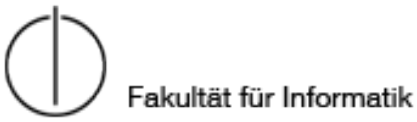


Monocular Reconstruction and Tracking in Non-Rigid Environments

Patrick Ruhkamp

Master of Science Thesis

Fakultät für Informatik
Technische Universität München
D-85747 Garching

2017

Technische Universität München
Fakultät für Informatik
CAMP - Chair for Augmented Reality and Medical Procedures

Master's thesis

Monocular Reconstruction and Tracking in Non-Rigid Environments

Patrick Ruhkamp, B.Sc.

Submission Date: 04. August 2017

Advisors:
Prof. Dr. Nasir Navab
Dr. Federico Tombari
Benjamin Busam, M.Sc.

Ich erkläre hiermit, dass ich die vorliegende Master's Thesis selbstständig und nur mit den angegebenen Hilfsmitteln angefertigt habe.

München, den 04. August 2017 _____

(Patrick Ruhkamp)

Acknowledgments

I would like to thank my supervisors Benjamin Busam and Dr. Federico Tombari for their endless support, time and helpful discussions. I appreciated the work at the excellent and distinguished research chair of Prof. Nassir Navab and thank him for this opportunity.
I dignify the encouragement and support of my family and friends, helping me to reach my goals.

Abstract

Simultaneously localizing a moving camera system and sparsely mapping its environment is already a well studied problem in robotics (SLAM). Likewise, structure from motion (SfM) approaches reconstruct the scene in a denser manner. For both, tracking and reconstruction, non-rigidity introduces substantial uncertainty. Tracking may drift or fail entirely whereby correct reconstruction is unfeasible. Assuming non-rigid deformations, the fusion of consecutive reconstructions of the scene into a common model over time is not meaningful, since older results may not represent the deformed shape correctly. Therefore, substantial outlier removal for robust camera pose tracking, and incorporation of non-rigid deformations during reconstruction are essential for describing such a scene with certain accuracy.

Most of the current methods utilize RGB-D cameras to obtain direct depth information of a scene. Usually an embedded deformation graph accounts for non-rigidity and brings the already observed reconstructions into alignment with the current observation. Image based methods often try to simplify the problem of non-rigid deformations, by e.g. fragmenting the overall deformation in rigidly deforming individual parts, assuming deformations to be rigid or making prior assumptions on the deformation. Furthermore, adequate initialization is mostly needed.

Here, a system for tracking the ego-movement of a monocular RGB camera in a non-rigid environment, while densely reconstructing the scene, is presented. Tracking is performed on robust sparse features. For reconstruction, dense stereo correspondences are computed. An optical flow based feature matching approach accounts for rigid and non-rigid transformations across reconstructions and constitutes the embedded deformation graph for alignment of depth information. Local per point transformations across the deformation graphs are computed over time, while accounting for transformations being as-rigid-as-possible (ARAP) and incorporating further regularization. Dual quaternion blending (DQB) transforms the dense reconstructions according to the underlying embedded deformation graph into alignment with the current observation. Reconstructions are integrated into a common octree based signed distance field (TSDF) and respective deformations are updated within it.

Contents

1 Introduction

In robotics the estimation of the current position with regards to the surroundings is a common problem. Simultaneous localization and mapping (SLAM) algorithms try to estimate the pose and build up a sparse representation of the scene in real-time [29, 36]. SLAM is usually not sufficient to reconstruct a scene in detail due to sparsity and real-time constaints. Structure from motion (SfM) approaches compute dense depth information from different camera poses and fuse them into a common reconstruction model [33, 120]. These approaches often require expensive computation and alignment of reconstruction frames, which makes them mostly unfeasible for real-time scenarios [150]. Newcombe et al. [98] proposed a live and dense reconstruction approach for small rigid scenes and other real-time approaches for dense reconstruction emerged as well [63, 100]. Non-rigid transformations and deforming objects in the scene elevate the the problem of tracking and reconstruction even further. Utilizing RGB-D cameras with direct depth information, the first real-time methods, accounting for non-rigid deformations, were introduced [64, 99]. For sole RGB information usually vast simplifications [158] or shape templates are needed [153] (see section 2.2 for further details) to account for non-rigidity.
In the algorithm proposed in this thesis, the position of a freely moving monocular camera within an unknown non-rigid scene needs to be tracked. With known camera poses, stereo comparisons between suitable consecutive frames enable to reconstruct the scene. Deformations are accounted for by non-rigid alignment between sparse sets of optical flow based matched feature points. This set of feature points, with their associated deformations, constitute the embedded deformation graph, which enables to deform the rest of the reconstruction via dual quaternion blending (DQB) accordingly. The algorithm shall only rely on suitable RGB image inputs and on adequate camera poses for the first frames for proper tracking initialization. More details on the aim of

this thesis are given in section 1.1.
To introduce the topics of tracking, reconstruction and non-rigidity some thoughts on possible application scenarios and a brief overview of the terms are given in the next section. A rough historical overview of visual reconstruction and tracking systems will be presented as well as a classification of the relevant literature. Possible future applications and advantages of a monocular non-rigid reconstruction system will be depicted to appeal the reader of the necessity and benefits of such a system. At the end of the introductory chapter, the structure of the rest of this thesis will be laid out to help the reader.

Everything changes
and nothing stands still.
(Heraclitus[1])

πάντα χωρεῖ καὶ οὐδὲν μένει

[1]As in Plato's Cratylus [43]

1.1 Goal of thesis

The goal of this thesis is to develop a monocular system capable of tracking its ego-movement and reconstructing the observed scene. This system only depends on the visual information obtained by a single freely moving camera. Different principles and methods proposed in literature are combined and further extended. While providing robust camera tracking, based on features as presented in recent SLAM-algorithms [30, 72, 93], and exploring the scene with a moving monocular camera in a SfM fashion [98], the 3D scene is reconstructed as a dense volumetric model by fusing depth information into a common actree based truncated sigend distance function (TSDF) model over time [101]. The approach also accounts for regularization and incorporation of non-rigid deformations [64, 99, 125, 153], thus enabling the reconstruction of such scenarios. The obtained depth information can be aligned by iterative closest points (ICP) methods [59, 116] to improve the overall alignment of the reconstructed scene and reducing drift. Additionally, the aim is to provide an algorithm which does not need prior template shapes or initialization but rather starts with an estimation of the camera pose and scene reconstruction. This is refined over time, resulting in accurate camera poses and a dense high quality 3D reconstruction of the scene. Non-rigidity deformation is accounted for by an embedded deformation graph [131], similar to current RGB-D approaches, where projective implicit correspondence association is perfomred between the model and the current depth information, to define the deormations nodes of the graph [64, 99, 153]. Here, an explicit correspondence association on a sparsely triangulated representation of the scene from matched ORB features is suggested. The deformation graph encodes the underlying rigid and non-rigid transformations and constitutes the basis for dual quaternion blending (DQB) of the model.

The reason for using a monocular camera instead of a stereo system, RGB-D cameras or more sophisticated sensors, is to provide a tracking and reconstruction approach applicable for a variety of scenes and environments. The scale ambiguity of monocular systems is of great benefit, while at the same time only being dependent on a simple, small and affordable camera system.

As proposed, the applicability of the use of a monocular RGB camera in a non-rigid scene for tracking and reconstruction shall be demonstrated. Furthermore, deformations accounted for by a deformation graph and subsequent DQB of the model shall be shown to be sufficient with a sparse explicit correspondence association approach.

1.2 Computer vision and today's applications

In many modern research fields such as robotics, augmented reality or autonomous driving, it is vital to obtain visual information from the surroundings. Drawing information from the scene is a challenging and likewise interesting field in modern computer vision. Advances in hardware performance and affordable GPUs have paved the track to tackle even more sophisticated problems in higher quality. Different sensors for recording the scene and obtain visual information can be used: (1) RGB-D sensors, which provide not only RGB input but also a depth map, (2) stereo RGB cameras, (3) monocular RGB cameras and (4) other sensors such as laser systems or structured light. One advantage of monocular systems is the scale ambiguity [36], thus being able to use them in different environments. A single camera system can be installed more easily compared to other complex systems and has low costs, while 3D reconstruction is somewhat more challenging.
Research in the area of monocular tracking goes back to the late 80's, when Harris and Pike presented their DROID system [49]. Real-time tracking and reconstruction do offer a variety of applications in mentioned fields. The ideas which are used in those already established areas can also be extended towards other fields such as the medical scope, with applications in computer aided surgery, image guidance during surgery or minimally invasive surgery (MIS). One specific field of interest is the biopsy and treatment of tumorous tissue.

"Cancer is a major public health problem in the United States and many other parts of the world. One in 4 deaths in the United States is due to cancer"[122]

Minimally invasive surgery (MIS) has many benefits for the patient compared to classical open surgery [79]. However, it is more challenging for the surgeon, since the laporoscopic camera can not provide depth information or detailed information of the surrounding operational setting [84]. Next to already existing methods for stereoscopic MIS, such as the *da Vinci* [65] system [126], monocular approaches emerged. The approach of using such monocular tracking and reconstruction systems may prove to be of great value for the patient and surgeon. This is just one of many possible application scenarios.
The problem of a moving system - here we are using a monocular camera system - within an unknown environment, has already been considered with different SLAM

approaches [30, 36] (compare Figure 1.1).

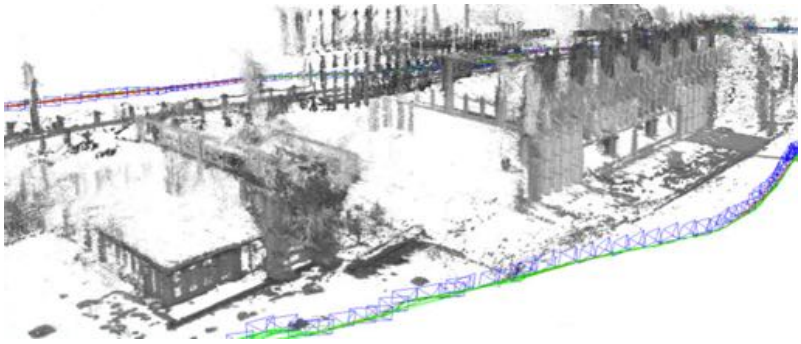


Figure 1.1: Example of a SLAM trajectory; As presented in [36].

Another research area on vision-based reconstruction of the environment is SfM (compare Figure 1.2). This approach traditionally uses offline global optimization techniques over the whole image sequence to reconstruct the scene [30], but new algorithms have also been shown to work in real time, as already mentioned. The non-rigid SfM is of particular interest for many applications, since often transformations may occur which are not limited to be rigid, e.g. movement of humans or deforming surfaces (compare Figure 1.3).

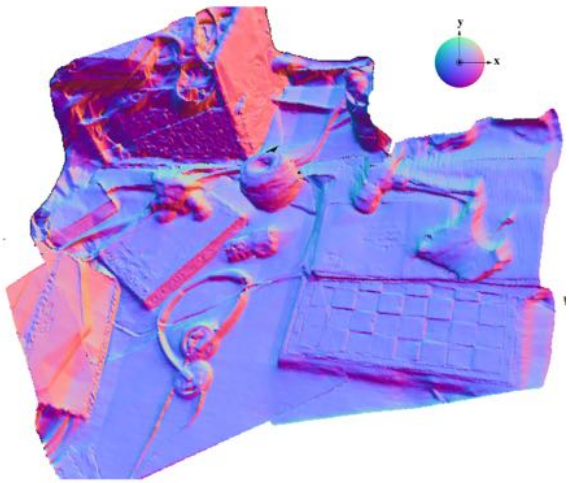


Figure 1.2: Example of a reconstructed Structure from Motion (SfM) scene; As presented in [98].

Figure 1.3: Example of a reconstructed non-rigid scene; As presented in [99].

Here, we have the combined problem for a vision based system within an unknown possibly non-rigid scene, which needs to localize itself with regards to the surrounding and simultaneously reconstruct the scene.

1.3 Problem formulation and contributions

The problem of tracking the egomovement of a monocular camera and reconstructing the observed scene is two-fold. Computing the depth information from stereo images is conditioned on estimating the relative camera pose of a stereo setup. In this case, the problem is even aggravated to non-rigid environments. Non-rigidity is imposed by deformable objects or surfaces, such as human tissue, the movement of humans or general deforming objects. This work lays out necessary principles for 3D reconstruction with monocular vision. Furthermore, a robust system for dense non-rigid reconstruction is presented. It involves robust feature tracking, camera pose estimation, dense 3D reconstruction and alignment of extracted depth information together with sophisticated deformation regularization based on an embedded deformation graph and DQB. This provides high quality reconstruction results for non-rigid scenes with a moving monocular camera.

1.4 Structure of thesis

To commence with an overview of recent developments in the research area of SLAM, SfM and 3D reconstruction, relevant papers will be mentioned in chapter 2. Beginning with a classification and definition of above named methods, current approaches, and how they are used in applications, are considered.

Preliminaries and fundamentals of monocular and stereo vision are introduced in chapter 3. Important methods for tracking and reconstruction are specified in 4. Furthermore, an overview of the pipeline for the proposed algorithm, suggested methods and the implementation of those is presented and discussed in detail subsequently. In section 5 quantitative results for the camera pose tracking and qualitative results for non-rigid reconstruction are presented and discussed.

2 Related work

A short overview, classification and definition of important terms in computer vision for this thesis, namely *Simultaneous Localization and Mapping* (SLAM), *Structure from Motion* (SfM) and *3D-Reconstruction*, will be given. After briefly covering the historic development of such methods, a discussion of relevant research papers on the current state of the art follows. Common application scenarios are described and potential future fields are considered. The last part of this chapter will restate the papers and methods most relevant for this thesis. Thus, the reader conceives a precise idea on terminology, gains a broad insight in current research and receives an introduction in the most important literature and used methods for this work on non-rigid reconstruction and tracking.
Analogous to the evolution of life on earth as depicted in Figure 2.1, we will see how tracking and reconstruction methods have evolved over time and led to current approaches and state-of-the-art methods in visual SLAM and 3D reconstruction.

In my opinion, all previous advances
in the various lines of invention
will appear totally insignificant when compared
with those which the present century will witness.
(Charles H. Duell[1])

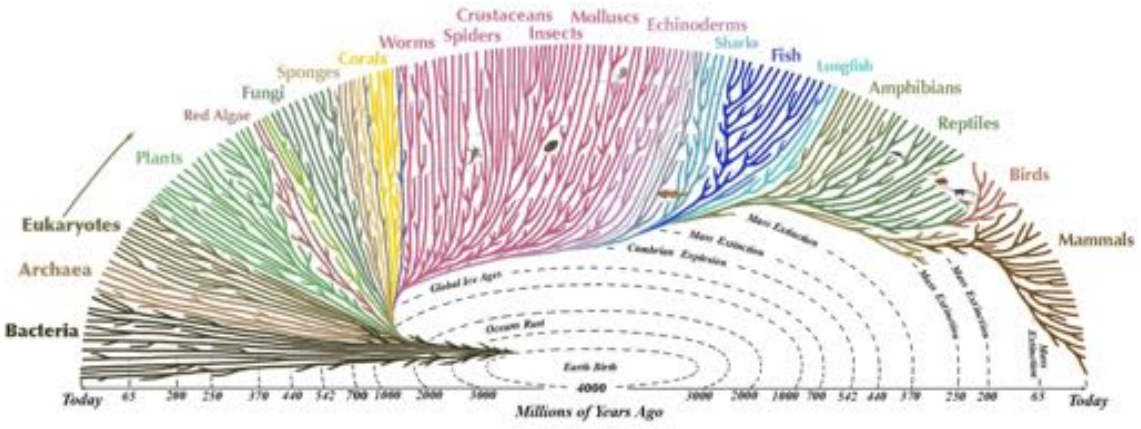


Figure 2.1: The great tree of life; Adopted from [161].

[1] Charles Holland Duell, Commissioner, U.S. Office of Patents, 1899 *Cortland, New York 13.04.1850, †Yonkers, New York 29.01.1920 [136]

2.1 Definition and classification of different approaches and terminology

Important components which are used in computer vision and applied in different areas, such as robotics or augmented reality, differ in terminology and may get confused, since in many cases they describe similar approaches for related problems. Among others these include the computation of the relative camera pose in space [100], reconstruction of 3-dimensional rigid [63] and non-rigid [64, 99] scenes or mapping the whole scenario within a global graph [36]. Therefore, a short general overview of the most relevant terms will be given.

2.1.1 Simultaneous localization and mapping

SLAM algorithms are often applied to localize the position of a camera system within an unknown scene and to map the surroundings for navigation purposes in robotics [36]. Localization and mapping approaches aim to build up a rudimentary representation of the scene for orientation, rather than reconstructing specific elements in fine detail [101]. SLAM can be further differentiated in feature based methods [93] and algorithms depending on whole image alignment [36]. In general, feature based methods can be considered faster, since computation is performed on a small set of image features. Relying on features might be a drawback for scenes with few geometric structures [36]. Then again, they do not rely on good initialization or are interfered by camera artifacts as for example rolling shutter.

Direct methods try to circumvent the limitations of feature based methods by incorporating the whole image within the optimization procedure, thus using much more information from the images and yielding to higher accuracy and level of detail [36].

Geometric calculations are usually performed based on the information of the current image and a defined key-frame. A key-frame is some suitable frame of an image sequence and can be obtained by heuristics, thresholds or more sophisticated methods (e.g. a new key-frame is selected after a certain global transformation is reached).

Commonly used in SLAM, is the creation of a scene map by fusing poses from the geometry into a global graph (compare Figure 1.1). Sophisticated algorithms also account for loop closures [36].

2.1.2 Reconstruction

Detailed acquisition and reconstruction of 3D geometries extend the approaches described above [98]. In general, scenes might consist of rigid and non-rigid geometries, whereas rigid elements are not deformable. As a result, transformations are limited to rotation, translation and scaling. Non-rigid bodies imply many more degrees of freedom, being highly difficult to track and reconstruct over time [63, 99]. The objective of reconstruction algorithms is to obtain a dense representation of the scene.
To reconstruct a static scene (compare Figure 1.2), besides other mentioned sensors, a freely moving camera can be used. This approach is commonly referred to as SfM, inasmuch as the structure is reconstructed from images taken at different points of view from the moving camera.
In general, the aim of SfM, alike SLAM, is to gain geometrical information from the environment. In extension to generated sparse scene representations from SLAM algorithms, SfM approaches reconstruct a dense model from the scene. Often a TSDF is used for representing the 3D object in a data structure, while fusing all obtained reconstructions from different points of view together [27, 101]. For offline approaches, so-called bundle adjustment can be used to perform a globally consistent reconstruction of the geometry [30]. Likewise for SLAM, here the position of the camera with regards to the scene needs to be estimated to be able to reconstruct the structure.
An even more challenging task is the reconstruction of deformable objects. For this purpose templates and strong prior assumptions are often used to ease the process [135, 153], or a preliminary initialization phase is required [44]. Recent approaches [99] compute warp fields from depth images and align deforming reconstructions with the model represented in a TSDF. Please note that for current real-time approaches [64, 99] RGB-D cameras are used. Here, the sensor directly provides a depth map aligned with visual information, which is not the case for a normal monocular camera.

2.2 Current state of the art

Several methods have been proposed in literature to extract information of the environment with vision based approaches. The recent emergence of commodity RGB-D sensors have accelerated research in robotics, leading to many real-time applications for SLAM. SfM and multi-view stereo methods are also in focus. To overcome small

application distances of RGB-D sensors, stereo and monocular camera methods seem promising [110]. Here, the focus will be on monocular systems since they are able to handle scenes with differing distances. Generally, non-rigidity within the scene is still a major challenge, which recent methods try to circumvent with prior initialization for template generation [153], or by using RGB-D sensors and regularization of the non-rigid deformation [64, 99].

2.2.1 The rise of real-time SLAM algorithms

Davison et al. [29] were the first to show real time capabilities for localization and mapping using monocular cameras and extended their work towards their MonoSLAM system [30]. They use an extended Kalman filter (EKF) to incrementally build up a sparse high quality feature map including a probabilistic estimate accounting for possible deviations. The system is mainly targeted for small rigid scenarios with various possible loops and large dynamic camera movements (e.g. moving freely within a room), rather than moving along a long trajectory. Therefore, they use a standard single full covariance version of EKF to allow the system to detect loop closures within the dynamic movement. No single drift aware loop closure after a long trajectory is applied, thus being able to perform long-term repeatable localization. Landmark features within monochrome images and the computed camera location are used for robust tracking during large camera movements. For depth perception a short initialization procedure - tracking a few frames of a standardized object - is required to obtain a template shape. One key contribution of their approach which improves runtime, is active feature search in a bottom-up fashion, leading to improved efficiency by lowering the amount of necessary image processing [30].
Parallel tracking and mapping (PTAM) [72] was the first approach to split tracking and mapping into two tasks, while using a passive monocular sensor for data auqisition. One thread continuously tracks the monocular camera movement, the other one creates a 3D depth map, which is denser compared to earlier approaches. Instead of producing a sparse map of high quality features, they aim for a somewhat denser map, but with lower quality features, still achieving real time computation for small scenarios [72].
MonoFusion by Pradeep et al. [111] first estimates the camera pose from a monocular input stream and secondly constructs depth maps by dense stereo matching between key frames. They incrementally update their reconstructed model in a signed distance

function (TSDF) using local L2- based regularization, leading to improved speed while providing high quality results comparable to KinectFusion [101] (see below: 2.2.2) [111].
Engel et al. [36] present a monocular direct SLAM algorithm with scale-aware alignment and drift reducing global pose graph creation. They track new keyframes with $se(3)$ elements for camera position and $sim(3)$ elements [8] for scale aware direct image alignment by incorporating the probabilistic depth uncertainty, thus reducing scale-drift and enable loop closure even for long trajectories [36]. Rather than using classical bundle adjustment over features, they optimize the problem on the pixel intensities directly. They proofed real-time capabilities with their CPU based implementation [36].
Mur-Artal et al. presented the ORB-SLAM [93] algorithm for monocular SLAM, and its later extension, ORB-SLAM2 [94] (for monocular, stereo and depth cameras). As in PTAM by Klein et al. [72] they use bundle adjustment over features to estimate the camera pose and sparsely reconstruct the environment. They use ORB features [115] for all SLAM tasks, introduce a novel space recognition technique enabling for lifelong tracking, and show results superior to other state of the art SLAM algorithms, even compared to dense methods such as LSD-SLAM [36] or DTAM [100]. They argue, that despite strong prior inputs for feature based bundle adjustment techniques, their approach still performs with higher accuracy compared to previously mentioned methods. Direct methods potentially poses photometric artifacts and are computationally expensive, resulting in only incremental upgrades of the map or discards of information leading to a reduced pose graph as in LSD-SLAM. They propose to combine feature based methods with direct methods to obtain denser reconstruction results [93].

2.2.2 From SLAM towards dense reconstruction

For augmented reality or similar applications Newcombe and Davison [98] understood that a sparse map as obtained by classical SLAM algorithms is not sufficient to describe a scene in an adequate precision. Therefore, they use a dense SfM approach to gain much denser depth maps being able to reconstruct a detailed surface mesh. Firstly, a sparse point map and the camera position are estimated similar to PTAM [72], adding new points to the map over time and polygonise those to obtain a first surface estimate. Secondly, images from camera positions partially overlapping the initial surface map

are fed into a dense reconstruction process, resulting in dense depth maps which are later integrated into the global surface model [98]. Depth estimation for monocular approaches is usually done by triangulation [111] and stereo comparisons [37, 110] between certain points from the current image and a previous keyframe [63].
By incorporating the whole image, rather than just using sparse features, Newcombe et al. [100] further improved robustness based on direct image alignment, thus even being able to provide dense 3D reconstructions of the scene in real time. However, they use global optimization for continuously regularizing the incrementally built up depth map, leading to higher computational complexity [111]. Their approach is based on previous ideas to build up 3D models by integrating new depth information into a volumetric model (TSDF) [27] and add further robustness and smoothness by a total variation L1 (TV-L1) regularizer [154]. By incorporation of the TSDF they extended their earlier dense reconstruction approach [98].
Pizzoli et al. [110] present an optimized probabilistic complete dense reconstruction method named regularized monocular depth estimation (REMODE). They argue for a monocular multi-view stereo approach, depth information is not only influenced by the accurately acquired image, but also by the correctly computed camera orientation. Bayesian estimation is used to account for uncertainty in the measurement, thus creating accurate depth maps in a sequential manner [110].
Another complete dense depth reconstruction method (CD-SLAM) by Huang et al. [63] outperforms the above approaches. Here, dense mapping and sparse batch tracking are separated into two threads. Tracking is based on the idea of Forster et al. [42] who just uses patches of the images to estimate the camera pose by minimizing a normalized error term, thus leading to high efficiency. For dense tracking, a new weighted gradient which incorporates depth disparities, is added to the photometric error term. This leads to enhanced results for non constant illumination conditions and more precise edge regions. The method provides depth maps in pixel resolutions which allows for detailed reconstruction [63].
Keller et al. [70] reduce computational complexity by proposing a different way of representing surfaces. They use a flat, point-based representation directly extracted from RGB-D data instead of a spatial data structure, leading to improved memory performance while maintaining high detailed tracking and reconstruction. Based on the idea of motion segmentation in [66], their approach can also handle occlusions and even reconstrcut moving objects [70]. Whelan et al. [147] extended this idea towards

a surfel representation in their real time dense SLAM system using RGB-D, as Stückler and Benke [129] did in their multi-resolution SLAM. Tracking is performed with photometric and geometrical information in a frame-to-model approach continuously fusing new information into a dense surface element (surfel) [109] based map. Local loop closures are performed frequently, together with a global surface recognition technique, which makes global pose graph optimization obsolete. Whelan et al. further extended this method to compute number and orientation of light sources within the scene for more realistic results beneficial for augmented reality [148].

Newcombe et al. [101] extended an earlier approach based on RGB cameras [98] for the use of an RGB-D camera which provides direct depth information. They show real-time capabilities for reconstruction of small rigid scenes. Camera pose tracking is performed by model-to-frame comparison to reduce the effect of drift from frame-to-frame methods. New depth maps are continuously integrated into a volumetric TSDF model, inspired by Curless and Levoy [27], thus providing high quality results.

Based on KinectFusion [101] Izadi et al. [66] extended the system to register objects leaving their initial position or moving through the scene and enable for extensive user interaction. Outliers acquired during iterative-closest-point (ICP compare 4.4.1) computation are identified as moving objects and segmented from the rest of the scene [66]. Perera et al. [108] refine this approach, being able to segment moving objects even if they do not leave the initial position completely. Assuming the latest reconstructed TSDF does not accurately describe the current RGB-D depth and camera pose information, in case an object is moving in the scene, the likelihood for a point not being static is calculated and subsequently segmented [108].

Thinking of mobile applications, real-time performance, accuracy and applicability are difficult to achieve. MobileFusion by Ondruska et al. [104] gives an idea of what the discussed approaches might be capable of. Their combined GPU/CPU implementation on a mobile phone is capable of computing dense 3D mesh models, while estimating the 6 DOF, using the internal RGB camera. Despite limited computation power, results are compelling and calculated in real-time. However, their method is limited to small static scenes with only global rigid camera movement and without non-rigid transformations [34].

2.2.3 Extension to non-rigidity

The majority of the mentioned methods above, lack the ability to reconstruct deformable non-rigid objects. Also other methods need a template of the objects in the scene as prior as in Yu et al. [153], or assume simplifications of the geometry as in [12] and [135]. Newcombe et al. [99] and Innmann et al. [64] were among the first to achieve real-time tracking and 3D non-rigid structure from motion reconstruction (NRSfM) with RGB-D cameras. Newcombe et al. extend their rigid reconstruction approach (KinectFusion [101]) by using a coarse warping field to optimize the scenario for rigid and non-rigid transformations at the same time by only incorporating the depth information. Innmann et al. [64] propose the utilization of RGB information, thus improving tracking robustness especially if there is only little geometric structure. Furthermore, they use sparse SIFT features to improve drift and tracking during fast movements.

Bregler et al. [12] extended the structure from motion problem to the non-rigid case by describing any object by a linear combination of basis shapes. Torresani et al. [135] used the same idea. However, these approaches simplify the ambiguous shape and motion problem [32]. Del Bue et al. [32] used an optimized version of RANSAC [41] with prior probability distributions of the degree of non-rigidity [68] for each feature point, enabling to distinguish between rigid and non-rigid movement, under the prerequisite only rigid points undergo Euclidean transformations. After segmentation, rigid points are used to compute the general translation and rotation of the scene, and the non-rigid shape is defined as a non-linear optimization problem [32].

Garg et al. [44] present a method to reconstruct NRSfM in a variational minimization approach by switching between solving the 3D surface shape and the camera motion. Their system is not applicable for real time since they require 2D image trajectories throughout the image sequence to estimate motion and use a batch process for optimization. Furthermore, they only show compelling results if there is minor rotation within the scene. They propose to incorporate the photometric information for future approaches [44].

VolumeDeform by Innmann et al. [64] present a method for reconstruction of non-rigid objects without the need of a template. However, the method still needs strong priors for initialization. They use a unified volumetric representation for the undeformed shape by incorporating a color and depth map for each frame, and fusing new

deformed information into a TSDF. Further, they integrate the global motion similar to DynamicFusion [99], but with higher accuracy by interpolating between grid points [64]. Deformation is applied to the vertices of the scene, which are obtained by an optimized marching cubes algorithm of the TSDF. The deformation graph is mainly updated by a dense depth correspondence matching, with additional use of sparse robust color features to make tracking more robust and account for drift. For color features, SIFT features are computed in a coarse to fine pyramid approach and further sorted to obtain the best ones. Aligning the non-rigid surface is done in an ICP-manner, incorporating the dense depth data and sparse color data. An additional regularization term utilizes an as-rigid-as-possible (ARAP) [125] (compare 4.5.1) approach to account for the highly under-constrained optimization problem. The resulting energy term, which accounts for the one-ring neighborhood around the isosurface, is minimized by a data parallel strategy, and applied within several hierarchy levels. This enables real-time computation also for fine grained resolutions. Despite imposing non-rigid tracking ability in real-time and improved robustness by incorporation of SIFT features, *VolumeDeform* still has issues with drift and is, due to the use of a uniform grid for representation purposes of the deformation, limited to small application environments [64].

