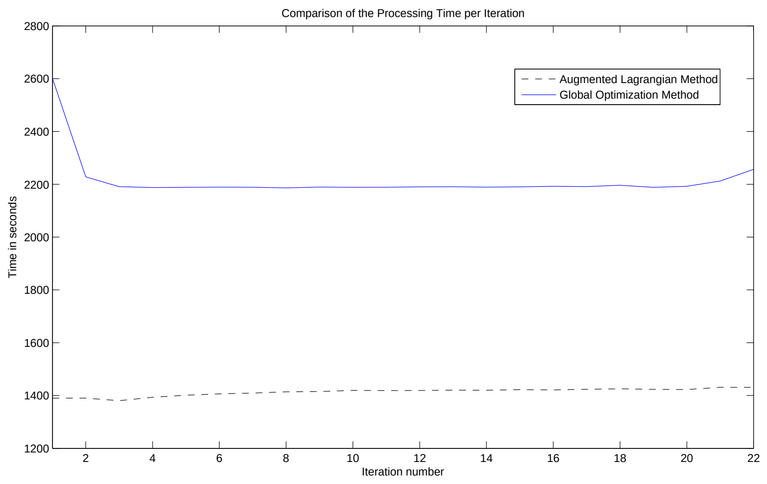


Figure 9.14: Evolution of the processing time per iteration using the Global Optimization and the Augmented Lagrangian Algorithm

Figure 9.14 compares the processing time per iteration step for the Global Optimization and the Augmented Lagrangian Algorithm. The purpose of this graph is not to evaluate the algorithmic processing speed, as none of the algorithms are optimized for speed or intended to be used in real-time applications. However, its purpose is to compare the speed of both algorithms, evaluated on the same computer under the same conditions.

As can be noted from Figure 9.14, the processing time per iteration step is about constant for both algorithms. However, the Augmented Lagrangian based method is considerably faster per iteration than the Global Optimization based technique. Furthermore, the analysis of the convergence of both algorithms, as presented above, taught that the Augmented Lagrangian method converges in fewer iterations to a stable solutions. Taking these 2 factors into account, it can be concluded that the Augmented Lagrangian - based reconstruction method is significantly faster than the Global Optimization based approach.

As a final result of the iterative optimizer, Figure 9.15 shows the reconstructed left and right proximity maps using the Augmented Lagrangian based stereo - motion algorithm. The reconstructed proximity field correlates extremely well with the physical nature of the scene. Foreground and background objects are clearly distinguishable. The depth gradients on the left and back walls can be clearly identified, despite the fact that there is very little texture on these walls. The occurrence of specular reflection on the poster does not cause erroneous reconstruction results. The only remaining errors on the proximity field are in fact due to border effects. Indeed, at the lower left of Figure 9.15(a) and the lower right of Figure 9.15(b), one can notice some areas where the regularization has smoothed out the proximity field not entirely correctly. The reason for this lies in the complete lack of initial proximity data in these areas (see Figure 9.2).

Due to the total absence of proximity information in these areas, the algorithm has smoothed out the solution from the neighboring regions. In general, this was performed correctly, but due to the lack of information, the algorithm estimated the direction of regularization wrongly at these 2 locations. As indicated before, this is quite a normal side-effect when using area-based optimization techniques, which is in general solved by extending the image canvas before the calculations or by cutting the image canvas after the calculations.

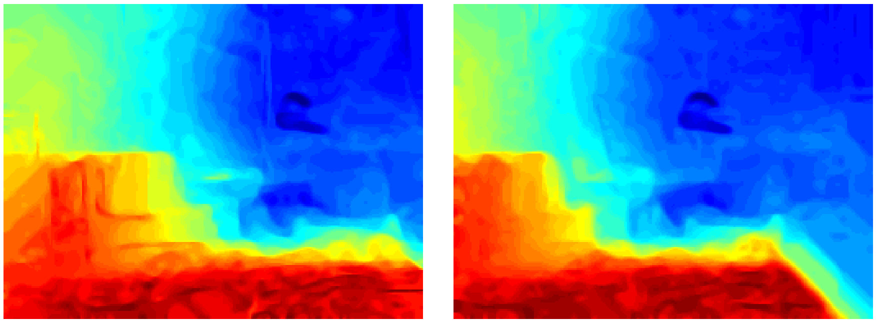

(a) Left proximity map, (d_1^l) (b) Right proximity map, (d_1^r)

Figure 9.15: Left and Right Proximity Maps using the Augmented Lagrangian Algorithm

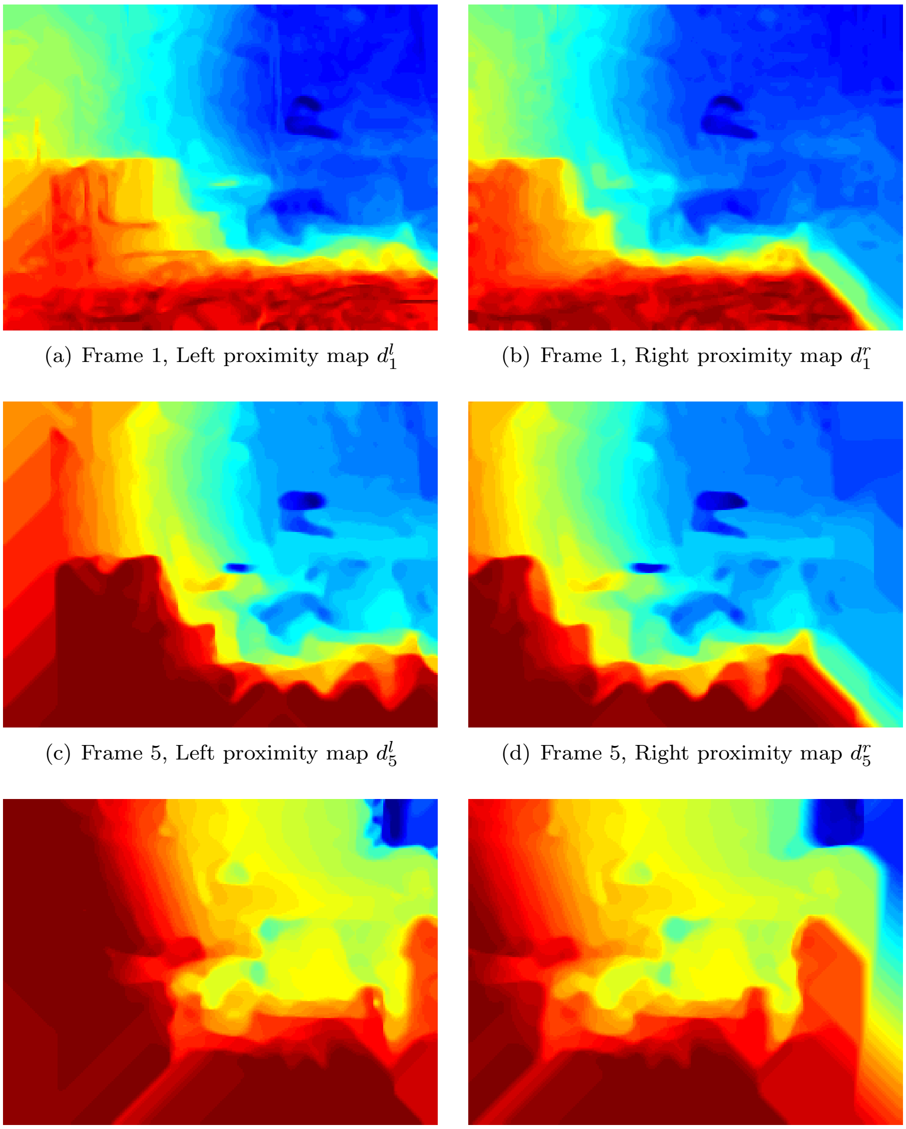
The result of Figure 9.15 can be compared to Figure 9.8, which shows the same output, but using the global optimization approach. From this comparison, it is evident that the result of the Augmented Lagrangian - based reconstruction technique is far superior to the one using global optimization. The global optimization result features numerous problems: erroneous proximity values, under-regularized areas, over-regularized areas, erroneous estimation of discontinuities, ... None of those problems are present in the result of the Augmented Lagrangian, as shown on Figure 9.15.

The reconstruction results for different frames of the *Desk* sequence are shown on Figure 9.16. As the camera translates along its optical axis, it approaches the different items in the environment, leading to a proximity field with higher values.

This results in some saturation in the proximity field, most notable on the left side of Figure 9.16(e). This is in fact not an error of the optimization scheme, but due to the limited dynamical range of the visualization.

As already discussed for Figure 9.15, a number of border effects can be noted for the fifth and the tenth frame as well. However, the general structure of the dense proximity field corresponds well to the physical reality, portrayed in Figure 9.1.

To show the applicability of the presented technique towards 3D modeling, the individual reconstruction results for all frames were integrated to form one consistent 3D representation of the imaged environment. Figure 9.17 shows 4 novel views of the 3D model which was reconstructed as such.



(a) Frame 1, Left proximity map d_1^l

(b) Frame 1, Right proximity map d_1^r

(c) Frame 5, Left proximity map d_5^l

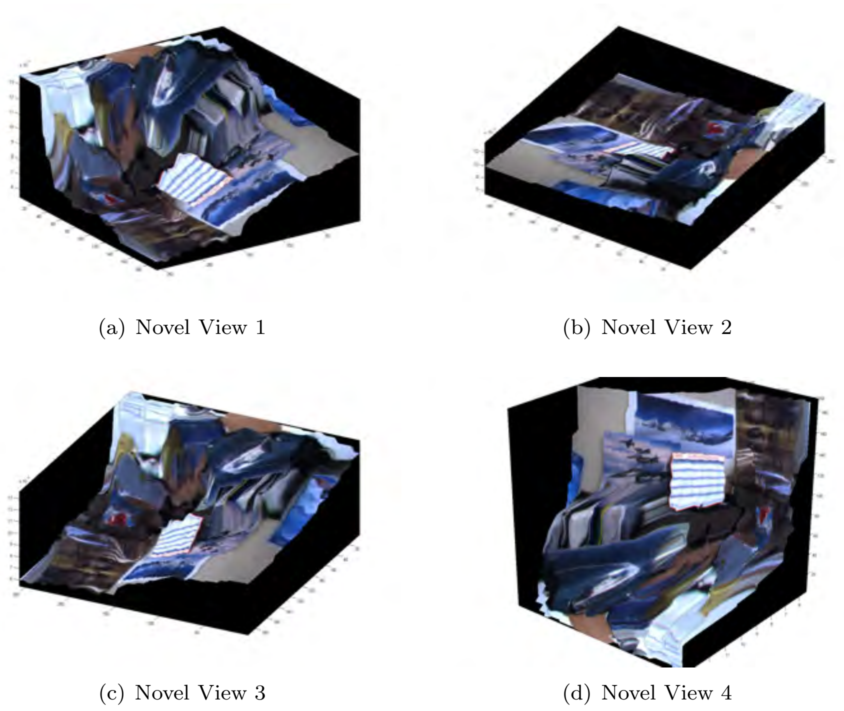
(d) Frame 5, Right proximity map d_5^r

(e) Frame 10, Left proximity map d_{10}^l

(f) Frame 10, Right proximity map d_{10}^r

Figure 9.16: Proximity Maps for different frames of the *Desk* sequence using the Augmented Lagrangian Algorithm

Figure 9.17 shows that a qualitative 3D model can be reconstructed using the Augmented Lagrangian - based stereo-motion reconstruction technique. Indeed, from the different novel viewpoints, the 3D structure of the office environment can be clearly deduced. There are no visible outliers and all items in the scene have been reconstructed, even those with very texture, as such fulfilling the requirement of dense reconstruction.


(a) Novel View 1 (b) Novel View 2

(c) Novel View 3 (d) Novel View 4

Figure 9.17: Reconstructed 3D Model of the Desk sequence using the Augmented Lagrangian Algorithm

9.4 Conclusions

In this chapter, we have evaluated the Augmented Lagrangian based stereo - motion reconstruction algorithm, presented in chapter 8. Therefore, we compared the performance of this approach on a validation sequence with a classical global optimization based approach. The global optimization - based approach defines the classical stereo, motion and regularization constraints, combining them all together in one single objective functional and then using a brute force approach to minimize this objective functional. This classical optimization methodology can hardly be called an "intelligent" approach, as it treats all constraints equally and thus does not discriminate between the different constraints, which is in contradiction with the physical reality that some constraints are more important than others. For example: the objective functional, as given by equation 9.3 considers 1 equation for the expressing the stereo constraint and no less than 4 equations for expressing the motion constraint. This will cause the motion constraint to be applied much harder than the stereo constraint, whereas the stereo constraint is in fact more reliable than the motion constraint, as it is not based upon the error-prone estimation of camera motion parameters.

Another problem of the global optimization based approach is that its formulation of the motion constraint depends heavily on the reliable estimation of a dense optical flow field for both left and right images, which is hardly evident. This expression of the motion constraint disregards the observations of chapter 6, where it was proven that the use of the image-derivatives based optical flow constraint, in conjunction with a proper depth parametrization, can lead to good reconstruction results in the monocular case. As such, it seems more appropriate to build on this experience and to use a similar formulation in the binocular case.

The result of this lack of a proper balancing mechanism between different constraints and the use of an impropriately defined motion constraint becomes evident when evaluating the end result of the proximity field optimization, as presented by Figure 9.9, showing numerous problems of over- and under-relaxation and structural inconsistencies. In contrast, the Augmented Lagrangian method does provide an extended framework of balancing the data and regularization terms, by updating the regularization parameter μ, and by updating the Lagrangian multipliers for each constraint at each iteration. Furthermore, it builds upon the experience accumulated in the monocular reconstruction case for defining a robust motion constraint. This leads to a much better reconstruction result, as shown in Figure 9.15. Not only does the Augmented Lagrangian based stereo - motion reconstruction technique converge to a reconstruction result with a much smaller overall error, the evaluation presented in this chapter also showed that it also converges faster than the classical global optimization based technique.

The only advantage which is offered by the global optimization technique, is that the motion parameters are also iteratively updated using this method. For the Augmented Lagrangian based approach, we have chosen not to do this, because this can lead to optimization problems. Indeed, due to the totally different scale and nature of a proximity field and a motion vector, it is dangerous to incorporate them both in a single optimization scheme. As the main goal of the stereo - motion reconstruction algorithm is structural reconstruction, it is in this case therefore better to focus on the structural aspect, as done by the Augmented Lagrangian based approach.

Chapter 10

Final conclusions and future work

10.1 Conclusions and discussion

The main contribution of this work lies in the development of a novel algorithm for dense 3D reconstruction from monocular image sequences, as presented in chapter 5. The general concept behind this algorithm is that it combines the robustness of traditional sparse structure from motion methods, as discussed in chapters 2 and 3 with the completeness of optical flow based dense reconstruction approaches which were discussed in section 2.5. This was achieved by defining an integrated framework, formulating the problem of fusing dense image data as an optimization problem. The variational approach which was set up in chapter 5 to solve the optimization problem considers two formulations. In chapter 6, the performance of each of these formulations was compared to one another, which led us to conclude that it is better to use the image derivatives based optical flow constraint, than to use the constant image brightness - based flow constraint, as is done by most classical approaches. A comparison to other state-of-the-art dense reconstruction techniques learned that the proposed dense reconstruction approach performs excellent. Without being the actual top performer for one specific quality measure, it succeeds at estimating a globally optimal reconstruction, which balances the accuracy and completeness measures. One of the disadvantages of the proposed algorithm is that it suffers from a slight over-relaxation, due to the automated process of estimating the diffusion parameter μ and therefore, the level of fine detail can sometimes be a bit lower. However, globally, our method delivers robust and reliable 3D reconstruction results, due to the relative absence of very large errors. As such, it proves to be a valuable candidate for the dense reconstruction of natural sequences.

As advocated in chapter 7, optimal depth perception requires the fusion of different depth cues. Therefore, we extended the monocular dense reconstruction algorithm to the binocular case, by integrating it in a dense stereo - motion reconstruction framework. The combination of temporal and spatial information makes it possible to reduce the uncertainty in the depth reconstruction result and to augment the precision, but this comes at the cost of an increased computational complexity, due to the large number of constraints to be considered. In chapter 8, we presented a novel solution to this problem, by simultaneously optimizing the left and right proximity field, using the methodology of the Augmented Lagrangian. Chapter 9 evaluates the performance of the proposed algorithm and compares the presented methodology to a more traditional global optimization based approach. The experiments presented in chapter 9 show that the quality of the results using the proposed methodology far exceeds the quality of the results of the traditional method. This allows us to conclude, that, also for the binocular case, the proposed 3D reconstruction algorithm presents an excellent reconstruction tool.

10.2 Future Work

Although this work showed the potential of the presented monocular and binocular 3D reconstruction techniques, the processing time required by each of those algorithms is still too high for application in practice. Therefore, further investigations should be carried out to decrease the required calculation time. This can be done in the first place by applying more intelligent scale-conscious numerical schemes. Indeed, the presented methodology does not apply any form of scale-based reasoning, meaning that all calculations are performed for all pixels at maximum resolution, leading to an excessive processing time. Multi-resolution methods like Multigrid could drastically speed up this computing process by introducing a hierarchy of scales and solving these from the coarsest scale to the finest scale. Furthermore, most of the algorithms described in this dissertation lend themselves very well to parallelization, which opens the door for more intelligent programming schemes targeted at multi-core CPUs, or even for GPUs.