Compelling real-time results for non-rigid reconstruction with a single RGB-D camera have been achieved by the GPU implementation by Zollhöfer et al. [159]. They claim to present the first general purpose non-rigid reconstruction approach, since the method does not rely on strong priors, physics or shape models. However, a short rigid sequence of the object of interest is required to acquire a template online. In the initialization period a multi-resolution grid is extracted with an optimized ICP approach by Nießner et al. [102]. Non-rigid registration is performed in a coarse to fine approach with ARAP [125] regularization and new data is fused into the model at the finest level [159].

Based on the ideas of Zollhöfer et al [159], Yu et al. [153] proposed a new monocular method for direct, dense non-rigid 3D reconstruction. Their approach also incorporates coarse to fine computation and ARAP regularization [125] accounting for non-rigid regularization of a mesh model. Further regularization and smoothness are applied to the model. The energy function is minimized in two steps, firstly for rotation and translations, and secondly for shape. Template acquisition is performed by calibrating the input frames using a multiple view stereo (MVS) algorithm [150], then depth

maps are computed using stereo comparisons [25] which are fused into a volumetric representation using a method from [139]. The authors show results comparable to, or even more accurate than Garg et al. [44]. The presented approach is superior to [44] in terms of scalability, being able to compute even long sequences. Despite the need of a template, their method can be applied to various close range scenes and objects by generating the input in a generic online way [153].

Contrary to using one single RGB-D sensor, Dou et al. [34] use several cameras in their Fusion4D work, based on methods presented by Newcombe et al. [99]. Despite their complex setup, they are able to reconstruct highly accurate non-rigid objects in real-time with topology changes.

Dynamicfusion by Newcombe et al. [99], VolumeDeform by Innmann et al. [64] and Fusion4D by Dou et al. [34] all incorporate an embedded deformation graph [131] to account for non-rigid deformations throughout the reconstruction process. They estimate correspondences between consecutive frames and compute per point local transformations for each node of the deformation graph by non-rigid alignment minimization with sophisticated regularization. Yu et al. [153] propose a similar non-rigid optimization for their template-based monocular RGB reconstruction approach. Since correspondence association may be difficult and expensive, Slavcheva et al. [123] proposed to perform non-rigid alignment directly in a SDF without the need of correspondences. They account for general rigid transformation between frames by their SDF-2-SDF method [124]. For non-rigid deformations, an approximately Killing vector field accounts for the 3D flow field between frames. Alignment is directly performed within the SDF.

2.3 Applications in new fields

Minimally invasive surgery (MIS) has many benefits compared to traditional open surgery, which include less pain, smaller trauma, lower infection risk and shorter hospital stay [79]. As reference see the well known professional MIS product *da Vinci* system [65]. Due to the setup of MIS - inflating the abdomen and access the area of interest through small incisions - surgeons get information from laparoscopic images during surgery. However, the images are lacking depth perception, which can only be assumed by the surgeons based on their experience. To further enhance MIS,

different approaches to include computational capabilities, have emerged in recent years, which in general are referred to as computer-aided surgery (CAS) [79]. The goal is to provide additional information for the surgeon during the medical procedure, e.g. by developing models of organs obtained from medical images beforehand, or track the surgical instruments and guide the surgeon [79].
Using computer vision to provide the surgeon with additional visual information is one way to integrate computer aided procedures. Vision-based 3D-reconstruction in real-time is quite challenging in the medical scope, induced by non-rigidity and lack of geometric structures [78, 79]. Thus, reconstruction and tracking become a lot more difficult, as they are almost unfeasible due to fast movements, occlusions and non-rigid properties and deformations.
Other application scenarios include augmented and virtual reality applications. The extension of KinectFusion for interaction by Izadi et al. [66] gives an idea for such scenarios. While tracking the camera pose and reconstructing the scene, the user is able to extensively interact within the scene by for example placing virtual objects within it. For archaeology or speleology easily obtainable dense reconstructions may be advantageous. Reconstructions of rooms or apartments might be useful for real-estate. Especially for non-rigid reconstruction, character animation and character scanning for movies and video games can be an interesting, since today they usually rely on relatively complex recording setups. Robotics, autonomous drones or vehicles are further possible application scenarios. Being able to reconstruction deforming objects would yield to easier information retrieval of the scene.

2.4 Focus and and most relevant existing approaches

Of particular interest for our tracking pipeline is the ORB-SLAM approach of Mur-Artal et al. [93]. Newcombe et al. lay out important principles for dense reconstruction [98] and for integration of depth information in a TSDF [27] in their KinectFusion method [101]. DynamicFusion [99], VolumeDeform [64], and Fusion4D [34] introduce how deformations within the scene can be accounted for with an embedded deformation graph [131]. Mentioned approaches for RGB-D data, and Yu et al. [153] for RGB, utilize sophisticated regularization [125] for non-rigid alignment, vital for meaningful non-rigid reconstruction results.

3 Preliminaries and monocular vision

In this chapter important preliminaries on monocular and stereo vision will be presented. A short outline of the methods used in this work will be depicted to give the reader a comprehensive overview. First, an overview and common every day examples will introduce the idea on how to extract depth information from images.
The fundamental principles on how a scene is transformed into an image with a projective pinhole camera will be given. We can extend the idea of a single moving camera to a virtual stereo camera system setup as discussed in section 3.3.
We will see how the relative transformation between the cameras of a stereo system can be described. For a simplified case, with known camera pose and the mathematical characteristics of the projective camera model, we will illustrate how the 3D position of a certain point observed in two corresponding images can be computed.
Having the way a cyclops (compare Figure 3.1) observes the world - as representative of a monocular camera - in mind, we will discuss the principles needed to describe the visual information in a mathematical way, suitable for computation.

Nature is written
in mathematical language.
(Galileo Galilei[1])

Figure 3.1: The Cyclops: 1914 (oil on canvas) by *Odilon Redon*[2], Rijksmuseum Kröller-Müller, Otterlo, Netherlands; Adopted from [149].

[1] *Galileo Galilei*, Italian Mathematician, Physicist and Philosopher, *Pisa, Italy 15.02.1564, †Florence, Italy) 08.01.1642 [19]

[2] *Odilon Redon*, or Bertrand-Jean Redon, French Artist, *Bordeaux, France 22.04.1840, †Paris, France 06.07.1916 [18]

3.1 Introduction and overview of monocular 3D reconstruction

The visual abilities of humans are generally different to the ones of a cyclops in Figure 3.1. The binocular vision of humans is the biological model of a stereo camera setup. Taking images from a scene with a monocular camera at distinct camera poses basically describes a generalized stereo setup, as discussed below.

The fundamental necessity for stereo 3D reconstruction is to obtain consecutive images of the desired scene with know relative position between the two cameras which form a stereo setup. This is a technique to acquire the depth information one might be familiar with from 3D movies, by empathizing the stereo vision capabilities of humans. This can also be done with a single moving camera, given its absolute camera poses in a common reference frame. Such a setting basically describes the stereo vision case, observing the scene from distinct camera positions, where two consecutive and suitable poses represent a virtual stereo camera setup.

As an example of stereo images, we take a look at the old stereogram in Figure 3.2(a). The scene in the stereo images is taken from different points of view, such that when one uses special equipment in order to view each part of the image with only one eye, the scene will be seen in 3D. This is the same way humans see the world, with the two eyes forming a stereo system.

(a) Stereogram as two single stereo images

(b) Stereogram with overlaid left and right images from (a)

Figure 3.2: Stereogram: Pictures are taken with a stereo camera from slightly different camera poses; When looked at with a stereoscope the 3D structure becomes visible; Adopted from [112].

The left and right view can be overlaid on each other (compare Figure 3.2(b)). This is what is usually done for 3D movies in cinemas. A scene is first recorded with a stereo camera. Then the synchronized images are projected simultaneously onto the screen, each with a different polarization (other methods exist). These polarized overlaid images are filtered by 3D glasses (compare Figure 3.3), splitting the left and right view for the eyes. Thus, the observer can see the scene in 3D.

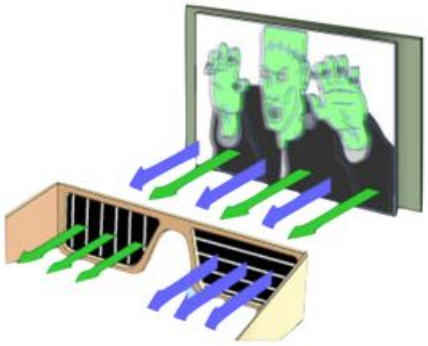


Figure 3.3: Polarizing 3D glasses: The stereo images are projected onto the screen with differently polarized light; The polarizing filters of the 3D glasses split the simultaniously projected images again for the left and right eye, respectively; Adopted from [35].

The human brain is capable of reconstructing the 3D scene from binocular vision directly. But how can an algorithm extract the depth information from stereo images and reconstruct the scene.

We distinguish between sparse and dense methods for reconstruction. For sparse reconstruction, given camera poses of a stereo system and sparse point correspondences, it is applicable to compute the point in 3D as the intersection of the back projected rays (see 4.3.2). For dense reconstruction purposes this is not reasonable, as will be discussed later.

The basic idea of dense depth computation from 2D images, is to compute the difference in pixel coordinates for known point correspondences in two views (see section 4.3.3). This is best explained by taking a look on the observed scene through the window of a moving train in Figure 3.4. Objects in the background stay at the same pixel coordinate as the picture is taken. Whereas close objects move along the x-direction during the exposure time, resulting in blur. This behavior of pixel displacement in one direction leads to the useful property of disparity. When observing a scene with a stereo system, distant objects will have the same pixel coordinate in both images. The closer an object, the larger the disparity will be, since the angle under which the object is observed from each camera will increase, resulting in different pixel coordinates in each image. With known disparity for a certain 2D point correspondence, the depth value can be computed. Also, take a look at the comic in Figure 3.5 for a humorous illustration of points at infinity.

In the following a short outline of the proposed algorithm is described to give the

reader a first impression of the methods and implementation: Before extracting dense depth information from stereo images, further preprocessing as well as the estimation of camera poses is necessary. Therefore, a feature based procedure for finding sparse point correspondences in stereo images will be discussed in section 4.1. With these point correspondences, which describe the general rigid displacement of the scene, the epipolar geometry can be estimated. The epipolar geometry describes the relation between image points of a stereo system (see section 4.2), from which the relative transformation between both cameras can be estimated up to scale (see section 4.2.3). Now, with known camera poses and sparse point correspondences, sparse 3D points can be triangulated as an initial sparse reconstruction in a common world reference frame (see section 4.3.2). With known 3D-2D sparse point correspondences the subsequent camera positions of the freely moving monocular camera can be computed in relation to the world reference frame (see section 4.3.1). Furthermore, with known camera poses, each stereo image pair can be rectified (see section 4.2.4) for dense 2D point correspondence and disparity computation (see 4.3.3). From the disparity, the 3D structure can be reconstructed (see section 4.3.4). Since the algorithm should work in non-rigid environments, sophisticated outlier removal and regularization needs to be considered in order to obtain robust camera poses.

Figure 3.4: View through a window of a moving train: Distant objects in the background keep the pixel position, whereas close objects appear blurred due to large movement; this relates to the disparity of the point and thus their depth; Adopted from [138].

The projected 3D point clouds from different views can now be integrated and fused

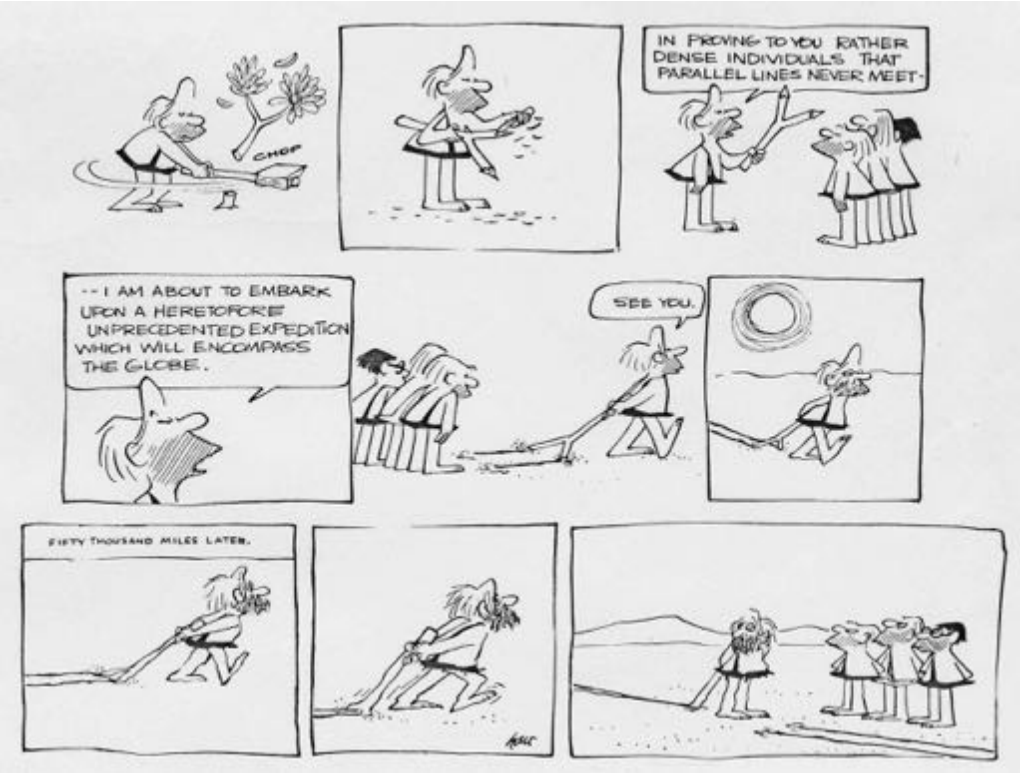


Figure 3.5: Comic by Johnny Hart[4], Parallel lines; Adopted from [95].

together to obtain a dense model (see section 4.4). For possible rigid transformations of the point clouds, classical ICP methods can be used. For the reconstruction of the non-rigid deformations, an embedded deformation graph based on sparse correspondences is introduced, since fully dense computation would be computational expensive (see section 4.4.3). Correspondences are established by an optical flow based feature matching (see section 4.4.2). For the alignment of non-rigidly deforming point clouds, the ARAP approach, together with sufficient regularization is taken into account (see section 4.5) in order to solve the non-linear optimization problem with the publicly available Ceres[3]solver.

Sparse correspondences with small local non-rigid transformations are the basis for the deformation nodes in the deformation graph. The point clouds are continuously integrated into the canonical volumetric model, represented as TSDF (see 4.4.6). Cor-

[3]Ceres Solver is an open source solver for non linear opimization problems developed by Google Inc. [2]

[4]*Johnny Hart*, US cartoonist, *Endicott, New York 18.02.1931, †Nineveh, New York, 07.04.2007 [67]

respondences between the canonical model and the deformation nodes are established and the whole model is deformed via DQB according to the deformation nodes (see section 4.4.5). DQB is a sophisticated method for interpolating unknown points between points with known transformations (transformations with rotation and translation). The following descriptions on important preliminaries, as well as some algorithms and mathematical background in section 4 are partly based on the text book of Hartley and Zisserman [53] which is also the basis for many algorithms of the well known computer vision library *OpenCV* [105], many of which are also used for the implementation of this work.

3.2 Image acquisition

In analogy to how the cyclops in Figure 3.1 sees the world, we will discuss the principles on the transformation of a 3D scene into a 2D image of a monocular camera. The basic idea of a camera is to project a 3D world onto a virtual 2D image plane, thus describing a mapping [53]:

$$(X, Y, Z)^T \mapsto (x, y)^T$$

A common camera model is the *pinhole camera* (compare 3.2.1). This section will cover how such a camera model forms an image, how 3D world points and points in an image are related and how they can be obtained.

3.2.1 Pinhole camera model

The earliest and simplest idea of a photographic camera is the *camera obscura* (see Figure 3.6). This model was first described by *Ibn al-Haitham*[5] in the 11th century and later studied by *Leonardo da Vinci*[6]. Light passes through the small hole in the front and generates an upside-down image of the scene [15]. It is the basis for the more sophisticated geometrical *pinhole camera* description.

The commonly used *pinhole camera* model (compare Figure 3.7) is a formal representation of Figure 3.6. Instead of a simple hole, a lens system is located in front of the camera center and the upside-down image is registered by a camera sensor. The obtained image looks as it was projected onto the virtual image plane, which lays in front of the camera center with a distance equal to the focal length $z = f$. Therefore, the pinhole camera model is also referred to as projective camera model.

The projective pinhole camera model describes the central projection of a world point $\mathbf{X}$ onto a virtual plane at $z = f$ with its principal point p in front of the camera center O, called (virtual) image plane or focal plane. f is the distance between the camera center or projection center O and the image plane [53].

[5] *Abu Ali al-Hasan Ibn al-Hasan Ibn al-Haitham Haitham*, Arabian Physicist and Mathematician, *Basra 965, †Kairo arround 1040, [16]

[6] *Leonardo da Vinci*, Artist, Researcher, Engineer, *Anchiano near Vinci (Near Florenz) 15.04.1452, †Palace Cloux (Today Clos-Lucé, Amboise) 2.05.1519 [17]

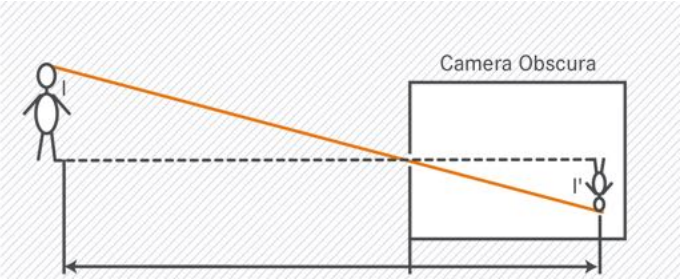


Figure 3.6: Schematic represenation of the *camera obscura*: as already described by *Ibn al-Haitham* in the 11th century and later by *Leonardo da Vinci* around 1500 A.D.; Adopted from [15].

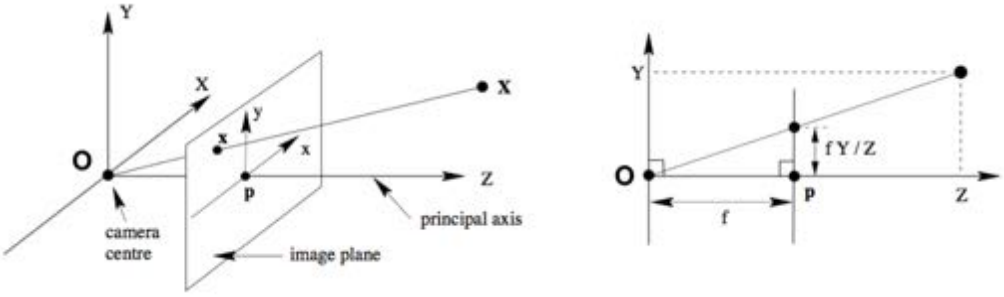


Figure 3.7: Pinhole camera model: A 3D world point **X** is projected onto the image plane as image point **x**; The principal axis of the camera, originating at the camera center O facing towards the image plane, intersects the image plane at the principal point p at the distance of the focal length f; Adapted from [53].

3.2.2 Projective geometry

When projectively transforming a world point to an image, homogeneous coordinates are commonly used. They were already introduced by *August Ferdinand Möbius*[7] at the end of the 19th century. Some point **x** on a plane can be represented with cartesian coordinates $(x, y) \in \mathbb{R}^2$. If we consider $\mathbb{R}^2$ as vector space, then (x, y) describes a vector. Some line in the plane can be represented as $ax + by + c = 0 = (ka)x + (kb)y + (kc)$ for some non-zero k. Thus, $(a, b, c)^T$ and $k(a, b, c)^T$ are equivalent with $\mathbf{x} = \{k\mathbf{x} | k \in \mathbb{R} \setminus \{0\}\}$. Vectors with such an equivalent relationship are called homogeneous vector. All such vectors (except $(0, 0, 0)^T$) in $\mathbb{R}^2$ form the projective space $\mathbb{P}^2$. Considering

[7] *August Ferdinand Möbius*, German Mathematician, most recognized for his work on projective geometry and homogenous coordinates (*Der barycentrische Calcül* [89]), *Schulpforta, Germany 17.11.1790, †Leipzig, Germany 26.09.1868 [20]

$(kx, ky, k)^T$ with some k as representation for a point $(x, y)^T$, then any point in $\mathbb{R}^2$ can be represented with homogeneous coordinates in $\mathbb{P}^2$, whereas some homogeneous point $(x_1, y_1, k)^T$ describes the point $(x_1/k, y_1/k)^T \in \mathbb{R}^2$.
Similar to above, points in $\mathbb{R}^3$ can also be represented with homogeneous coordinates in $\mathbb{P}^3$, where some 3D point $\mathbf{X} = (X, Y, Z)^T \in \mathbb{R}^3$ can be represented with $\mathbf{X} = (X_1, Y_1, Z_1, W_1)^T \in \mathbb{P}^3$ whereas the inhomogeneous coordinates are: $X = X_1/W_1$, $Y = Y_1/W_1$, $Z = Z_1/W_1$ [53].
Describing the mapping from 3D to 2D more formally for a projective camera model, a world point $\mathbf{X} = (X, Y, Z)^T$ is mapped to an image point $\mathbf{x} = (fX/Z, fY/Z)^T$, or:

$$(X, Y, Z)^T \mapsto (fX/Z, fY/Z)^T$$

which describes a central projection mapping from $\mathbb{R}^3$ to $\mathbb{R}^2$ [53].
In the following, a definition of the projective camera P will be given. We will see it consists of intrinsic parameters K and extrinsic parameters, namely rotation and translation.
Using homogeneous coordinates, the central projection can be represented as a linear mapping:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

With $\mathbf{X} = (X, Y, Z, 1)^T$ as a homogenized representation for the world point, $\mathbf{x}$ a 3-vector homogenized to $x_3 = Z$ for the image point, P a 3×4 camera projection matrix and $[I \mid \mathbf{0}]$ describing a 3×3 identity matrix with an appended fourth column with zeros,

$$P = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} = \mathrm{diag}(f, f, 1)[I \mid \mathbf{0}]$$

we get:

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

Since the principal point not necessarily equals the origin of coordinates in the image plane, the offset needs to be taken into account, resulting in:

$$(X,Y,Z)^T \mapsto \left(\frac{fX}{Z}+p_x, \frac{fY}{Z}+p_y\right)^T \tag{3.1}$$

with $\left(p_x, p_y\right)^T$ as the coordinates of the principal point, thus homogenized with $x_3 = Z$ yields:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX+p_x \\ fY+p_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Now, with

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

as the camera calibration matrix[53]. Hence,

$$\mathbf{x} = K\,[I \mid \mathbf{0}]\mathbf{X}_{cam} \tag{3.2}$$

The world coordinate frame and the camera frame are related by a rotation and translation. If $\tilde{\mathbf{X}} \in \mathbb{R}^3$ represents the world point and $\tilde{\mathbf{X}}_{cam} \in \mathbb{R}^3$ the same point in the camera frame, then:

$$\tilde{\mathbf{X}}_{cam} = R\left(\tilde{\mathbf{X}} - \tilde{O}\right)$$

with $\tilde{O}$ as the camera center. Or,

$$\tilde{\mathbf{X}}_{cam} = \begin{bmatrix} R & -R\tilde{O} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{O} \\ 0 & 1 \end{bmatrix} \mathbf{X}$$

Together with equation (3.2) we get [53]:

$$\mathbf{x} = KR\left[I \mid -\tilde{O}\right]\mathbf{X}$$

Thus, the general pinhole camera $P = KR\left[I \mid -\tilde{O}\right]$ has 9 DOF: 3 for the intrinsic camera parameters K with f ,p_x and p_y, and 3 both for R and $\tilde{O}$ as the extrinsic camera parameters, describing the camera orientation and position relative to the world frame. While nowadays almost all sensors have square pixels, we still account for different pixel width and height, leading to 10 DOF [53]:

$$K = \begin{bmatrix} f_x & & p_x \\ & f_y & p_y \\ & & 1 \end{bmatrix} \tag{3.3}$$

where

- f_x is the focal length measured in the width of a pixel
- f_y is the focal length measured in the height of a pixel
- p_x, p_y are the coordinates of the principal point

One can also write: $\tilde{\mathbf{X}}_{cam} = R\tilde{O} + t$, thus:

$$P = K\left[R \mid t\right]$$

with $t = -R\tilde{O}$ [53].

3.3 Stereopsis

Figure 3.8: Comic by Johnny Hart, "THE LAW OF PERSPECTIVE": The scale ambiguity which arises from vision based depth computation, especially for monocular approaches, can be anticipated by this illustration; From the sole image information one can not distinctively predict the absolute scale of objects in the scene; Adopted from [155].

Stereopsis describes the perception of depth with visual information obtained from two distinct positions of the same scene, usually with binocular vision as e.g. humans do [21]. Extracting the 3D structure of a scene and computing the camera motion within it, go hand in hand with each other [132].

A stereo camera system basically mimics the binocular vision of humans, where the two eyes represent a stereo configuration (compare Figure 3.9).

As depicted in the comic sketch by *Johnny Hart* (compare Figure 3.8), perspective vision and extracting the correct depth information can be challenging, especially when using monocular vision. The scale ambiguity can be both beneficial since monocular vision is applicable in many environments and crucial by adding uncertainty. Classical stereo vision limits the complexity, since the base line, or distance between the two camera centers, is known.

Marr [85] argued that a primitive way of describing vision is "*[...] to know what is where [...]*". Taking his idea of visual perception, the methods to extract the three-dimensional information of a scene with a moving monocular camera and how to represent it, will be explained. We need to know the camera poses and thereafter compute the positions of all points in 3D for dense reconstruction. In this section we will first discuss how monocular cameras and stereo camera configurations are related. Simplified introductory examples for depth computation and arising challenges will be considered subsequently.

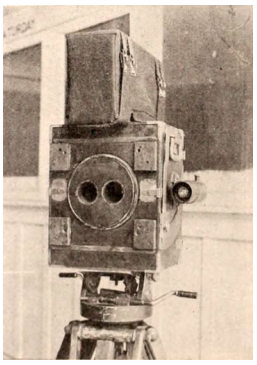


Figure 3.9: Stereoscopic camera: Stereo camera from the first 3D movie "The Power of Love", 1922 [160]; Adopted from [38].

3.3.1 From monocular to stereo

A stereo system consists of two cameras, transformed by a rotation and translation. In an ideal case (compare Figure 3.10) the cameras are only transformed by a translation in x-direction, facing parallel aligned in the same direction (translation in y-direction would lead to vertical stereo). A moving monocular camera can represent such a system, if the transformation between two camera poses is known. Therefore, the general case will be discussed.

Two distinct cameras in the world frame are related by a rigid transform $G \in SE(3)$ of the Euclidean group, which consists of a global rotation $R \in SO(3)$ of the special orthogonal group[8]and translation $t \in \mathbb{R}^3$ in 3D (compare Figure 3.11):

$$G = \begin{pmatrix} R & t \\ 0^T & 1 \end{pmatrix}$$

R can be defined by a 3D Rotation Matrix (see below), and the 3D translation as $t = (t_X, t_Y, t_Z)^T$. We can now describe how two cameras, or, as in our case, one camera at two distinct points of view or poses, are related.

[8]For a valid rotation in $SO(3)$ the orthogonality must hold: $\{R \in \mathbb{R}^{3\times3} | R^T R = I, \det(R) = 1\}$ [145]

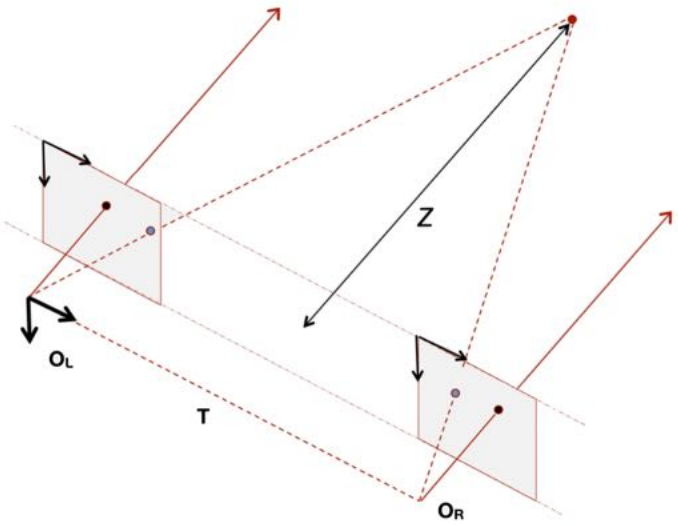


Figure 3.10: Canonical stereo camera setup: Two cameras are frontal parallel aligned and displaced by the baseline T in only one direction (which equals a translation t with $t_Y, t_Z = 0$) without any rotation; Adopted from [76].

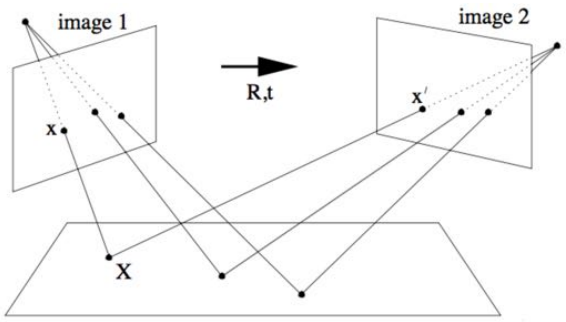


Figure 3.11: Projective transformation with rotation R and translation t between two distinct points of view; The point $\mathbf{X}$ is represented as $\mathbf{x}$ and $\mathbf{x}'$ in both images, respectively; With known R and t this setup describes an oversymplified stereo system; Adopted from [53].

As an introductary example for transformations, consider the following affine transformations, which describe a rotation by θ around the z-axis without translation, by ψ around the X-axis and by φ around the Y-axis, respectively [28, 53]:

$$G_Z(\theta) = \begin{pmatrix} R_z & t \\ 0^T & 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$G_X(\psi) = \begin{pmatrix} R_x & t \\ 0^T & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi & 0 \\ 0 & \sin\psi & \cos\psi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$G_Y(\varphi) = \begin{pmatrix} R_y & t \\ 0^T & 1 \end{pmatrix} = \begin{pmatrix} \cos\varphi & 0 & \sin\varphi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\varphi & 0 & \cos\varphi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Now, all above rotations in 3D can be represented by a sequence of individual transformations [28]: $G = G_X(\psi)\, G_Y(\varphi)\, G_Z(\theta)$.

An arbitrary rotation can be described by the composition of rotations about three axes (Euler's rotation theorem). Hence, it can be represented by a 3×3 matrix [28, 145]:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Translation t by $t = (t_X, t_Y, t_Z)^T$ is described by:

$$x' = x + t_X$$

$$y' = y + t_Y$$

$$z' = z + t_Z$$

To describe a general displacement of a point in 3D by rotations and translations with a product of matrices, homogeneous coordinates can be used. By extending the general solution for rotations from above and stacking the translation vector, results in a homogeneous $4x4$ matrix describing a generalized displacement [28]:

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} R_{11} & R_{12} & R_{13} & t_X \\ R_{21} & R_{22} & R_{23} & t_Y \\ R_{31} & R_{32} & R_{33} & t_Z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

We want to justify R is a rotation by definition and thus $R \in SO(3)$. For simplicity we will denote the original point (x, y, z) with $\mathbf{x}$ subscript i, with $i = 1, 2, 3$ for each element and analogous for (x', y', z'). Taking a closer look at the two conditions for the rotation matrix R, we first wish to obtain an orthogonal transformation[9] $R : \mathbb{R}^3 \mapsto \mathbb{R}^3$. Therefore, the length of a vector needs to be identical after the rotation as is given by any matrix $\in SO(3)$, thus:

$$\sum_i \mathbf{x}_i' \mathbf{x}_i' = \sum_i \mathbf{x}_i \mathbf{x}_i$$

leading to:

$$\sum_i \left(R_{ij}\mathbf{x}_j\right)\left(R_{ik}\mathbf{x}_k\right) = \sum_i \mathbf{x}_i \mathbf{x}_i$$

We can rewrite this as:

$$\begin{aligned} \sum_i R_{ij}\left(\mathbf{x}_j R_{ik}\right)\mathbf{x}_k &= \sum_i R_{ij}\left(R_{ik}\mathbf{x}_j\right)\mathbf{x}_k \\ &= \sum_i R_{ij} R_{ik} \mathbf{x}_j \mathbf{x}_k \\ &= \sum_i \mathbf{x}_i \mathbf{x}_i \end{aligned}$$

and introducing the Kronocker delta δ_{jk} in order the equation above holds:

$$R_{ij} R_{ik} = \delta_{jk}$$

for $j, k = 1, 2, 3$ and

$$\delta_{jk} = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j \end{cases}$$

we get the orthogonality condition which assures $R^T = R^{-1}$ and $R^T R = I$ [145].
Further, the eigenvalues of an orthogonal rotation matrix need to satisfy the condition:

[9] Preserving length of vectors and angles between them [145]

One of the eigenvalues is 1, whereas the other two are complex conjugates of the form $e^{i\theta}$ and $e^{-i\theta}$

Or one of the special cases:

- All eigenvalues are 1
- One of the eigenvalues is 1, whereas the other two are -1

Additionally, to represent pure rotation by an orthogonal matrix, the determinant of the rotation matrix has to equal 1: $\det(R) = 1$ since otherwise also reflections would be possible.
To sum this up, we get the following conditions for a valid rotation matrix R [145]:

$$R \quad \text{is} \quad \text{a} \quad \text{rotation} \in SO(3) \Leftrightarrow \left(R^T = R^{-1}\right) \wedge (\det(R) = 1)$$

With known rotation and translation the camera pose can be defined within a world reference frame. Now a single moving camera can be described as a stereo system with known transformation between two distinct camera poses. This is a necessary basis for the following sections.

3.3.2 Depth computation

Figure 3.12: Land measurement: Ancient example of land measurement; Copper engraving from 1607; Adopted from [118].

The idea of calculating the distance between some point and the point of view goes back a long way in land measurement (compare Figure 3.12). Triangulation is used to compute the distance to a certain point by knowing the baseline between two points of view and the angle in which the point can be seen. We first look at the simple and ideal case for simplification. We will see later that real world scenarios are more complex and require further prior computation. These methods will be discussed subsequently. Later, important relations between stereo images described by the epipolar geometry will help us to compute dense reconstructions (see sections 4.2 and 4.3).

In Figure 3.13 a simplified geometry is illustrated. Let's take this as an introductory example for stereo depth computation. A world point $\mathbf{X}$ can be seen on two distinct image frames as $\mathbf{x}$ and $\mathbf{x}'$, with camera centers O_L and O_R, baseline b between them, and focal length f. By locating the origin of each image coordinate center at the base point, the difference of the relative position of the intersecting rays of $\mathbf{x}$ and $\mathbf{x}'$ and the base point, namely d and d', are given by their x-coordinates. Thus, the disparity is given by: $\mathbf{d} = \mathbf{x} - \mathbf{x}'$. We can observe two similar triangles $O_L O_R \mathbf{X}$ and $\mathbf{x}\mathbf{x}'\mathbf{X}$, thus [28,

96]:

$$\frac{z}{b} = \frac{z+f}{T+\mathbf{d}}$$
$$z = \frac{Tf}{\mathbf{d}} \tag{3.4}$$

Note the relationships between the disparity and other entities. Disparity is inversely proportional to the depth, leading to low disparities for distant points and high disparities for close points. Furthermore, it is proportional to the base line, thus resulting in higher disparity for far-off camera centers [96]. The uncertainty of a reconstructed point is depicted in Figure 3.14. The smaller the baseline or more precisely the smaller the angle between the two points of view, the higher the resulting uncertainty of the reconstruction will be [28]. Thus, a certain minimum offset between both images is needed for providing reliable depth information.
These principles will be discussed in more detail in section 4.3 for sparse and dense 3D reconstruction.

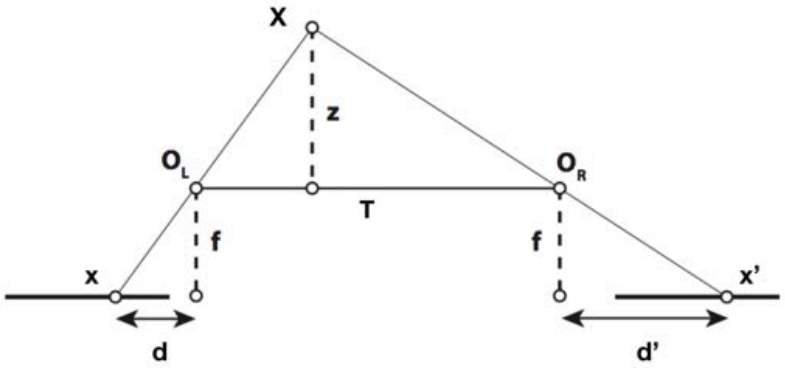
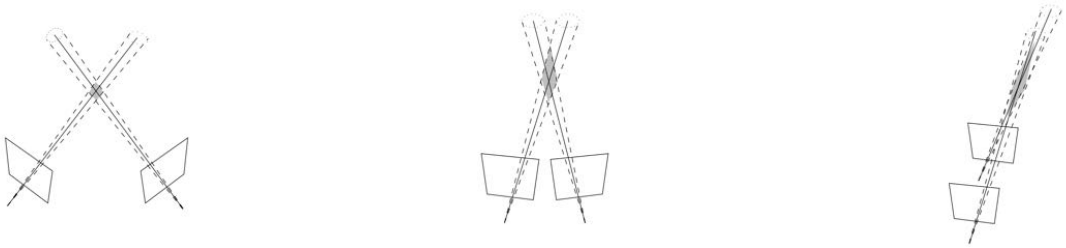


Figure 3.13: Simplified depth computation: Similar triangles can be observed between the world point **X**, the two camera centers and the image points; d and d' describe the disparity; From the similar triangle relations the depth can be calculated; Adopted from [22].

3.3.3 Motion parallax

As a simple introductory problem of computing depth information from two distinct images we take a brief look at the motion parallax (compare Figure 3.15). Here, two neighboring points in one image $\mathbf{x}'_1$ and $\mathbf{x}'_2$ can not be distinguished in the other view.



(a) Sufficient baseline (b) Small baseline (c) Inadequate baseline

Figure 3.14: Uncertainty of depth: Illustration of the uncertainty of depth computation in relation to camera pose; With sufficient baseline 3.14(a) the depth uncertainty is small; For poor stereo camera configurations, the uncertaintly gets very large; Adopted from [53].

They reside on the ray which goes through the principal point or camera center and the image point. Without specific point correspondences, depth information can not be reliably obtained [53]. As discussed later, this is one of the reasons why triangulation is not suitable for dense reconstruction (see 4.3.2). This example also illustrates the importance of distinguishable image content. Without being able to find corresponding points in the stereo images based on their pixel description (e.g. pixel intensities of a small patch in the image or image features), the reconstruction cannot be determined.

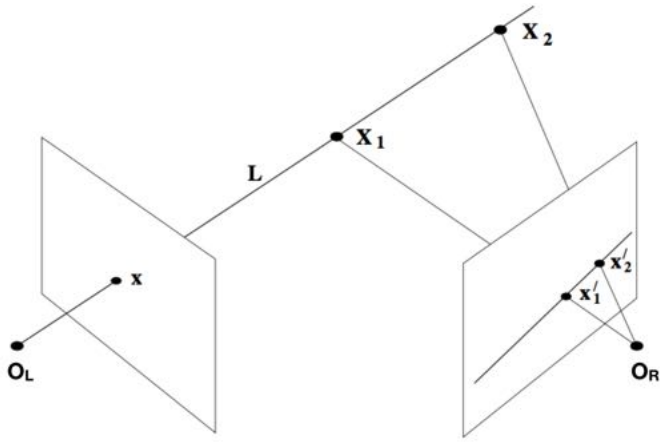


Figure 3.15: Motion parallax: Illustration of the motion parallax; Depending on the camera poses - especially with large rotations - point correspondences are difficult to estimate; Adopted from [53].

4 Methods and implementation

As in Figure 4.1 illustrated, we will now take a closer look on the methods proposed for this thesis and how they are implemented.

First, the extraction and matching of visual information based on feature points will be explained (see section 4.1). The epipolar geometry, presented in section 4.2, describes the relation between corresponding visual information in a stereo system. This relation can be used to estimate two essential unknowns needed for reconstruction: the relative transformation between camera poses up to scale, and aid to easily determine stereo point correspondences. With known stereo point correspondences - which point in the second image corresponds to a certain point in the first one - their position in 3D can be computed. As for a moving monocular camera, the camera pose needs to be known within the global reference frame. Methods to compute robust camera poses and how dense reconstruction is performed will be discussed in section 4.3. In the next sections (4.4, 4.5), techniques to align reconstructions from different points of view and how potential non-rigid deformations can be accounted for and integrated efficiently into a common reconstruction model will be detailed.

After proposed methods have been explained, an overview of the tracking and reconstruction pipeline presented here will be discussed (see section 4.23). Further details and implementation considerations will be described for all important functional entities. A workflow diagram of the algorithm is presented to clarify the sequence of individual procedures.

An algorithm for tracking and reconstruction in non-rigid environments with a monocular camera is proposed. We will see how suggested methods enable to extract robust sparse feature points for reliable camera pose tracking. An extensive outlier removal and sophisticated feature matching is proposed accounting for filtering out non-rigid transformations in the scene. By robust camera tracking we can set up a virtual stereo camera system by consecutive key frames. With known camera poses, depth information can be computed by stereo comparisons between key frames, and fully dense disparity maps are obtained as suggested here. Based on an optical flow based feature matching scheme, a novel explicit correspondence association for the constitution of an embedded deformation graph for subsequent deformation of the reconstruction via DQB is introduced.

If you think it's simple,
then you have misunderstood the problem.
(Bjarne Stroustrup[1])

Figure 4.1: Star Wars: The Empire Strikes Back in 1981: An operator prepares the model of an AT-AT; Taken from [47].

[1] *Bjarne Stroustrup*, Professor for Computer Science, famous for the development of C++, *Aarhus, Denmark 30.12.1950 [128]

4.1 Extracting visual image information

Being able to extract visual information from images is a fundamental task in computer vision. For many applications it is necessary to identify characteristic patches or regions in the image. Especially in the scenario presented in this work, it is essential to identify such image regions and match them e.g. in a stereo image pair, for estimating the epipolar geometry (see 4.2) and computing the 3D structure of the scene (see 4.3.3). For humans this is an easy task. One would simply find a distinctive part of an image and describe its appearance, as for example the eye or the paw of the squirrel in Figure 4.2.

Figure 4.2: Picture with characteristic image regions: The highlighted areas describe image patches with characteristic visual information; Such areas can be the basis for feature detectors and descriptors; Adopted from [105].

Probably the most obvious way an algorithm can find such information in an image is to detect edges or corners, since they usually have a large variation in intensity [50]. Detecting such areas is referred to as feature detection. Describing the detected feature point is called feature description. Now, the feature descriptions around detected feature points with interesting and distinctive image information can be matched throughout different images.

To improve the amount of features, its stability, performance and robustness for matching, more sophisticated methods instead of solely relying on corners or edges have been proposed. One of such feature methods is the oriented FAST and rotated BRIEF

(ORB) feature detector, which is also used in the implementation of this work (see 4.1.1).

4.1.1 ORB features

ORB is a feature detector introduced by Rublee et al. [115] with the aim to provide an efficient open-source alternative to SIFT [81] or SURF [5]. It is based on the FAST [114] keypoint detector and the visual descriptor BRIEF [24] (Binary Robust Independent Elementary Features).
For ORB, the FAST keypoint detector is modified such that it accounts for some shortcomings, resulting in oriented FAST (oFAST). A Harris corner detection [50] is introduced for quality improvement, a scale pyramid as for SIFT enables scale invariance and the calculation of an intensity centroid provides rotational invariance. Since BRIEF descriptors work poorly with rotations, the rotation-aware BRIEF (rBRIEF) has been proposed. The computed BRIEF descriptors are "steered" according to the orientation of the keypoints. ORB features can be computed, processed and matched fast due to its binary representation, while accounting for rotation and scale invariance [115].
Since SIFT and SURF are patented, ORB has the major advantage of being publicly available and free of charge [105]. ORB is also the basis of the well known and robust ORB-SLAM approach of Mur-Artal et al. [93].

4.1.2 Feature matching

Feature matching is the task of finding correct correspondences of feature points, characterized by their feature descriptors, in different images. Since here ORB features are used, binary matching methods will be discussed, mostly ignoring representatives of vector features such as SIFT [81] or SURF [5]. In general, binary feature matching can be performed quite fast, since it relies on the Hamming distance[2]. The binary Hamming distance calculation can efficiently be done by bitwise XOR operation followed by a bit count [54]. Improving the matching of vector features is often done by the well established ratio test [81], comparing the best and second best match, and with efficient approximate nearest neighbor searches based on FLANN, finding matches using the L2

[2]The Hamming distance equals the number of non-identical symbols within two strings of the same length [46]

norm as distance measure [91]. Such procedures are not suitable for binary descriptors (L2 norm compared to Hamming distance). By introducing locality sensitive hashing (LSH), FLANN based approximate nearest neighbor search is achievable for binary descriptors. Given some distance measure for the similarity of two descriptors, the idea of LSH involves that near objects will have a high probability to be hashed to similar values, collected in hash buckets [48]. Lv et al. [152] proposed a method to build up a fast search structure with LSH suitable for fast approximate nearest neighbor search with FLANN [92].

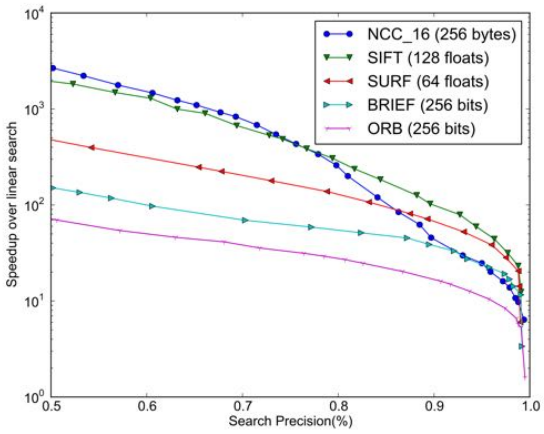


Figure 4.3: Feature matching speedup: Speedup of feature matching with LSH and FLANN in comparison to linear matching for different feature descriptors; While suffering from precision loss, speedups for all feature descriptors can be achieved; Note that speedups for ORB features are the lowest, and in particular not noteworthy for accurate matchings; Adopted from [92].

However, taking the computational effort for indexing and sorting the features with LSH and building a FLANN based search tree into account, such a procedure is only desirable for very large datasets. For matching of a single stereo image one might want to consider this. Also compare Figure 4.3 for the experimental evaluation of runtime for LSH methods compared to linear search. In this particular case a variation of LSH, namely hierarchical clustering algorithm was used for ORB, since it showed the best performance [92]. Please note that the presented speedups were obtained for a very large dataset with 80 million images. The speedup for ORB features is the lowest for all considered features, while suffering from lower matching accuracy since only approximate neighbors might be found.

To sum this up, there is a trade off between runtime and accuracy, whereas for very large datasets clearly runtime is in focus but for stereo matching the accuracy is crucial and speedups will be low compared to e.g. brute force matching. Also note, that ORB features are the fastest features to compute and match among the commonly used [107]. This behavior also conforms with evaluations of feature matching procedures done for this work. Speedups are only minor for stereo matching compared to brute-force and matching accuracy suffers.
Accuracy of feature matching is vital for subsequent computation in the algorithm presented here. Therefore, matching is performed in a left-right consistency fashion. All feature points in the first image are matched with features in the other image and vice versa. Only matches which occur in both matching sets are considered as inliers. Furthermore, matches which will be identified as outliers during the computation of the epipolar geometry (see section 4.2) will be removed. The remaining matches can be used for a sparse reconstruction of the scene by triangulation (see section 4.3.2). Feature points with too high reprojection errors will also be considered as outliers and removed from the set of feature matches. Employing this extensive procedure for outlier removal based on several error measures, we receive a robust set of feature matches from which the rigid camera pose transformation between the two frames can be estimated (see sections 4.2.3 and 4.3.1).

4.2 Epipolar geometry

Until now, we have only considered the case of simplified geometries. Further preprocessing is necessary to enable the reconstruction of dense depth information from 2D stereo images. The epipolar geometry explains the relationship between corresponding points in stereo images (see sections 4.2.1 and 4.2.2), and in particular it describes the relative displacement between the stereo cameras (the relative camera pose can be estimated from the epipolar geometry up to scale: see section 4.2.3), which is essential for computing 3D reconstructions from stereo images. It is difficult to compute the relative camera position of a stereo system, since the general displacement of the camera is not constant. Always two consecutive keyframes are considered as a stereo system. For computing the depth information, it is essential to find stereo correspondences between the stereo images while estimating the camera poses. The process of finding those correspondences and utilizing them to compute their 3D depth is referred to as stereo matching [132]. Epipolar geometry comes in handy for finding these associations. The epipolar geometry between two images is the intersection between the image frames and the pencil of all planes with a common baseline [53]. It describes the projective geometry of these two images, while it is only dependent on the intrinsic camera parameters and the relative position of both views. Given a world Point $\mathbf{X}$ and the two image points $\mathbf{x}$ and $\mathbf{x}'$, the homogeneous 3×3 fundamental matrix F of rank 2 (see section 4.2.1) describes this geometry, while satisfying $\mathbf{x}'^T F \mathbf{x} = 0$ (epipolar constraint) [53].

As shown in Figure 4.4, the world point, both camera centers and both image points are coplanar (plane denoted as π). The back-projected coplanar rays of the image points in π intersect in $\mathbf{X}$.

If we have the image point $\mathbf{x}$ in the first view, how can we match the corresponding point $\mathbf{x}'$ in the second view (compare Figure 4.5)? The epipolar plane π is defined by the baseline and the ray through $\mathbf{x}$. Further, $\mathbf{x}'$ lies in π. $\mathbf{x}'$ lies on the epipolar line l' of x, defined by the intersection of π with the image plane, through the epipole e', which describes the intersection of the image plane and the baseline. The epipolar line l', is the back-projected ray through $\mathbf{x}$ in the second image. Thus, the search space for $\mathbf{x}'$ is reduced to be 1-dimensional along the epipolar line l' [53]. Compare Figure 4.6 for such a case with matched feature points and their respective corresponding epipolar lines for an undistorted stereo image pair.

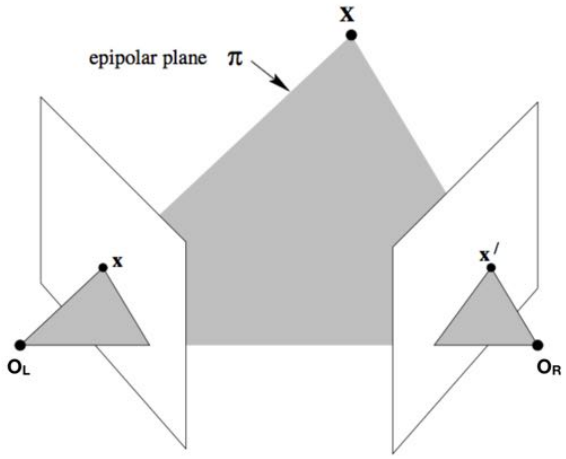


Figure 4.4: Epipolar geometry and intersecting epipolar plane: The camera centers, two corresponding image points and the associated epipoles define a plane; The intersection of the epipolar plane with the image planes define the corresponding opposite epipolar line; The 3D world point resides on the epipolar plane at the intersection of the rays originating at the camera centers passing through the image points, respectively; Adopted from [53].

4.2.1 Fundamental matrix

The described relation from above can be formulated as a mapping from a point $\mathbf{x}$ in one image to the epipolar line l' in the second $\mathbf{x} \mapsto l'$. This projective mapping from points to lines is a (singular) correlation, represented by the fundamental matrix F. The fundamental matrix encodes the epipolar geometry between two views. The fundamental matrix F can be derived algebraically, according to the notation of Zhang [156]. Given an image point $\mathbf{x}$, the set of possible world points lies on the back projected ray. The ray can be represented by a line through two points in 3D, the camera center O, where $PO = 0$ and the point P^+x, where P^+ is the pseudo-inverse of P. More precisely: $P^+ = P^T \left(PP^T\right)^{-1}$, for which $PP^T = I$. The point $P^+\mathbf{x}$ lies on the ray, since it projects to $\mathbf{x}$, with $P\left(P^+\mathbf{x}\right) = I\mathbf{x} = \mathbf{x}$. Therefore, the ray can be formed by the line between those two points [53]

$$\mathbf{X}(\lambda) = P^+\mathbf{x} + \lambda O$$

The scalar λ specifies the two particular points on the ray, $P^+\mathbf{x}$ at $\lambda = 0$ and the camera center O at $\lambda = \infty$. These points are described in the second camera P'

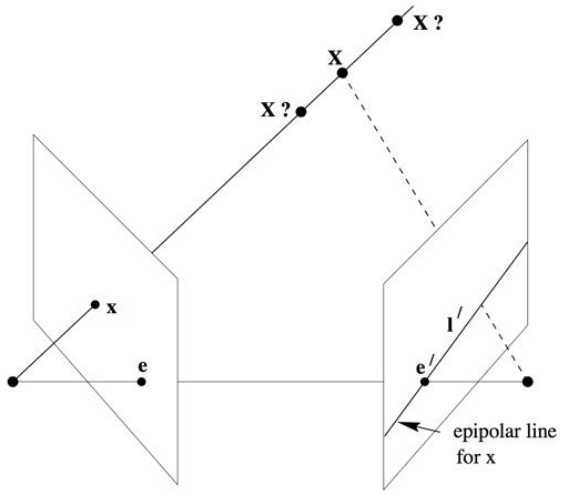


Figure 4.5: Epipolar geometry and point correspondences along the epipolar line: Without known correspondences, the intersection of the rays (as in Figure 4.4) can not easily be determined; The correspondence search space is restricted to 1D along the epipolar line; Adopted from [53].

as $P'P^{+}\mathbf{x}$ and $P'O$. The line connecting the two projected points is the epipolar line $l' = (P'O) \times (P'P^{+}\mathbf{x})$. The point $P'O$ (projection of the first camera center) is the epipole e' in the second image. Leading to:

$$l' = \left[e'\right]_x \left(P'P^{+}\right)\mathbf{x} = F\mathbf{x}$$

with F as the fundamental matrix:

$$F = \left[e'\right]_x \left(P'P^{+}\right)$$

and the skew-symmetric matrix $[e']_\times = \begin{bmatrix} 0 & -e'_3 & e'_2 \\ e'_3 & 0 & -e'_1 \\ -e'_2 & e'_1 & 0 \end{bmatrix}$ of $e' = \left(e'_1, e'_2, e'_3\right)^T$.

Consider the ideal calibrated stereo system where the first camera describes the world origin, with cameras:

$$P = K\left[I \mid \mathbf{0}\right]$$

$$P' = K'\left[R \mid t\right]$$

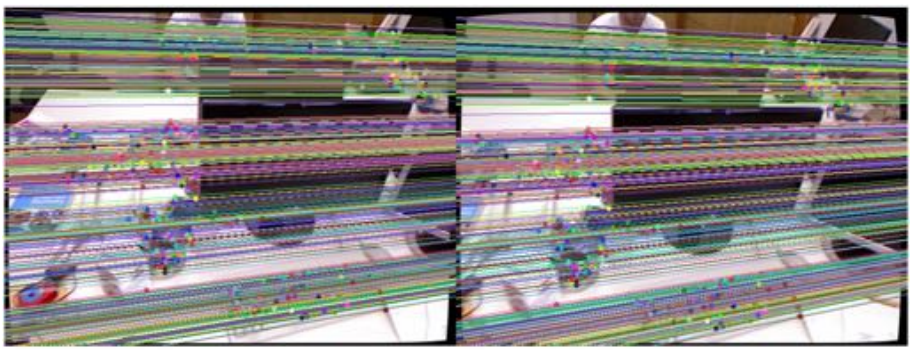


Figure 4.6: Example of epipolar lines before rectification: Feature points and corresponding epipolar lines as computed by the algorithm presented here; Input from TUM RGB-D dataset: [130].

$$\text{where } P^{+} = \begin{bmatrix} K^{-1} \\ 0^{T} \end{bmatrix} \text{ and } O = \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}$$

Thus, the fundamental matrix F can be written as:

$$F = \left[P'O\right]_x P'P^{+} = \left[K't\right]_x K'RK^{-1} = K'^{-T}\left[t\right]_x RK^{-1} = K'^{-T}R\left[R^{T}t\right]_x K^{-1} = K'^{-T}RK^{T}\left[KR^{T}t\right]_x \tag{4.1}$$

Considering the epipoles, which are the images of the respective other camera center

$$e = P\begin{bmatrix} -R^{T}t \\ 1 \end{bmatrix} = KR^{T}t$$

$$e' = P'\begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} = K't$$

the fundamental matrix becomes:

$$F = \left[e'\right]_x K'RK^{-1} = K'^{-1}\left[t\right]_x RK^{-1} = K'^{-T}R\left[R^{T}t\right]_x K^{-1} = K'^{-T}RK^{T}\left[e\right]_x$$

To sum up what has been described above and conclude the most important properties of the fundamental matrix: For a stereo camera system, with distinct camera centers $O \neq O'$, the fundamental matrix F is defined by the homogeneous matrix $F \in \mathbb{R}^{3\times3}$

with $rank(F) = 2$ and 7 DOF, satisfying

$$x'^T F x = 0$$

for all corresponding points $\mathbf{x} \longleftrightarrow \mathbf{x}'$ (epipolar constraint) [53].
$l' = Fx$ is the corresponding epipolar line to $\mathbf{x}$, and $l = F^T\mathbf{x}'$ is the corresponding epipolar line to $\mathbf{x}'$. Regarding both epipoles, following holds: $Fe = 0$ and $F^T e' = 0$, respectively. Note, that if F is the fundamental matrix for the camera pair (P, P'), F^T is the fundamental matrix in vice versa order (P', P) [53].
Despite F having 9 values, there are only 8 independent ratios, since common scaling is not significant. Due to $\det(F) = 0$, this results in 7 DOF [53].

4.2.2 Essential matrix

When the intrinsic parameters of the camera in both views are known (strong calibration case), we can reduce the number of corresponding points needed to compute the epipolar geometry. This results in the specialized form of the fundamental matrix with normalized image coordinates, namely the essential matrix [80]. The essential matrix E is dependent on 5 parameters describing the relative pose displacement between both views, 3 for the 3D Rotation and 2 for the direction of translation [53].
Assuming a camera matrix $P = K[R \mid t]$ and an image point $\mathbf{x} = P\mathbf{X}$ with known camera intrinsics K, we can use the inverse intrinsic matrix K^{-1} to get the image point in normalized coordinates $\hat{\mathbf{x}} = K^{-1}\mathbf{x}$. Thus, $\hat{\mathbf{x}} = [R \mid t]\mathbf{X}$ is the image point in normalized coordinates. We consider the normalized camera $P = [I \mid \mathbf{0}]$ and the second camera $P' = [R \mid t]$ [53]. Figuratively speaking, the normalized camera P represents the origin of coordinates and the second camera P' describes the point of view relative to the origin displaced by R and t.
Similar to the fundamental matrix (compare equation (4.2.1)) we define the essential matrix E as

$$\hat{\mathbf{x}}'^T E \hat{\mathbf{x}} = 0 \tag{4.2}$$

with normalized image coordinates for a set of corresponding normalized points $\hat{\mathbf{x}} \longleftrightarrow \hat{\mathbf{x}}'$ [53].
After substituting $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$, we get $\mathbf{x}'^T K'^{-T} E K^{-1}\mathbf{x} = 0$. Bringing this together with the definition for the fundamental matrix $\mathbf{x}'^T F\mathbf{x} = 0$, we get the relation between the

fundamental and essential matrix:

$$E = K'^T F K$$

Nistér [103] presented an efficient numerical solution for E with the five point algorithm which is also the basis of the *OpenCV* implementation [105].