The main focus of the 3D reconstruction in this work has been on the production of dense depth or proximity maps. The 3D modeling aspect, where multiple individual reconstructions are integrated into one consistent 3D model, has received considerably less attention. In this dissertation, we used the well-known Iterative Closest Point algorithm for this purpose, but this method is quite old and extremely slow. A natural extension of the presented research work, would be to use the results obtained by the monocular and binocular reconstruction techniques as input for a more modern 3D modeling algorithm. In combination with the aforementioned decrease in processing time, this would mean that real-time 3D modeling becomes possible, even with a simple household camera.

Appendices

Appendix A

Feature Detection, Description and Matching

A.1 Feature Detection

Since the early work of Moravec [98] for stereo matching, many point extractors have been proposed in the literature of Computer Vision. Few comparison studies has been done for these approaches. See for example the ones of 2000 for grey value images [126] and color images [45]. The most popular feature detector is probably the Harris and Stephens detector [48], which has been used first for stereo purposes and then for image retrieval. The Harris corner detector is such a popular interest point detector due to its strong invariance to [126]: rotation, scale, illumination variation and image noise. The Harris corner detector is based on the local auto-correlation function of a signal; where the local auto-correlation function measures the local changes of the signal with patches shifted by a small amount in different directions.
Given a shift $(\Delta x, \Delta y)$ and a point (x, y), the distance function is defined as,

$$c\,(x,y) = \sum_{W} \left[I(x_i, y_i) - I(x_i + \Delta x, y_i + \Delta y) \right]^2 \tag{A.1}$$

where I denotes the image function and (x_i, y_i) are the points in the window W centered on (x_i, y_i). The shifted image is approximated by a Taylor expansion truncated to the first order terms,

$$I(x_i + \Delta x, y_i + \Delta y) \approx I(x_i, y_i) + \left[\begin{array}{cc} I_x(x_i, y_i) & I_y(x_i, y_i) \end{array} \right] \left[\begin{array}{c} \Delta x \\ \Delta y \end{array} \right] \tag{A.2}$$

where $I_x(x_i, y_i)$ and $I_y(x_i, y_i)$ denote the partial derivatives in x and y, respectively.
Substituting approximation A.2 into A.1 yields:

$$c\left(x,y\right)=\left[\begin{array}{cc}\Delta x & \Delta y\end{array}\right]\left[\begin{array}{cc}\sum\limits_{W} I_x(x_i,y_i)^2 & \sum\limits_{W} I_x(x_i,y_i)I_y(x_i,y_i)\\ \sum\limits_{W} I_x(x_i,y_i)I_y(x_i,y_i) & \sum\limits_{W} I_y(x_i,y_i)^2\end{array}\right]\left[\begin{array}{c}\Delta x\\ \Delta y\end{array}\right] \tag{A.3}$$

or shorter:

$$c\left(x,y\right)=\left[\begin{array}{cc}\Delta x & \Delta y\end{array}\right]\mathbf{H_C}\left(x,y\right)\left[\begin{array}{c}\Delta x\\ \Delta y\end{array}\right] \tag{A.4}$$

where matrix $\mathbf{H}_C\left(x,y\right)$ captures the intensity structure of the local neighborhood. A set of features can then be obtained by computing the local maxima of the function $Det(\mathbf{H}_C(x,y)) - k.Trace(\mathbf{H}_C(x,y))^2$ then . A modified version of this detector has been proposed in [125] by improving the computation of the spatial derivatives with precise gaussian derivatives. This precise version allows to gain in repeatability, as demonstrated in [126]. The precise version of the Harris detector has been also extended to deal with color images in [97], where it gets a better repeatability [45].

Despite all these improvements, the Harris corner detector also suffers from some limitations: the Harris detector (and its precise or color version) is invariant to rotation but is very sensitive to changes in image scale. Recent works proposed derived or new detectors to achieve scale invariance. The problem of identifying an appropriate scale for feature detection has been studied by Lindeberg who has described it as a problem of selection of a characteristic scale [81]. From these considerations, several works on scale invariance have been proposed for local features. In [80], Lindeberg has found a stable keypoint location in scale space by searching for 3D maxima of a function based on the Laplacian normalized with the scale. On the other hand, Lowe considered the local extrema in scale-space of Differences of Gaussian (DoG) images [84]. Such points of interest are often called DoG points. As demonstrated by Lowe and evaluated later in [93], the DoG approach represents a close approximation of the Laplacian one, which is successfully compared to other functions (the Gradient and the standard Harris functions). Based on the DoG detector [84], Lowe has proposed the Scale Invariant Feature Transform approach (SIFT) [85] for describing the local neighborhood of such points. In the following, the SIFT methodology, developed by Lowe, is introduced in order to give the reader some background information about the feature detection, description and matching processes which are used in the sparse reconstruction algorithm, as described in chapter 3.

SIFT features are located at scale-space maxima/ minima of a difference of Gaussian function. The scale space of an image is defined as a function, $L(x,y,\sigma)$, that is produced from the convolution of a *variable-scale* Gaussian, $G(x,y,\sigma)$, with an input image, $I(x,y)$:

$$L(x,y,\sigma) = G(x,y,\sigma) \otimes I(x,y) \tag{A.5}$$

where $G(x,y,\sigma)$ is defined by equation C.2.

To efficiently detect stable keypoint locations in scale space, Lowe proposes in [85] to use scale-space extrema in the difference-of-Gaussian function convolved with the image, $D(x, y, \sigma)$, which can be computed from the difference of two nearby scales separated by a constant multiplicative factor k:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) \otimes I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{A.6}$$

In order to detect the local maxima and minima of $D(x, y, \sigma)$, each sample point is compared to its eight neighbors in the current image and nine neighbors in the scale above and below. It is selected only if it is larger than all of these neighbors or smaller than all of them.

An important issue is to determine the frequency of sampling in the image and scale domains that is needed to reliably detect the extrema. The scale-space difference-of-Gaussian function has a large number of extrema and that it would be very expensive to detect them all. Fortunately, it is possible to detect the most stable and useful subset even with a coarse sampling of scales.
Once a keypoint candidate has been found by comparing a pixel to its neighbors, the next step is to perform a detailed fit to the nearby data for location, scale, and ratio of principal curvatures. This information allows points to be rejected that have low contrast (and are therefore sensitive to noise) or are poorly localized along an edge. Lowe uses the Taylor expansion (up to the quadratic terms) of the scale-space function, $D(x, y, \sigma)$, shifted so that the origin is at the sample point:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D^T}{\partial \mathbf{x}^2}\mathbf{x} \tag{A.7}$$

where D and its derivatives are evaluated at the sample point and $\mathbf{x} = (x, y, \sigma)^T$ is the offset from this point. The location of the extremum, $\hat{\mathbf{x}}$, is determined by taking the derivative of this function with respect to $\mathbf{x}$ and setting it to zero, giving:

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1} \frac{\partial D}{\partial \mathbf{x}} \tag{A.8}$$

If the offset $\hat{\mathbf{x}}$ is larger than 0.5 in any dimension, then it means that the extremum lies closer to a different sample point. In this case, the sample point is changed and the interpolation performed instead about that point. The final offset $\hat{\mathbf{x}}$ is added to the location of its sample point to get the interpolated estimate for the location of the extremum. The function value at the extremum, $D(\hat{\mathbf{x}})$, is useful for rejecting unstable extrema with low contrast. This can be obtained by substituting equation A.8 into A.7, giving:

$$D(\hat{\mathbf{x}}) = D + \frac{1}{2}\frac{\partial D^T}{\partial \mathbf{x}}\hat{\mathbf{x}} \tag{A.9}$$

All extrema with a value of $|D(\hat{\mathbf{x}})|$ less than a certain threshold are discarded.

For stability, it is not sufficient to reject keypoints with low contrast. The difference-of- Gaussian function will have a strong response along edges, even if the location along the edge is poorly determined and therefore unstable to small amounts of noise. A poorly defined peak in the difference-of-Gaussian function will have a large principal curvature across the edge but a small one in the perpendicular direction. The principal curvatures can be computed from a 2×2 Hessian matrix, $\mathbf{H}$, computed at the location and scale of the keypoint:

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \tag{A.10}$$

In order to discard edge responses, it suffices to check that the ratio of principal curvatures is below some threshold, r:

$$\frac{Trace(\mathbf{H})^2}{Det(\mathbf{H})^2} < \frac{(r+1)^2}{r} \tag{A.11}$$

Since the vector of gradients consists of differences of intensity values, it is invariant to affine changes in intensity. Due to these advantageous properties, the SIFT feature detector was selected for use throughout this work.

A.2 Feature Description

Several feature descriptors [85],[70][11] have been proposed in the literature. The simplest descriptor is a vector of image pixels. As an alternative, the local intensity variation matrix $\mathbf{H_C}$ can be used as a feature descriptor. Distribution based descriptors [157] use histograms to represent different characteristics of appearance or shape, which typically results in very robust descriptors. A simple descriptor is the distribution of the pixel intensities represented by a histogram. The SIFT descriptor, as introduced by Lowe in [85] is a distribution based descriptor.

The SIFT descriptor assigns to each feature an image location, a scale, an orientation and a description vector, based on local image properties, as discussed in the following paragraphs. The keypoint descriptor is represented relative to the scale and orientation and is therefore invariant to image scale and rotation.

The scale of the keypoint is used to select the Gaussian smoothed image, L, with the closest scale, so that all computations are performed in a scale-invariant manner. For each image sample, $L(x,y)$, at this scale, the gradient magnitude, $m(x,y)$, and orientation, $\theta(x,y)$, are computed using pixel differences:

$$\begin{aligned} m_g(x,y) &= \sqrt{\left(L(x+1,y)-L(x-1,y)\right)^2 + \left(L(x,y+1)-L(x,y-1)\right)^2} \\ \theta_g(x,y) &= \tan^{-1}\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right) \end{aligned} \tag{A.12}$$

An orientation histogram is formed from the gradient orientations of sample points within a region around the keypoint. The orientation histogram has 36 bins covering the 360 degree range of orientations. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a scale σ that is 1.5 times that of the scale of the keypoint. Peaks in the orientation histogram correspond to dominant directions of local gradients. The highest peak in the histogram is detected, and then any other local peak that is within 80% of the highest peak is used to also create a keypoint with that orientation. Therefore, for locations with multiple peaks of similar magnitude, there will be multiple keypoints created at the same location and scale but different orientations.

The previous operations have assigned an image location, scale, and orientation to each keypoint. These parameters impose a repeatable local 2D coordinate system in which to describe the local image region, and therefore provide invariance to these parameters. The next step is to compute a descriptor for the local image region that is highly distinctive yet is as invariant as possible to remaining variations, such as change in illumination or 3D viewpoint.

To achieve this, first the image gradient magnitudes and orientations are sampled around the keypoint location, using the scale of the keypoint to select the level of Gaussian blur for the image. In order to achieve orientation invariance, the coordinates of the descriptor and the gradient orientations are rotated relative to the keypoint orientation.

A Gaussian weighting function with scale σ equal to one half the width of the descriptor window is used to assign a weight to the magnitude of each sample point. The purpose of this Gaussian window is to avoid sudden changes in the descriptor with small changes in the position of the window, and to give less emphasis to gradients that are far from the center of the descriptor, as these are most affected by misregistration errors.

The keypoint descriptor allows for significant shift in gradient positions by creating orientation histograms over 4×4 sample regions. It is important to avoid all boundary affects in which the descriptor abruptly changes as a sample shifts smoothly from being within one histogram to another or from one orientation to another. Therefore, trilinear interpolation is used to distribute the value of each gradient sample into adjacent histogram bins. The descriptor is formed from a vector containing the values of all the orientation histogram entries. A 4×4 array of histograms with 8 orientation bins each is used in general, which means that the feature vector constructed for each keypoint has $4 \times 4 \times 8 = 128$ elements.

Finally, the feature vector is modified to reduce the effects of illumination change. First, the vector is normalized to unit length. A change in image contrast in which each pixel value is multiplied by a constant will multiply gradients by the same constant, so this contrast change will be canceled by vector normalization. A brightness change in which a constant is added to each image pixel will not affect the gradient values, as they are computed from pixel differences. Therefore, the descriptor is invariant to affine changes in illumination. However, non-linear illumination changes can also occur due to camera saturation or due to illumination changes that affect 3D surfaces with differing orientations by different amounts. These effects can cause a large change in relative magnitudes for some gradients, but are less likely to affect the gradient orientations. Therefore, the influence of large gradient magnitudes is reduced by thresholding the values in the unit feature vector to be each no larger than a threshold value of 0.2, and then renormalizing to unit length. This means that matching the magnitudes for large gradients is no longer as important, and that the distribution of orientations has greater emphasis.

A.3 Feature Matching

Matching is the third step in the feature correlation finding process: a given feature is associated with one or more features in other images. Important aspects of matching are metrics and criteria to decide whether two features should be associated, and data structures and algorithms for matching efficiently.

The main problem is that point descriptors are subject to different kinds of noises: in practice, they are sensitive to image acquisition (sensors and sampling errors), to numerical errors, to points of interest delocalization, etc. These considerations show the importance of the similarity measure which must be carefully chosen for the considered descriptor to achieve best performances.

A standard metric for computing a similarity score between two descriptors consisting of image intensity samples, is cross-correlation. For this technique, all features within a certain disparity limit are compared over the two images. In the course of the matching process, there are often several candidate matches for each feature. Initially, the one that is most correlated in image intensities at the corner positions is selected. The strength of the match is obtained by cross-correlation of image intensity over two pixel patches centered on each feature:

$$C = \sum_{i,j \in patch} \left(I_2\left(i,j\right) - I_1\left(i,j\right)\right)^2, \tag{A.13}$$

where $I_n\left(i,j\right)$ is the image intensity at coordinate (i,j) in the n^{th} image. The match with the maximum strength is stored for each corner from the first to the second image. The same process is then reversed from the second to the first image. Matches are only accepted into the initial set if they exhibit a maximum in both comparisons. This has the effect of removing matches which are ambiguous because they have multiple candidate matches. Correlation matching can work quite well when there is only a small change in illumination. However, for significant changes in perspective and lighting, the intensities of corresponding points can undergo large changes, resulting in the failure of correlation matching.

Therefore, SIFT features are a better candidate for matching purposes. Multiple feature matchers [85][125][24] have been proposed in the literature for vector-based descriptors as used in SIFT. These approaches usually compute the similarity between descriptors by calculating the Euclidian or Mahalanobis distance d_M.

$$d_M(\mathbf{v_1}, \mathbf{v_2}) = \sqrt{\left(\mathbf{v_1} - \mathbf{v_2}\right)^T \mathbf{\Sigma^{-1}} \left(\mathbf{v_1} - \mathbf{v_2}\right)}, \tag{A.14}$$

with $\mathbf{v_1}$ and $\mathbf{v_2}$ two (SIFT-) descriptor vectors. The involved covariance matrix $\mathbf{\Sigma}$ takes the different magnitudes, possible correlations and variability of the feature components into account. The best candidate match for each feature is then found by finding its nearest neighbor in the other image using the descriptor vector and image location, scale and orientation information. As the SIFT keypoint descriptor has a 128-dimensional feature vector, this process would require a lot of processing time when using an exhaustive search algorithm. Therefore, solutions have been proposed to speed up this process.

Matching can be performed using a k-D tree, which is a balanced binary tree. It can be thought of as a coarse grained density map of the distribution of data points. To build the k-D tree, the database is split on the dimension with the largest variance. This continues until each node on the tree is categorized. Each search through the tree is first performed on the k-d tree and search times grow logarithmically with the number of data points. The used k-D tree has a very large dimension (128) corresponding to the dimensionality of the SIFT descriptor. The nearest neighbor is defined as the keypoint with minimum Euclidean distance for the invariant descriptor vector. Even with the k-D tree approach, matching can take a long time, due to the search through a high-dimensional vector space as offered by the descriptor vector of the SIFT-features. To address this problem, Lowe used in [85] the Best-Bin-First (BBF) algorithm introduced by Beis in [10]. This is approximate in the sense that it returns the closest neighbor with high probability. The BBF algorithm uses a modified search ordering for the k-D tree algorithm so that bins in feature space are searched in the order of their closest distance from the query location. This priority search order requires the use of a heap-based priority queue for efficient determination of the search order. An approximate answer can be returned with low cost by cutting off further search after a specific number of the nearest bins have been explored. However, if the cutoff threshold is chosen too low, the k-D tree based feature matching process tends to loose points quite quickly. Therefore, re-initialization with new keypoints is required to keep the number of features high enough when processing long image sequences.

Appendix B

Methods for estimating the Optical Flow

The main goal of the optical flow estimation is to obtain smooth flow fields on uniform regions, while maintaining the motion information on the boundaries and on the edges of the moving objects. The main problem in obtaining a precise optical flow field is to maintain the discontinuities [108].

In the literature, there exist numerous computational approaches for estimating optical flow, they are mainly classified into the following approaches: frequency based methods, correlation based methods, model based methods, differential methods and others. The boundaries between each class of methods are not always clear. In general, differential methods have been proved to be among the best optical flow estimation techniques [8], which is why this section focusses on the differential methods.