Again, considering $P = [I \mid \mathbf{0}]$ and $P' = [R \mid t]$, let $[t]_\times$ be the skew symmetric matrix

$$[t]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

such that $[t]_\times x = t \times x$ for all points x. Thus, the fundamental matrix can be defined as:

$$F \equiv K^{-T} [t]_\times R K^{-1}$$

Now, for the case of normalized image points, meaning they have already been multiplied by K, we get from $\mathbf{x}'^T F \mathbf{x} = 0$ to the similar equation with normalized coordinates and the essential matrix: $\hat{\mathbf{x}}'^T E \hat{\mathbf{x}} = 0$. Therefore, it follows $F \equiv [t]_\times R$.

For the essential matrix, besides having rank 2, the following cubic constraint must hold:

$$EE^T E - \frac{1}{2} trace\left(EE^T\right) E = 0$$

Rewriting equation (4.2) to $\tilde{\mathbf{x}}' \tilde{E} = 0$ with:

$$\tilde{\mathbf{x}}' \equiv \left[\hat{\mathbf{x}}_1\hat{\mathbf{x}}'_1, \hat{\mathbf{x}}_2\hat{\mathbf{x}}'_1, \hat{\mathbf{x}}_3\hat{\mathbf{x}}'_1, \hat{\mathbf{x}}_1\hat{\mathbf{x}}'_2, \hat{\mathbf{x}}_2\hat{\mathbf{x}}'_2, \hat{\mathbf{x}}_3\hat{\mathbf{x}}'_2, \hat{\mathbf{x}}_1\hat{\mathbf{x}}'_3, \hat{\mathbf{x}}_2\hat{\mathbf{x}}'_3, \hat{\mathbf{x}}_3\hat{\mathbf{x}}'_3\right]^T$$

$$\tilde{E} \equiv [E_{11}, E_{12}, E_{13}, E_{21}, E_{22}, E_{23}, E_{31}, E_{32}, E_{33}\ ]^T$$

By using the representation $\tilde{\mathbf{x}}'$ for all five points needed, a 5×9 matrix is constructed. By SVD four vectors $\tilde{X}, \tilde{Y}, \tilde{Z}, \tilde{W}$ spanning the right nullspace of the matrix from above, the four vectors can be formed to four 3×3 matrices X, Y, Z, W, thus the essential matrix is: $E = xX + yY + zZ + wW$ with unknown scalar values x, y, z, w. Since the scalars can only be defined up to a common scale, let $w = 1$. Inserting this definition of the essential matrix into the nine equations from the cubic constraint, a 9×20

coefficient matrix is constructed, corresponding to a monomial vector:

$$\left[x^3, y^3, x^2y, xy^2, x^2z, x^2, y^2z, y^2, xyz, xy, xz^2, xz, x, yz^2, yz, y, z^3, z^2, z, 1\right]$$

Applying Gauss-Jordan elimination leads to an upper triangle form. The resulting equations are rearranged to two 4×4 matrices and determinants are extracted. A 10th degree polynomial for the determinants is obtained. Solving for the real roots and back-substituting them, all unknowns can be calculated, leading to the essential matrix [60, 103].
Note that the 5-point algorithm can robustly handle planar and non-planar structures and find a unique solution for the pose (see 4.2.3). The fundamental matrix is generally not able to find a unique solution, neither providing reliable pose estimation. This behavior is referred to as planar structure degeneracy [103]. With unfixed camera intrinsics, any homography between the two cameras can be realized, which leads to a degenerated fundamental matrix for the uncalibrated case. Therefore, establishing the calibrated case and computing the essential matrix is desirable.

4.2.3 Pose recovery based on epipolar geometry

Given an essential matrix (see section 4.2.2) describing the epipolar geometry between two stereo images, the rotation and translation between the two cameras P and P' can be estimated (up to scale) [103].
The essential matrix $E = [t]_x R$ has 3 DOF for both, the rotation and translation. However, there is a general scale ambiguity, leading to 5 DOF of E [53]. Induced by the reduced DOF compared to the fundamental matrix, we get further constraints: A necessary condition of the 3×3 matrix E is that two of its singular values are equal and the third is zero [62].
E is now decomposed $E = [t]_x R = SR$ with S as a skew-symmetric matrix. In the following the orthogonal matrix W and the skew-symmetric Z will be considered:

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

S can be written as $S = kUZU^T$, where U is orthogonal. We can write (up to sign),

$Z = \text{diag}(1,1,0)W$ and up to scale, $S = U\text{diag}(1,1,0)WU^T$. Thus,

$$E = SR = U\text{diag}(1,1,0)\left(WU^TR\right) = U\text{diag}(1,1,0)V^T$$

being the SVD of E [51, 53].

Given the SVD of E as $U\text{diag}(1,1,0)V^T$ we get two possible factorization (ignoring signs) of $E = SR$

$$S = UZU^T \quad R = UWV^T \quad or \quad R = UW^TV^T \tag{4.3}$$

By writing R as UXV^T with some rotation matrix X we get

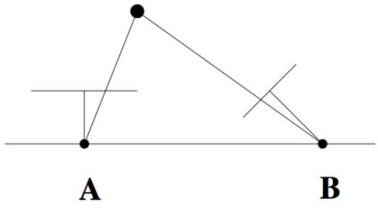
$$E = SR = U\text{diag}(1,1,0)V^T = \left(UZU^T\right)\left(UXV^T\right) = U(ZX)V^T$$

with $ZX = \text{diag}(1,1,0)$ and X being a rotation matrix, thus $X = W$ or $X = W^T$. It follows that the factorization describes the camera translation t up to scale by $S = [t]_x$. Since the Frobenius norm of $S = UZU^T$ is $\sqrt{2}$, meaning that it includes scale, the condition $\| t \| = 1$ has to hold. This is common and normalizes the camera baseline [53].

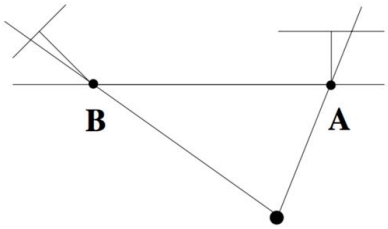
Due to: $St = 0$, it follows that $t = U(0,0,1)^T = u_3$ is the last column of U. Unfortunately, the sign of E and thus t can not be determined, leading to four possible solutions for the camera P'. However, only one of those solutions describes the case that the world point $\mathbf{X}$ is in front of both image planes (compare Figure 4.7) [53, 83, 141].

Induced by the unknown scale ambiguity, pose recovery based on the epipolar geometry is not sufficient for consecutive image stereo pairs with varying baseline. For each stereo pair one would be concerned with unknow scaling factors for the computed translation. Therefore, it is necessary to consider methods accounting for the absolute pose in a world reference frame with respect to the current 3D representation of the scene, rather than solely between stereo images themselves (see section 4.3.1).

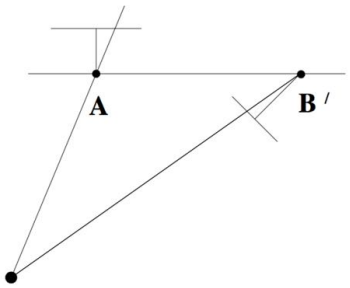
[2] The Frobenius norm is the matrix norm of an $m \times n$ matrix A, defined as: $\|A\|_F \equiv \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$ [144]



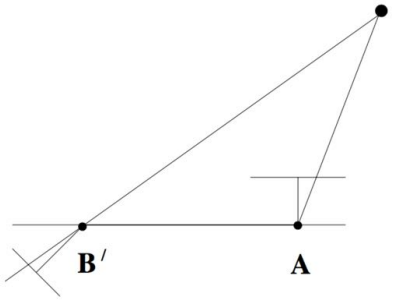
(a) Correct estimation with 3D point in front of both image planes



(b) With switched sign of t, the stereo configuartion is reversed



(c) Camera B is rotated by 180 deg to B' (compare equations (4.3) for two solutions for the second camera rotation)



(d) Again, also for the reversed case the, camera B is rotated

Figure 4.7: Four possible solutions for pose recovery from the essential matrix: As there are two possible solutions for the camera rotation and translation each, four possible camera configurations are possible; Only one (here (a)) describes the case that all 3D points are in front of both camera image planes; Adopted from [53]

4.2.4 Rectification

The epipolar geometry has already made it significantly easier for finding point correspondences between two images (compare Figure 4.6). One point in the first image is restricted to the 1D search space of its corresponding epipolar line in the second image. This search can be further simplified by rectifying both images in a way that the epipolar lines in the images are horizontally aligned. Given point correspondences (e.g. matched feature points) the stereo images can be warped resulting in the characteristics mentioned (compare Figure 4.8).

The stereo images can be rectified in different manners. For the uncalibrated case, the image warping relies on the epipolar geometry without considering the camera intrinsics (see 4.2.4.1 for the uncalibrated case). This procedure can be used for simply

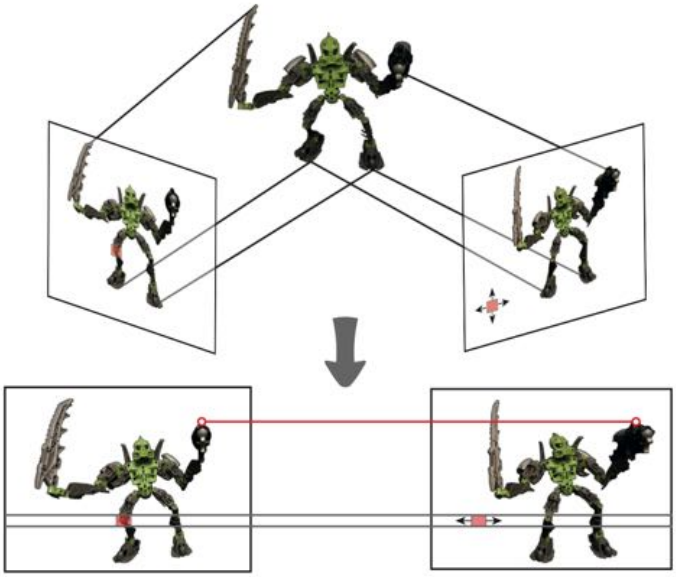


Figure 4.8: Overview of simplified rectification process: Before rectification, point correspondence search has to be performed across the whole image; After rectification corrsponding points are located on a 1D scanline; Adopted from [22]

finding stereo correspondences within the stereo images. For dense reconstruction the calibrated case is of particular interest. By not only aligning the epipolar lines in parallel, but also bringing the virtual image planes of the two cameras in frontal parallel position (compare canonical camera setup in Figure 3.10), this procedure allows to reconstruct 2D points within the general world reference frame (see 4.2.4.2 for the calibrated case). Both cameras are virtually rotated around the cameras optical center, such that the camera center is horizontally aligned and the image planes face frontal parallel towards the scene. By using the calibrated method, 3D reconstruction can be performed up to a simple similarity transform, namely scaling, whereas for the uncalibrated case a projective ambiguity will result in incorrect reconstructions (see 4.2.4.3: Reconstruction ambiguity and Figure 4.12). Both methods will be discussed in the following.

4.2.4.1 Rectification based on epipolar geometry for the uncalibrated case

For uncalibrated cameras, the rectification can be performed making use of the epipolar geometry, in particular the fundamental matrix. The resulting image projections have epipolar lines parallel to the $x-axis$, thus disparities between the images are in x direction only [53]. Figure 4.9 illustrates such a procedure, depending on the matched feature points which satisfy the epipolar geometry described by the fundamental matrix, the images can be warped, such that the scanlines are parallel.

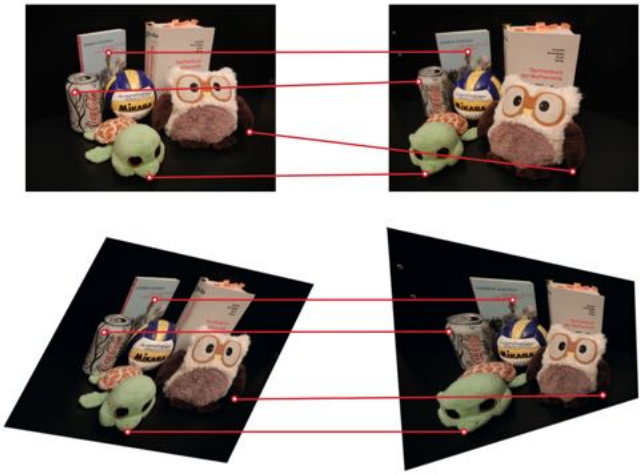


Figure 4.9: Comparison of sparse point correspondences before and after rectification: Sparse point correspondences can be established in the non rectified images; After rectification corresponding points reside on 1D scanlines, thus enabling for dense correspondence search; Adopted from [22]

To transform the epipolar lines to be parallel with the $x-axis$, the epipole should move to infinity, in particular $(1,0,0)^T$. We want to estimate the transformation for each image which brings them into coplanar images, called homography H. The homography should act as a rigid transformation of the image, such that the neighborhood around the point x_0 will only be transformed by rotation and translation and thus looks the same after transformation. For x_0 one can take the image center for example [53]. For the uncalibrated case the rectification is actually a planar perspective transformation encoded as homography matrices [76].

To transform the epipole $(f,0,1)^T$ to infinity $(f,0,0)^T$ we consider the transformation:

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1/f & 0 & 1 \end{bmatrix}$$

G maps some point $(x,y,1)^T$ to $(\hat{x},\hat{y},1)^T = (x,y,1-x/f)^T$. For the case $\mid x/f \mid < 1$ we can write:

$$(\hat{x},\hat{y},1)^T = (x,y,1-x/f)^T = (x(1+x/f+\ldots),y(1+x/f+\ldots),1)^T$$

Looking at the Jacobian:

$$\frac{\partial(\hat{x},\hat{y})}{\partial(x,y)} = \begin{bmatrix} 1+2x/f & 0 \\ y/f & 1+x/f \end{bmatrix}$$

and neglecting higher order terms and setting $x = y = 0$ this becomes the identity matrix $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, meaning that the mapping at x_0 is approximately the identity [53].
We get the mapping H for a point x_0 and the epipole e, $H = GRT$, with translation T of x_0 to the origin, rotation R around the origin bringing the epipole e' to a point $(f,0,1)^T$ on the x-axis, and G a mapping to take $(f,0,1)^T$ to infinity [53].
The epipole of one image has been moved to infinity in order to rectify it and to bring the epipolar lines in parallel orientation by a transformation. Now the transformation for the second image has to be determined to match with the epipolar lines. Therefore we take two images, for which H is used to transform image J, and H' for image J', respectively. For any pair of epipolar lines l and l' we get $H^{-T}l = H'^{-T}l'$. We aim to find the matching transformation H for the transformation H' obtained as described above [53].
Let J and J' be two images with the fundamental matrix $F = [e']_x M$, then the two homographies match if and only if

$$H = \left(I + H'e'a^T\right)H'M \tag{4.4}$$

with $M := PP'^{+}$ for some arbitrary vector [53].
Taking a closer look at the case, that H' brings the epipole e' to infinity $(1,0,0)^T$,

leading to

$$\begin{aligned} H &= \left(I + H' e' a^T\right) H' M \\ &= \underbrace{\left(I + (1,0,0)^T a^T\right)}_{H_A} \underbrace{H' M}_{H_0} \\ &= H_A H_0 \end{aligned} \tag{4.5}$$

with $H_A = \begin{bmatrix} a & b & c \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

For $\hat{\mathbf{x}}_i' = H' \mathbf{x}_i'$ and $\hat{\mathbf{x}}_i = H_0 \mathbf{x}_i$, we want to solve for the parameter in H_A, thus we set up the minimization term:

$$\min \sum_i d\left(H_A \hat{\mathbf{x}}_i, \hat{\mathbf{x}}_i'\right)^2$$

With $\hat{\mathbf{x}}_i = (\hat{x}_i, \hat{y}_i, 1)^T$ and $\hat{\mathbf{x}}_i' = \left(\hat{x}_i', \hat{y}_i', 1\right)^T$, and known H' and M, we can substitute into the formula from above:

$$\min \sum_i \left(a\hat{x}_i + b\hat{y}_i + c - \hat{x}_i'\right)^2 + \left(\hat{y}_i - \hat{y}_i'\right)^2$$

resulting in following, since $\left(\hat{y}_i - \hat{y}_i'\right)^2$ is constant:

$$\min \sum_i \left(a\hat{x}_i + b\hat{y}_i + c - \hat{x}_i'\right)^2$$

Solving the minimization, yields a, b and c, from which we can set H_A and subsequently compute H with (4.5) and (4.4) [53].

Applying the described algorithm to the introductory example with non rectified epipolar lines (compare Figure 4.6), we get the rectified image projections as in Figure 4.10. Note that now corresponding points only vary in x-direction as indicated by the red parallel lines.

Figure 4.10: Example of epipolar lines after rectification for the uncalibrated case: From the input Figure 4.6, now the epipolar lines are horizontally aligned; Point correspondences only differ in x-direction as indicated by the red lines; TUM RGB-D dataset: [130]

4.2.4.2 Rectification for the calibrated case with frontal parallel cameras

For the calibrated case, with known camera intrinsics and camera pose, it is possible to not only align the epipolar lines to be parallel by warping the image to the rectified image projection, but to virtually bring the cameras in the canonical configuration (compare Figure 3.10), describing a perfect stereo setup with aligned cameras being frontal parallel. This procedure is depicted in Figure 4.11, bringing the cameras virtually into the canonical camera configuration. Further, the $y-axis$ is shifted perpendicular to the camera center line, making epipolar lines horizontally and ensuring disparity for points in infinity equals 0. If necessary, the images can be rescaled [133].

The calibrated rectification process is beneficial, since it allows to find 2D stereo correspondences efficiently - as it does for the uncalibrated case - while being able to reconstruct the 3D points in the world reference frame without projective ambiguity. This is assured since the new camera intrinsics and extrinsics for the now rectified images are known. With known transformation from the original camera to the virtual rectified, the 3D points can be projected and directly transformed into the world reference frame. This is not the case for the uncalibrated scenario, which relies solely on the epipolar geometry and simply warping the images for alignment. Here, the cameras are actually transformed to conform with the canonical stereo camera setup.

The relative transformation between the two cameras of the stereo system, with given absolute rotation and translation in the world reference frame, is defined by $R = R_r R_l^T$

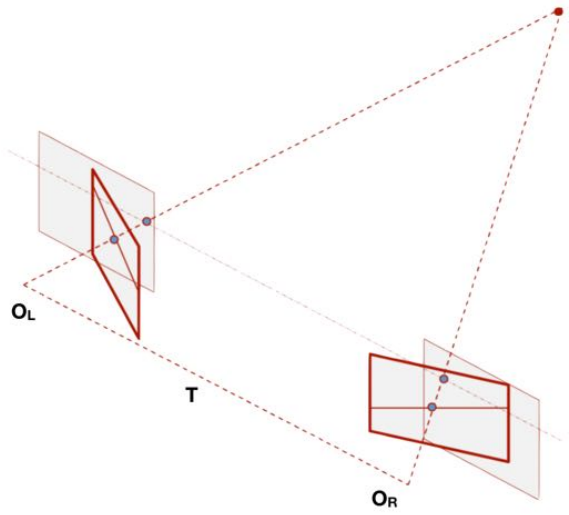


Figure 4.11: Process of rectification for the calibrated case: As illustrated by the solid bold image planes, the original cameras are virtually rotated and shifted such that they correspond to the canonical stereo setup (compare Figure 3.10); After rectification epipolar lines are parallel and the new virtual camera poses are known (indicated as transparent image planes); Adopted from [76]

and $t = t_l - R^T t_r$ for the relative rotation and translation, with subscripts r and l accounting for the rotation and translation of the left and right camera in the world reference frame, respectively. Given the camera intrinsics and the relative transformation between the considered stereo setup, a rotation matrix R_{Rect} rotates the left camera around its optical center. This brings the left epipole to infinity and making the epipolar lines horizontal.

$$R_{Rect} = \begin{pmatrix} e_1^T \\ e_2^T \\ e_3^T \end{pmatrix}$$

with $e_1 = \frac{t}{\|t\|}$, $e_2 = \frac{1}{\sqrt{t_x^2 + t_y^2}} \left[-t_y, t_x, 0\right]^T$ and $e_3 = e_1 \times e_2$.

By transforming the right camera with R and t from above, we get:

$$P_l = R^T P_r + t$$

and applying R_{Rect}, leads to:

$$R_{Rect}P_l = R_{Rect}R^T P_r + R_{Rect}t$$

It can be shown, that

$$R_{Rect}t = \begin{pmatrix} \|t\| \\ 0 \\ 0 \end{pmatrix}$$

which means, now there is only translation in x-direction, thus having a perfect canonical stereo setup. Note that now t can be substituted with T describing the baseline with $t_X, t_Y = 0$ in compliance with Figure 4.11 and Figure 3.10.
To complete the rectification process, all image points need to be projected in the camera frame, transformed into the virtual rectified camera frame and backprojected to the new rectified images [76].
Note, that stereo rectification for calibrated cameras only leads to meaningful results if the rotation between the considered stereo system is not too large and translation is mainly in x-direction (or y-direction for vertical stereo). Otherwise the newly rectified image points would move out of the field of view of the rectified virtually rotated cameras.
The rectified images can now be used to find dense stereo correspondences easily and compute their disparity, which is necessary for a dense 3D reconstruction of the scene (see section 4.3.3).

4.2.4.3 Reconstruction ambiguity

As already mentioned, for vision based methods, the reconstruction will always be ambiguous, at least to some extent. Without knowing the general placement of a scene, the absolute orientation and position cannot be determined. Furthermore, without having some object with true and known length or true baseline between stereo cameras, the general scale of the scene can not be obtained. For calibrated cameras the ambiguity relates to general rotation, translation and scaling of the whole scene (compare Figure 4.12: left). Without camera intrinsics, the reconstruction must only comply with the corresponding image points. Such a case is depicted in Figure 4.12 (right) where the focal length of the cameras is altered, leading to a

different reconstructions while the rays of the 3D points still intersect the image plane at the identical position [53].

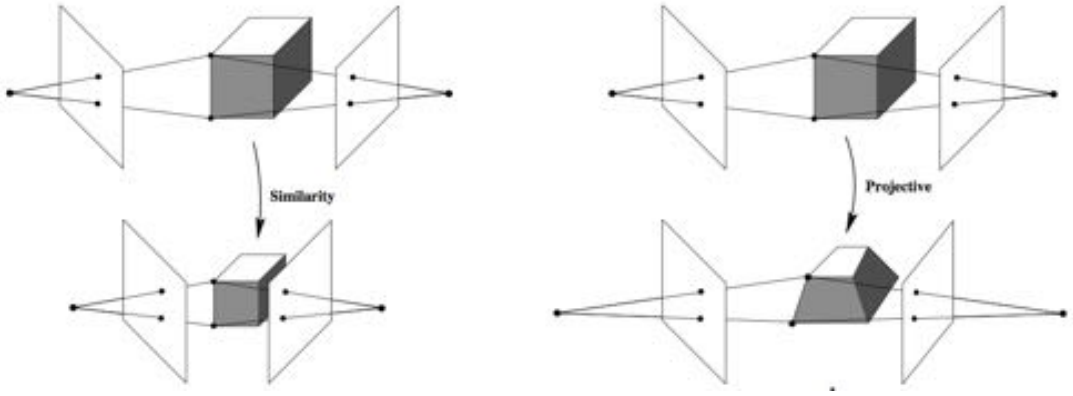


Figure 4.12: Reconstruction ambiguity: For the calibrated case (left), the reconstruction can be computed up to an unknown similarity transform, namely scaling; With uncalibrated cameras and unknwon intrinsics (right), the true appearence of the scene cannot be reconstructed; Adopted from [53]

It is obvious that only the rectification for calibrated cameras can lead to meaningful 3D reconstructions, since firstly ambiguity is reduced to some similarity, and secondly the reconstruction can be transformed to the world reference frame due to known new rectified virtual cameras.

4.3 3D reconstruction

For 3D reconstruction it is essential to know the intrinsic and extrinsic camera parameters as well as 2D point correspondences from different views. In section 4.2 the epipolar geometry has been considered, how the relative camera pose can be estimated up to scale, and how computing dense stereo correspondences is reduced to a 1-dimensional problem. This knowledge can now be extended for computing a 3D reconstruction.

Since the relative camera pose estimation between two cameras, based on the essential matrix, can only provide translation up to an unknown scale, this procedure cannot be used to estimate the global camera pose from subsequent stereo comparisons with a monocular camera. The camera poses need to be known within the world reference frame relative to the 3D scene. The first two camera poses can first be estimated based on the epipolar geometry as an initialization. Their unknown scale of the baseline define the overall scale ambiguity of the reconstruction. In relation to their initially built up sparse feature point cloud, we can now compute new camera poses with the perspective-n-point algorithm (PnP).

We distinguish between *sparse* (see section 4.3.2) and *dense* (see section 4.3.4) reconstruction. For sparse reconstruction one might want to find a single 3D point from a known 2D stereo correspondence, e.g. matched feature points. This can be achieved via triangulation, computing the intersection of image rays from the optical center of each camera through the corresponding image coordinate (see section 4.3.2). However, such methods would not be applicable for dense reconstruction since the rays will in general not intersect in 3D but are rather skew. Without dense 2D correspondences reconstruction would categorically not be possible for a stereo setup as there is no unique intersection between two rays from one image to one of the other image, from the set of all image rays. Computing a dense ray intersection from more than two views would be possible, but it is computationally expensive and some uncertainty of the correct 3D point remains. Newcombe et al. [100] propose a similar approach for a rigid scene. However, they take hundreds of views and compute a probability distribution for each 3D point from ray intersections over all input frames.

To circumvent the above mentioned shortcomings, the rectified images from the calibrated case in section 4.2.4.2 are used to find dense stereo correspondences and compute a so called disparity map (see section 4.3.3). A disparity map, or depth map,

is a 2.5D map, where each valid pixel holds the disparity value for its image coordinate. From the disparity, it is easy to compute the depth value, as will be shown in section 4.3.3, thus enabling to compute a dense 3D reprojection from the current view (see section 4.3.4).

4.3.1 Perspective-n-point algorithm

Given n 3D-2D correspondences, the 6 DOF camera pose and the camera intrinsics can be computed with the well known direct linear transform (DLT) algorithm [53]. With known camera intrinsics, the camera pose estimation can be reduced to the perspective-n point algorithm (PnP) with a minimum of three 3D-2D correspondences (P3P) [151]. Figure 4.13 illustrates the process of finding the camera pose with known point correspondences [106].

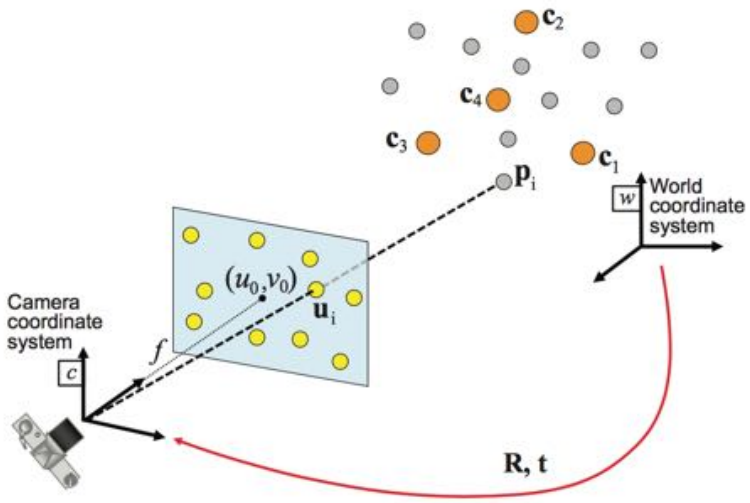


Figure 4.13: PnP Problem: Given a set of 3D-2D point correspondences, the 6 DOF camera pose (R and t) w.r.t. the world reference frame is computed; Adopted from [106].

Besides an iterative method based on Levenberg-Marquardt optimization [105], the minimal approach for three corresponding points [151], a direct-least-squares-based method [56] or the combined computation of the extrinsic camera parameters and the focal length [106], the method proposed by Lepetit, et al. [77] (EPnP) works

in the general case of planar and non-planar scenes, with a complexity of $O(n)$. For issues of robustness and reducing the influence of outliers, a RANSAC scheme can be implemented.

4.3.2 Sparse reconstruction

After computing the pose (see sections 4.2.3 and 4.3.1) for each camera of a stereo system, and with given intrinsics, the 3D position for 2D point correspondences can be computed via triangulation. As already mentioned before (compare 3.3.2), the computation of the 3D coordinates of a point from two images is more complicated in real applications. Besides rectifying images and computing disparity maps as described in 4.3.3, one might want to compute the depth of a certain pixel in two distinct images without the need to compute the epipolar geometry. Here, for a given world point $\mathbf{X}$ and the corresponding image points $\mathbf{x}$ and $\mathbf{x}'$ with two distinct cameras P and P' and known intrinsics, a method for obtaining the 3D point will be described. The difficulties for correct depth estimates by triangulation arising in practice are not obvious. One would assume the case in Figure 4.4 where the image rays of corresponding points intersect at the 3D world point $\mathbf{X}$. However, when back-projecting the image points back in 3D space, the rays are often skew and do not intersect in $\mathbf{X}$ (compare Figure 4.14).