Differential methods estimate optical flow from spatio-temporal derivatives of image intensity or filtered versions of the image. This gives a dense flow field whose values are available throughout the image plane, the optical flow is recovered based on local spatial-temporal changes in image intensity. The fast emerging use of PDE-based image processing methods such as nonlinear diffusion filtering for image enhancement and restoration [169], has motivated many researchers to apply similar ideas to estimate optical flow.

Many differential methods can be expressed in terms of a variational problem where the optical flow minimizes some energy. These energy functionals normally consist of two terms: a data term, which comes from image data, e.g., intensity constancy assumption; and a regularization term that contains the certain constraints on the flow field. The functional to be minimized is expressed as

$$E(u,v) = \int_{\Omega} (\alpha E_{data}(u,v) + E_{regularization}(u,v))d\Omega = \int_{\Omega} F d\Omega \qquad \text{(B.1)}$$

where $E_{data}(u,v)$ is the optical flow constraint which belongs to the data term , usually chosen as

$$E_{data}(u,v) = (\nabla I \cdot \mathbf{u} + I_t)^2 \tag{B.2}$$

For the regularization term $E_{regularization}(u,v)$, the smoothness constraint is most considered. The most used and simple candidate is the Tikhonov quadratic regularization. The pioneering work by Horn and Schunck [62] introduced the Tikhonov regularization term to constrain the optical flow solution. This regularization assumes that the optical flow field is smooth. Smoothing takes place isotropically ignoring the discontinuities. It has the effect of propagating the velocity information into the areas of uniform image intensity. It uses a global smoothness term in order to compute an optical flow field from sparse local motion information.

$$E_{regularization}(u,v) = \|\nabla u\|^2 + \|\nabla v\|^2 = u_x^2 + u_y^2 + v_x^2 + v_y^2 \tag{B.3}$$

where $\nabla u, \nabla v$ measure how rapidly the velocity is changing across the image. u_x, u_y, v_x, v_y are partial derivatives of u and v along x and y directions respectively.

It smoothes the flow field isotropically without taking into account the discontinuities of the flow field. It tends to blur the flow field at genuine motion boundaries. This explains the reason why the Horn and Schunck approach creates a rather blurred optical flow field. However, in practice, the motion field is not always smooth, discontinuities are inevitable due to boundaries, occlusions, etc. Therefore, many researchers work on improving the Horn and Schunck approach.

Following Lucas and Kanade [87], Barron et al [9] implemented a weighted least squares fit of a local optical flow constraint for $\mathbf{u}$ in each small spatial neighborhood Ω by minimizing

$$\min \sum_{\mathbf{x}\in\Omega} W^2(\mathbf{x})[\nabla I(\mathbf{x},t)\mathbf{u} + I_t(\mathbf{x},t))]^2 \tag{B.4}$$

where $W(\mathbf{x})$ denotes a window function introduced in order to give stronger constraints at the center of the neighborhood than those at the periphery.

As an alternative to the global quadratic smoothness constraint by Horn and Schunck, Nagel and Enkelmann proposed in [105] an orientation smoothness constraint in which the smoothness requirement is only imposed orthogonal to the intensity gradient in order to handle motion discontinuities. The problem is formulated as the minimization of the following functional:

$$E(\mathbf{v}) = \int_\Omega (I_1(x-u,y-v) - I_2(x,y))^2 dxdy + \mu \int_\Omega trace((\nabla\mathbf{u})^T D(\nabla I_1)(\nabla\mathbf{u}))dxdy \tag{B.5}$$

where μ is a positive constant, $\nabla\mathbf{u} = \begin{pmatrix} u_x & v_x \\ u_y & v_y \end{pmatrix}$, and $D(\nabla I_1)$ is a regularized projection matrix in the direction perpendicular of ∇I_1 :

$$D(\nabla I_1) = \begin{pmatrix} p & q \\ q & r \end{pmatrix} = \frac{1}{|\nabla I_1|^2 + 2\upsilon^2} \begin{pmatrix} \left(\frac{\partial I_1}{\partial y}\right)^2 + \upsilon^2 & -\frac{\partial I_1}{\partial x}\frac{\partial I_1}{\partial y} \\ -\frac{\partial I_1}{\partial x}\frac{\partial I_1}{\partial y} & \left(\frac{\partial I_1}{\partial x}\right)^2 + \upsilon^2 \end{pmatrix}, \tag{B.6}$$

with υ a regularization parameter.

Appendix C

Image De-Noising and Interpolation

A raw camera image can be regarded as a 2-dimensional function. Image processing algorithms as the ones discussed further in this text operate on these 2-dimensional functions to extract useful information. However, when working with raw camera images, two problems often occur:

1. The image function is a highly discontinuous function. This is partly due to discontinuities in the imaged scene, but often also due to noise superposed on the image data.

2. The image function is a discrete function, only defined at the pixel coordinate points. In general, it is desirable to work with continuous functions.

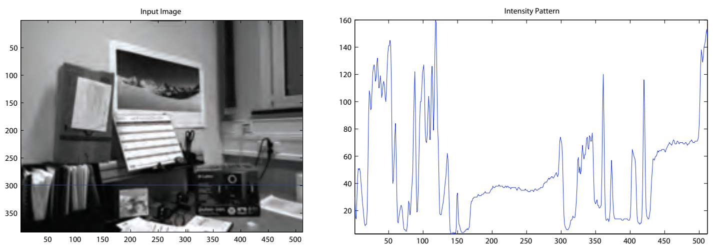


Figure C.1: The Camera Image as a Discrete Discontinuous function of Space: the left figure shows the raw camera image; the right figure shows a cross section of the intensity pattern along the blue line in the left figure

Figure C.1 illustrates both problems by showing a cross section along a horizontal line in the image. It can be noted that the 1-dimensional function of image intensities is highly discontinuous in some regions.

Both problems present serious problems for a dense reconstruction algorithm. Image discontinuities due to noise are propagated in the processing pipeline and often lead to even larger errors in the reconstructed depth field. Therefore, the image needs to be smoothed before any further processing. Due to the fact that the image is a discrete function, it is only possible to apply integer arithmetics on the image coordinates, which limits the computational possibilities drastically. As such, all images first need to be turned into continuous functions, which are more easy to work with. To be clear, the reason for doing this is *not* that we aspire to achieve an improved sub-pixel precision on the end result. By making the image function continuous, no real new data is added, so we cannot hope to improve the basic algorithm accuracy.

To address the first issue, the image is convolved with a Gaussian function G:

$$I_{blurred}(x,y) = I(x,y) \otimes G(x,y,\sigma) = \sum_{j=1}^{height} \sum_{k=1}^{width} I(j,k)G(x-j, y-k), \tag{C.1}$$

with the 2-dimensional Gaussian function defined as:

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{C.2}$$

For image smoothing, the standard deviation σ of the Gaussian distribution is a fixed parameter, such that $G(x,y,\sigma) = G(x,y)$. The result is a smoothed , more continuous image, as shown on Figure C.2*d* and *e*, when compared to the non-blurred versions of Figure C.2*b* and *c*. Note that to obtain Figure C.2*d* and *e*, the amount of Gaussian smoothing was exaggerated to show the effect more clearly.

To turn the image function into a continuous function, the image is interpolated. The interpolation problem can be stated as follows: *Given a set of discrete data f_k, build a continuous function $f(x)$, which optimally fits the model described by the discrete data.* Many image interpolation algorithms have been proposed [172, 160, 71, 64]. The most simple ones are linear algorithms of the form:

$$I_{interpolated}(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^2} I(\mathbf{k})\phi_{\text{int}}(\mathbf{x}-\mathbf{k}), \tag{C.3}$$

where an interpolated value $I_{interpolated}(\mathbf{x})$ at some (perhaps non-integer) coordinate $\mathbf{x}(x,y)$ is expressed as a linear combination of the samples $I(\mathbf{k})$ evaluated at integer coordinates $\mathbf{k} = (k_1, k_2) \in \mathbb{Z}^2$, the weights being given by the values of the synthesis function $\phi_{\text{int}}(\mathbf{x}-\mathbf{k})$.

Several synthesis functions have been proposed. The synthesis function associated to nearest-neighbor interpolation is the simplest of all, since it is made of a square pulse. Its expression is given by:

$$\phi^{nearest-neighbor}(x) = \begin{cases} 0 & x < -\frac{1}{2} \\ 1 & -\frac{1}{2} \leqslant x < \frac{1}{2} \\ 0 & \frac{1}{2} \leqslant x \end{cases} \tag{C.4}$$

Its main problem is that it is discontinuous, thus has no regularity. In fact, for any coordinate $\mathbf{x}$ where it is desired to compute the value of the interpolated function $\mathfrak{I}$ there is only one sample $I(\mathbf{k})$ that contributes.

The bilinear interpolant is made of the (continuous-signal) convolution of a square pulse with itself, which yields a triangle, sometimes also named a hat or a tent function. This interpolant is continuous, but not differentiable, which is a problem for some of the algorithms used in this work which require second order derivatives. Therefore, we use bi-cubic interpolation using cubic splines to interpolate the image function. Bi-cubic spline interpolation operates by building a smooth surface between each set of 4 neighboring pixels, which are considered the corners of a regular grid, through the knowledge of the intensity values $I(\mathbf{x})$ and the derivatives $I_x(\mathbf{x})$, $I_y(\mathbf{x})$ and $I_{xy}(\mathbf{x})$ at the corner points. This interpolated surface can be written as:

$$I_{interpolated}(x,y) = \sum_{i=0}^{3}\sum_{i=0}^{3} a_{ij}x^i y^j, \tag{C.5}$$

with a_{ij} 16 coefficients which need to be determined by expressing constraints on $I(\mathbf{x})$ and the derivatives $I_x(\mathbf{x})$, $I_y(\mathbf{x})$ and $I_{xy}(\mathbf{x})$ for each of the 4 corner points of the grid. Bi-cubic spline interpolation delivers a smooth interpolated image function with continuous first and second order derivatives, as required for some of the algorithms discussed later in this text. Figure C.2 illustrates the interpolation result by comparing the discrete raw data in Figures C.2*b* and *c* with the continuous function built by bi-cubic spline interpolation in Figures C.2*f* and *g*.

In order not to overload the notations, the notation I is used throughout this text to denote both the blurred form of the image $I_{blurred}$ and the continuous interpolated form of the image function $I_{interpolated}$.

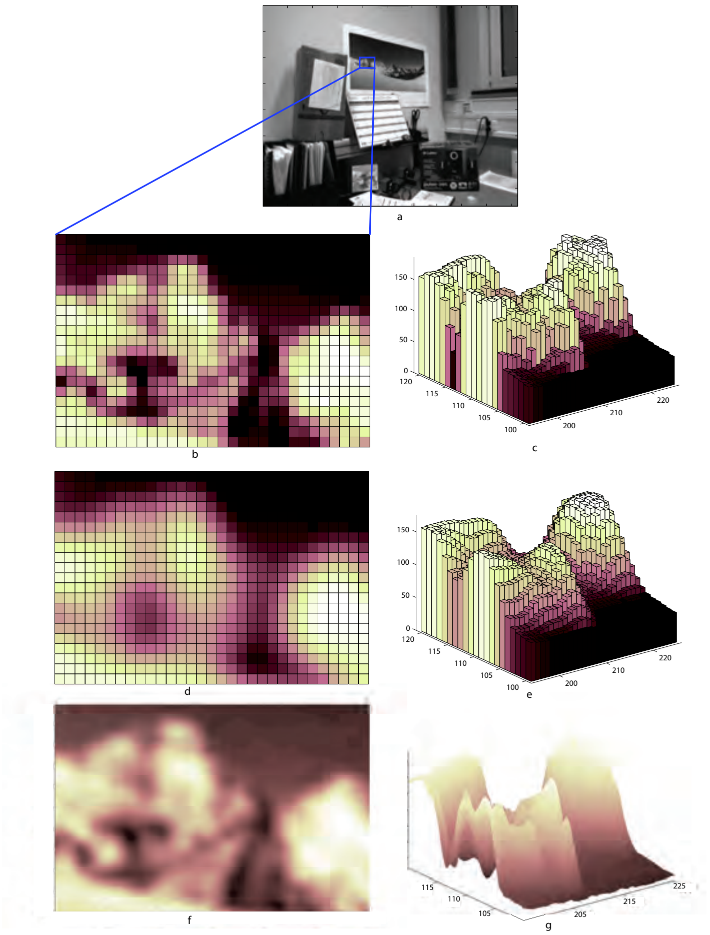


Figure C.2: The Effect of image blurring and interpolation operation with a) the original input image with the region of interest indicated by the blue rectangle; b) a zoomed view of the selected region of interest; c) the intensity function as a 3D function; d and e) the blurred versions of b and c; f and g) the interpolated versions of b and c.

Appendix D

Depth from sparse motion-augmented stereo

D.1 Introduction

Stereo vision is one of the most active fields of research within the computer vision community. Hence, stereo algorithms have gained great maturity over the past decades. The focus of this research work is *not* to develop new stereo vision algorithms, but rather to investigate if and how the addition of SfM motion data could improve existing stereo algorithms. Therefore, it is not our goal to overwhelm the reader with a detailed explication of stereo algorithms. Instead, a short overview is given of the working principles of different stereo algorithms. For this, we base ourselves on the excellent taxonomy [124] on dense two-frame stereo vision algorithms, written by Scharstein and Szeliski.

Scharstein ans Szeliski based their taxonomy on the observation that most stereo algorithms perform the following four steps [124]:

1. Matching cost computation:
   The most popular matching costs are the *Absolute intensity Differences (AD)* and the *Squared intensity Differences (SD)*. In [13], Birchfield and Tomasi proposed a matching cost that is insensitive to image sampling as an extension to these algorithms. Instead of comparing the value of each pixel to shifted pixel values, they compare to a linearly interpolated function of the other image.

2. Cost support aggregation:
   Comparison of raw pixel values for disparity calculation is prone to errors. Therefore, most local methods aggregate the matching cost by summing or averaging over a support region in the Disparity Space Image $C(x, y, d)$. In general, aggregation with a fixed support region can be performed using 2D or 3D convolution. In practice mostly rectangular windows are used which can be implemented efficiently using box-filters. Shiftable windows can also

be implemented efficiently using a sliding min-filter. Another aggregation method is *iterative diffusion* [123] [132]. Using this technique, the weighted values of neighboring pixels' costs are repeatedly added to each pixel's cost value.

3. Disparity computation and optimization:
   For local methods, this processing step is trivial: they simply choose the disparity associated with the minimum cost value. This is the so-called *Winner-Take-All (WTA)* principle. For global methods on the other hand, this processing step is the most important one. These methods are often formulated as an energy minimization problem with a data term $E_{data}(d)$ representing how well the disparity function d fits with the input image pair and a smoothness term $E_{smoothness}(d)$ enforcing smoothness constraints, to form an energy function:

   $$E(d) = E_{data}(d) + \lambda E_{smoothness}(d) \tag{D.1}$$

   Numerous techniques exist to solve equation D.1. Among the more popular ones, one can find simulated annealing and - more recently - graph cut methods. A more efficient approach towards computing time are dynamic programming techniques. These approaches work by computing the minimum-cost path through the matrix of all pairwise matching costs between two corresponding scanlines. Bobick and Intille presented in [15] such a dynamic programming approach, which is used as a reference algorithm in this work. Another optimization technique used as a reference is the scanline optimization approach introduced by Scharstein and Szeliski in [124]. Unlike regular dynamic programming, this method is asymmetric and does not utilize visibility or ordering constraints. Instead, a disparity value is assigned to each point such that the overall cost along the scanline is minimized.

4. Disparity refinement:
   Most stereo algorithms only estimate one (integer) disparity value for each image pixel. Sub-pixel precision can be achieved by fitting correlation curves to the integer-sampled matching costs. Other refinement techniques aim to clean up mismatches, to fill holes, to fit surfaces, ... In the course of this work, we do not consider these "post-processing" steps, as they are often application dependent.

D.2 The proposed methodology

Considering the different steps of classical stereo processing algorithms, as discussed in the former section, it is clear that motion information is mostly useful in the first step: matching cost computation. In this step, the stereo algorithm searches a range of disparities and accords a matching cost to each disparity for each pixel. However, next to the stereo disparity, SfM could also deliver an estimate of the new disparity level for each pixel. Figure D.1 explains how this is possible.

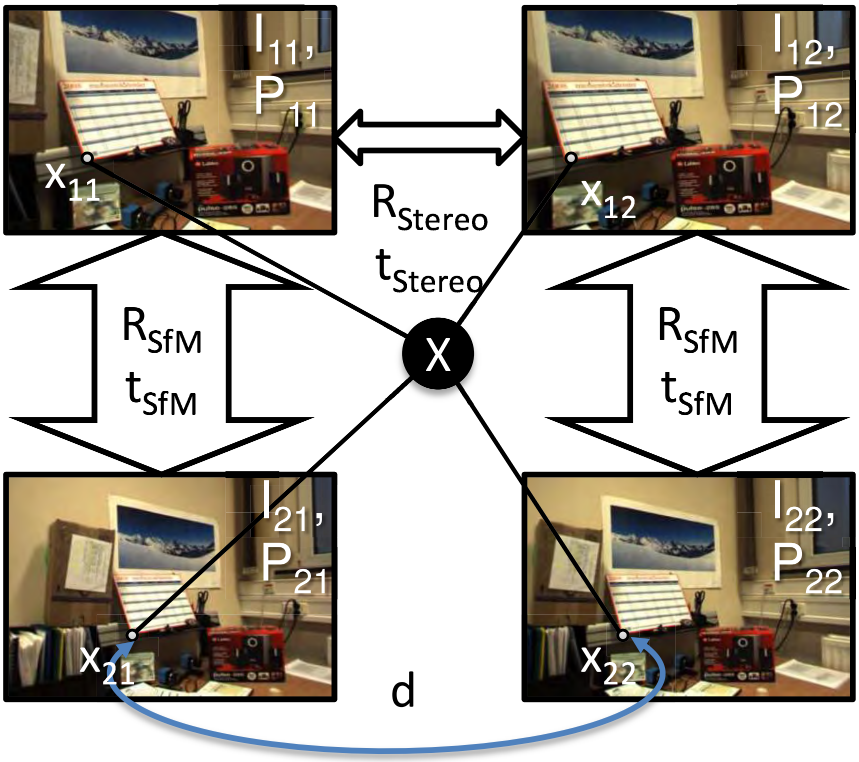


Figure D.1: Relations between 2 consecutive stereo frames: one 3D point X is visible in the 4 images with projections x_{11} and x_{12} in the first left/right stereo frame and projections x_{21} and x_{22} in the second stereo frame.

Consider 2 left and right stereo images I_{11} and I_{12}, shot at time t_0 by a calibrated stereo rig such that R_{Stereo} and t_{Stereo} are known. In both images, the projections x_{11} and x_{12} of the same 3D point X are visible. These points, together with a lot of other feature points, are matched by a classical dense stereo algorithm. At time $t_0 + k$, 2 new stereo images, I_{21} and I_{22}, are shot by the same stereo vision system. Structure from Motion is applied on the left images and right images. This leads to an estimation of the inter-frame camera motion, R_{SfM} and t_{SfM}, and camera projection matrices $P_{11}, P_{12}, P_{21}, P_{22}$ for the 4 cameras. When now performing the disparity search for the projection of the same 3D point X in the new image I_{21}, it is now possible to estimate the position of the projection of X in image I_{22}, x_{22}, by:

1. Projecting point x_{21} to image I_{11}: x_{11}
2. Projecting point x_{11} to image I_{12}: x_{12}
3. Projecting point x_{12} to image I_{22}: x_{22}

The projections of points x_{21} x_{22} can be written as:

$$x_{21} = P_{21} X_{21} \tag{D.2}$$

$$x_{22} = P_{22} X_{22} \tag{D.3}$$

Observing the stereo and SfM transformation equations,

$$X_{11} = \begin{pmatrix} R_{SfM} & t_{SfM} \\ 0 & 1 \end{pmatrix} X_{21} = T_{SfM} X_{21} \tag{D.4}$$

$$X_{11} = \begin{pmatrix} R_{Stereo} & t_{Stereo} \\ 0 & 1 \end{pmatrix} X_{12} = T_{Stereo} X_{12} \tag{D.5}$$

$$X_{12} = \begin{pmatrix} R_{SfM} & t_{SfM} \\ 0 & 1 \end{pmatrix} X_{22} = T_{SfM} X_{22} \tag{D.6}$$

we obtain x_{22}:

$$x_{22} = P_{22} T_{SfM}^{-1} T_{Stereo}^{-1} T_{SfM} P_{21}^{-1} x_{21} \tag{D.7}$$

The disparity can then be calculated directly from the pixel positions x_{21} and x_{22}:

$$d_{SfM} = \left(P_{22} T_{SfM}^{-1} T_{Stereo}^{-1} T_{SfM} P_{21}^{-1} - I \right) x_{21} \tag{D.8}$$

The disparity, estimated through SfM as presented above, is then used as an extra term for the cost function:

$$Cost_{Total} = Cost_{SAD/SSD} + \left(d_{SfM}^2 - d_{Stereo}^2 \right) \tag{D.9}$$

It is evident that the computational cost of the matching cost computation is proportional to *ImageWidth* × *ImageHeight* × *NumberOfDisparities*, which rises with the requested maximum disparity, which is a measure for the precision. An obvious advantage would be if we could constrain the disparity search over a smaller domain. Due to the fact that many stereo algorithms process whole scanlines at a time, this is however not straightforward to implement.

D.3 Evaluation Methodology

In this work, we used the stereo evaluation code developed by Scharstein and Szeliski to accompany their taxonomy paper [124]. This evaluation module enables to evaluate many stereo algorithms with a multitude of parameter settings. As it is not our goal to describe the stereo vision algorithms in detail, we refer the reader to [124] for a more in-depth explication of the stereo algorithm parameters.

We extended the existing evaluation module, such that SfM results are taken into account when evaluating the matching cost. Subsequently, we ran tests over a number of algorithms and parameter settings with and without using the SfM data to evaluate the effect of introducing SfM data in the matching cost function.

To evaluate the performance of the stereo (+SfM) algorithms, quantitative measures are needed to estimate the quality of the computed correspondences.

Two general approaches to this are to compute error statistics with respect to some ground truth data [8] and to evaluate the synthetic images obtained by warping the reference or unseen images by the computed disparity map [148]. In the current version of the software, the following two quality measures are computed based on known ground truth data:

- RMS (Root-Mean-Squared) error (measured in disparity units) between the computed disparity map $d_C(x,y)$ and the ground truth map $d_T(x,y)$:

$$R = \sqrt{\frac{1}{N}\sum_{(x,y)} |d_C(x,y) - d_T(x,y)|^2} \tag{D.10}$$

  where N is the total number of pixels.

- Percentage of bad matching pixels:

$$B = \frac{1}{N}\sum_{(x,y)} (|d_C(x,y) - d_T(x,y)| > \delta_d) \tag{D.11}$$

  where $\delta_d = 1.0$ is a disparity error tolerance.

One problem of evaluating the stereo + SfM performance is that most existing datasets or sequences available on the net focus on one of both applications, so there are not a lot of standard sequences for moving stereo vision. To remedy this, a custom motion-stereo sequence was recorded. This sequence is shown in Figure D.2 and was shot using a BumbleBee2 stereo vision system, developed by Point Grey. This stereo vision system consists of 2 1/3" progressive scan color CCD cameras. This stereo vision system is capable of outputting real-time high-accuracy depth range images, which are used as ground truth input.

In total, 166 different parameter settings for the stereo algorithms were compared. In order to summarize the results, 33 algorithms / parameter settings were selected from this list. These algorithms are listed in table D.1 and were selected on the basis that:

1. they give a good representation of all stereo algorithm options.
2. they provide good results using the classic stereo algorithm (without SfM).

The reason for the second constraint is that adding SfM data is no magical method for making totally erroneous stereo measurements become better again. Tests have shown that when the stereo algorithm fails (e.g. due to erroneous parameter settings), adding SfM data yields no improvement to the situation. This is to be expected, as one should not forget that the proposed method still uses stereo matching as base information.

Table D.1 explains all chosen algorithms and parameter settings as used by the evaluation module presented in [124].

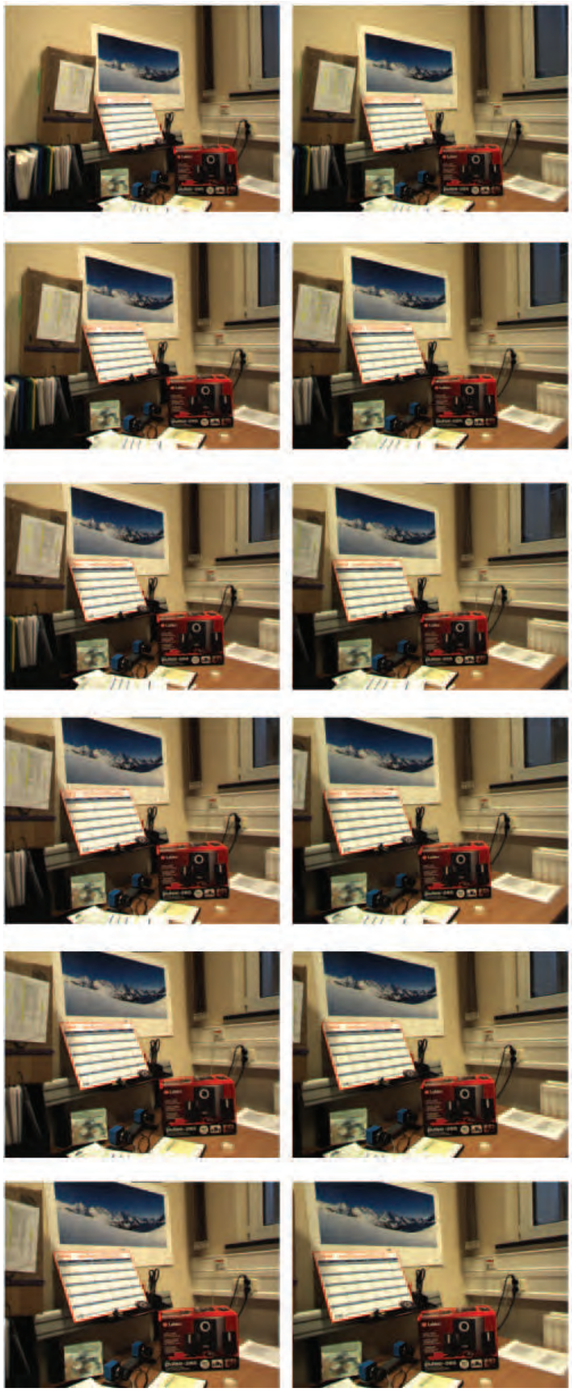


Figure D.2: Some images of a Motion-Stereo sequence.

Table D.1: List of selected algorithms. SAD: Sum of Absolute Differences; SSD: Sum of Squared Differences; BT: Birchfield-Tomasi; DP=Dynamic Programming; SO: Scanline Optimization;

Algorithm	Matching Cost	Aggregation	Optimization	Parameters	Time (s)
DPm1o20s0020	SAD	-	DP	*occlusion cost = 20, smoothness* = 20	10.937
DPm1o20s0020b	SAD + BT	-	DP	*occlusion cost = 20, smoothness* = 20	12.046
DPm1o20s0020bt20	SAD + BT	-	DP	*occlusion cost=20, smoothness*=20, *truncation*=20	11.875
DPm1o20s0020t20	SAD	-	DP	*occlusion cost=20, smoothness*=20, *truncation*=20	10.906
DPm1o20s0200	SAD	-	DP	*occlusion cost = 20, smoothness* = 200	10.922
DPm1o50s0020	SAD	-	DP	*occlusion cost = 50, smoothness* = 20	10.891
DPm1o80s0020	SAD	-	DP	*occlusion cost = 80, smoothness* = 20	10.891
SAD09	SAD	Rect. Win.	-	*window size = 9*	9.359
SAD09b	SAD + BT	Rect. Win.	-	*window size = 9*	10.407
SAD09bt50	SAD + BT	Rect. Win.	-	*window size = 9, truncation* = 50	10.438
SAD09t50	SAD	Rect. Win.	-	*window size = 9, truncation* = 50	9.343
SAD11	SAD	Rect. Win.	-	*window size = 11*	9.453
SADbf20	SAD	Bin. Filter	-	*number of iterations = 20*	283.015
SADdiff030	SAD	Diff. Filter	-	*number of iterations = 30*, $\lambda_{diffusion} = 0.15$	14.890
SADmemb01	SAD	Memb. Filter	-	$\beta_{diffusion} = 0.1$, $\lambda_{diffusion} = 0.15$	54.031
SADmf09	SAD	Min Filter	-	*window size = 9, min filter = 9*	10.422
SADmf09b	SAD + BT	Min Filter	-	*window size = 9, min filter = 9*	11.453
SADmf09bt50	SAD + BT	Min Filter	-	*window size = 9, min filter = 9, truncation* = 50	11.562
SADmf09t50	SAD	Min Filter	-	*window size = 9, min filter = 9, truncation* = 50	10.453
SADmf11	SAD	Min Filter	-	*window size = 11, min filter = 11*	10.547
SOm1s0020	SAD	-	SO	*smoothness* = 20	18.656
SOm1s0020b	SAD + BT	-	SO	*smoothness* = 20	19.735
SOm1s0020bt50	SAD + BT	-	SO	*smoothness* = 20, *truncation* = 50	19.468
SOm1s0020t50	SAD	-	SO	*smoothness* = 20, *truncation* = 50	18.344
SOm1s1000	SAD	-	SO	*smoothness* = 1000	18.438
SSD09	SSD	Rect. Win.	-	*window size = 9*	9.453
SSD09b	SSD + BT	Rect. Win.	-	*window size = 9*	10.281
SSD09bt50	SSD + BT	Rect. Win.	-	*window size = 9, truncation* = 50	10.266
SSD09t50	SSD	Rect. Win.	-	*window size = 9, truncation* = 50	9.265
SSD09mf09	SSD	Min Filter	-	*window size = 9, min filter = 9*	10.188
SSD09mf09b	SSD + BT	Min Filter	-	*window size = 9, min filter = 9*	11.359
SSD09mf09bt50	SSD + BT	Min Filter	-	*window size = 9, min filter = 9, truncation* = 50	11.359
SSD09mf09t50	SSD	Min Filter	-	*window size = 9, min filter = 9, truncation* = 50	10.188

D.4 Results

Figure D.13 shows for all frames of the sequence the depth map obtained by stereo and stereo + SfM. As can be noted, the differences between both depth maps are in general minor and visually not clearly distinguishable. Therefore, it is necessary to analyze the results using both methodologies quantitatively.

For this series of tests, 4 quantitative measures are taken into consideration and evaluated for each of the 166 base stereo algorithms / parameter settings. These are:

1. The percentage of bad matching pixels according to equation D.11
2. The RMS (Root-Mean-Squared) error (measured in disparity units) between the computed disparity map $d_C(x, y)$ and the ground truth map $d_T(x, y)$ according to equation D.10
3. The Final Energy left in the disparity image according to equation D.1
4. Total Execution Time in seconds

The reference image for this image sequence is frame 12, which corresponds to the top row image of Figure D.13. Figures D.3 through D.8 show the sorted quantitative results for different frames of the sequence. These figures do not allow a per-algorithm evaluation as there are too many considered algorithms, they are meant to show the general usefulness of the proposed stereo+SfM method. As it is impossible to describe all these stereo algorithms on the graphs, they were numbered from 1 to 166, which explains the caption for the x-axis of Figures D.3 to D.8.

When comparing the bad pixel ration (top left graph) on image D.3, it is clear that the proposed stereo + SfM approach (indicated by a dash-dotted blue line) consistently delivers significantly better results than using stereo only (continuous black line). The same holds true for the RMS error (top right graph), although the difference is less clear due to the scaling. For the final energy present in the disparity image after optimization (bottom left graph), adding SfM information does not seem to yield any significant improvement. Of course, adding an additional matching cost for taking SfM data into account requires more processing time. As indicated by the bottom right graph of image D.3, incorporating SfM data adds about 3 seconds to the total stereo+SfM processing time. Note that the goal of this study was to perform a comparative evaluation and we were therefore more interested in relative timing than in achieving the fastest possible implementation. Therefore, all algorithms were run in *debug* mode without optimizations. Most stereo algorithms implemented by Scharstein and Szeliski in their evaluation module run roughly 10 times faster when built in *release* mode and the more simple algorithms (e.g. SAD or SSD with a rectangular window) achieve real-time performance at over 20 frames per second using dedicated hardware (FPGA, GPU, ...). The added time penalty for using SfM data is not that high, so if the SfM processing scales in the same way towards optimization as stereo processing, the final framerate should still be reasonable, even using SfM data.

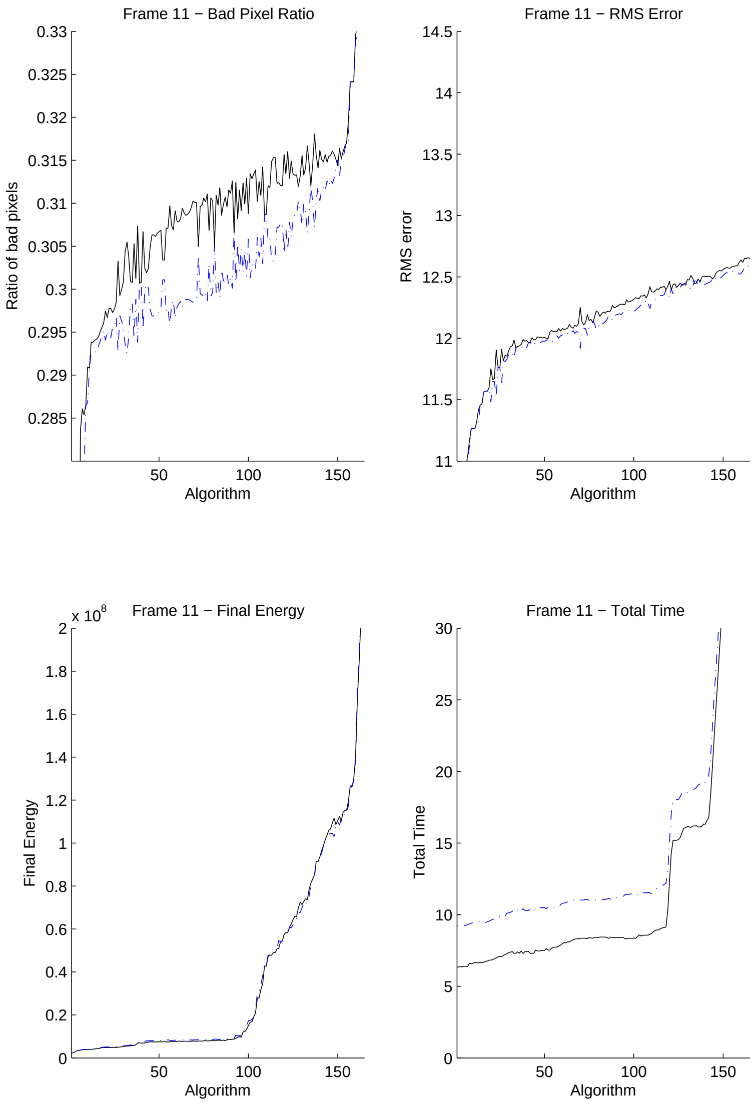


Figure D.3: Pure Stereo and Stereo + SfM results compared for 166 different base stereo algorithms / parameter settings. Results using stereo only are indicated by a continuous black line; results using stereo + sparse SfM are indicated by a dash-dotted blue line. Top Left: Bad pixel ratio according to equation D.11; Top Right: RMS error according to equation D.10; Bottom Left: Final Energy left in the disparity image according to equation D.1; Bottom Right: Total Execution Time in seconds for each algorithm. These are the results for frame 11 from the sequence shown in Figure D.2, where the reference image is image 12.