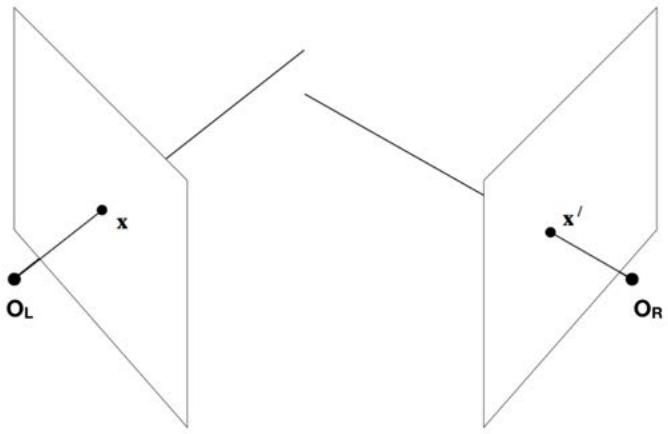


Figure 4.14: Skew image rays during triangulation: Due to noisy image aquisition, small errors in the pixel coordinates can lead to non intersecting skew rays in 3D; Adopted from [53].

The skew rays are a result of errors in the image acquisition leading to uncertainties

in the image coordinates. Thus, the back-projected rays will not intersect in general such that no point $\mathbf{X}$ satisfies $\mathbf{x} = P\mathbf{X}$ and $\mathbf{x}' = P'\mathbf{X}$, simultaneously. Furthermore, these image points violate the epipolar constraint $\mathbf{x}'^T F\mathbf{x} = 0$. The violation of the epipolar constraint is depicted in Figure 4.15, where the image points do not lie on the corresponding epipolar line.

Figure 4.15: Violation of the epipolar constraint in 2D: As a result from noisy image aquisition $\mathbf{x}$ and $\mathbf{x}'$ do not satisfy the epipolar constraint, thus they do not lie on the corresponding epipolar line; Adopted from [53].

It is not reasonable to minimize errors in the 3D projective space due to computational costs. Thus, the geometry of the image points will be under consideration. As discussed, the point correspondences $\mathbf{x} \longleftrightarrow \mathbf{x}'$ do not satisfy the epipolar constraint. In practice, the correct image points for the correspondences $\bar{\mathbf{x}} \longleftrightarrow \bar{\mathbf{x}}'$ should be close to the erroneous points, satisfying $\bar{\mathbf{x}}'^T F\bar{\mathbf{x}} = 0$. Therefore, a geometric error function is introduced, minimizing the distance between the measured image points and the ones fulfilling the epipolar constraint:

$$C\left(\mathbf{x},\mathbf{x}'\right) = d\left(\mathbf{x},\hat{\mathbf{x}}\right)^2 + d'\left(\mathbf{x}',\hat{\mathbf{x}}'\right)^2 \tag{4.6}$$

subject to $\hat{\mathbf{x}}'^T F\hat{\mathbf{x}} = 0$ with d as the Euclidean distance. This is equivalent to the minimization of the reprojection error for $\hat{\mathbf{X}}$ (compare Figure 4.16).

We will now focus on the geometry depicted in Figure 4.17. We search for corresponding points $\hat{\mathbf{x}} \longleftrightarrow \hat{\mathbf{x}}'$ that minimize the geometric error from Figure 4.16. Any two corresponding points in the images must lie on the epipolar line in order to satisfy the epipolar constraint. Thus, we can take any pair of points on the epipolar lines. Here, we take the points $\mathbf{x}_\perp$ and $\mathbf{x}'_\perp$ being the closest points from $\mathbf{x}$ and $\mathbf{x}'$ on the epipolar lines, respectively. We can therefore write $d\left(\mathbf{x},\hat{\mathbf{x}}\right) = d\left(\mathbf{x},l\right)$ with $d\left(\mathbf{x},l\right)$ being the perpendicular distance between $\mathbf{x}$ and the epipolar line l. The same applies for the

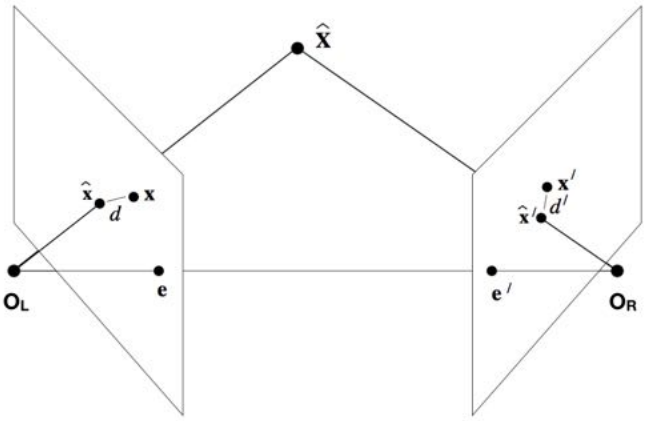


Figure 4.16: Minimization of the reprojection error in 3D: To find the optimal 3D point $\hat{\mathbf{X}}$ from noisy point correspondences, we seek to minimize the reprojection error between the estimated and noisy image point correspondences; Adopted from [53].

Figure 4.17: Minimization of the reprojection error in 2D: Since solving the reprojection in 3D is not suitable, the perpendicular distance between the estimated epipolar lines and the noisy point correspondences can be minimized in 2D; Adopted from [53].

point $\mathbf{x}'$ accordingly. Thus, we write the minimization problem as:

$$C_l\left(\mathbf{x}, \mathbf{x}'\right) = d\left(\mathbf{x}, l\right)^2 + d'\left(\mathbf{x}', l'\right)^2$$

By minimizing the objective, the point correspondences and thus the 3D point can be computed. For further details refer to Hartley and Zisserman [53].

Herewith, single 3D points can be computed from e.g. matched 2D feature correspondences, resulting in a sparse reconstruction of the scene or object (compare Figure 4.18). For denser reconstructions triangulation is not suitable, leading to further processing of the given input images as discussed in the following (see section 4.3.3).

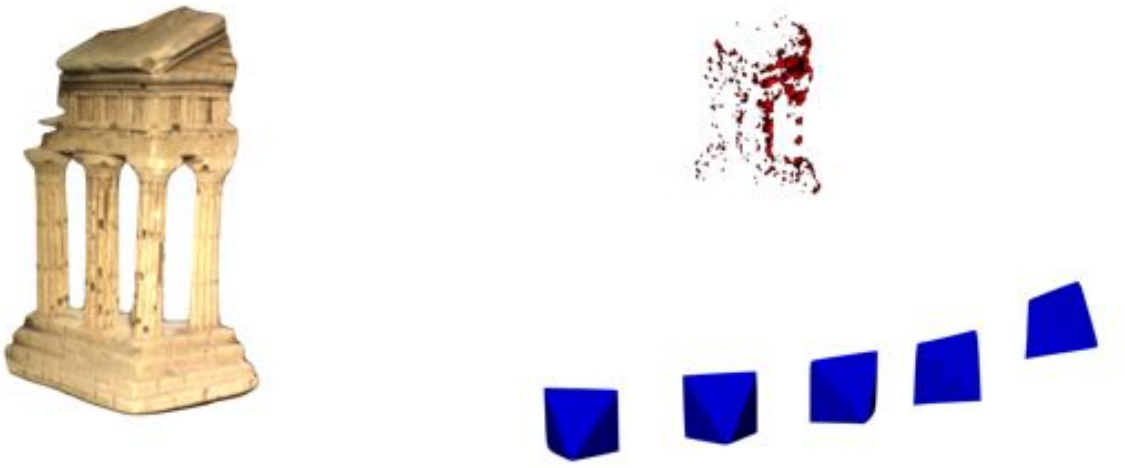


Figure 4.18: Example of sparse reconstruction with the algorithm presented here: Images of the temple on the left side are taken at different camera positions; The algorithm presented here performs a sparse reconstruction based on feature points; This reconstruction (red points in 3D) is used for camera pose tracking within the common world reference frame (camera poses indicated by blue pyramids); Please note the image has been processed for better visibility; Input from Middleburry MVS Dataset [120].

4.3.3 Stereo correspondences and disparity map

The search domain for corresponding image points has already been restricted to parallel orientated epipolar lines. We now focus on how corresponding image regions and points from the first image can be found in the second image in a dense manner. The information of a single pixel is not suitable for robust matching. Therefore, a search window or block around the considered pixels can be incorporated. This is for example the basis of the block-matching algorithm of Konolige [74]. The algorithm only works well for regions with strong textures and disparity map results are generally too poor for our purposes. For such local block-matching approaches, depth discontinuities cannot be accurately reconstructed, due to the use of a sliding window for finding the disparities [58, 119]. In order to correctly estimate a per-pixel disparity, additional regularization and smoothness constraints need to be incorporated, as is done in global methods [58]. Unfortunately this leads to an NP-hard problem [10] and approximate solutions such as graph cuts [73] are still computationally expensive.

Hirschmueller [57] proposed a semi-global matching (SGM) method, combining local and global matching concepts. The global cost function is optimized in 1D along several scanlines and accounts for depth discontinuities and edges by regularization. The costs are aggregated along multiple scanlines, and the correct disparity is computed [57]. Thereby, the complexity is kept linear to the number of pixels and disparities. With the incorporation of the left-right consistency check [74], SGM produces reliable results. The algorithm performs further post processing of the disparity map in terms of filters and interpolation of missing disparity values accounting for occlusions [57].

The OpenCV implementation of SGM [105] incorporates a pixel dissimilarity measure proposed by Bierchfield and Tomasi [7] as matching costs, instead of the originally proposed mutual information. Furthermore, this implementation uses either five or eight directions for the scan lines. This implementation will be used in the algorithm presented in this thesis.

The methods described above are the basis for extracting dense structural information from images. Based on the computed disparity, we can reproject a dense representation of the scene back in 3D. The obtained disparity maps can be thought of a 2.5D representation of the scene. With known camera intrinsics and pose, each pixel in the disparity map can be projected into the world reference frame with its disparity accounting for the depth (see section 4.3.4).

Since the quality of disparity maps obtained by SGM is still not suitable for high quality reconstruction, a weighted least squares filter with left-right consistency consideration from the contribution modules of OpenCV is implemented [39, 88]. Resulting depth maps are considerably denser and describe the scene more accurately.

4.3.4 Dense reconstruction by reprojection of disparity

The OpenCV function [105] for the rectification process with calibrated cameras (see section 4.2.4.2) provides a so-called reprojection matrix Q:

$$Q = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 1 & 0 & \frac{-1}{t_X} & \frac{\left(c_x - c'_x\right)}{t_X} \end{bmatrix}$$

where f ,c_x and c_y are the intrinsics for the left rectified camera, c'_x the image center in x-direction for the right rectified camera, and t_X the relative translation between both optical centers, which now equals the baseline T, since we have a perfect canonical stereo setup (compare Figure 3.10).
For homogeneous 2D points and the associated disparity, the reprojection matrix can directly reproject a point to homogeneous 3D points [11]:

$$Q \begin{bmatrix} p_x \\ p_y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix}$$

Thus, dehomogenization gives the 3D coordinates.
An example of such a reprojection to 3D is shown for the disparity map (after filtering) in Figure 4.19.

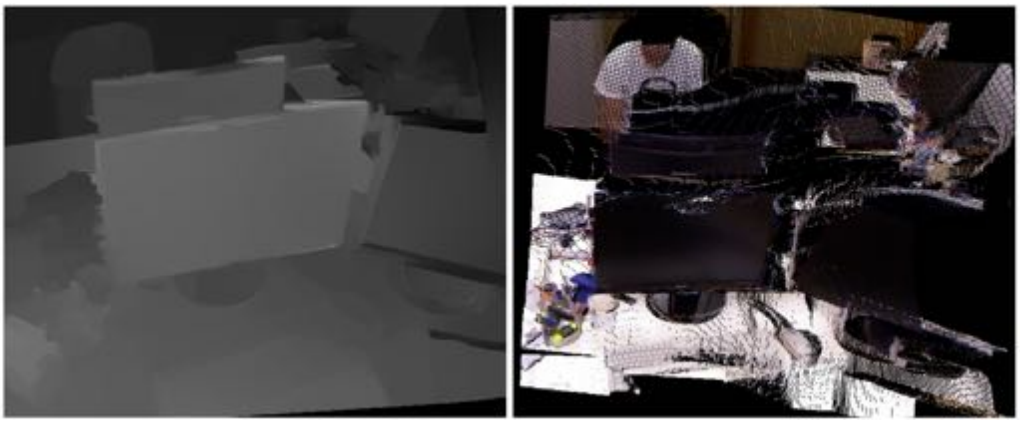


Figure 4.19: Example of 3D reconstrcution from filtered disparity map; Input from TUM RGB-D dataset: [130].

The reprojection matrix Q accounts for the left rectified camera. To bring the reconstructed 3D points from the rectified camera frame in the common world reference frame, they first need to be transformed back to the non-rectified camera frame. This camera frame is now at the origin of the world reference frame, since we assumed the camera pose to be at the origin during rectification and computed the relative transformation of the stereo system for simplification. Thus, the reconstructed 3D points need

to be transformed to their final position in the world reference frame by the camera pose of the left camera [3].

4.4 Point cloud alignment and non-rigidity

Until now only single point clouds have been considered. For a dense model, point clouds from all available camera poses should be aligned and fused together. Results can be continuously integrated into a commonTSDF representation(see section 4.4.6), which will be called the canonical model. In the previous sections the focus was on extracting robust rigid feature point correspondences as foundation for reliable camera pose estimation as needed for dense reconstruction. Points in the background can for instance be treated as such. Now, not only rigid parts but rather non-rigidly deforming points should be considered. Under the assumption that there is only limited deformation between two frames of a stereo setup and that deformations are as-rigid-as-possible (see section 4.5.1), the scene can be reconstructed. However, between several point clouds rigid and non-rigid transformations can occur. Therefore, the rigid transformation between the previous and the current point cloud needs to be accounted for by rigid ICP alignment (see 4.4.1). Please note there will only be minor rigid misalignment since all point clouds are already in the world reference frame. However, especially for monocular systems, drift might occur [127]. To reduce drift, it is beneficial to align the sampled canonical model to the current point cloud, whereas finding correspondences can be challenging. Drift will not be fully solved as this is one of the main issues with all SLAM methods [36].

A feature based optical flow approach can find non-rigidly deforming point correspondences. This is not the case for the previously discussed feature matching (see section 4.1.2), since non-rigid correspondences would be considered as outliers, for which the epipolar constraint does not hold true between keyframes. The non-rigid transformations for the sparse point correspondences between the old and the current point cloud are computed as described in section 4.5. The resulting local transformations are represented as dual quaternions (see section 4.4.4) with some transformation consisting of point-wise rotation and translation. These underlying sparse correspondences and their associated transformations define the deformation support nodes of the embedded deformation graph, used for DQB (see 4.4.3 and 4.4.5) to deform the whole model accordingly.

4.4.1 Iterative closest points - ICP

The idea of the iterative closest point algorithm (ICP) is to rigidly align two sets of points in space, which are at least partially overlapping but not necessarily completely coincident, in an iterative way [45]. After estimating the depth from world points projected on the image planes of two or more distinct camera positions, we obtained the world point position described by $\mathbf{X} = (X, Y, Z)$ as its coordinates in a global world frame. The aim is to align different 3D world point clouds of the same scene obtained from different views [59, 116]. The ICP algorithm, described by Rusinkiewicz and Levoy [116], consists of six steps [90]:

1. Point selection: characteristic points in each point cloud are selected
2. Matching: selected points need to be matched
3. Matching agreement: filter matched points
4. Weighting: filtered matches are weighted
5. Term definition: Set up ICP term
6. Minimization: minimize ICP term to obtain best alignment of point clouds

Consider a data structure P, representing a 3D point cloud with points $p \in \mathbb{R}^3$. Given two point clouds P with $p \in P$ as source and Q with $q \in Q$ as target, we want to find correspondences between them and compute the transformation A to align the later to the source [59]. Commonly the Euclidean distance is used to determine the distance between the points of both point clouds [6]. Correspondence search is accelerated by kd-trees to achieve an algorithm complexity of $O(N \log N)$. In PCL [117] the open source library FLANN [91] is used.
Several error metrics are used to minimize the alignment error of the point clouds among others, the most common are point-to-point and point-to-plane:

$$E_{point-to-point}(A) = \sum_{k=1}^{N} w_k \|Ap_k - q_k\|^2 \tag{4.7}$$

$$E_{point-to-plane}(A) = \sum_{k=1}^{N} w_k \left((Ap_k - q_k) \cdot n_{q_k}\right)^2 \tag{4.8}$$

where A is the transformation consisting of global rotation R and a translation t, (p_k, q_k) is the k-th correspondence pair of N, w_k a weighting factor and n_{q_k} the surface normal at point q_k [59].

4.4.2 Optical flow

In section 4.1.2 we already described how features can be robustly matched between images. However, this method is restrictive in terms of outliers not confirming with the specified error measures (e.g. epipolar constraint: see equation 4.2.1, and reprojection error). Since for simple feature matching it is hard to distinguish between possible outliers and inliers, without employing further regularization, here an optical flow based radius feature matching is proposed. For a set of feature points in one image, first their displacement in the other image is estimated by sparse optical flow. Then, all feature points within a certain radius around the computed displacement position in the second image are considered for matching and the best possible match is regarded as match. Thus, not only feature points with rigid transformation, but rather rigid and non-rigid transformations between images are included in this feature matching approach.

Optical flow methods aim to compute the pixel displacement of corresponding points between one image to another. For some image point $I(x, y, t)$ at time t, find the pixel $I(x+u, y+v, t+1)$ in the image at time $t+1$, with the displacement u and v. Using dense optical flow estimation methods [40] would be beneficial, but not applicable due to expensive computation, and the assumption to have only very small displacements within the image, which is not true, since we only compare key frames, rather than every frame [11].

The Lucas-Kanade (LK) method [82] is used for sparse optical flow estimation. It assumes brightness consistency, temporal persistence meaning small displacements and spatial coherence as close points move consistently. It can handle larger displacements with a coarse-to-fine pyramidal approach together with a window search [9]. This also solves the aperture problem. By defining some $n \times n$ search window, this results in the

following set of equations:

$$\underbrace{\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_{n\cdot n}) & I_y(p_{n\cdot n}) \end{bmatrix}}_{A=n\cdot n\times 2} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{d=2\times 1} = -\underbrace{\begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_{n\cdot n}) \end{bmatrix}}_{b=n\cdot n\times 1}$$

where I_x and I_y are spatial derivatives and I_t the derivative over time between the images.

For this over-constrained system the least-squares minimization $\|Ad-b\|^2$ can be solved as:

$$\underbrace{\left(A^TA\right)}_{2\times 2}\underbrace{d}_{2\times 1} = \underbrace{A^Tb}_{2\times 2}$$

which can be written as

$$\underbrace{\begin{bmatrix} \sum I_xI_x & \sum I_xI_y \\ \sum I_xI_y & \sum I_yI_y \end{bmatrix}}_{A^TA}\begin{bmatrix} u \\ v \end{bmatrix} = -\underbrace{\begin{bmatrix} \sum I_xI_t \\ \sum I_yI_t \end{bmatrix}}_{A^Tb}$$

leading to

$$\begin{bmatrix} u \\ v \end{bmatrix} = \left(A^TA\right)^{-1}A^Tb$$

With $\left(A^TA\right)$ having rank 2 and thus being invertible, the objective can be solved. One robust possibility is the case if $\left(A^TA\right)$ has two large eigenvectors, which occurs in image regions with corners. Thus, it is worthwhile to consider corner points for optical flow estimation in order to obtain good results [11]. Here, ORB features (see section 4.1.1) are used, which rely on the oriented FAST keypoint detection [114], which again employs the Harris corner detection [50] to find keypoints.

Now, with known displacement for the detected feature points in the first image, a radius matching in the second image is performed if the error metric from the optical flow method is reasonable [105]. At the displaced location the best feature match is found, enabling the optical flow results to be further refined for all rigidly and non-rigidly transformed and displaced feature points.

4.4.3 Deformation nodes

As in DynamicFusion [99], Fusion4D [34] and VolumeDeform [64] a reduced number of point correspondences from the 3D reconstruction can sufficiently explain local transformations needed in order to align two point clouds or meshes non-rigidly [131], called deformation graph G_V with V being the set of all nodes v_i of the graph. Newcombe et al. [99] as well as Innmann et al. [64] establish correspondences by performing a projective association method incorporating a model-to-frame point-plane minimization based on their RGB-D input, while Dou et al. [34] perform a more complicated procedure similar to the global patch collider [140] using decision trees. Here, the already established sparse point correspondences from the proposed optical flow based radius feature matching method define the nodes of the deformation graph. Point correspondences from consecutive stereo image pairs can first be reconstructed in 3D. The point-wise deformations can be computed by minimizing their distance in 3D space, while accounting for regularization and local deformation consistency (details in section 4.5). These local transformations can be described by quaternions (see section 4.4.4) and subsequently be used for interpolating the deformation of the remaining points in the cloud via DQB (see section 4.4.5).

Each deformation node has some influence on its neighborhood within the point cloud. Thus, the remaining points can be transformed in 3D space according to their associated deformation nodes, which serve as anchor points of the deformation (see section 4.4.5). As described in [131], the k-nearest neighbors within a specified radius around the deformation nodes are influenced by its deformation. DynamicFusion [99] argues this approach can describe the overall deformation well enough, while introducing a hierarchical scheme of deformation graphs due to computational complexity. They work on a relatively fine scaled graph, established by projective correspondence association. Here, the known sparse correspondences form the deformation graph ensuring explicit correspondences and computational reduction due to sparsity. However, poorly defined areas in the graph may reduce the quality of DQB.

4.4.4 Quaternions

Besides the matrix notation (see 3.3.1), and the axis-angle notation with Euler angles, rotations can also be described by quaternions [61]. Quaternions extend complex

numbers and can be thought of as a four- component vector with three different imaginary parts. They are usually denoted by symbols with dots. A quaternion, as presented by Hamilton[3]in 1901 [13] a quaternion $\dot{q}$ (do not confuse with temporal derivative in physics) is given by:

$$\dot{q} = q_0 + iq_1 + jq_2 + kq_3$$

and has a real part q_0 and three imaginary parts q_x, q_y and q_z [61]. The set of all quaternions is denoted by $\mathbb{H}$ [113], defined by $i^2 = j^2 = k^2 = ijk = -1$.
Note, that in general the product $\dot{q}\dot{r}$ does not have the same form as $\dot{r}\dot{q}$, thus $\dot{r}\dot{q} \neq \dot{q}\dot{r}$ (non-commutative).
A quaternion of norm one is called a unit quaternion. By dividing the non zero quaternion by its norm, we get the unit quaternion of this quaternion $q = \frac{\dot{q}}{\|\dot{q}\|} \in \mathbb{H}_1$.
Considering four parameters e_0, e_1, e_2, e_3 as Euler axis-angle parameters, describing a rotation about some axis u, with:

$$e_0 \equiv \cos\left(\frac{\theta}{2}\right)$$
$$\mathbf{e} \equiv \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = u \sin\left(\frac{\theta}{2}\right),$$

we can write this as a quaternion: $(e_0, \mathbf{e}) = e_0 + e_1 i + e_2 j + e_3 k$ and substitute e_i with q_i for $i = 0, 1, 2, 3$ to result in the previous notation of quaternions [143]. For

$$\dot{q} = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right) u$$

with $\|u\| = 1$ that $\|\dot{q}\| = 1$, then the sandwich product $\dot{q}p\dot{q}^\star$ rotates the point p around the axis u by the angle θ, where $\dot{q}^\star$ is the conjugate defined as $\dot{q}^\star = q_0 - q_1 i - q_2 j - q_3 k$.
One advantage of quaternions is that only 4 values are needed to represent a rotation rather than 9 as for a rotation matrix [55, 113]. Furthermore, the gimbal lock of Euler-angles is avoided, they can easily be projected to $SO(3)$ by normalization, the concationation of several rotations is cheaper[71] and interpolation can be performed efficiently [23].

[3]Sir William Rowan Hamilton, Irish Mathematician, *Dublin, Ireland 04.08.1805, †02.09.1865 [14]

4.4.5 Dual quaternion blending

Dual quaternion blending (DQB) is a sophisticated method for interpolating the new position of points with unknown transformation with respect to points with known transformationss (rotation and translation). Dual quaternions - more precisely unit dual quaternions - are a convenient representation of rigid transformations accounting for rotation and translation [23]. For the interpolation process, each point in the model is associated to certain anchor points and weights. DQB is commonly used for mesh deformations e.g. for character animations [69]. Figure 4.20 illustrates such an application, where the transformation of the underlying skeleton is known. Each vertex of the mesh is associated with a certain weight to the corresponding part of the skeleton as anchor points. As the skeleton is transformed, the rest of the mesh deforms accordingly using the efficient parametrization with dual quaternions.

In the scenario presented here, the anchor points for DQB are not as clearly determinable as for an underlying skeleton. The deformation nodes (see section 4.4.3) are the basis for DQB, as they abstract the point-wise transformations, which are estimated by the minimization scheme discussed in the next section (see section 4.5), between two point clouds in order to align them. Similar approaches have been proposed by Newcombe et al. [99] and Dou et al. [34] for RGB-D applications. Admittedly, they use a denser graph of deformation nodes. The aim is to deform the whole point cloud according to the reliable sparse deformation nodes from known correspondences, while reducing the computational effort of non-rigid alignment.

As already mentioned, rigid transformations can be concisely expressed by unit dual quaternions having many benefits compared to other representations [71]. Dual quaternions were already introduced in 1873 by Clifford [26]. They consist of two unit quaternions represented as dual number[4]:

$$\hat{q} = \dot{q}_r + \dot{q}_d \epsilon$$

where $\dot{q}_r$ is the unit quaternion denoted as real part and $\dot{q}_d$ the unit quaternion denoted as dual part [71].

As presented by Shoemake [121], quaternions can be utilized to interpolate rotations

[4] Dual numbers are numbers $z = x + \epsilon y$, where $x, y \in \mathbb{R}$ and $\epsilon^2 \neq 0$ [142]

easily, which by the incorporation of dual quaternions can be extended to rigid transformations with rotation and translation [23]. In general quaternions cannot describe non-rigid deformations with shearing and scaling [69]. However, we consider the transformations for single point correspondences, which can be described by individual per point rotations and translations in 3D. The non-rigidity is implicitly encoded within the whole set of local transformations, as each deformation node constitutes an individual rigid transformation [99], other points in between are interpolated by DQB.

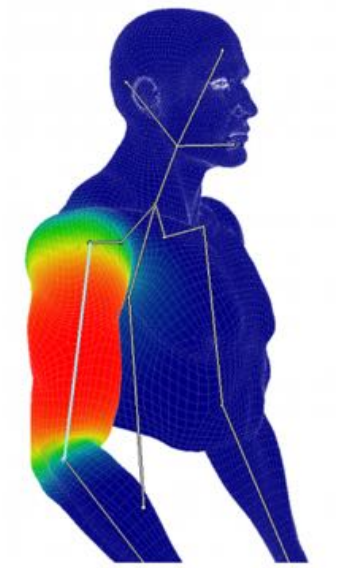


Figure 4.20: Example of DQB for character animation: Each vertex in the mesh is associated to the underlying skeleton with a certain weight; According to the movement of the skeleton, the rest of the mesh will deform; Adopted from [137].

The DQB is defined as:

$$DQB(p_c) \equiv \frac{\sum_{k \in N(p_c)} w_k(p_c)\,\hat{q}_{kc}}{\left\| \sum_{k \in N(p_c)} w_k(p_c)\,\hat{q}_{kc} \right\|} \tag{4.9}$$

where p_c is some 3D point or vertex of the point cloud under consideration without known local transformation, $N(p_c)$ the k-nearest neighbors (knn) around p_c in the deformation graph, $w_k(p_c)$ the weight of p_c for each of the associated k anchor points in the deformation graph, and $\hat{q}_{kc} \in \mathbb{H}_1$ the k unit dual quaternions describing the rigid transformation of the deformation nodes within the knn range [69, 99].

Finding appropriate weights can be challenging [4], especially when no proper meshes of the point clouds are available. This issue has been addressed differently. Dou et al. [34] (Fusion4D) rely on fixed weights, whereas Newcombe et al. [99] (DynamicFusion) take the distance between deformation nodes and the interpolated point into account. Similar to the as-rigid-as-possible approach for neighboring points [125] (see section 4.5.1), cotangent weights [86] could be established [137]. However, this is not suitable for the deformation graph considered here due to sparsity which requires a simpler distance measure. As discussed in [97] where different weights are examined for Laplacian mesh optimization, the

weighting scheme choice definitely does have influence on the mesh deformation. However, they are mainly identified for very detailed reconstruction. Thus, it is an acceptable assumption to rely on a simple weighting scheme based on the distance similar to [99].

4.4.6 Common model representation and fusion

The truncated signed distance function (TSDF) is a volumetric surface representation where each voxel retains the signed distance to the nearest surface of the 3D object [108]. This data structure can be used to efficiently represent and extract the reconstructed scene [64, 101]. The fusion of multiple reconstruction frames from different poses into a common TSDF model is inspired by Curless and Levoy [27]. Newcombe et al. [101] already employed this method for rigid reconstruction and later extended it for non-rigid deformations [99], which is also used in a modified approach by Innmann et al. [64]. Fusion4D [34] modified the fusion of frames, accounting for large deformations by reseting the current volumetric model representation. We will integrate new depth information similar to [99] into a TSDF. Non-rigid deformations are accounted for as discussed in sections 4.5 and employed as already described in section 4.4.3. Previous reconstruction frames are deformed according to the deformation graph and relevant voxels in the TSDF are updated. The structure of the TSDF will be based on an octree to reduce complexity for sparsely represented areas within the scene [87, 146].

4.5 Non-rigid deformation minimization

For non-rigid reconstruction the obtained point clouds, which will be continuously fused together in a common canonical model representation (see section 4.4.6), need to be aligned with the current point cloud according to the non-rigid and rigid transformations. As introduced by Sumner et al. [131] and applied in recent approaches [34, 64, 99], a deformation graph can describe the local transformations between one point cloud and a deformed one. To construct such a deformation graph, correspondences between the two point clouds need to be established. Contrary to referred approaches using RGB-D cameras [34, 64, 99], explicit sparse point correspondences are found as described in section 4.4.2. By computing the necessary local transformation for alignment of the deformation graphs in 3D, the rest of the non-deformed point cloud can be transformed accordingly via DQB (see section 4.4.5) with the nodes of the deformation graph as anchor points (see section 4.4.3).
We aim to find per point local transformations for alignment between point clouds in order to deform the current point cloud and further the canonical model non-rigidly to the current observation. Therefore we propose the following non-linear functional similar to [34, 64, 99] for RGB-D and [153] for RGB cameras:

$$E_{total}(V) = E_{Reg}(V) + E_{Temp}(V) + E_{ARAP}(V) \tag{4.10}$$

where $E_{Reg}(V)$ accounts for spatial regularization across all vertices $v_i \in V$, $E_{Temp}(V)$ encourages smooth deformations (see 4.5.2), and $E_{ARAP}(V)$ ensures local deformations to be as-rigid-as-possible (see 4.5.1).
All terms will be discussed in the following.

4.5.1 As-rigid-as-possible

Given an undeformed shape V and the shape V' after deformation, we wish the deformation to be as-rigid-as-possible [125] as presented in other non-rigid reconstruction approaches [34, 64, 99, 153]. We introduce the ARAP-term in our model, to allow for small local transformations of vertices v_i during deformation while restricting them to be as rigid as possible. This is necessary for regularization of the non-rigid reconstruction problem [64]. In particular we want to align two deformation graphs which explain the overall deformation of a point cloud to another (see 4.4.3).

If the deformation of a small cell - the one-ring neighborhood around v_i - is rigid, there exists a local rotation R_i which satisfies:

$$v'_i - v'_j = R_i \left(v_i - v_j \right)$$

$\forall j \in \mathbf{N}(i)$, where $v_i \in \mathbb{R}^3$ is some vertex position of the mesh V before, and v'_i after deformation with the mesh V', $\mathbf{N}(j)$ the one-ring neighborhood around the vertex and R_i some small local rotation [125].

We can minimize the condition from above, finding the best local rotations for alignment. This allows for small non-rigid deformations while preventing and limiting large local deformations. We get the least squares problem for all vertices $v_i, i = 1...n$ [125]:

$$E_{ARAP}\left(V'\right) = \sum_{i=1}^{n} \sum_{j \in \mathbf{N}(i)} w_{ij} \left\| \left(v'_i - v'_j \right) - D_i \left(v_i - v_j \right) \right\|^2 \tag{4.11}$$

where w_{ij} is a weight between the considered points (will be discussed below) and D_i are local per point rigid transformations. Please note that in the original paper of Sorkine et al. [125] only local rotations are considered. They refer to Horn [61], who uses unit quaternions to represent those rotations.

Also Yu et al. [153] only account for the case of local rotations without translation. In Fusion4D [34] rigid transformations with rotation and translation are considered. This seems more meaningful, since local transformations may not be reduced to rotations only. Furthermore, obtaining local transformations with rotation and translation for each deformation node is desirable for proper utilization of DQB.

Again, determining the associated weights is difficult in our case, as discussed in section 4.4.5. A cotangent weight [86] as proposed by [125] is not applicable [1]. Sorkine et al. [125] argue the weight is important to make the energy minimization independable from the underlying non-uniform mesh. They only provide a qualitative comparison to naive constant weights. DynamicFusion [99] and Fusion4D [34] both use a simple distance based weight as will be used here, as well.

4.5.2 Regularization

Besides accounting for local transformations being as-rigid-as-possible with the ARAP-term, spatial and temporal smooth deformations need to be assured [153].

4.5.2.1 Spatial regularization

The spatial regularization term ensures smooth deformations between V and V' by taking the local neighborhood of surrounding vertices into account.

$$E_{Reg}\left(V'\right) = \sum_{i=1}^{n} \sum_{j \in \mathbf{N}(i)} w_{ij} \left\| \left(v'_i - v'_j\right) - \left(v_i - v_j\right) \right\| \tag{4.12}$$

While the ARAP term 4.11 accounts for possible local point-wise transformations between frames to be as-rigid-as-possible, the spatial regularization encourages smooth deformations for the neighborhood of points. w_{ij} again accounts for the weight between neighboring points. Figuratively speaking, this can be thought of a measure how tight points are coupled together. The closer points are, the stronger their link is.

4.5.2.2 Temporal regularization

For reducing drift, guarantee smooth deformations across time and avoiding frame-to-frame flickering, the overall deformation between all vertices V' at time t and $t-1$ are regularized.

$$E_{Temp}\left(V'\right) = \sum_{i=1}^{n} \left\| \left(v_i'^{t} - v_i'^{t-1}\right) \right\|^2 \tag{4.13}$$

The temporal regularization of the shape can be interpreted by the assumption, that deformations between consecutive frames are only small.

4.6 Tracking and reconstruction pipeline

At the beginning of this section an overview of the proposed pipeline for monocular tracking and reconstruction in non-rigid environments is given (see section 4.6.1). We will see how presented methods are integrated into the pipeline in more detail in section 4.6.2. An in depth discussion of the implementation follows and is illustrated in a flow diagram in section 4.6.3. Furthermore, encountered difficulties and considerations of the implementation and its specialties, particularly with regards to non-rigidity, are examined. We will follow a coarse to fine approach for describing the pipeline.

4.6.1 Overview of pipeline

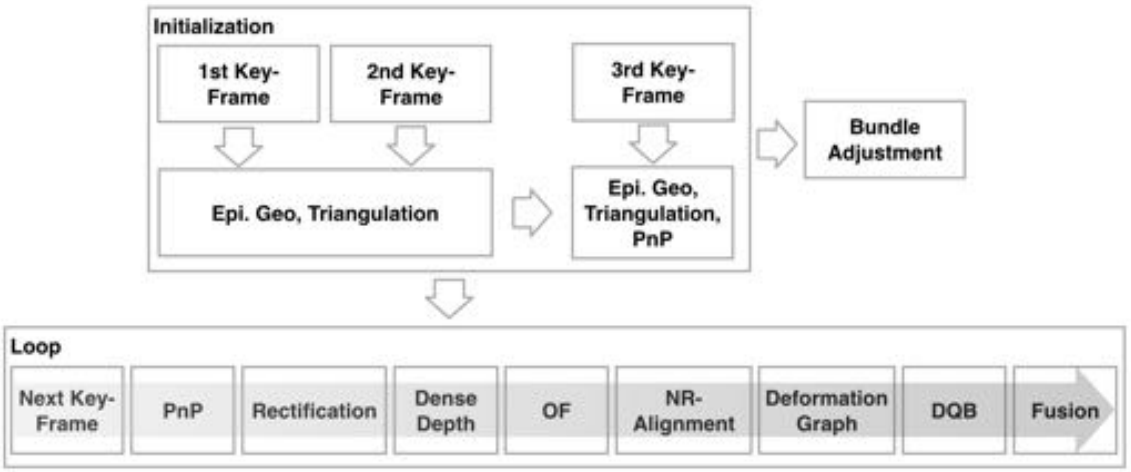


Figure 4.21: Overview of proposed pipeline

The pipeline consists of an initialization phase for tracking, with a subsequent bundle adjustment for the first three key frames, and a loop for continuous tracking and reconstruction. For the initialization of camera poses, the epipolar geometry enables to recover the relative transformation based on robustly matched ORB features. A sparse triangulated reconstruction of these feature points provides 3D-2D point correspondences. Features across the second and third key frame can be matched and the 3D-2D correspondences can be associated with the third frame for pose estimation via PnP. The successive bundle adjustments helps to further refine the camera poses, whereas the first camera is kept at the origin of the world reference frame.

With initial camera poses and the rudimentary sparse reconstruction we can start the continuous tracking and reconstruction loop, whereas further camera poses can be computed via PnP. With rectified key frames dense depth information can be computed. An optical flow based radius feature matching gives the foundation for a sparse feature based deformation graph. The per-point local transformations are computed across key frames by non-rigid alignment and the deformation graph is constructed for DQB. After deforming the dense 3D point cloud accordingly, respective voxels in the TSDF are updated. New depth information are integrated into a common TSDF. A reconstruction of the non-rigidly deforming scene can be obtained via marching cubes of the TSDF.

4.6.2 Details of pipeline

4.6.2.1 Details of initialization

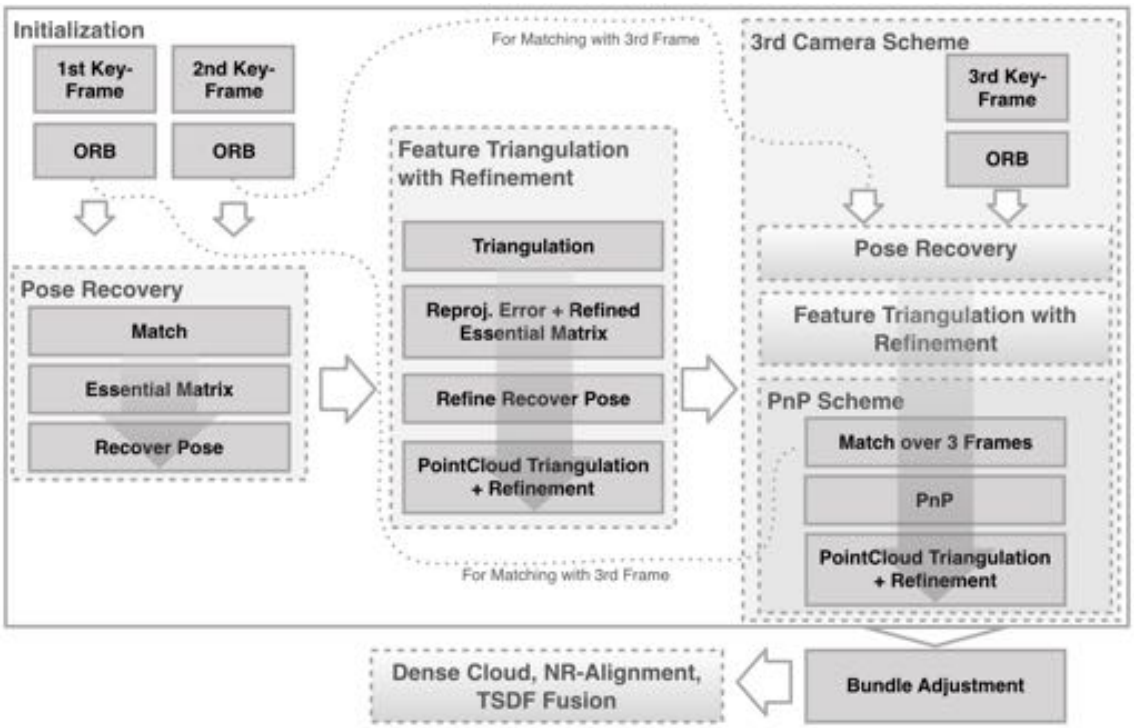


Figure 4.22: Details of proposed pipeline (Initialization)

Pose Recovery: ORB features are extracted from the first two frames (4.1.1). Matched features (4.1.2) are used to estimate the epipolar geometry (4.2) by computing the essential matrix with the 5-point algorithm (4.2.2). With the essential matrix, the relative camera pose can be obtained up to scale (4.2.3).

Feature Triangulation with Refinement: The first two camera poses and matched features can be used to triangulate an initial sparse reconstruction (4.3.2). Outliers can be identified by individual reprojection error. Inliers are used to compute a refined essential matrix. From the refined essential matrix, also a refined camera pose is estimated. Now, robust initial camera poses can be assumed and the remaining feature point inliers are again triangulated for sparse reconstruction.

3rd camera scheme: *Pose recovery* and *Feature Triangulation with Refinement* is also performed between frames two and three for reliable feature matches.

PnP Scheme: Matches between all three frames are compared whereas only feature points with a triangulated point between frames one and two are considered subsequently. Thus, sparse 3D-2D correspondences between image points of the third frame and the sparse reconstruction from key frame one and two can be established. Given the correspondences, the third camera position can be computed relative to the scene via PnP (4.3.1).

Bundle Adjustment: Since the reliable initialization of the first three camera poses is vital for subsequent tracking and reconstruction, we optimize the computed camera positions by minimizing the reprojection error of the sparsely triangulated feature points.

4.6.2.2 Details of continuous tracking and reconstruction

Continuous Camera Tracking and Mapping: With initialized camera poses for the first three key frames, there is no need for the computation of the epipolar geometry for tracking. Robust feature matches can be obtained between the new key frame and the two old ones. A sparse reconstruction can be triangulated with the already computed camera poses for key frame $i-2$ and $i-1$ (outlier filtering by reprojection error).

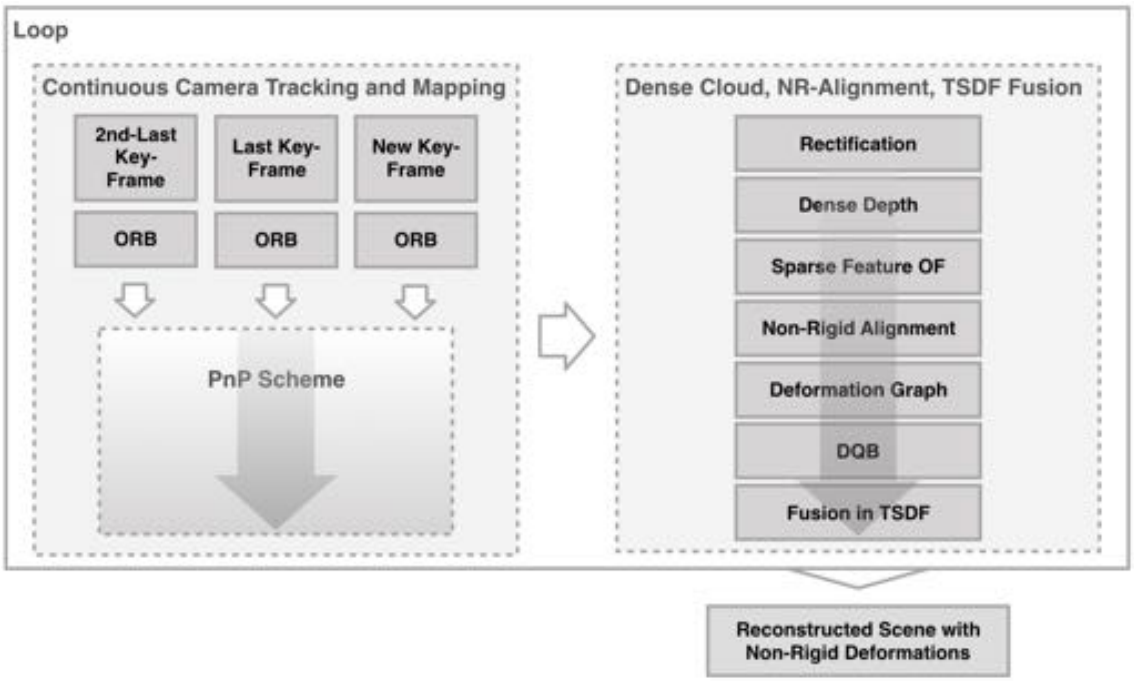


Figure 4.23: Details of proposed pipeline (Continuous tracking and reconstruction)

3D-2D correspondences are associated with the new key frame i and the camera pose is computed via PnP scheme. The camera pose is further refined by bundle adjustment over the last three key frames.

Dense Cloud, NR-Alignment, TSDF Fusion: With known camera poses and intrinsics the images can be rectified as for the calibrated case (4.2.4.2). By SGM of the rectified images dense depth information is obtained (4.3.3). Dense 3D point clouds are computed and transformed into the world coordinate frame (4.3.4). Between consecutive virtual stereo setups an optical flow based radius feature matching (4.4.2) is performed. Unlike previous feature matching with extensive outlier removal, we now also want to find parts in the scene which transform non-rigidly and therefore would not be considered as inliers of our rigid feature estimation approach. Therefore, we estimate the sparse optical flow for feature points from one image to the next. Within a specified radius around the obtained displaced pixel coordinates we find the best match with feature points in the next image (4.4.2). Matched features can be triangulated to a sparse representation of the scene for rigidly and non-rigidly transforming points. This

procedure is performed between consecutive stereo pairs and explicit correspondences are established (e.g. between key frame $i-2$ and $i-1$, and $i-1$ and i) resulting in deformation graphs for each frame (4.4.3). For explicit correspondences a non-rigid alignment is performed by minimizing their distance in 3D space, together with sophisticated regularization (4.5). Sparse per-point local transformations between consecutive deformation graphs constitute the embedded deformation information for the rest of the dense point clouds, which is deformed via DQB (4.4.5). New depth data is integrated into the common TSDF scene representation. After bringing old depth information into alignment with the current shape, respective voxels in the TSDF are updated (4.4.6). At the end of the pipeline, a dense reconstruction can be obtained from the TSDF, e.g. by marching cubes.

4.6.3 Realization and implementation details of pipeline

4.6.3.1 Tracking initialization

In the following all parts of the pipeline will be discussed in detail. To ease the reading process references to respective sections will not be given, since they have already been presented. Also note, that some descriptions have partly been delineated already and are now discussed in more detail, following a coarse to fine approach.

Extract ORB: Extract ORB features in a coarse to fine approach with multiple pyramid levels. Features below certain score (Harris score) are not considered.

Feature matching: Right-left and left-right matching for outlier removal. Use a geometric constraint for further outlier removal: geometric distance should not be larger than the image size. After outlier removal, compute the average geometric distance between matched feature points and remove those below and above a certain threshold.

Compute essential matrix: Initialization of RANSAC is not determinable and computation of essential matrix dependent on outliers. Therefore, the essential matrix is computed iteratively until our criteria (see later) are reached. Firstly, compute initial essential matrix. Secondly, remove outliers from RANSAC scheme from set of matched

feature points and compute the refined essential matrix again with RANSAC. Get the number of remaining inliers and compute ratio between initial feature matches and inliers. Compute the average epipolar error (average SSD between points and epipolar lines). Accept refined essential matrix if both error measures are below threshold.

Recover pose (from essential matrix): Take the refined essential matrix and compute the relative camera pose by decomposition for rotation and translation. Perform cheiriality check [52] for correct pose. Triangulate feature match inliers for sparse reconstruction. Filter outliers based on reprojection error and compute the relative camera pose again on reduced set of inliers. Again, compute R and t and perform cheiriality check. Triangulate remaining set of feature matches with estimated camera pose and save final sparse reconstruction for further processing.

3rd camera scheme: For the third camera the epipolar geometry and pose recovery is computed just as for the first two cameras, in order to obtain robust feature matches. The resulting feature matches are compared with those between key frames one and two, and matches for identical points in image two are considered as inliers. For all inliers the corresponding triangulated points from the previous step are utilized for 3D-2D correspondences between the sparse reconstruction and image points in the new image. Given the correspondences, the third camera pose, with regards to the reconstruction and within the world reference frame, is solved via EPnP algorithm. The computed camera pose is accepted if the average reprojection error is below a certain threshold. To further refine the camera poses, a bundle adjustment is performed. We implement the bundle adjustment (with the Ceres solver) such that the reprojection error is minimized by only optimizing the third camera pose and 3D points, while fixating cameras one and two.
Please note, that a robust initialization of the tracking system is vital for further camera pose tracking as well as reconstruction. Therefore, we spend quite some computational effort for outlier removal and camera pose optimization to avoid ill-posed camera initialization. Despite substantial uncertainties induced by the nature of vision based tracking systems, especially in non-rigid environments, we can obtain robust rigid feature matches by removing outliers based on several error measures in combination with RANSAC schema. Also note, that the methods proposed here are able to compute the epipolar geometry and thereafter the camera poses in a very robust way, also being

able to handle planar and non planar scene. Computing the epipolar geometry by the fundamental matrix may not be advisable [157]. The result is prone to outliers and number of considered points [53, 157]. Furthermore, given planar point correspondences (as might be the case for points in the background), the planar degeneracy prohibits from computing a suitable fundamental matrix [53]. These limitations are circumvented by computing the essential matrix with calibrated cameras. Please note, that Mur-Artal et al. [93] also account for degenerated cases in their robust ORB-SLAM method. They compute both the fundamental matrix and a projective homography, and decide based on an error ratio which one describes the scene best. This is one of the advantages of ORB-SLAM, which makes them more robust compared to many other feature based SLAM approaches.

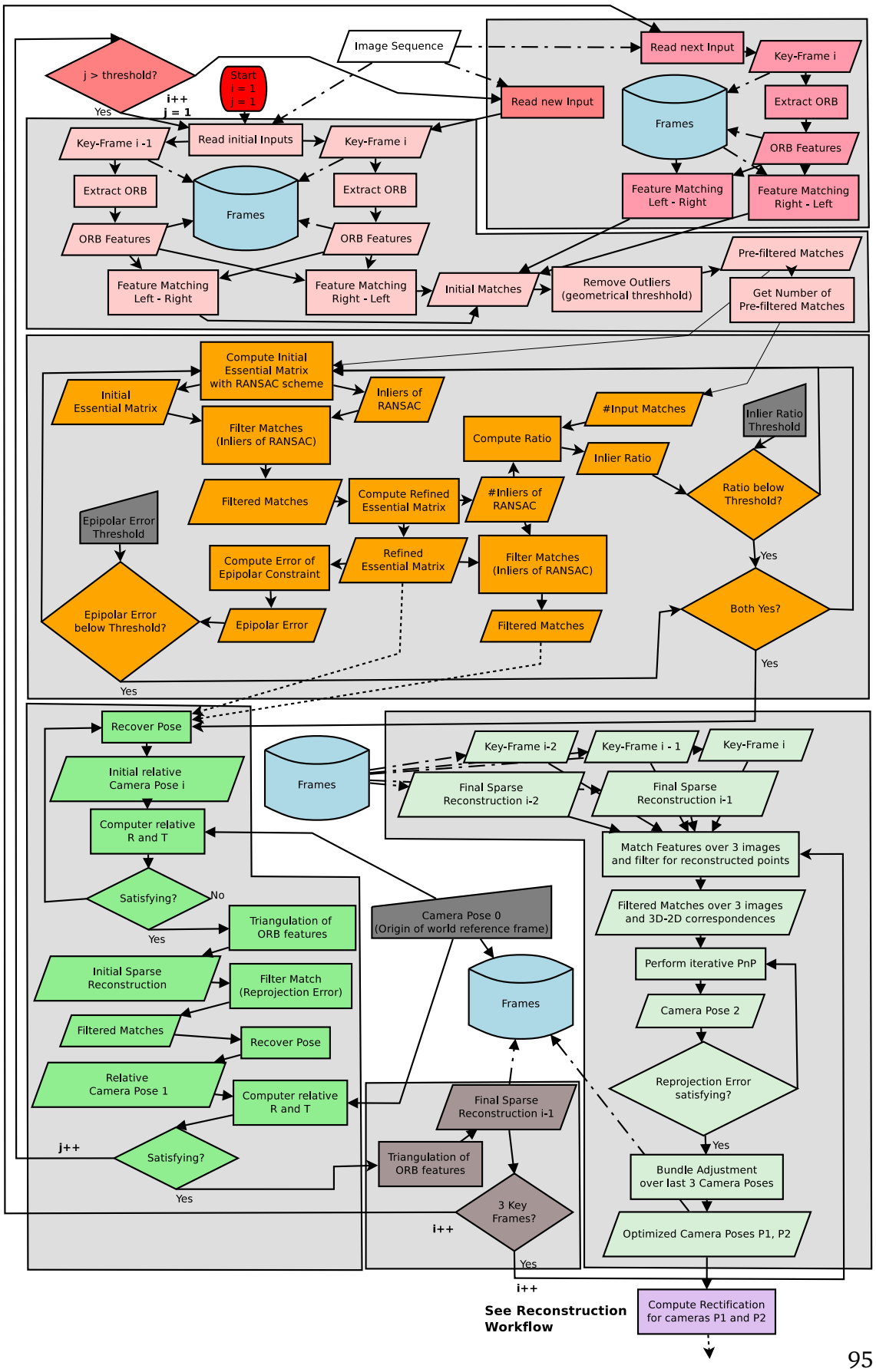


Figure 4.24: Flow diagram for tracking initialization: For details please refer to the text.

4.6.3.2 Continuous tracking

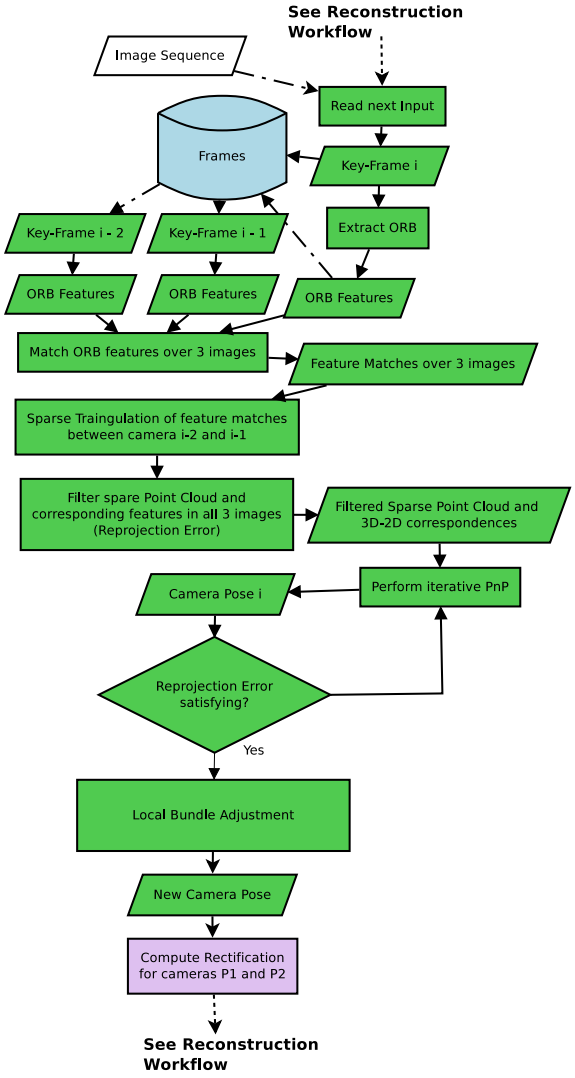


Figure 4.25: Flow diagram for continuous tracking: For details please refer to the text.

PnP scheme: Given ORB features in three consecutive images, they can be matched across these images, similar to discussed above. Matches between the last two cameras ($i-2$ and $i-1$) can be triangulated with known camera poses in the world reference frame. Based on the reprojection error outliers can be removed from the sparse reconstruction and the sets of matched features for all three frames. Given the 3D-2D

correspondences, we can again compute the new camera pose via EPnP.
We have found that most of the implemented methods for PnP in OpenCV [105] do not compute reliable camera poses. Also the incorporation of RANSAC schema for outlier removal does not bring notable accuracy improvement. The iterative algorithm for PnP, minimizing the reprojection error based on Levenberg-Marquardt optimization fails completely with uninitialized camera poses. Presumably this is due to estimating an initial camera pose by DLT [53] internally, and subsequent divergence to local minimum. When providing an adequate initial camera pose (e.g. pose of last camera), the iterative method showed good accuracy for some cases, probably when only selecting planar points during RANSAC. Since EPnP accounts for planar and non-planar points, we took this as the method of choice. However, camera poses are often still far off correct values. As for the the camera pose initialization, a sparse bundle adjustment over the last three frames, whereas now we keep the first two camera poses and the 3D points constant, thus optimizing the third camera pose, is performed as well.

4.6.3.3 Reconstruction, non-rigid alignment and fusion

Rectification: With given camera poses, the virtually rectified cameras as for the calibrated case are computed. During rectification the reprojection matrix Q is also computed. The baseline of the rectified cameras, as well as the mapping of the rectified images is computed as error measures.

Dense depth: Since disparity maps do not show compelling results for SGM, a filtering scheme, already proposed as contribution module to OpenCV, is introduced. It is based on a weighted least squares filtering and yields high qualitative and fully dense disparity maps (compare Figure 5.7). Disparity computation is performed from the left to the right image and vice versa. The block size can be set to be very small enabling high details since filtering still accounts substantially for undefined pixels.
The filtered disparity map can now be projected into 3D with the reprojection matrix. Outliers with very high depth values are considered as outliers. Finally, the dense cloud is transformed from the virtually rectified camera frame into the original one (left camera frame is always assumed). For simplification during rectification we assumed one of the cameras to be at the origin of some reference frame and only accounted for relative displacement of the second camera. Therefore, the point cloud still needs to

be transformed into the global world reference frame by the absolute world camera pose of the left camera. This transformation is essential to properly align the point clouds within the world reference frame. Now the point cloud can be integrated into the common canonical represented in a TSDF.

Optical flow based radius feature matching: Due to our extensive outlier removal during feature matching, only points with rigid transformations were considered in the previous methods. Since we also want to account for deformations in the reconstruction, we propose a feature matching which is based on prior sparse optical flow computation. We use the sparse LK optical flow algorithm to compute the displacement for all feature points between consecutive images. Estimated flows for points above the so called spatial gradient matrix error [9] are considered as outliers.

At the displaced pixel coordinate we search for all available feature points within a specified radius. We sort the matches based on their Hamming distance (not geometrical distance) and take the best match. Matched features are triangulated and outliers are filtered based on the reprojection error. Since here we do not assume precise reprojections due to non-rigidity, the reprojection error is rather large, just accounting for badly estimated optical flow and subsequent matched features.

Deformation graph: The optical flow based radius feature matching can be performed across multiple frames (e.g. frame $i-2$ to $i-1$ and frame $i-1$ to i). We can search for feature matches present in both matching sets from frame $i-1$. Thus, we get a set of sparse 3D correspondences with rigid and non-rigid transformations. They represent the deformation between consecutive point clouds and constitute the embedded deformation graph. To account for general rigid displacement between the sparse clouds (e.g. due to depth uncertainty from triangulation with small baselines) they are firstly aligned by rigid point-to-point ICP.