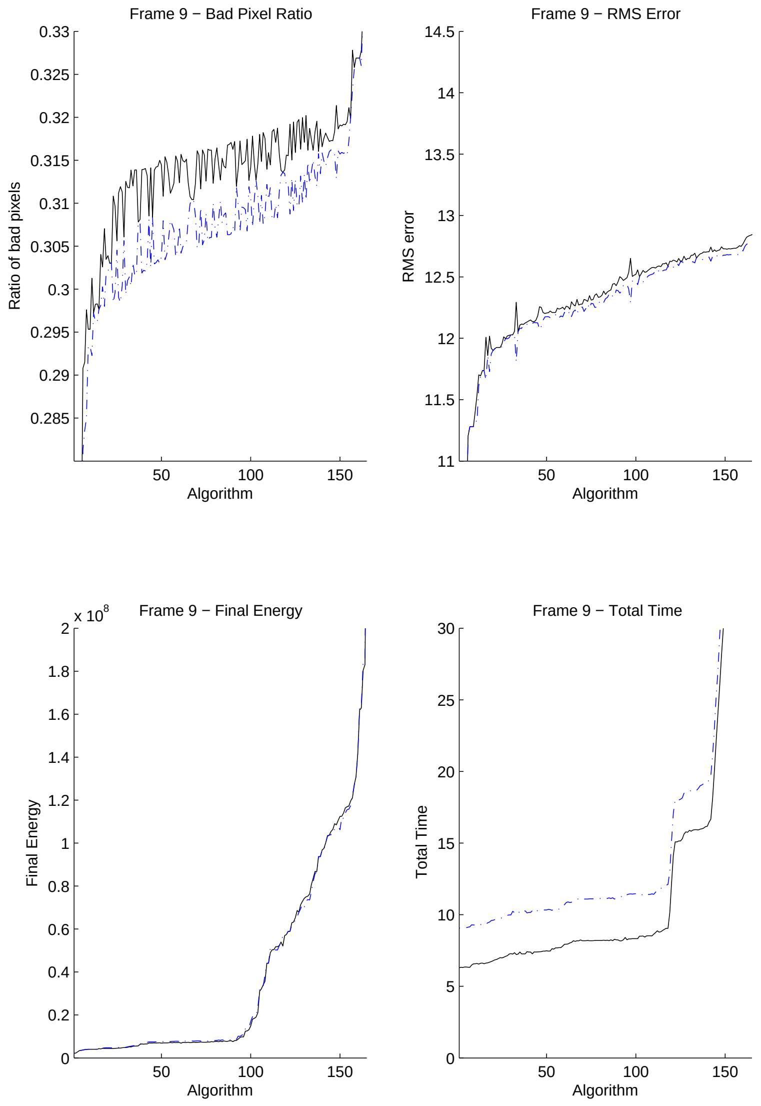


Figure D.4: Pure Stereo and Stereo + SfM results compared for 166 different base stereo algorithms / parameter settings. Results using stereo only are indicated by a continuous black line; results using stereo + sparse SfM are indicated by a dash-dotted blue line. Top Left: Bad pixel ratio according to equation D.11; Top Right: RMS error according to equation D.10; Bottom Left: Final Energy left in the disparity image according to equation D.1; Bottom Right: Total Execution Time in seconds for each algorithm. These are the results for frame 9 from the sequence shown in Figure D.2, where the reference image is image 12.

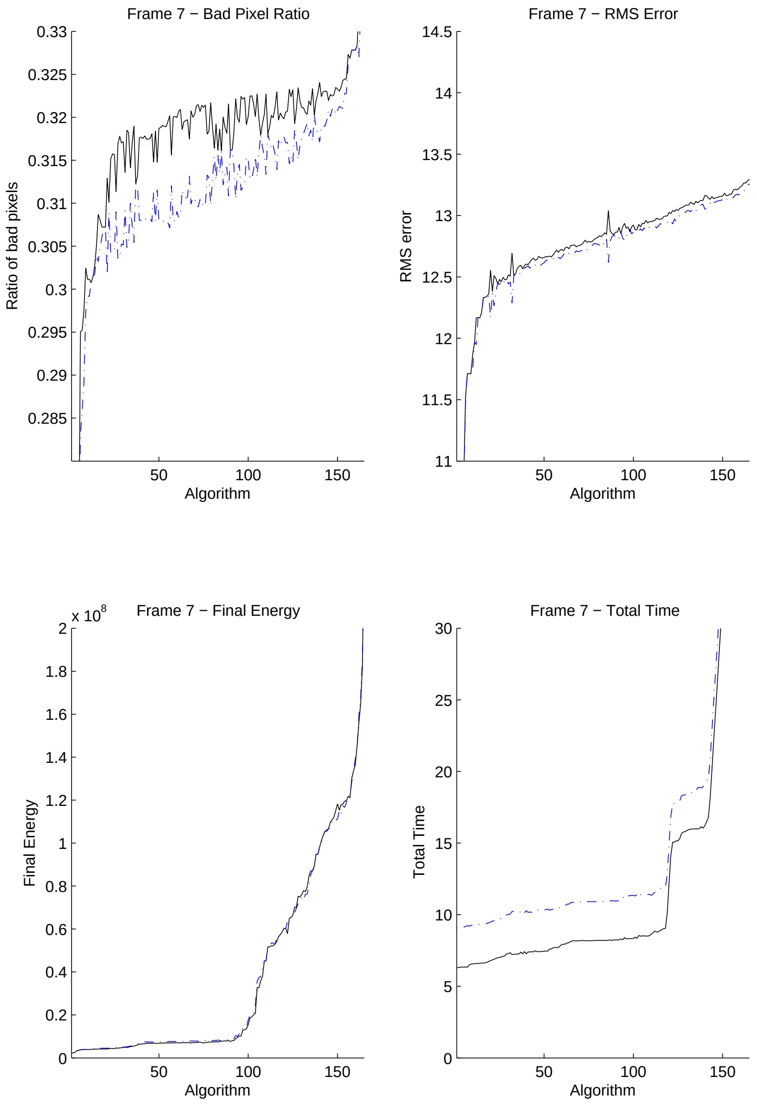


Figure D.5: Pure Stereo and Stereo + SfM results compared for 166 different base stereo algorithms / parameter settings. Results using stereo only are indicated by a continuous black line; results using stereo + sparse SfM are indicated by a dash-dotted blue line. Top Left: Bad pixel ratio according to equation D.11; Top Right: RMS error according to equation D.10; Bottom Left: Final Energy left in the disparity image according to equation D.1; Bottom Right: Total Execution Time in seconds for each algorithm. These are the results for frame 7 from the sequence shown in Figure D.2, where the reference image is image 12.

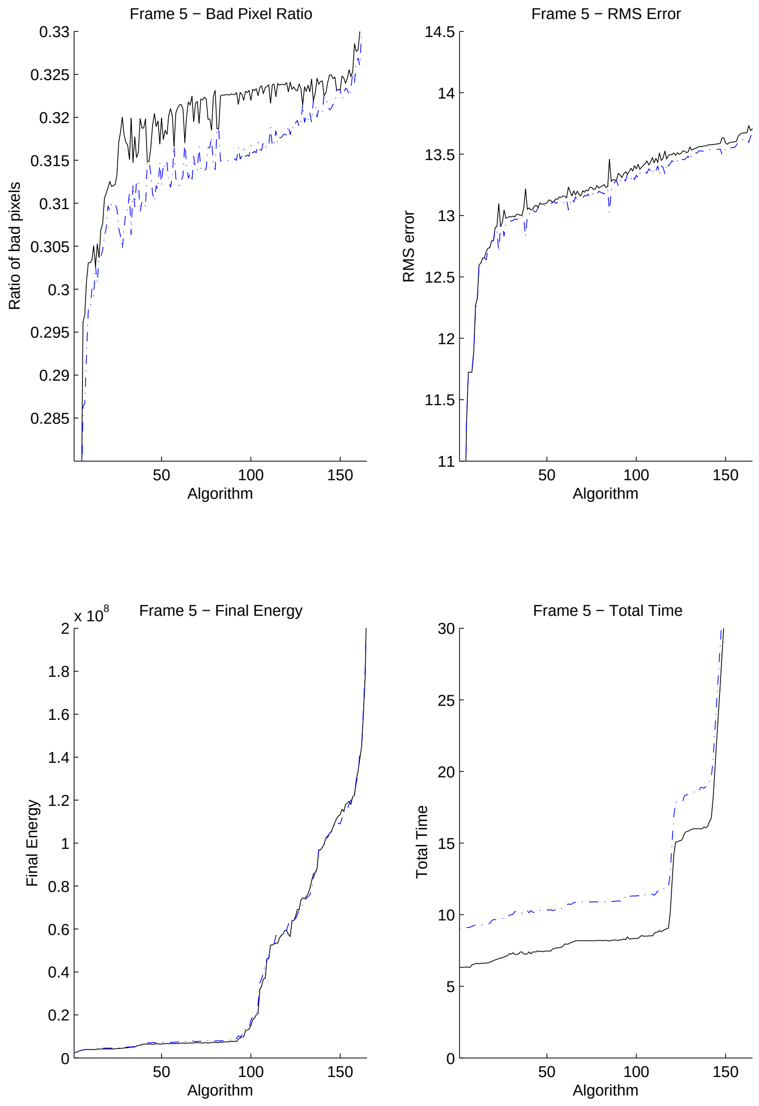


Figure D.6: Pure Stereo and Stereo + SfM results compared for 166 different base stereo algorithms / parameter settings. Results using stereo only are indicated by a continuous black line; results using stereo + sparse SfM are indicated by a dash-dotted blue line. Top Left: Bad pixel ratio according to equation D.11; Top Right: RMS error according to equation D.10; Bottom Left Final Energy left in the disparity image according to equation D.1; Bottom Right: Total Execution Time in seconds for each algorithm. These are the results for frame 5 from the sequence shown in Figure D.2, where the reference image is image 12.

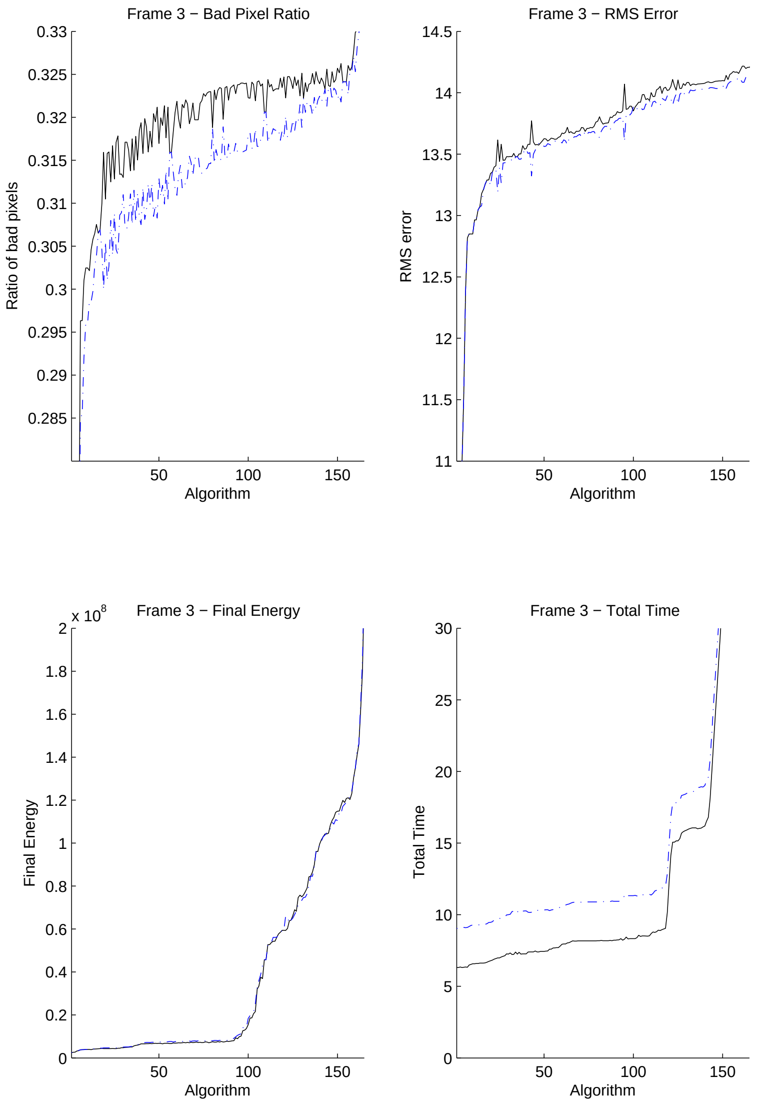


Figure D.7: Pure Stereo and Stereo + SfM results compared for 166 different base stereo algorithms / parameter settings. Results using stereo only are indicated by a continuous black line; results using stereo + sparse SfM are indicated by a dash-dotted blue line. Top Left: Bad pixel ratio according to equation D.11; Top Right: RMS error according to equation D.10; Bottom Left: Final Energy left in the disparity image according to equation D.1; Bottom Right: Total Execution Time in seconds for each algorithm. These are the results for frame 3 from the sequence shown in Figure D.2, where the reference image is image 12.

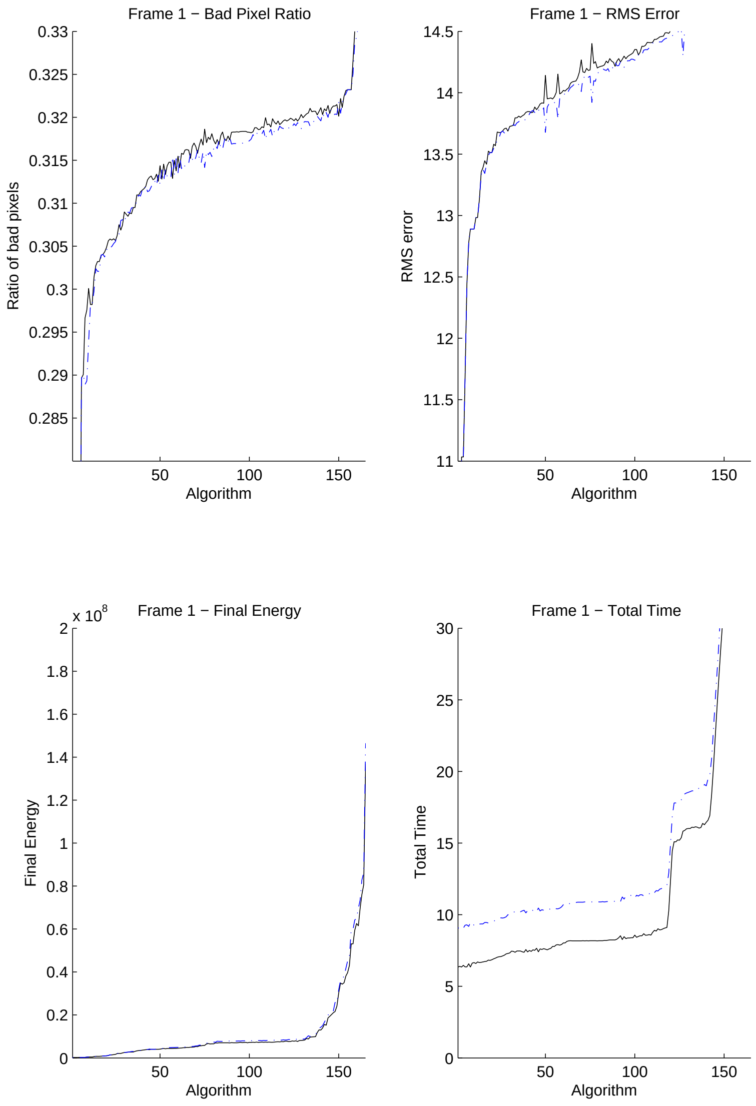


Figure D.8: Pure Stereo and Stereo + SfM results compared for 166 different base stereo algorithms / parameter settings. Results using stereo only are indicated by a continuous black line; results using stereo + sparse SfM are indicated by a dash-dotted blue line. Top Left: Bad pixel ratio according to equation D.11; Top Right: RMS error according to equation D.10; Bottom Left: Final Energy left in the disparity image according to equation D.1; Bottom Right: Total Execution Time in seconds for each algorithm. These are the results for frame 1 from the sequence shown in Figure D.2, where the reference image is image 12.

When comparing figures D.3 through D.8, it becomes clear that the advantage of using SfM data reduces when the difference between the reference image (image 12) and the observed image becomes large. In this case, SfM is no longer able to accurately estimate the camera motion and using the resulting SfM data yields few benefits over normal stereo. This is the situation on Figure D.8, where the difference between black (stereo) and blue (stereo + SfM) curves is negligible (except for the timing of course). To the same extent, the advantage of using SfM data reduces when the difference between the reference image and the observed image is too small. There thus exists an optimal inter-frame gap or framerate which can be estimated by calculating the GRIC criterion using equation 3.1.

Figure D.9 shows the sorted bad pixel ratio averaged over the 11 frames of the sequence shown in Figure D.2, according to equation D.11, for the selection of 33 different base stereo algorithms / parameter settings with and without using sparse SfM data. From this graph, we can note that optimization methods like Dynamic Programming and Scanline Optimization are certainly valuable, as their results can generally be found to the right, which means the bad pixel ratio is low. However, the benefit of using SfM data in conjunction with an optimization step like Dynamic Programming or Scanline Optimization is also lower: The blue and black lines are more separated for *“simple”* algorithms like SAD window than for algorithms using Dynamic Programming or Scanline Optimization. It thus seems that adding SfM data into the stereo matching process is mostly useful for *“simple”* stereo algorithms.

Figure D.10 shows the sorted RMS error averaged over the 11 frames of the sequence shown in Figure D.2, according to equation D.10 for a selection of 33 different base stereo algorithms / parameter settings with and without using sparse SfM data. Here, the result is a bit more diverse, but still, it can be noted that optimization-based methods like Dynamic Programming and Scanline Optimization can generally be found to the right, which means their RMS error is low and that simpler methods can be found to the left. Concerning the RMS error, the benefit of using SfM data is about equal among all algorithms.

Figure D.11 shows the sorted final energy averaged over the 11 frames of the sequence shown in Figure D.2, according to equation D.1 for a selection of 33 different base stereo algorithms / parameter settings with and without using sparse SfM data. It is clear that using SfM data yields no better results as far as the final energy present in the disparity image is concerned. This also wasn’t expected as adding the SfM information changes nothing to the energy minimization process itself, only the input differs.

Figure D.12 shows the sorted total execution time averaged over the 11 frames of the sequence shown in Figure D.2, for a selection of 33 different base stereo algorithms / parameter settings with and without using sparse SfM data. The added time penalty for incorporating SfM data is about 3 seconds, regardless of the algorithm. These time measurements were taken without any optimization and in debug mode, only to compare the relative processing time of stereo only versus stereo + SfM. As can be observed, the (sparse) SfM calculation needs less time than the (dense) stereo calculation, but joined together, the extra SfM calculation adds a significant time penalty to the total processing time.

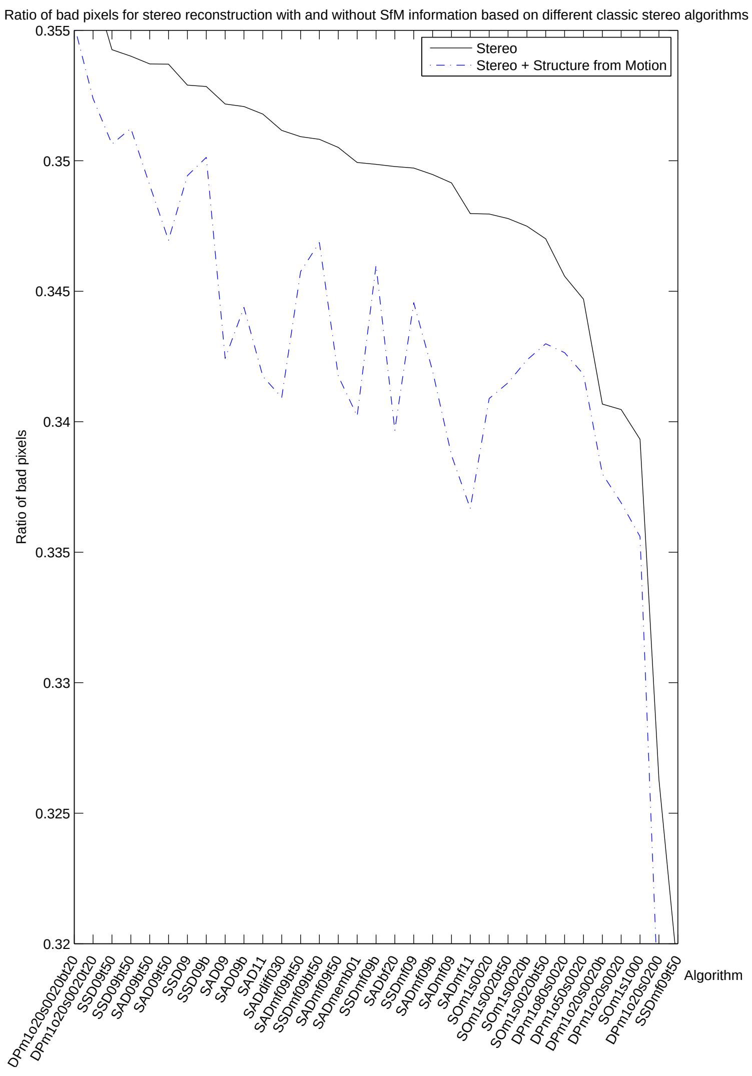


Figure D.9: Bad pixel ratio according to equation D.11 for a selection of 33 different base stereo algorithms / parameter settings with and without using sparse SfM data. Results obtained using stereo only are indicated by a continuous black line; results obtained using stereo + sparse SfM are indicated by a dash-dotted blue line. These are the averaged results over the 11 frames from the sequence shown in Figure D.2.

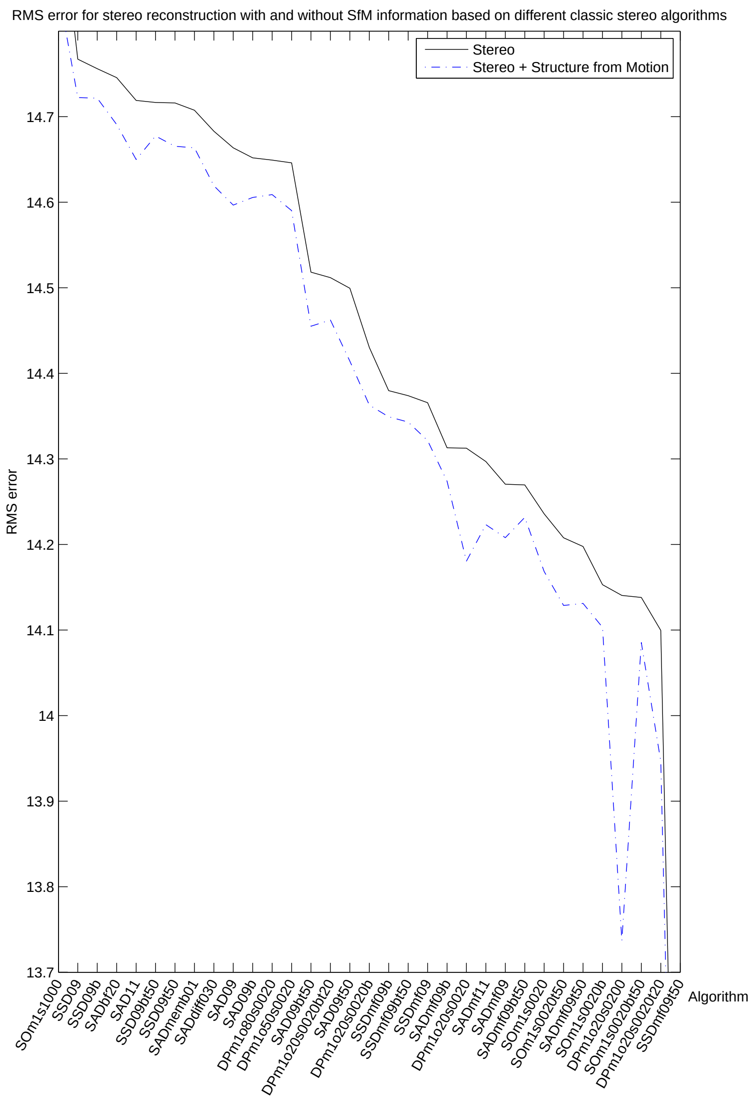


Figure D.10: RMS error according to equation D.10 for a selection of 33 different base stereo algorithms / parameter settings with and without using sparse SfM data. Results obtained using stereo only are indicated by a continuous black line; results obtained using stereo + sparse SfM are indicated by a dash-dotted blue line. These are the averaged results over the 11 frames from the sequence shown in Figure D.2.

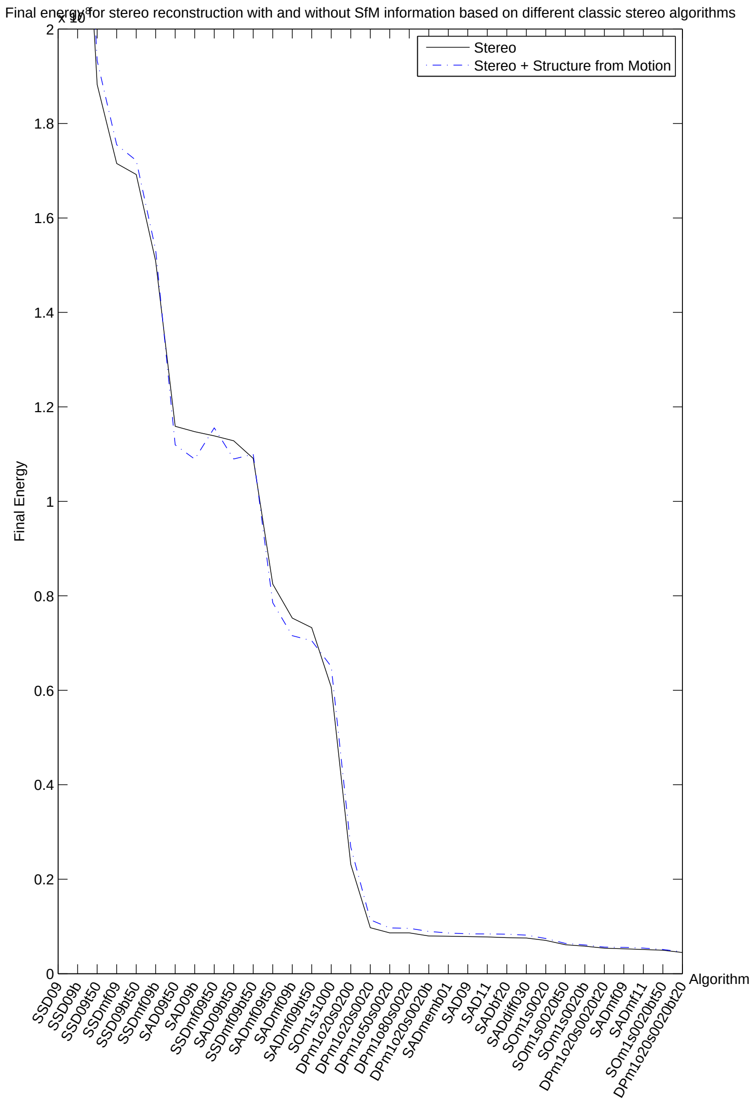


Figure D.11: Final energy according to equation D.1 for a selection of 33 different base stereo algorithms / parameter settings with and without using sparse SfM data. Results obtained using stereo only are indicated by a continuous black line; results obtained using stereo + sparse SfM are indicated by a dash-dotted blue line. These are the averaged results over the 11 frames from the sequence shown in Figure D.2.

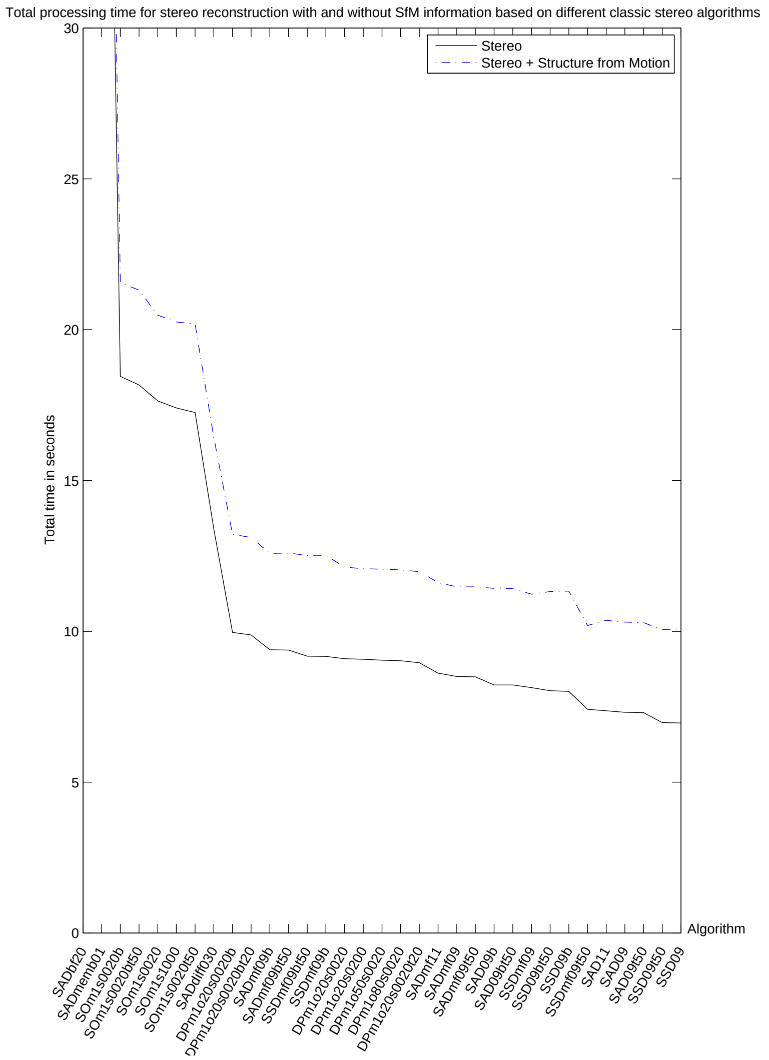


Figure D.12: Total execution time for a selection of 33 different base stereo algorithms / parameter settings with and without using sparse SfM data. Results obtained using stereo only are indicated by a continuous black line; results obtained using stereo + sparse SfM are indicated by a dash-dotted blue line. These are the averaged results over the 11 frames from the sequence shown in Figure D.2.

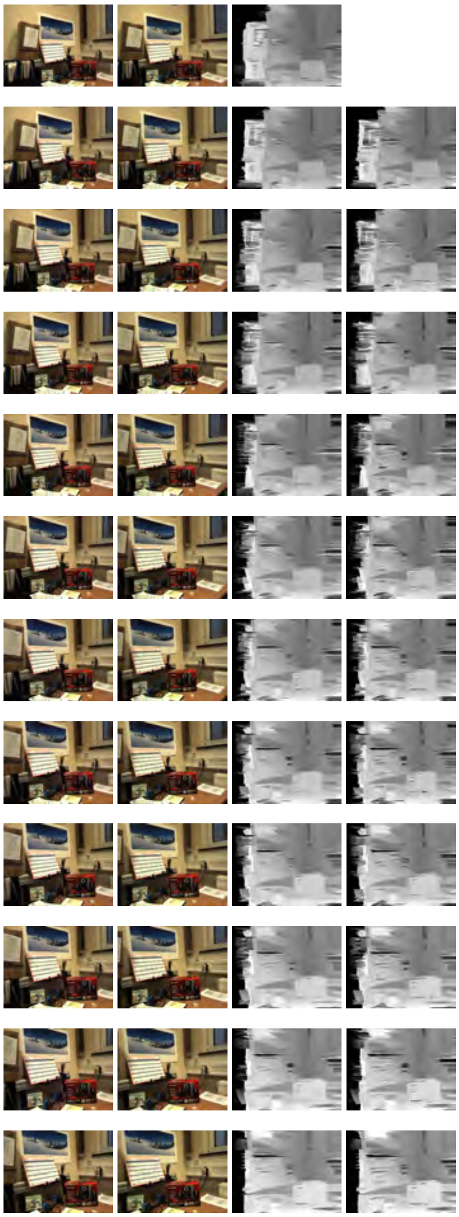


Figure D.13: Evaluation of stereo results over a sequence of 12 images showing from left to right: the left input image, the right input image, the disparity map as calculated using stereo alone and finally the disparity map as calculated by the proposed SfM aided stereo algorithm. Note that for the first (reference) frame, no SfM data can be obtained, so in this case there is no result. The underlying stereo algorithm used for this evaluation is the Dynamic Programming approach with SAD matching (parameters: *occlusion cost* = *20*, *smoothness* = 20).