The decoupling of rigid ICP and non-rigid alignment proved to be meaningful, since convergence has been faster with individual optimization, and the general rigid transformation tend to be very small, as the coupled non-rigid part accounted well enough for alignment. This behavior would require substantial and hardly determinable weighting of the respective terms.

Non-rigid alignment: With pre-aligned deformation nodes, we now compute local per point transformations which bring them into alignment. To regularize the non-linear optimization problem, we introduce the ARAP term, limiting local transformations being as-rigid-as-possible together with spatial and temporal regularization. Since we need to establish the local neighborhood of the deformation nodes for our regularization, we search for knn neighbors in the cloud with the help of an kd-tree. Only points within a certain geometric distance are considered as neighbors. Points with too few neighbors will not be processed further. For ARAP we also find a weight, based on the distance between the point and their neighbors. This ensures points are closely together to deform similarly. We now optimize our proposed energy term (with Ceres solver) and find local per point transformations for all deformation nodes.

DQB: For each deformation node we initialize a unit dual quaternion with the rotation and translation as computed from above. We fill a kd-tree with the previous dense point cloud and the deformation graph and find their respective knn deformation nodes within a radius threshold for all reconstructed points. A weight is associated based on the geometric distance between the query point and the deformation node. DQB is performed for all suitable points. The same is done for the current reconstructed canonical model (for details see below: Fusion in TSDF).

Fusion in TSDF: In order to achieve a fully dense reconstruction, consecutive depth information is integrated into a common model represented in a TSDF. As deformations may occur, it is not applicable to simply add new point clouds to the model. After deformation, the already integrated depth information may not comply with the current scene. Therefore, we extract a mesh form the current view of the model and convert it to a point cloud. The point cloud is deformed via DQB and transformed 3D points are updated in the TSDF. The previous point cloud is also deformed and integrated into the TSDF. The current point cloud can easily be integrated as well. Thus, we get a dense reconstruction of the scene while accounting for deformations. Transformed parts of the reconstruction are sufficiently represented in the TSDF by updating the model, deforming the old point cloud and integrating the new one. Depending on the weight of a marching cubes algorithm we can extract detailed reconstructions.

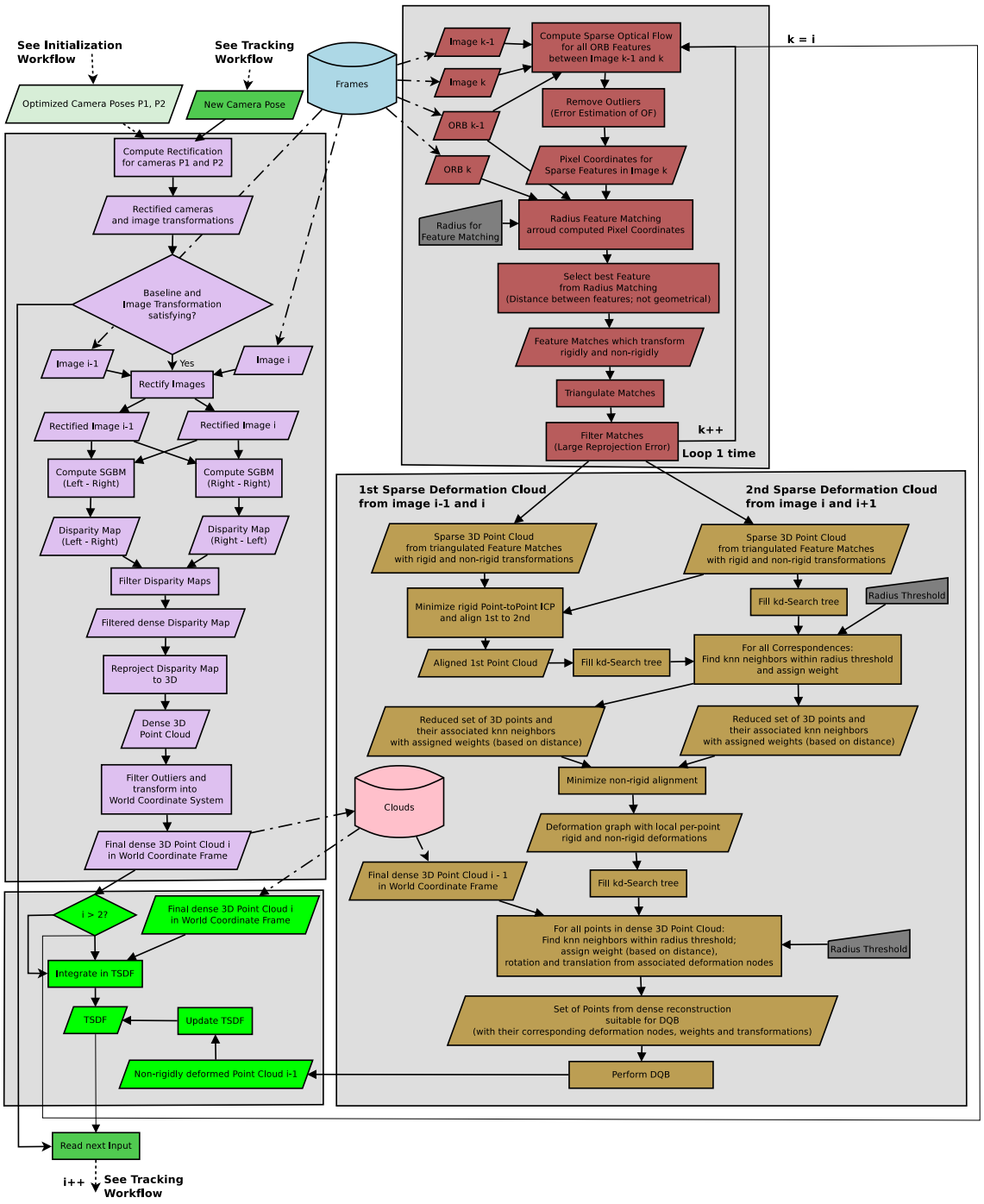


Figure 4.26: Flow diagram for reconstruction and non-rigid alignment: For details please refer to the text.

4.7 Summary

We can now estimate the relative camera pose, rectify stereo images, find correspondences and compute depth information for a stereo system. These ideas can be elevated to multiple cameras, in our case to a single moving camera, taking images from multiple points of view. In literature this is often referred to as multiple view stereo (MVS) [120] or structure from motion (SfM) [98]. Taking the information of a scene from multiple viewpoints into account, we can now estimate the actual structure of a scene or object more accurately (compare Figure 4.27).

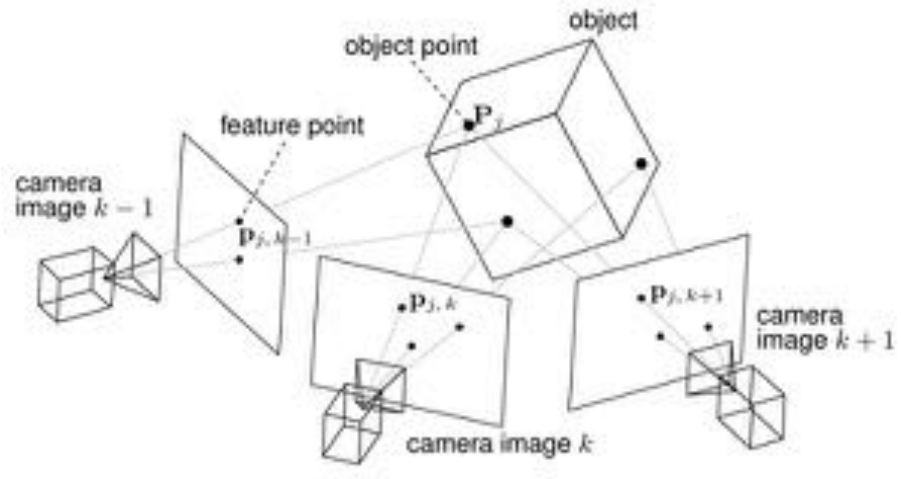


Figure 4.27: Object seen from multiple camera poses: Over time an object is seen from different camera poses; Now the object can accuratley be reconstructed by fusing individual reconstructions into a common model; Adopted from [75]

A sophisticated feature matching and outlier removal to obtain rigidly transforming feature points for camera pose tracking has been proposed. PnP algorithms and a suggested sparse local bundle adjustment consecutively compute new camera poses. High quality and fully dense disparity maps are extracted by stereo matching and enable detailed 3D reprojection of the observed scene with a freely moving monocular RGB camera. An optical flow based feature matching approach enables to extract rigidly and non-rigidly transforming points from the scene as a basis for an embedded deformation graph. Deformation of the remaining model is proposed to be performed with this underlying graph via DQB. Results are fused within a common octree based TSDF to improve reconstruction results, integrate depth information from different camera

poses, and retrieving the final reconstruction via marching cubes, while minimizing memory footprint and computational effort for surface extraction.

5 Results and discussion

Results for camera tracking and reconstruction obtained from the algorithm presented here are given and discussed in the following. In particular we analyze results from a self obtained dataset with non-rigid movement of a person. While most of the scene is static, it is more challenging than one would presume at a first glance. Lightning conditions induce reflections, occlusions occur and large parts of the scene do not contain structure (e.g. white wall).
A quantitative evaluation of the camera pose tracking with known ground truth is presented. Challenges and issues leading to cases where the camera trajectory cannot be computed robustly are discussed. For the results of the reconstruction, only a qualitative presentation of results is meaningful, since no ground truth for deformations in the scene is available.
Due to the lack of publicly available and suitable datasets with appropriate camera movement, ground truth and eligible non-rigid deformations, the dataset presented was produced during the composition of this thesis.

With this new finding Dawn[1] has shown
that Ceres[2] contains key ingredients for life.
(Simone Marchi[3])

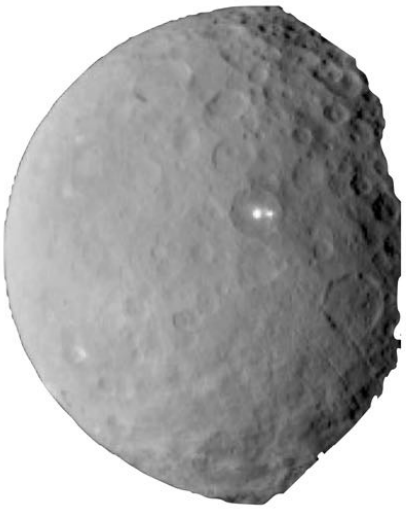


Figure 5.1: The dwarf planet Ceres: Photograph taken by NASA's Dawn spacecraft on February 19, 2015; Adopted from [134].

[1] Dawn spacecraft, launched by NASA in 2007 and has been orbiting Ceres since 2015 [31]
[2] Ceres, dwarf planet and the largest asteroid in the main asteroid belt; Named after the ancient roman grain goddess and patron goddess of Sicily [134]
[3] Simone Marchi, Senior research scientist at Southwest Research Institute [31]

5.1 Results

5.1.1 Quantitative evaluation of tracking

In Figure 5.2 the computed camera poses and triangulated sparse ORB feature matches are illustrated for the first six key frames of an image sequence. Parts of the original image are transparently embedded into the scene manually, to give a better idea which triangulated points correspond to which part of the input (for reference: input image at the right bottom). This example gives a first impression of the dataset we will consider in the following. Key frames were manually selected with a constant translation in x-direction only, to have a known ground truth for camera pose estimation. Non rigidity is induced by human movement. Data set was recorded during composition of this thesis.

Figure 5.2: Sparsely reconstructed scene with camera poses for six key frames: Transparent parts of the input image (see bottom right) are manually embedded.

For analyzing the tracking accuracy, we computed the camera poses for a short non-rigid scene with known offset between camera positions. Since pose recovery from the essential matrix, as it is done for camera pose initialization, only gives the translation up to scale (norm of translation vector is one), we define the ground truth as the delta between camera poses to be 1.0 (in x-direction). Figure 5.3 and Figure 5.4 give an

error analysis of camera poses for 16 and 11 key frames of the same scene, respectively. As absolute camera poses are not meaningful due to some unknown common scaling, the difference between absolute consecutive camera poses is considered. Ground truth for the delta between camera poses would be one in x-direction and zero for y- and z-direction. The standard deviation of the camera pose delta is illustrated as well as their mean values (refer to caption of figure for results). At the bottom, also the camera poses from the reconstruction are illustrated to give a visual impression. Initialization and continuous tracking was performed as described in section 4.23.

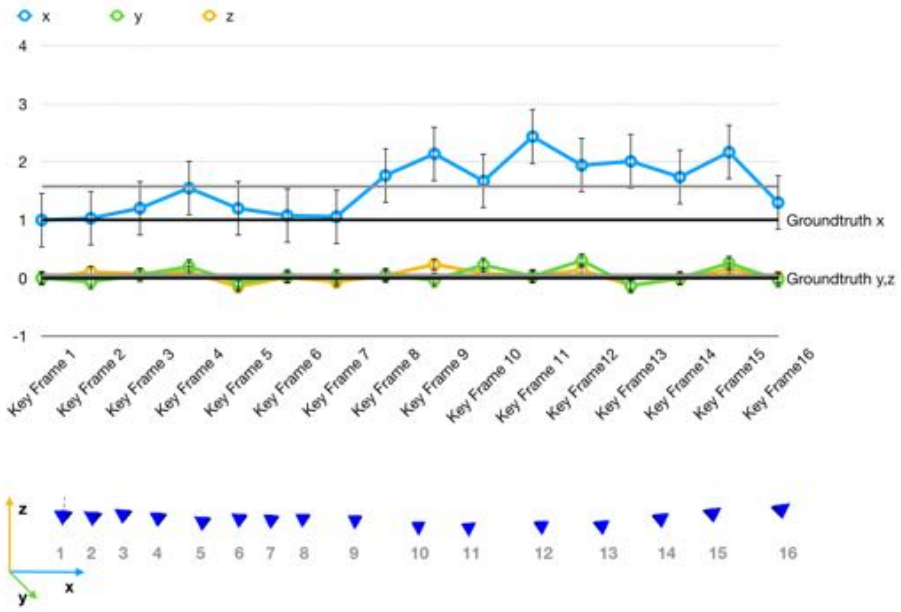


Figure 5.3: Trajectory error analysis for 16 key frames: Computed values for camera pose deltas are illustrated for each axis, as well as their standard deviation (error bars) and mean values (gray lines); Mean (x-dir.): 1.58, mean (y-dir.): 0.05, mean (z-dir.): 0.04; Std.dev. (x-dir.): 0.47, std.dev. (y-dir.): 0.13, std.dev. (z-dir.): 0.10

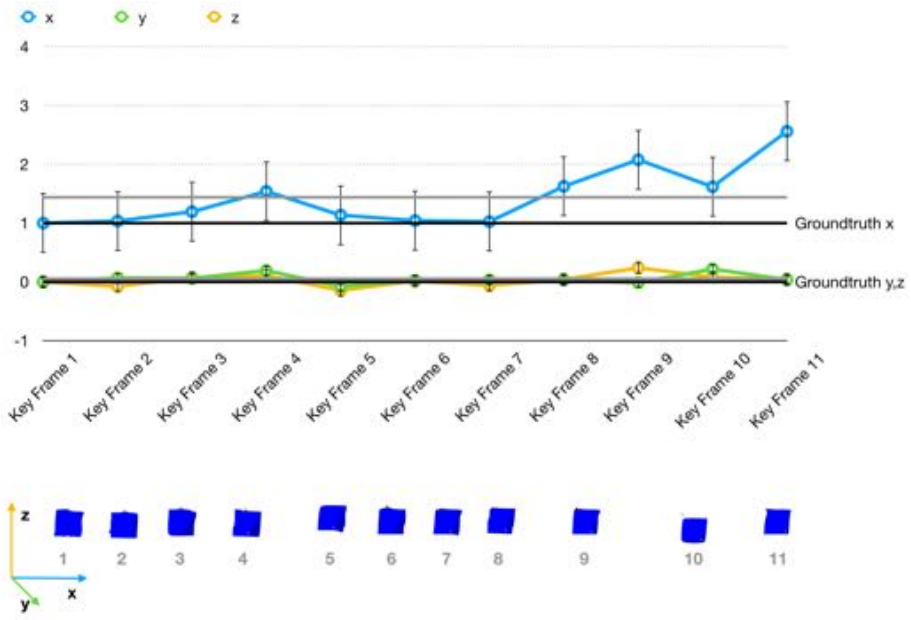


Figure 5.4: Trajectory error analysis for 11 key frames: Trajectory error analysis for 16 key frames: Computed values for camera pose deltas are illustrated for each axis, as well as their standard deviation (error bars) and mean values (gray lines); Mean (x-dir.): 1.44, mean (y-dir.): 0.05, mean (z-dir.): 0.03; Std.dev. (x-dir.): 0.51, std.dev. (y-dir.): 0.09, std.dev. (z-dir.): 0.10

Figure 5.5 illustrates the filtered matched feature points used for camera tracking with their displacement throughout three consecutive images.

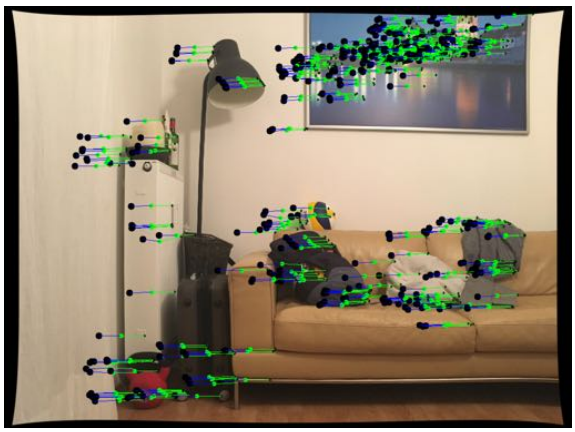


Figure 5.5: Displacement of tracking points: Feature points for tracking are projected into the same image; Displacements are indicated by their connecting lines.

As an example of poor camera pose estimation refer to Figure 5.6. Here, minor errors in the camera calibration lead to major drift and completely off camera poses. This illustrates the importance of correct camera intrinsics, since for pose computation reliable reprojection results are assumed. Obviously, a detailed error analysis is not meaningful for this trajectory.

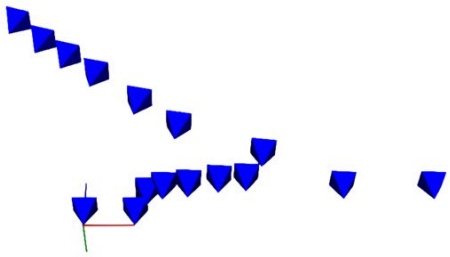


Figure 5.6: Example of poorly estimated camera poses: Since camera pose estimation and refinement relies on the minimization of the reprojection error, minor errors in the camera calibration can lead to wrong camera pose results.

5.1.2 Qualitative evaluation of reconstruction results

As already discussed in 4.3.4 and illustrated in Figure 4.19, a disparity map filtering with left-right and right-left disparity map incorporation was performed. See Figure 5.7 for a qualitative comparison between disparity maps before and after filtering. Clearly, filtering improves the quality of the disparity map which leads to more accurate reprojection results (compare Figure 5.8).

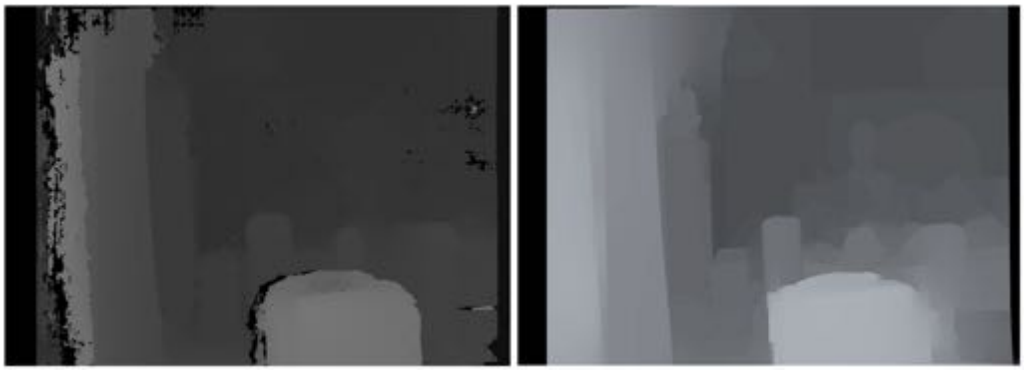


Figure 5.7: Comparison of disparity map filtering: Disparity maps from an identical scene before (left) and after (right) filtering.

Figure 5.8: Reprojection of fully dense disparity map and triangulated feature points (green): The scene projected is highly detailed and depth continuities are preserved.

Figure 5.9 shows results after fusing the first two computed point clouds into the TSDF. The reconstruction is obtained via marching cubes of the TSDF. The result shows holes and sparsity, since only two point clouds have been integrated so far.

Figure 5.9: Reconstruction after marching cubes of the TSDF with two integrated point clouds: Original input image is shown at the bottom.

Due to inaccuracies in camera pose estimation and subsequent stereo comparisons between key frames, the reprojected point clouds results suffer also from inaccuracies as illustrated in Figure 5.10. As common, reprojection results are influenced by the

accuracy of disparity computation. Here, not perfectly computed camera poses elevate the problem of reliable reconstruction as well. As depicted, consecutive point clouds from the reconstruction pipeline show deviations in alignment. This is not only due to rigid alignment errors, but rather because some unknown scale ambiguity is induced by uncertain camera poses. Stereo rectification is based on given camera poses from the tracking pipeline. Disparity maps are computed from given rectified images and virtually rectified cameras. Thus, given inaccurate camera poses, reprojection of the disparity maps will give uncertain results. Especially the baseline between camera poses has influence on the scale of the reprojection. This issue cannot be accounted for by rigid ICP alignment of the point clouds due to unknown scaling and ambiguity.

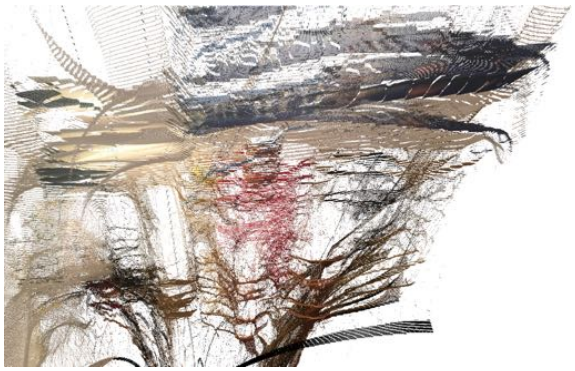


Figure 5.10: Erroneous point clouds from consecutive key frames due to uncertain camera poses: Induced by incorrectly estimated camera results, some unknown scale and reprojection ambiguity results in not aligned point clouds.

Figure 5.11 shows the reconstruction of a rigid scene. Some noise is noticeable, which has not been filtered out here, despite perfect camera poses. The noise suggests errors in the disparity computation.

Figure 5.11: Reconstruction of rigid scene with perfect camera poses and noise: As the marching cubes weight has been set to minimum, also voxels filled only once with noise are considered for reconstruction.

Figure 5.12 and Figure 5.13, show reconstruction results without accounting for non-rigidity. The moving head is poorly represented as old reconstruction results are not deformed via DQB based on the deformation graph. This weakness for reconstruction of non-rigidly transforming point clouds justifies the algorithm proposed here.

Figure 5.12: Reconstruction of non-rigid scene without deforming already obtained depth information appropriately: Subsequent point clouds are simply integrated into the TSDF without accounting for non-rigidity.

Figure 5.13: Reconstruction of non-rigid scene without appropriate deformation: Another example of bad reconstruction results without incorporation of non-rigid transformations.

The nodes of the embedded deformation graph, constituted by the optical blow based radius feature matching, are illustrated in Figure 5.14.

Figure 5.14: Nodes of the embedded deformation graph

After non-rigid alignment of consecutive deformation graphs, the rest of the reprojected

3D point cloud can be deformed accordingly via DQB. In Figure 5.15 a point cloud before (red) and after DQB is shown. One may notice, that parts of the body have moved, since the red points are not visible any more. Some parts of the cloud seem to have been deformed falsely. This may be due to minor inaccuracies during non-rigid alignment. The deformation of the point cloud is also shown in detail in Figure 5.16 and Figure 5.17 for a later key frame. One can easily see the movement of the head from its prior position (red point cloud).

Figure 5.15: Point cloud before (red) and after DQB: Moving body parts are deformed via DQB; Some parts of the point cloud seem to have been slightly transformed falsely.

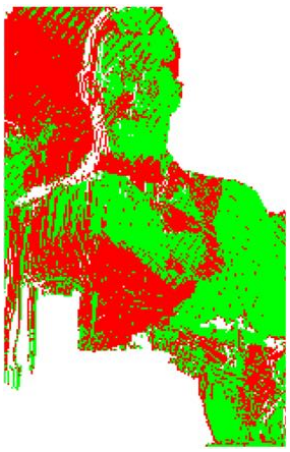


Figure 5.16: Detailed example of deformation via DQB: The red point cloud is deformed via DQB towards the green one; The displacement and deformation of 3D points is clearly visible, e.g. the left ear.

Figure 5.17: Detailed example of deformation via DQB(2): Similar to above with actual color information.

For proper reconstruction results of non-rigid scenes, the point cloud deformation via DQB as seen above can now be applied throughout an image sequence. Thus, old depth information is transformed to align with the current scene and respective voxels of the TSDF are updated. New depth information can further be integrated. See Figure 5.18 for such a reconstruction result. Please note, that here camera poses have been estimated by the proposed tracking approach. Thus, nose and not perfectly aligning

point clouds result from ambiguity and inaccuracies of camera poses. Despite these defects, the non-rigidly transforming parts, e.g. human head and arm, are adequately accounted for, as the reconstruction results suggests. Compare with Figure 5.12, where old point clouds are not deformed according to the movement of the head, resulting in odd shapes of the background. Details of the resonstruction are also shown in Figure 5.18 and Figure 5.20 for anotherrun of the algorithm.

Figure 5.18: Reconstruction of non-rigid scene: Despite non-perfect camera poses and resulting noise and reconstruction ambiguity, non-rigidly transforming parts in the scene are sufficiently accounted for.

Figure 5.19: Reconstruction of non-rigid scene (Details)

Figure 5.20: Reconstruction of non-rigid scene (Details 2)

As indicated by Figure 5.21, the weight between points and deformation nodes for DQB, proper initialization of local transformations, and the influence of each deformation node on the point cloud are important for adequate results of DQB. The circular shape of he clouds illustrate the large influence of individual deformation nodes. Additionally, induced by sparsity of the deformation graph, the deformed point cloud (green) shows deficient deformations.

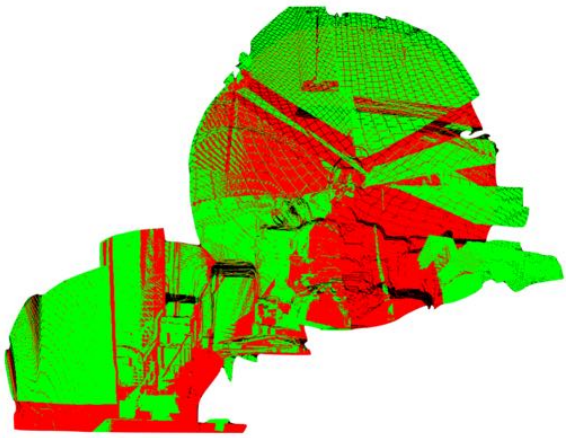


Figure 5.21: Bad results for DQB: Point cloud before (red) and after (green) deformation; The deformed cloud shows clutter due to inappropriate weights and too large distance threshold for deformation association.

5.2 Discussion and outlook

As already mentioned, it was difficult to obtain a publicly available dataset for monocular tracking with suitable non-rigid deformations and groundtruth, while given appropriate camera movement. It has been shown, that monocular reconstruction and tracking is extremely challenging, especially in non-rigid environments. Extensive and effective outlier removal has been proposed, making it suitable to track the camera movement robustly in non-rigid environments. Several error measures and filtering stages are incorporated. As Figure 5.2, Figure 5.3 and Figure 5.4 for camera pose tracking suggest, camera poses can be reliably computed on the sparse set of filtered ORB features. Drift of camera poses is a common problem for SLAM and frame-to-frame camera tracking. The qualitative illustrations of camera poses show appealing results.

With known camera poses, it has been shown how high quality disparity maps can be obtained. Initial disparity maps from SGM stereo matching are filtered in a left-right and right-left consistency assumption (compare Figure 5.7). With these fully dense disparity maps, detailed point clouds are computed by reprojection (compare Figure 5.8). They show a fully dense representation of the scene in high detail while preserving depth continuities and object boundaries.

As indicated by poor reconstruction results in Figure 5.12 for non-rigid transformations in the scene, it is necessary to consider such cases.

The optical flow based feature matching scheme accounts for rigid and non-rigid transformations of feature points. The resulting sparse deformation graph explains the underlying transformations well, and suitable deformation has been shown to be possible via DQB effectively and accurately on the reprojected point clouds of a monocular camera. This is affirmed by the results in Figure 5.15, Figure 5.16 and Figure 5.17 for comparisons of point clouds before and after deformation via DQB based on a sparse deformation graph.

The integration of consecutive results in a TSDF improves reconstruction results. Memory consumption is reduced by an octree based TSDF, which is also beneficial for the computational effort of marching cubes of the TSDF.