Bibliography

[1] L. Alvarez, C. Cuenca, A. Salgado, and J. Sanchez. A variational approach to 3d cylindrical geometry reconstruction from multiple views. *ELCVIA*, 6(2):54–66, September 2007.

[2] L. Alvarez, R. Deriche, J. Weickert, and J. Sanchez. Dense disparity map estimation respecting image discontinuities: A PDE and scale-space based approach. In *IAPR International Workshop on Machine Vision Applications*, 2000.

[3] L. Alvarez, R. Deriche, J. Weickert, and J. Sánchez. Dense disparity map estimation respecting image discontinuities: A PDE and scale-space based approach. *Journal of Visual Communication and Image Representation*, 13:3–21, 2002.

[4] Luis Alvarez, Rachid Deriche, Tho Papadopoulo, and Javier Snchez. Symmetrical dense optical flow estimation with occlusions detection. *International Journal of Computer Vision*, 75(3):371–385, December 2007.

[5] G. Aubert, R. Deriche, and P. Kornprobst. Computing optical flow via variational techniques. *SIAM J. Math. Anal.*, 60(1):156–182, 1999.

[6] S. Ayer and H. S. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, page 777, Washington, DC, USA, 1995. IEEE Computer Society.

[7] Simon Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *Proceedings of the 1998 IEEE Conference on Computer Vision and Pattern Recognition*, pages 434 – 441, June 1998.

[8] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12:43–77, 1994.

[9] J. L. Barron, D. J. Fleet, S. S. Beauchemin, and T. A. Burkitt. Performance of optical flow techniques. Technical Report TR-299, Dept. of Computer Science, University of Western Ontario, Ontario, N6A 5B7, Canada, 1993.

[10] J. Beis and D. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, pages 1000–1006, 1997.

[11] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24):509 – 522, April 2004.

[12] S.A. Berrabah, G. De Cubber, V. Enescu, and H. Sahli. Mrf-based foreground detection in image sequences from a moving camera. In *International Conference on Image Processing (ICIP 2006)*, pages 1125–1128, Atlanta, USA, 2006.

[13] S. Birchfield and C. Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *ICCV*, page 489495, 1999.

[14] A. Blake and H. Blthoff. Does the brain know the physics of specular reflection? *Nature*, 343:165–168, 1990.

[15] A. F. Bobick and S. S. Intille. Large occlusion stereo. *IJCV*, 33(3):181200, 1999.

[16] Richard. P Brent. *Algorithms for Minimization without Derivatives.* Prentice-Hall, Englewood Cliffs, New Jersey, 1973.

[17] M. Brown and D. Lowe. Recognising panoramas. In *Proceedings of the 9th International Conference on Computer Vision*, volume 2, pages 1218–1225, Nice, France, October 2003.

[18] N. Bruno and J.E. Cutting. Minimodularity and the perception of layout. *Journal of Experimental Psychology*, 117(2):161–70, 1988.

[19] Anna R Bruss and Berthold K. P Horn. Passive navigation. *Computer Vision, Graphics and Image Processing*, 21:3–20, January 1983.

[20] Heinrich H. Bulthoff and Hanspeter A. Mallot. Integration of depth modules: stereo and shading. *J. Opt. Soc. Am. A*, 5(10):1749, 1988.

[21] Michelangelo Buonarroti. Doni tondo. Galleria degli Uffizi, 1503. Oil and tempera on panel.

[22] Johannes Burge, Mary A. Peterson, and Stephen E. Palmer. Ordinal configural cues combine with metric disparity in depth perception. *J. Vis.*, 5(6):534–542, 6 2005.

[23] Rodrigo L. Carceroni and Kiriakos N. Kutulakos. Multi-view scene capture by surfel sampling: From video streams to non-rigid 3d motion, shape and reflectance. *Int. J. Comput. Vision*, 49(2-3):175–214, 2002.

[24] Gustavo Carneiro and Allan D. Jepson. Flexible spatial models for grouping local image features. *Computer Vision and Pattern Recognition*, 2:747–754, 2004.

[25] T.F. Coleman and Y. Li. On the convergence of reflective newton methods for large-scale nonlinear minimization subject to bounds. *Mathematical Programming*, 67(2):189–224, 1994.

[26] T.F. Coleman and Y. Li. An interior, trust region approach for nonlinear minimization subject to bounds. *SIAM Journal on Optimization*, 6:418445, 1996.

[27] T.F. Coleman and Y. Li. A reflective newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization*, 6(4):1040–1058, 1996.

[28] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust Region Methods.* MPS-SIAM Series on Optimization 1. MPS-SIAM, 2000.

[29] N.D. Cook, A. Yutsudo, N. Fujimoto, and M. Murata. On the visual cues contributing to pictorial depth perception. *EMPIRICAL STUDIES OF THE ARTS*, 26(1):69–92, 2008.

[30] J.E. Cutting and P.M. Vishton. *Handbook of perception and cognition*, chapter Perceiving layout and knowing distances: The integration, relative potency, and contextual use of different information about depth, pages 69–117. Academic Press, 1995.

[31] G. de Haan and P.W.A.C. Biezen. Sub-pixel motion estimation with 3-d recursive search block-matching. *Signal Processing: Image Communication*, 6:229–239, 1994.

[32] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[33] Fulvio Domini, Corrado Caudek, and Peter Skirko. Temporal integration of motion and stereo cues to depth. *Perception and Psychophysics*, 65(1):48–57, Jan 2003.

[34] B. A. Dosher, G. Sperling, and S. Wurst. Tradeoffs between stereopsis and proximity luminance covariance as determinants of perceived 3d structure. *Vision Research*, 26:973–990, 1986.

[35] Fabian Ernst, Piotr Wilinski, and Cornelius W. A. M. van Overveld. Dense structure-from-motion: An approach based on segment matching. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part II*, pages 217–231, London, UK, 2002. Springer-Verlag.

[36] O. Faugeras and R. Keriven. Variational principles, surface evolution, PDE's, level set methods and the stereo problem. Technical Report 3021, INRIA, France, October 1996.

[37] O. Faugeras and T. Papadopoulo. A nonlinear method for estimating the projective geometry of 3 views. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, page 477, Washington, DC, USA, 1998. IEEE Computer Society.

[38] Olivier D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 563–578, London, UK, 1992. Springer-Verlag.

[39] Olivier D. Faugeras and Renaud Keriven. Complete dense stereovision using level set methods. In *ECCV 98: Proceedings of the 5th European Conference on Computer Vision*, volume 1, pages 379–393, London, UK, 1998. Springer-Verlag.

[40] Andrew W. Fitzgibbon and Andrew Zisserman. Automatic camera recovery for closed or open image sequences. In *5th European Conference on Computer Vision*, volume 1, pages 311–326, Freiburg, Germany, June 1998.

[41] G. E. Forsythe, M. A. Malcolm, and C. B. Moler. *Computer Methods for Mathematical Computations,*. Prentice-Hall, 1976.

[42] Friedrich Fraundorfer and Horst Bischof. A novel performance evaluation method of local detectors on non-planar scenes. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, volume 3, pages 33–33. IEEE Computer Society, June 2005.

[43] Pascal Fua. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In *In 12th. International Joint Conference on Artificial Intelligence*, pages 1292–1298, 1991.

[44] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *CVPR*, 1:1–8, 2007.

[45] V. Gouet, P. Montesinos, R. Deriche, and D. Pele. Evaluation de detecteurs de points d'interet pour la couleur. *Reconnaissance des formes et Intelligence Artificielle*, 2:257–266, 2000.

[46] Michael Grabner, Helmut Grabner, and Horst Bischof. Fast approximated sift. In *ACCV 2006: Asian Conference on Computer Vision*, pages 918–927. Springer-Verlag, 2006.

[47] K. J. Hanna. Direct multi-resolution estimation of ego-motion and structure from motion. In *Workshop on Visual Motion*, pages 156–162, October 1991.

[48] C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[49] R. Hartley. Lines and points in three views: A unified approach. In *ARPA94*, pages II:1009–1016, 1994.

[50] R. I. Hartley and P. Sturm. Triangulation. In *In American Image Understanding Workshop*, pages 957–966, 1994.

[51] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, New York, NY, USA, second edition, 2004.

[52] Richard I. Hartley. Estimation of relative camera positions for uncalibrated cameras. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 579–587, London, UK, 1992. Springer-Verlag.

[53] Richard I. Hartley. Computation of the quadrifocal tensor. In *ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume I*, pages 20–35, London, UK, 1998. Springer-Verlag.

[54] D. J. Heeger. Optical flow using spatiotemporal filters. *International Journal of Computer Vision*, 1:279–302, 1988.

[55] David J. Heeger and Allan D. Jepson. Subspace methods for recovering rigid motion i: Algorithm and implementation. *International Journal of Computer Vision*, 7(2):95–117, January 1992.

[56] Joachim Heel. Direct estimation of structure and motion from multiple frames. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1990.

[57] M. R. Hestenes. Multipler and gradient methods. *Journal of Optimization Theory and Applications*, 4:303–320, 1969.

[58] A. Heyden. Reconstruction from image sequences by means of relative depths. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, page 1058, Washington, DC, USA, 1995. IEEE Computer Society.

[59] J. Hillis, S. Watt, M. Landy, and M. Banks. Slant from texture and disparity cues: Optimal cue combination. *Journal of Vision*, 4:967–992, 2004.

[60] J. M. Hillis, M. O. Ernst, M. S. Banks, and M. S. Landy. Combining sensory information: Mandatory fusion within, but not between, senses. *Science*, 298(5598):1627 – 1630, November 2002.

[61] Horn. *Robot Vision.* MIT Press, Cambridge, Ma, 1979.

[62] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[63] Berthold K. P. Horn and E. J. Weldon. Direct methods for recovering motion. *International Journal of Computer Vision*, 1:51–76, 1988.

[64] Harry C. Andrews Hsieh S. Hou. Cubic splines for image interpolation and digital filtering. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(6):508–517, December 1978.

[65] R.A. Jacobs and I. Fine. Experience-dependent integration of texture and motion cues to depth. *Vision Research*, 39:4062–4075, 1999.

[66] M. Jancosek and T. Pajdla. Segmentation based multi-view stereo. In *CVWW*, 2009.

[67] Xiaoyi Jiang and Hanspeter Bieri. *Integrated Image and Graphics Technologies*, volume 762 of *The International Series in Engineering and Computer Science*, chapter 3D Imaging and Applications, pages 331–349. Springer Netherlands, 2004.

[68] E. B. Johnston, B. G. Cumming, and M. S. Landy. Integration of stereopsis and motion shape cues. *Vision Res.*, 34(17):2259–2275, 1994.

[69] E.B. Johnston, B.G Cumming, and A.J. Parker. Integration of depth modules: Stereopsis and texture. *Vision Research*, 33(5/6):813–826, 1993.

[70] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004.

[71] Robert G. Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(6):1153–1160, December 1981.

[72] D.C. Knill. Ideal observer perturbation analysis reveals human strategies for inferring surface orientation from texture. *Vision Research*, 38(17):2635–2656, 1998.

[73] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *ECCV*, volume 3, pages 82–96, 2002.

[74] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part III*, pages 65–81, London, UK, 2002. Springer-Verlag.

[75] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. *Int. J. Comput. Vision*, 38(3):199–218, 2000.

[76] M. Landy, L. Maloney, E. Johnsten, and M. Young. Measurement and modeling of depth cue combinations: in defense of weak fusion. *Vision Research*, 35:389–412, 1995.

[77] Sang-Beom Lee, Kwan-Jung Oh, and Yo-Sung Ho. Segment-based multi-view depth map estimation using belief popagation from dense multi-view video. In *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, pages 193–196, Istanbul, May 2008.

[78] L. Li and J.H. Duncan. 3-d translational motion and structure from binocular image flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(7):657–667, 1993.

[79] Ping Li, Dirk Farin, Rene K. Gunnewiek, and Peter H. N. de With. On creating depth maps from monoscopic video using structure from motion. In *Proc. IEEE Workshop on Content Generation and Coding for 3D-television*, 2006.

[80] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):77–116, 1998.

[81] Tony Lindeberg. Scale-space theory: A basic tool for analysing structures at different scales. *J. of Applied Statistics*, 21(2):224–270, 1994.

[82] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(5828):133135, September 1981.

[83] M.I.A. Lourakis and A.A. Argyros. The design and implementation of a generic sparse bundle adjustment software package based on the levenberg-marquardt algorithm. Website: http://www.ics.forth.gr/ lourakis/sba/ 340, Institute of Computer Science - FORTH, Heraklion, Crete, Greece, August 2004.

[84] David G. Lowe. Object recognition from local scale-invariant features. *Proceedings of the International Conference on Computer Vision-*, 2:1150–1157, 1999.

[85] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):pp. 91–110, 2004.

[86] Ye Lu, J.Z. Zhang, Q.M.J. Wu, and Ze-Nian Li. A survey of motion-parallax-based 3-d reconstruction algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 34(4):532–548, November 2004.

[87] B. Lucas and T. Kanade. An iterative image registration technique with applications in stereo vision. In *Proceeding of the DARPA Image Understanding Workshop*, pages 121–130, 1981.

[88] Wallace James MacLean. *Recovery of Egomotion and Segmentation of Independent Object Motion Using the EM-Algorithm.* Doctor of philosophy, University of Toronto, 1996.

[89] L.T. Maloney and M.S. Landy. A statistical framework for robust fusion of depth information. *Proceedings of the SPIE: Visual Communications and Image Processing*, Part 2:1154–1163, 1989.

[90] David Marr. *Vision: a computational investigation into the human representation and processing of visual information.* W.H. Freeman, San Francisco, 1982.

[91] F. X. Martinez, Jenny Benois-Pineau, and Dominique Barba. Extraction of the relative depth information of objects in video sequences. In *ICIP (1)*, pages 948–952, 1998.

[92] Ignacio S McQuirk. An analog vlsi chip for estimating the focus of expansion. Technical report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1996.

[93] Krystian Mikolajczyk and Cordelia Schmid. Indexing based on scale invariant interest points. In *Proceedings of the 8th International Conference on Computer Vision*, volume 1, pages 525–531, Vancouver, Canada, 2001.

[94] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(10):1615–1630, 2005.

[95] Steven Mills. Stereo-motion analysis of image sequences. In *Proceedings of Digital Image & Vision Computing: Techniques and Applications (DICTA'97)*, pages 515–520, December 1997.

[96] Amar Mitiche and Souad Hadjres. Mdl estimation of a dense map of relative depth and 3d motion from a temporal sequence of images. *Pattern Anal. Appl.*, 6(1):78–87, 2003.

[97] P. Montesinos, V. Gouet, and R. Deriche. Differential invariants for color images. In *Proceedings of 14 th International Conference on Pattern Recognition*, pages 838–841. IEEE Computer Society, 1998.

[98] Hans Moravec. Towards automatic visual obstacle avoidance. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, page 584, August 1977.

[99] Pierre Moreels and Pietro Perona. Evaluation of features detectors and descriptors based on 3d objects. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 1, pages 800–807. IEEE Computer Society, 2005.

[100] Franck Morier, Henri Nicolas, Jenny Benois-Pineau, Dominique Barba, and Henri Sanson. Relative depth estimation of video objects for image interpolation. In *ICIP (1)*, pages 953–957, 1998.

[101] M. Concetta Morrone, John Ross, David C. Burr, and Robyn Owens. Mach bands are phase dependent. *Nature*, 324(6094):250–253, November 1986.

[102] D. Mumford and J. Shah. Boundary detection by minimizing functionals. In *Proc. IEEE Conf. Comput. Vision Patt. Recogn*, San Francisco, CA, USA, June 1985.

[103] Don Murray and James J. Little. Using real-time stereo vision for mobile robot navigation. *Autonomous Robots*, 8(2):161–171, 2000.

[104] H. H. Nagel. Displacement vectors derived from second-order intensity variations in image sequences. *Computer Graphics and Image Processing*, 21:85–117, 1983.

[105] H.H. Nagel and W. Enkelmann. An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(5):565–593, 1986.

[106] M. Nawrot and R. Blake. The interplay between stereopsis and structure from motion. *Perception and Psychophysics*, 49:230–244, 1991.

[107] S. Negahdaripour and B. K. P. Horn. Direct passive navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:168–176, 1987.

[108] P. Nesi. Variational approach to optical flow estimation managing discontinuities. *Image and Vision Computing*, 11(7):419–439, 1993.

[109] Jerry D. Nguyenkim and Gregory C. DeAngelis. Disparity-based coding of three-dimensional surface orientation by macaque middle temporal neurons. *J. Neurosci.*, 23(18):7117–7128, 2003.

[110] Jorge Nocedal and Stephen J. Wright. *Numerical optimization.* Springer series in operations research. Springer, 2 edition, 1999.

[111] S. Osher and J.A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79:dec/49, 1988.

[112] Sylvain Paris, François Sillion, and Long Quan. A surface reconstruction method using global graph cut optimization. *International Journal of Computer Vision*, 66(2):141–161, February 2006.

[113] M. Pollefeys, R. Koch, M. Vergauwen, and L. Van Gool. Automated reconstruction of 3d scenes from sequences of images. *ISPRS Journal Of Photogrammetry And Remote Sensing*, 55(4):251–267, 2000.

[114] J.-P. Pons, R. Keriven, and O. Faugeras. Modelling dynamic scenes by registering multiview image sequences. In *International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 822–827, 2005.

[115] M. J. D. Powell. *A method of nonlinear constraints in minimization problems*, chapter Optimization. London. Academic Press, 1969.

[116] M. Proesmans, L. van Gool, E. Pauwels, and A. Oosterlinck. Determination of optical flow and its discontinuities using non-linear diffusion. In *ECCV*, pages 295–304. Springer Verlag, 1994.

[117] Ruigang Yang Henrik Stewenius David Nister Qingxiong Yang, Liang Wang. Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 31(3):1–13, March 2009.

[118] Andreas M. Rauschecker, Samuel G. Solomon, and Andrew Glennerster. Stereo and motion parallax cues in human 3d vision: Can they vanish without a trace? *Journal of Vision Res.*, 6(12):1471–1485, 12 2006.

[119] B.J. Rogers and M.E. Graham. Similarities between motion parallax and stereopsis in human depth perception. *Vision Research*, 22:261–270, 1982.

[120] Brian J. Rogers and Thomas S. Collett. The appearance of surfaces specified by motion parallax and binocular disparity. *The Quarterly Journal of Experimental Psychology Section A*, 41(4):697–717, 1989.

[121] Carsten Rother and Stefan Carlsson. Linear multi view reconstruction and camera recovery using a reference plane. *International Journal of Computer Vision*, 49(2):117141, 2002.

[122] Sébastien Roy. Stereo without epipolar lines: A maximum-flow formulation. *Int. J. Comput. Vision*, 34(2-3):147–161, 1999.

[123] D. Scharstein and R. Szeliski. Stereo matching with nonlinear diffusion. *IJCV*, 28(2):155174, 1998.

[124] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vision*, 47(1-3):7–42, 2002.

[125] Cordelia Schmid and Roger Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, May 1997.

[126] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 2(37):151–172, 2000.

[127] Paul R. Schrater and Daniel Kersten. How optimal depth cue integration depends on the task. *International Journal of Computer Vision*, 40(1):71–89, October 2000.

[128] Steven Seitz and Charles Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 25(3):1067– 1073, November 1999.

[129] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 519–528, Washington, DC, USA, 2006. IEEE Computer Society.

[130] Hicham Sekkati and Amar Mitiche. Joint dense 3d interpretation and multiple motion segmentation of temporal image sequences: a variational framework with active curve evolution and level sets. In *ICIP*, pages 553–556, 2004.

[131] Hicham Sekkati and Amar Mitiche. A variational method for the recovery of dense 3d structure from motion. *Robotics and Autonomous Systems*, 55(7):597–607, July 2007.

[132] J. Shah. A nonlinear diffusion model for discontinuous disparity and half-occlusion in stereo. *CVPR*, 1:3440, 1993.

[133] A. Shashua and M. Werman. On the trilinear tensor of three perspective views and its underlying geometry. In *Proc. of the International Conference on Computer Vision*, Boston MA, June 1995.

[134] Jinxiang Chai Sing Bing Kang, Richard Szeliski. Handling occlusions in dense multi-view stereo. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'01)*, volume 1, page 103, 2001.

[135] Sudipta N Sinha, Jan-Michael Frahm, Marc Pollefeys, and Yakup Genc. Gpu-based video feature tracking and matching. In *EDGE 2006, workshop on Edge Computing Using New Commodity Architectures*, Chapel Hill, May 2006.

[136] Gregory G. Slabaugh, W. Bruce Culbertson, Thomas Malzbender, Mark R. Stevens, and Ronald W. Schafer. Methods for volumetric reconstruction of visual scenes. *Int. J. Comput. Vision*, 57(3):179–199, 2004.

[137] Gregory G. Slabaugh, Ronald W. Schafer, and Mat C. Hans. Image-based photo hulls. In *First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT02)*, pages 704–708, 2002.

[138] Natalia Slesareva, Thomas Bhler, Kai Uwe Hagenburg, Joachim Weickert, Andrs Bruhn, Zachi Karni, and Hans-Peter Seidel. Robust variational reconstruction from multiple views. *SCIA*, 4522:173–182, 2007.

[139] D. Snow, P. Viola, and R. Zabih. Exact voxel occupancy with graph cuts. In *CVPR00*, volume I, pages 345–352, 2000.

[140] C. Strecha, R. Fransens, and L. Van Gool. Combined depth and outlier estimation in multi-view stereo. In *Computer Vision and Pattern Recognition*, volume 2, pages 2394– 2401, 2006.

[141] C. Strecha and L. Van Gool. Pde-based multi-view depth estimation. In *First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT02)*, page 416, 2002.

[142] C. Strecha and L. J. Van Gool. Motion - stereo integration for depth estimation. In *ECCV (2)*, pages 170–185, 2002.

[143] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. *CVPR*, 1:1–8, 2008.

[144] Christoph Strecha, Rik Fransen, and Luc Van Gool. Wide-baseline Stereo from Multiple Views: a Probabilistic Account. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 552–559, 2004.

[145] Christoph Strecha, Tinne Tuytelaars, and Luc Van Gool. Dense matching of multiple wide-baseline views. In *ICCV 03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 1194, Washington, DC, USA, 2003. IEEE Computer Society.

[146] Peter Sturm and Bill Triggs. A factorization based algorithm for multi-image projective structure and motion. In *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, volume 2 of *Lecture Notes in Computer Science*, pages 709 – 720. Springer-Verlag, April 1996.

[147] G. Sudhir, S. Banerjee, K. K. Biswas, and R. Bahl. Cooperative integration of stereopsis and optic flow computation. *J. of the Optical Society of America A*, 12(12):2564–2572, 1995.

[148] R. Szeliski. Prediction error as a quality metric for motion and stereo. In *ICCV*, page 781788, 1999.

[149] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, November 1992.

[150] P. Torr and D. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *Intl. J. of Computer Vision*, 24(3):271–300, 1997.

[151] P. H. S. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *Int. J. Comput. Vision*, 50(1):35–61, 2002.

[152] P. H. S. Torr and A. Zisserman. Mlesac: a new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156, 2000.

[153] Philip H.S. Torr, Richard Szeliski, and P. Anandan. An integrated bayesian approach to layer extraction from image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):297–303, 2001.

[154] B. Triggs. The geometry of projective reconstruction i: Matching constraints and the joint image. In *Proc. International Conference on Computer Vision*, pages 338–343, 1995.

[155] Bill Triggs, Philip F. Mclauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment – a modern synthesis. *Lecture Notes in Computer Science*, 1883:298 – 372, January 2000.

[156] Doris Y. Tsao, Wim Vanduffel, Yuka Sasaki, Denis Fize, Tamara A. Knutsen, Joseph B. Mandeville, Lawrence L. Wald, Anders M. Dale, Bruce R. Rosen, David C. Van Essen, Margaret S. Livingstone, Guy A. Orban, and Roger B. Tootell. Stereopsis activates v3a and caudal intraparietal areas in macaques and humans. *Neuron*, 39(3):555–568, July 2003.

[157] T. Tuytelaars and K. Mikolajczyk. A survey on local invariant features. Technical report, K.U.Leuven, ESAT-PSI, May 2006.

[158] R. Tylecek and R.Sara. Depth map fusion with camera position refinement. In *CVWW*, 2009.

[159] S Ullman. *The Interpretation of Visual Motion.* MIT Press, Cambridge, Ma, 1979.

[160] Michael Unser. Splines a perfect fit for signal and image processing. *IEEE Signal Processing Magazine*, 1:22–38, November 1999.

[161] S. Vedula, S. Baker, S.M. Seitz, and T. Kanade. Shape and motion carving in 6d. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume II, pages 592–598, 2000.

[162] Sundar Vedula and Simon Baker. Three-dimensional scene flow. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(3):475–480, 2005.

[163] Thierry Vieville and Olivier D. Faugeras. Robust and fast computation of unbiased intensity derivatives in images. In *Proceedings of the Second European Conference on Computer Vision*, pages 203 – 211, 1992.

[164] G. Vogiatzis, P. Torr, and R. Cipolla. Multi-view stereo via volumetric graph-cuts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 391–398, 2005.

[165] H. Vu, R. Keriven, P. Labatut, and J.-P Pons. Towards high-resolution large-scale multi-view stereo. *CVPR*, 1, 2009.

[166] J. Y. A. Wang and E. H. Adelson. Representing Moving Images with Layers. *The IEEE Transactions on Image Processing Special Issue: Image Sequence Compression*, 3(5):625–638, September 1994.

[167] Z. Wang and Z. Zheng. A region based stereo matching algorithm using cooperative optimization. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

[168] A.M. Waxman and J.H. Duncan. Binocular image flows: Steps towards stereo-motion fusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):715–729, 1986.

[169] J. Weickert. *Anisotropic Diffusion in Image Processing.* B. G. Teubner, Stuttgart, Germany, 1998.

[170] A. E Welchman, A. Deubelius, V. Conrad, H. H. Blthoff, and Z. Kourtzi. 3d shape perception from combined depth cues in human visual cortex. *Nature Neuroscience*, 8(6):820–827, May 2005.

[171] Ryan White and David Forsyth. Combining cues: Shape from shading and texture. Technical Report UCB/EECS-2005-14, Electrical Engineering and Computer Sciences, University of California at Berkeley, November 2005.

[172] George Wolberg. *Digital Image Warping.* IEEE Computer Society Press, Los Alamitos, California, 1990.

[173] Joshua A. Worby. Multi-resolution graph cuts for stereo-motion estimation. Master's thesis, University of Toronto, 2007.

[174] Yalin Xiong and Steven A. Shafer. Dense structure from a dense optical flow sequence. *Computer Vision and Image Understanding*, 69(2):222–245, 1998.

[175] Feng Xu, Guihua Er, Xudong Xie, and Qionghai Dai. 2d-to-3d conversion based on motion and color mergence. In *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, pages 205 – 208, Istanbul, May 2008.

[176] Lixin Yang. *Structure and motion analysis: variational Methods and Related PDE models.* PhD thesis, Vrije Universiteit Brussel, 2004.

[177] Ruigang Yang and Marc Pollefeys. Multi-resolution real-time stereo on commodity graphics hardware. In *CVPR*, pages 211–217, 2003.

[178] A. J. Yezzi and S. Soatto. Structure from motion for scenes without features. In *Proc. 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 525–532, 532, Los Alamitos, CA, USA, June 2003. IEEE Computer Society Press.

[179] Anthony Yezzi and Stefano Soatto. Stereoscopic segmentation. *Int. J. Comput. Vision*, 53(1):31–43, 2003.

[180] Chang Yuan. *Motion segmentation and dense reconstruction of scenes containing moving objects observed by a moving camera.* PhD thesis, University of Southern California, April 2007.

[181] A.L Yuille. Shape from shading, occlusion and texture. Technical report, Massachusetts Institute of Technology, Cambridge, Artificial Intelligence Lab, May 1987.

[182] A. Zaharescu, E. Boyer, and R.P. Horaud. Transformesh: a topology-adaptive mesh-based approach to surface evolution. In *ACCV*, 2007.

[183] Y. Zhang and C. Kambhamettu. Integrated 3d scene flow and structure recovery from multiview image sequences. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 674–681, Hilton Head Island, SC, USA, June 2000.

[184] Y. Zhang and C. Kambhamettu. On 3-d scene flow and structure recovery from multiview image sequences. *Systems, Man, and Cybernetics, Part B*, 33(4):592–606, August 2003.

[185] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *Int. J. Comput. Vision*, 13(2):119–152, 1994.

[186] Song Chun Zhu and Alan Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):884–900, 1996.

List of publications

Book contributions

1. G. De Cubber. *Emerging robotics technology for humanitarian de-mining and risky interventions.* Human Victim Detection and Stereo-based Terrain Traversability Analysis for Behavior-Based Robot Navigation. Woodhead Publishing Limited, 2009.

ISI Journal papers

1. G. De Cubber, H. Sahli. A PDE-based approach for Dense 3D Structure and Motion Estimation from Monocular Image Sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence, submitted, 2010.*
2. G. De Cubber, S. Berrabah, D. Doroftei, Y. Baudoin, H. Sahli. Combining Dense Structure From Motion and Visual SLAM in a Behavior-Based Robot Control Architecture. *Journal of Advanced Robotic Systems, accepted, 2010.*
3. V. Enescu, G. De Cubber, K. Cauwerts, H. Sahli, E. Demeester, D. Vanhooydonck, M. Nuttin. Active stereo vision-based mobile robot navigation for person tracking. *Integrated Computer-Aided Engineering, vol. 13, nr. 3, pp. 203-222 , 2006.*
4. G. De Cubber, S. Berrabah, H. Sahli. Color-Based Visual Servoing Under Varying Illumination Conditions. *Robotics and Autonomous Systems, vol. 47, nr. 4, pp. 225-249, 2004.*

Other Peer-Reviewed Journal papers

1. G. De Cubber. Dense 3D structure and motion estimation as an aid for robot navigation. *Journal of Automation, Mobile Robotics and Intelligent Systems, Volume 2, N 4, pp 14-18, 2008.*

2. D. Doroftei, E. Colon, G. De Cubber. A behaviour-based control and software architecture for the visually guided Robudem outdoor mobile robot. *Journal of Automation, Mobile Robotics and Intelligent Systems, Volume 2, N 4, pp 19-24, 2008.*

Conferences

1. G. De Cubber. On-line and Off-line 3D Reconstruction for Crisis Management Applications. IARP Workshop RISE 2010, Sheffield, United Kingdom, January 2010.

2. D. Doroftei, G. De Cubber, E. Colon and Y. Baudoin. Behavior Based Control For An Outdoor Crisis Management Robot. Third International Workshop on Robotics for risky interventions and Environmental Surveillance-Maintenance, RISE2009- Brussels, Belgium, January 2009.

3. G. De Cubber, Daniela Doroftei, L. Nalpantidis, G. C. Sirakoulis, A. Gasteratos. Stereo-based Terrain Traversability Analysis for Robot Navigation. Third International Workshop on Robotics for risky interventions and Environmental Surveillance-Maintenance, RISE2009- Brussels, Belgium, January 2009.

4. G. De Cubber, G. Marton. Human Victim Detection. Third International Workshop on Robotics for risky interventions and Environmental Surveillance-Maintenance, RISE2009- Brussels, Belgium, January 2009.

5. G. De Cubber, D. Doroftei, G. Marton. Development of a visually Guided Mobile Robot for Environmental Observation as an Aid for Outdoor Crisis Management Operations. IARP Workshop on Environmental Maintenance and Protection, July 22 23, 2008, Baden-Baden, Germany.

6. G.De Cubber, G. Sirakoulis. Intelligent Robots need Intelligent Vision Visual 3D Perception. RISE 2008, Benicssim, Spain, 7-8 January 2008.

7. G. De Cubber. Dense 3D structure and motion estimation as an aid for robot navigation. ISMCR2007, Warsaw, Poland, 21-23 June 2007.

8. D. Doroftei, E. Colon, G. De Cubber. A behaviour-based control and software architecture for the visually guided Robudem outdoor mobile robot. ISMCR2007, Warsaw, Poland, 21-23 June 2007.

9. S. Berrabah, G. De Cubber, V. Enescu, H. Sahli. MRF-based foreground detection in image sequences from a moving camera. International Conference on Image Processing (ICIP 2006), pp. 1125-1128 , Atlanta, USA, 2006.

10. S. Berrabah, K. Cauwerts, G. De Cubber, T. Geerinck, W. Mattheyses, I. Ravyse, H. Sahli, M. Shami, P. Soens, W. Verhelst, P. Verhoeve. Audio-Visual Signal Processing: Speech and emotion processing for human-machine interaction. Second annual IEEE BENELUX/DSP Valley Signal Processing Symposium (SPS-DARTS 2006), pp. , Antwerp, Belgium, 2006.

11. V. Enescu, G. De Cubber, K. Cauwerts, S. Berrabah, H. Sahli, M. Nuttin. Active Stereo Vision-based Mobile Robot Navigation for Person Tracking. International Conference on Informatics in Control, Automation and Robotics, vol. 2, pp. 32-39, Barcelona, Spain, 2005.

12. G. De Cubber, S. Berrabah, H. Sahli. A Bayesian Approach for Color Constancy based Visual Servoing. 11th International Conference on Advanced Robotics, vol. 2, pp. 983-990, Coimbra, Portugal, 2003.

13. G. De Cubber, H. Sahli, E. Colon, Y. Baudoin. Visual Servoing under Changing Illumination Conditions. International Workshop on Attention and Performance in Computer Vision, vol. 1, pp. 47-54, Graz, Austria, 2003.

14. G. De Cubber, H. Sahli, F. Decroos. Sensor Integration on a Mobile Robot. ISMCR 02 12th International Symposium on Measurement and Control in Robotics, vol. 1, pp. 89-92, Bourges, France, 2002.

15. G. De Cubber, H. Sahli, H. Ping, E. Colon. A Colour Constancy Approach for Illumination Invariant Colour Target Tracking. IARP Workshop on Robots for Humanitarian Demining, pp. 63-68, Vienna, Austria, 2002.

VARIATIONAL METHODS FOR DENSE DEPTH RECONSTRUCTION FROM MONOCULAR AND BINOCULAR VIDEO SEQUENCES

Geert De Cubber

WWW.VUBPRESS.BE

9 789054 877103

ISBN 978 90 5487 710 3