Drift of the camera poses is a common problem for SLAM. State-of-the-art approaches usually perform global optimization such as global bundle adjustment [93] and pose graph optimization with loop closure [36]. Such methods are not suitable for the ap-

proach presented here, since camera pose optimization after obtaining a large number of key frames would only optimize the sparse feature reconstruction. Projected 3D point clouds between key frames are continuously integrated into the common TSDF, thus they would not be optimized. However, one solution would be to first perform a full tracking of the image sequence, optimize the camera poses and then run the reconstruction in a second stage.
Camera tracking results have been shown for a challenging scene. Drift occurs quite naturally and is unfeasible to prevent, especially for frame-to-frame tracking. Here, tracking is performed on triangulated feature points. However, depth uncertainty gives potentially inaccurate 3D points for sparse reconstruction. As a consequence, this will lead to erroneous subsequent camera poses. Particularly for a small baseline and distant points in the scene, the uncertainty increases. This is also the case and one of the reasons for camera pose drift in the results presented in Figure 5.3 and Figure 5.4. In Figure 4.18 camera poses for a scene with more distinguishable structure and more suitable camera poses is illustrated. This example shows better accuracy for camera pose tracking, but will not be considered here in detail since we aim for other application scenarios. Camera pose tracking can be improved by global optimization approaches, as already mentioned above, or placing robust and fixed markers in the scene [146]. Model-to-frame tracking, e.g. by projective ICP alignment, can also lead to more robust camera poses, but is not applicable for the reconstruction approach presented, as we cannot be certain of the reconstruction results of the model. The use of inertia sensors and incorporation of this information into a Kalman filter for filtering of bad poses may be considered.
Stereo and RGB-D cameras have the major advantage over monocular approaches to either have a constant baseline between cameras which is beneficial for reconstruction accuracy, or they give direct depth information, respectively. Misalignment of point clouds reconstructed by stereo cameras or given by RGB-D can easily be accounted for by rigid ICP methods. Furthermore, integration of point clouds into a common TSDF over time refines the reconstruction well enough.
As shown in section 5.1.1 frame to frame tracking leads to inaccuracies in camera poses, as it is common for SLAM approaches. These inaccuracies have direct influence on the computed disparity maps and thus also on reprojection results of the point clouds. Simple rigid alignment is not suitable, since some unknown scale ambiguity results in arbitrary scaled point clouds. For example, consider incorrectly estimated camera

poses for a given stereo image pair. Disparity calculation will be erroneous due to deficient rectification. The incorrect disparity map together with an incorrect baseline will give inaccurate reprojected point clouds. Stereo cameras can also be beneficial for camera pose tracking, since triangulation of features points can always be performed with known camera poses between the stereo setup, leading to higher accuracy of the sparse reconstruction, reduced frame-to-frame drift and less computational effort, e.g. computation of epipolar geometry.

Yu et al. [153] and Newcombe et al. [99] perform model-to-frame tracking for their non-rigid reconstruction approaches for RGB and RGB-D cameras, respectively. However, in the case of RGB-D cameras, one can be certain to obtain correct and detailed depth information suitable for camera pose tracking. Thus, similar to Dou et al. [34], also the embedded deformation graph can be estimated on a finer scale, beneficial for non-rigid deformation of the model. For the approach of [153] an extensive multiple view reconstruction of the static scene needs to be performed prior to non-rigid optimization, to obtain a high quality model beforehand. The embedded deformation graph of the approach presented here is coarser compared to mentioned literature, which is beneficial for computational effort. However, the graph may be too sparse, not accounting for the deformations well enough, leading to unsatisfying deformations via DQB, especially for parts with no or little structure, since deformation nodes are defined by ORB features. Deformation computation may be a hardly accountable problem with the algorithm presented here. For reconstruction purposes it is assumed that transformations between key frames are rigid, but at the same time we need to compare point clouds between key frames for non-rigid deformations. This problem could also be circumvented with stereo cameras.

It has been shown, that camera tracking can be performed in non-rigid environments on a sparse set of feature points, obtained by extensive outlier removal. Furthermore, fully dense and highly detailed disparity maps yield to accurate point cloud representations of the scene. The proposed procedure for the formation of a sparse embedded deformation graph, and its utilization to serve as basis for deformation of the remaining point cloud via DQB, has sufficiently been proven to adequately account for non-rigidity and lead to appealing reconstructions. An octree based TSDF has been used to integrate new information of the scene in a common representation, while reducing the memory consumption of regular grid based TSDFs.

Instead of current methods accounting for non-rigidity by en embedded deformation

graph, direct non-rigid alignment in a TSDF without the need of correspondences as recently proposed by Slavcheva et al. [123] may be of major advantage.
For further investigations, the use of a stereo camera system with fixed baseline, non-rigid pre-alignment by a sparse embedded deformation graph and subsequent DQB, and incorporation of direct non-rigid alignment inside an TSDF for refinement and details, may be beneficial. Tracking can still be performed on robust sparse ORB features. This approach would also lead to major savings in computational effort, since the epipolar geometry and the relative camera pose is already given for a calibrated stereo system.

List of Figures

Literature

[1] J. Achenbach, E. Zell, and M. Botsch. Accurate Face Reconstruction through Anisotropic Fitting and Eye Correction. *Vision, Modeling and Visualization*, 2015. DOI: 10.2312/vmv.20151251 (see p. 86)

[2] S. Agarwal, K. Mierle, and Others *Ceres Solver* URL: http://ceres-solver.org (see p. 26)

[3] A. Bajenov, D. Kapashi, and S. Chordia *Augmenting Videos with 3D Objects* 2016 URL: https://web.stanford.edu/class/cs231a/prev_projects_2016/augmenting-videos-3d__5_.pdf (see p. 75)

[4] I. Baran and J. Popović. Automatic rigging and animation of 3D characters. *ACM Transactions on Graphics*, **26**: 2007. DOI: 10.1145/1276377.1276467 (see p. 83)

[5] H. Bay, T. Tuytelaars, and L. Van Gool. "SURF: Speeded up robust features" in: *European Conference on Computer Vision*. 2006. DOI: 10.1007/11744023_32 (see p. 47)

[6] P. Besl and N. McKay. A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**: 239–256, 1992. DOI: 10.1109/34.121791 (see p. 77)

[7] S. Birchfield and C. Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**: 401–406, 1998. DOI: 10.1109/34.677269 (see p. 73)

[8] C. A. Blanchette, S. Leonardos, and J. Gallier "Motion Interpolation in SIM(3)" 2014 URL: https://pdfs.semanticscholar.org/a9cd/29dec93e55c21886d2a41155ed1762330440.pdf (see p. 14)

[9] J.-y. Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000. URL: http://robots.stanford.edu/cs223b04/algo_tracking.pdf (see pp. 78, 98)

[10] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**: 1222–1239, 2001. DOI: 10.1109/34.969114 (see p. 72)

[11] G. Bradski and A. Kaehler. *Learning OpenCV-Computer Vision with the OpenCV Library*. 2008. URL: http://shop.oreilly.com/product/9780596516130.do (see pp. 74, 78, 79)

[12] C. Bregler, A. Hertzmann, and H. Biermann. "Recovering non-rigid 3D shape from image streams" in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. vol. 2 Hilton Head Island, SC, USA: IEEE, 2000. 690–696 DOI: 10.1109/CVPR.2000.854941 (see p. 17)

[13] E. Bright and W. Hamilton. *Elements of quaternions*. New York, NY, USA, 1901. URL: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Elements+of+Quaternions#0 (see p. 81)

[14] Britannica *William-Rowan-Hamilton* URL: https://www.britannica.com/biography/William-Rowan-Hamilton (visited on 08/02/2017) (see p. 81)

[15] Brockhaus Online Encyclopedia *Camera obscura* 2016 URL: https://uni-lmu.brockhaus.de/sites/default/files/pdfpermlink/camera-obscura-38872e95.pdf (visited on 08/02/2017) (see pp. 28, 29)

[16] Brockhaus Online Encyclopedia *Haitham* 2016 URL: https://uni-lmu.brockhaus.de/sites/default/files/pdfpermlink/haitham-15281ef0.pdf (visited on 08/02/2017) (see p. 28)

[17] Brockhaus Online Encyclopedia *Leonardo da Vinci* 2016 URL: https://uni-lmu.brockhaus.de/sites/default/files/pdfpermlink/leonardo-da-vinci-446232e0.pdf (visited on 08/02/2017) (see p. 28)

[18] Brockhaus Online Encyclopedia *Redon, Odilon* 2016 URL: https://uni-lmu.brockhaus.de/sites/default/files/pdfpermlink/redon-odilon-a1b4b2bb.pdf (visited on 08/02/2017) (see p. 22)

[19] Brockhaus *Galilei, Galileo* 2016 URL: https://uni-lmu.brockhaus.de/sites/default/files/pdfpermlink/galilei-galileo-2d80e7d1.pdf (visited on 08/02/2017) (see p. 22)

[20] Brockhaus *Möbius, August Ferdinand* 2017 URL: https://uni-lmu.brockhaus.de/sites/default/files/pdfpermlink/moebius-august-ferdinand-423efff9.pdf (visited on 08/02/2017) (see p. 29)

[21] Brockhaus *Stereoskopisches Sehen* 2017 URL: https://uni-lmu.brockhaus.de/sites/default/files/pdfpermlink/stereoskopisches-sehen-4b05d193.pdf (see p. 33)

[22] B. Busam *Projective Geometry and 3D point cloud matching (Master's Thesis)* 2014 (see pp. 40, 59, 60)

[23] B. Busam, M. Esposito, B. Frisch, and N. Navab. "Quaternionic Upsampling: Hyperspherical Techniques for 6 DoF Pose Tracking" in: *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016. 629–638 DOI: 10.1109/3DV.2016.71 (see pp. 81–83)

[24] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. "BRIEF: Binary Robust Independent Elementary Features" in: *European Conference on Computer Vision*. vol. LNCS, 6314 2010. 778–792 DOI: 10.1007/978-3-642-15561-1_56 (see p. 47)

[25] N. D. F. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. "Using Multiple Hypotheses to Improve Depth-Maps for Multi-View Stereo" in: *Computer Vision ECCV 2008*. 2008. 766–779 DOI: 10.1007/978-3-540-88682-2_58 (see p. 19)

[26] P. Clifford *Preliminary Sketch of Biquaternions* 1873 URL: http://onlinelibrary.wiley.com/doi/10.1112/plms/s1-4.1.381/full (see p. 82)

[27] B. Curless and M. Levoy. "A volumetric method for building complex models from range images" in: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM Press, 1996. 303–312 DOI: 10.1145/237170.237269 (see pp. 12, 15, 16, 20, 84)

[28] R. Davies. *Computer and Machine Vision, Theory, Algorithms, Practicalities*. ed. by Academic Press El Sevier, 2012. URL: https://www.elsevier.com/books/computer-and-machine-vision/davies/978-0-12-386908-1 (see pp. 35, 36, 39, 40)

[29] A. J. Davison. "Real time simultaneous localisation and mapping with a single camera" in: *IEEE International Conference on Computer Vision (ICCV)*. vol. 2 IEEE, 2003. 1403–1410 DOI: 10.1109/ICCV.2003.1238654 (see pp. 1, 13)

[30] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**: 1052–1067, 2007. DOI: 10.1109/TPAMI.2007.1049 (see pp. 4, 6, 12, 13)

[31] M. C. De Sanctis, E. Ammannito, H. Y. McSween, A. Raponi, S. Marchi, F. Capaccioni, M. T. Capria, F. G. Carrozzo, M. Ciarniello, S. Fonte, M. Formisano, A. Frigeri, M. Giardino, A. Longobardo, G. Magni, L. A. McFadden, E. Palomba, C. M. Pieters, F. Tosi, F. Zambon, C. A. Raymond, and C. T. Russell. Localized aliphatic organic material on the surface of Ceres. *Science*, **355**: 719–722, 2017. DOI: 10.1126/science.aaj2305 (see p. 104)

[32] A. Del Bue, X. Lladó, and L. Agapito. "Non-rigid metric shape and motion recovery from uncalibrated images using priors" in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. New York, NY, USA: IEEE, 2006. DOI: 10.1109/CVPR.2006.209 (see p. 17)

[33] A. Delaunoy and M. Pollefeys. "Photometric bundle adjustment for dense multi-view 3D modeling" in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, 2014. DOI: 10.1109/CVPR.2014.193 (see p. 1)

[34] M. Dou, J. Taylor, P. Kohli, V. Tankovich, S. Izadi, S. Khamis, Y. Degtyarev, P. Davidson, S. R. Fanello, A. Kowdle, S. O. Escolano, C. Rhemann, and D. Kim. Fusion4D. *ACM Transactions on Graphics - Proceedings of ACM SIGGRAPH 2016*, **35**: 2016. DOI: 10.1145/2897824.2925969 (see pp. 16, 19, 20, 80, 82–86, 119)

[35] Editor in Chief *physics.org - Your guide to physics on the web* 2017 URL: http://www.physics.org/article-questions.asp?id=56 (visited on 06/22/2017) (see p. 24)

[36] J. Engel, T. Schöps, and D. Cremers. "LSD-SLAM: Large-Scale Direct monocular SLAM" in: *European Conference on Computer Vision*. vol. 8690 LNCS PART 2 2014. DOI: 10.1007/978-3-319-10605-2_54 (see pp. 1, 5, 6, 11, 14, 76, 117)

[37] J. Engel, J. Sturm, and D. Cremers. "Semi-dense visual odometry for a monocular camera" in: *Proceedings of the IEEE International Conference on Computer Vision*. Sydney, NSW, Australia: IEEE, 2013. DOI: 10.1109/ICCV.2013.183 (see p. 15)

[38] FOX Film Corporation *Exhebitors Herald* 1922 URL: http://archive.org/stream/exhibitorsherald15exhi_0#page/n263/mode/1up (visited on 08/02/2017) (see p. 34)

[39] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Transactions on Graphics*, **27**: 1, 2008. DOI: 10.1145/1360612.1360666 (see p. 73)

[40] G Farnebäck. "Two-frame motion estimation based on polynomial expansion" in: *13th Scandinavian Conference*. 1 Halmstad, Sweden: Springer Berlin Heidelberg, 2003. 363–370 DOI: 10.1007/3-540-45103-X_50 (see p. 78)

[41] M. a. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, **24**: 381–395, 1981. DOI: 10.1145/358669.358692 (see p. 17)

[42] C. Forster, M. Pizzoli, and D. Scaramuzza. "SVO: Fast semi-direct monocular visual odometry" in: *IEEE International Conference on Robotics and Automation*. Hong Kong, China: IEEE, 2014. DOI: 10.1109/ICRA.2014.6906584 arXiv: arXiv:1606.05830v2 (see p. 15)

[43] H. N. Fowler. *Plato: Cratylus, Parmenides, Greater Hippias, Lesser Hippias*. Loeb Class Harvard University Press, 1926. DOI: 10.4159/DLCL.plato_philosopher-cratylus.1926 (see p. 3)

[44] R. Garg, A. Roussos, and L. Agapito. "Dense Variational Reconstruction of Non-rigid Surfaces from Monocular Video" in: *IEEE Conference on Computer Vision and Pattern Recognition*. Portland, OR, USA: IEEE, 2013. DOI: 10.1109/CVPR.2013.168 (see pp. 12, 17, 19)

[45] M. Greenspan and M. Yurick. "Approximate K-D tree search for efficient ICP" in: *Fourth International Conference on 3-D Digital Imaging and Modeling, 2003*. Banff, Alta., Canada: IEEE, 2003. 442–448 DOI: 10.1109/IM.2003.1240280 (see p. 77)

[46] R. W. Hamming. Error Detecting and Error Correcting Codes. *Bell System Technical Journal*, **29**: 147–160, 1950. DOI: 10.1002/j.1538-7305.1950.tb00463.x (see p. 47)

[47] Happysheldon.com © 2017 All Right Reserved *Happy Sheldon* URL: https://www.happysheldon.com/news/32-historic-photos-of-movie-scenes-you-ve-never-seen-before/ (see p. 45)

[48] S. Har-Peled, P. Indyk, and R. Motwani. Approximate Nearest Neighbor: Towards Removing the Curse of Dimensionality. *Theory of Computing*, **8**: 321–350, 2012. DOI: 10.4086/toc.2012.v008a014 (see p. 48)

[49] C. Harris and J. Pike. 3D positional integration from image sequences. *Image and Vision Computing*, **6**: 87–90, 1988. DOI: 10.1016/0262-8856(88)90003-0 (see p. 5)

[50] C. Harris and M. Stephens. "A Combined Corner and Edge Detector" in: *Procedings of the Alvey Vision Conference 1988*. Alvey Vision Club, 1988. DOI: 10.5244/C.2.23 URL: http://www.bmva.org/bmvc/1988/avc-88-023.html (see pp. 46, 47, 79)

[51] R. Hartley "An investigation of the essential matrix" 1993 URL: http://axiom.anu.edu.au/$\sim$hartley/Papers/Q/Q.pdf (see p. 57)

[52] R. I. Hartley. Cheirality. *International Journal of Computer Vision*, **26**: 41–61, 1998. DOI: 10.1023/A:1007984508483 (see p. 93)

[53] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. New York, NY, USA: Cambridge University Press, 2003. URL: http://dl.acm.org/citation.cfm?id=861369 (see pp. 27–32, 35, 41, 50–52, 54, 56–58, 60–62, 66, 68–71, 94, 97)

[54] M Hassaballah, A. A. Abdelmgeid, and H. A. Alshazly. *Image Feature Detectors and Descriptors*. ed. by A. I. Awad and M. Hassaballah vol. 630 Studies in Computational Intelligence Springer International Publishing, 2016. 11–46 DOI: 10.1007/978-3-319-28854-3 (see p. 47)

[55] D. Hearn and M. P. Baker. *Computer Graphics - C version*. 2nd Editio 1996. 652 URL: http://www.hiteshpatel.co.in/ebook/cg/Computer_Graphics_C_Version.pdf (see p. 81)

[56] J. A. Hesch and S. I. Roumeliotis. "A Direct Least-Squares (DLS) method for PnP" in: *2011 International Conference on Computer Vision*. Barcelona, Spain, 2011. 383–390 DOI: 10.1109/ICCV.2011.6126266 (see p. 68)

[57] H Hirschmuller. Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **30**: 328–341, 2008. DOI: 10.1109/TPAMI.2007.1166 (see p. 73)

[58] H. Hirschmüller. Semi-Global Matching Motivation, Developments and Applications. *Proceedings of the 53rd Photogrammetric Week*, 173–184, 2011. URL: http://www.robotic.dlr.de/fileadmin/robotic/hirschmu/pw2011hh.pdf(see p. 72)

[59] D. Holz, A. E. Ichim, F. Tombari, R. B. Rusu, and S. Behnke. Registration with the Point Cloud Library: A Modular Framework for Aligning in 3-D. *IEEE Robotics and Automation Magazine*, **22**: 110–124, 2015. DOI: 10.1109/MRA.2015.2432331 (see pp. 4, 77, 78)

[60] Hongdong Li and R. Hartley. "Five-Point Motion Estimation Made Easy" in: *18th International Conference on Pattern Recognition (ICPR'06)*. IEEE, 2006. 630–633 DOI: 10.1109/ICPR.2006.579 (see p. 56)

[61] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, **4**: 629 –642, 1987. DOI: 10.1364/JOSAA.4.000629 (see pp. 80, 81, 86)

[62] T. S. Huang and O. D. Faugeras. Some Properties of the E Matrix in Two- View Motion Estimation. **11**: 1310–1312, 1989. DOI: 10.1109/34.41368 (see p. 56)

[63] X. Huang, L. Fan, J. Zhang, Q. Wu, and C. Yuan. "Real Time Complete Dense Depth Reconstruction for a Monocular Camera" in: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. Las Vegas, NV, USA: IEEE, 2016. 674–679 DOI: 10.1109/CVPRW.2016.89 (see pp. 1, 11, 12, 15)

[64] M. Innmann, M. Zollhöfer, M. Nießner, C. Theobalt, and M. Stamminger. "Volumedeform: Real-time volumetric non-rigid reconstruction" in: *European Conference on Computer Vision*. 2016. DOI: 10.1007/978-3-319-46484-8_22 (see pp. 1, 4, 11–13, 17–20, 80, 84, 85)

[65] Intuitive Surgical Inc *Da Vinci MIS System* URL: http://intuitivesurgical.com (visited on 11/16/2016) (see pp. 5, 19)

[66] S. Izadi, A. Davison, A. Fitzgibbon, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, and D. Freeman. "KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera" in: *Proceedings of the 24th annual ACM symposium on User interface software and technology*. New York, New York, USA: ACM Press, 2011. 559 DOI: 10.1145/2047196.2047270 (see pp. 15, 16, 20)

[67] John Hart Studios *BC Comic Strip - John Hart Studios* 2017 URL: http://johnhartstudios.com/meet-the-artists/remembering-johnny/ (visited on 01/18/2017) (see p. 26)

[68] T. KIM and K.-S. HONG. "ESTIMATING APPROXIMATE AVERAGE SHAPE AND MOTION OF DEFORMING OBJECTS WITH A MONOCULAR VIEW" in: *International Journal of Pattern Recognition and Artificial Intelligence*. vol. 19 04 2005. 585–601 DOI: 10.1142/S0218001405004150 (see p. 17)

[69] L. Kavan, S. Collins, J. Žára, and C. O'Sullivan. Geometric skinning with approximate dual quaternion blending. *ACM Transactions on Graphics*, **27**: 105:1–105:23, 2008. DOI: 10.1145/1409625.1409627 (see pp. 82, 83)

[70] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. "Real-Time 3D Reconstruction in Dynamic Scenes Using Point-Based Fusion" in: *Proceedings of the 2013 International Conference on 3D Vision*. IEEE, 2013. DOI: 10.1109/3DV.2013.9 (see p. 15)

[71] B. Kenwright. "A Beginners Guide to Dual-Quaternions: What They Are, How They Work, and How to Use Them for 3D Character Hierarchies" in: *The 20th International Conference on Computer Graphics, Visualization and Computer Vision*. 2012. 1–13 URL: http://wscg.zcu.cz/wscg2012/short/A29-full.pdf (see pp. 81, 82)

[72] G. Klein and D. Murray. "Parallel tracking and mapping for small AR workspaces" in: *IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR*. IEEE, 2007. 1–10 DOI: 10.1109/ISMAR.2007.4538852 (see pp. 4, 13, 14)

[73] V. Kolmogorov, P. Monasse, and P. Tan. Kolmogorov and Zabih's Graph Cuts Stereo Matching Algorithm. *Image Processing On Line*, **4**: 220–251, 2014. DOI: 10.5201/ipol.2014.97 (see p. 72)

[74] K. Konolige. "Small Vision Systems: Hardware and Implementation" in: *Robotics Research*. London: Springer London, 1998. 203–212 DOI: 10.1007/978-1-4471-1580-9_19 (see pp. 72, 73)

[75] C. Kurz, T. Thormahlen, and H.-P. Seidel. Visual Fixation for 3D Video Stabilization. *Journal of Virtual Reality and Broadcasting*, **8**: 2011. DOI: 10.20385/1860-2037/8.2011.2 (see p. 101)

[76] D. Lee *Lecture Slides - Brigham University Provo, Utah, Department for Electrical and Computer Engineering, Stereo Calibration and Rectification Calibration - Robotic Vision* Provo, Utah, 2015 URL: http://ece631web.groups.et.byu.net/Lectures/ECEn63114-CalibrationandRectification.pdf (see pp. 35, 60, 64, 65)

[77] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An Accurate O(n) Solution to the PnP Problem. *International Journal of Computer Vision*, **81**: 81–155, 2009. DOI: 10.1007/s11263-008-0152-6 (see p. 68)

[78] B. Lin, Y. Sun, and J. Sanchez. "Vesselness Based Feature Extraction for Endoscopic Image Analysis" in: *IEEE International Sympolsium on Biomedical Imaging*. Beijing, China: IEEE, 2014. DOI: 10.1109/ISBI.2014.6868114 (see p. 20)

[79] B. Lin, Y. Sun, X. Qian, D. Goldgof, R. Gitlin, and Y. You. Video-based 3D reconstruction, laparoscope localization and deformation recovery for abdominal minimally invasive surgery: a survey. *The International Journal of Medical Robotics and Computer Assisted Surgery*, **12**: 158–178, 2016. DOI: 10.1002/rcs.1661 (see pp. 5, 19, 20)

[80] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, **293**: 133–135, 1981. DOI: 10.1038/293133a0 (see p. 54)

[81] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, **60**: 91–110, 2004. DOI: 10.1023/B:VISI.0000029664.99615.94 (see p. 47)

[82] B. D. Lucas and T. Kanade. "An Iterative Image Registration Technique with an Application to Stereo Vision" in: *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI'81 San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1981. 674–679 URL: http://dl.acm.org/citation.cfm?id=1623264.1623280 (see p. 78)

[83] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision*. vol. 26 Interdisciplinary Applied Mathematics New York, NY: Springer, 2004. DOI: 10.1007/978-0-387-21779-6 (see p. 57)

[84] N. Mahmoud, I. Cirauqui, A. Hostettler, C. Doignon, L. Soler, J. Marescaux, and J. M. Montiel. "ORBSLAM-based endoscope tracking and 3d reconstruction" in: *International Workshop on Computer-Assisted and Robotic Endoscopy*. 2017. DOI: 10.1007/978-3-319-54057-3_7 (see p. 5)

[85] D. Marr. Vision: A computational investigation into the human representation and processing of visual information. *Journal of Mathematical Psychology*, **27**: 107–110, 1983. DOI: 10.1016/0022-2496(83)90030-5 (see p. 34)

[86] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. "Discrete Differential-Geometry Operators for Triangulated 2-Manifolds" in: *MATHVISUAL*. Springer Berlin Heidelberg, 2003. 35–57 DOI: 10.1007/978-3-662-05105-4_2 (see pp. 83, 86)

[87] S. Miller *GitHub CPU-TSDF* 2015 URL: https://github.com/sdmiller/cpu_tsdf/ (visited on 08/02/2017) (see p. 84)

[88] D. Min, S. Choi, J. Lu, B. Ham, K. Sohn, and M. N. Do. Fast Global Image Smoothing Based on Weighted Least Squares. *IEEE Transactions on Image Processing*, **23**: 5638–5653, 2014. DOI: 10.1109/TIP.2014.2366600 (see p. 73)

[89] A. F. Möbius. *Der barycentrische Calcül*. 1827. DOI: 10.3931/e-rara-14538 (see p. 29)

[90] F. Monnier, B. Vallet, N. Paparoditis, J.-P. Papelard, and N. David. Registration of terrestrial mobile laser data on 2D or 3D geographic database by use of a non-rigid ICP approach. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, **II-5/W2**: 193–198, 2013. DOI: 10.5194/isprsannals-II-5-W2-193-2013 (see p. 77)

[91] M. Muja and D. G. Lowe. "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration" in: *VISAPP 2009 - Proceedings of the Fourth International Conference on Computer Vision Theory and Applications*. Lisboa, Portugal, 2009. 331–340 URL: http://dblp.uni-trier.de/rec/bib/conf/visapp/2009-1 (see pp. 48, 77)

[92] M. Muja and D. G. Lowe. "Fast Matching of Binary Features" in: *2012 Ninth Conference on Computer and Robot Vision*. Toronto, ON, Canada: IEEE, 2012. 404–410 DOI: 10.1109/CRV.2012.60 (see p. 48)

[93] R. Mur-Artal, J. M. Montiel, and J. D. Tardos. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, **31**: 1147–1163, 2015. DOI: 10.1109/TRO.2015.2463671 (see pp. 4, 11, 14, 20, 47, 94, 117)

[94] R. Mur-Artal and J. D. Tardos. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 2017. DOI: 10.1109/TRO.2017.2705103 (see p. 14)

[95] A. Naeve *The Knowledge Management Research Group - KTH ROYAL INSTITUTE OF TECHNOLOGY* 2017 URL: http://kmr.nada.kth.se/VML/Matematik-inscannat/Comic-strips/BC/ (visited on 07/10/2017) (see p. 26)

[96] N. Navab and C. Unger *Lecture Slides - Technical University of Munich, Chair for Computer Aided Medical Procedures and Augmented Reality - 3D Computer Vision II Winter Term 2010/2011* 2010 URL: http://campar.in.tum.de/twiki/pub/Chair/TeachingWs10Cv2/3D_CV2_WS_2010_Rectification_Disparity.pdf (see pp. 39, 40)

[97] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa. "Laplacian mesh optimization" in: *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia - GRAPHITE '06*. New York, New York, USA: ACM Press, 2006. 381–389 DOI: 10.1145/1174429.1174494 (see p. 83)

[98] R. A. Newcombe and A. J. Davison. "Live dense reconstruction with a single moving camera" in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA, USA: IEEE, 2010. DOI: 10.1109/CVPR.2010.5539794 (see pp. 1, 4, 6, 12, 14–16, 20, 101)

[99] R. A. Newcombe, D. Fox, and S. M. Seitz. "DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time" in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Boston, MA, USA: IEEE, 2015. DOI: 10.1109/CVPR.2015.7298631 (see pp. 1, 4, 7, 11–13, 17–20, 80, 82–86, 119)

[100] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. "DTAM: Dense tracking and mapping in real-time" in: *IEEE International Conference on Computer Vision*. Barcelona, Spain: IEEE, 2011. DOI: 10.1109/ICCV.2011.6126513 (see pp. 1, 11, 14, 15, 67)

[101] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. "KinectFusion: Real-time dense surface mapping and tracking" in: *IEEE International Symposium on Mixed and Augmented Reality*. Basel, Switzerland: IEEE, 2011. DOI: 10.1109/ISMAR.2011.6092378 (see pp. 4, 11, 12, 14, 16, 17, 20, 84)

[102] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. "Real-time 3D Reconstruction at Scale Using Voxel Hashing" in: *ACM Transactions on Graphics*. vol. 32 6 2013. DOI: 10.1145/2508363.2508374 (see p. 18)

[103] D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**: 756–770, 2004. DOI: 10.1109/TPAMI.2004.17 (see pp. 55, 56)

[104] P. Ondruska, P. Kohli, and S. Izadi. MobileFusion: Real-Time Volumetric Surface Reconstruction and Dense Tracking on Mobile Phones. *IEEE Transactions on Visualization and Computer Graphics*, **21**: 1251–1258, 2015. DOI: 10.1109/TVCG.2015.2459902 (see p. 16)

[105] OpenCV Dev. Team *OpenCV* 2017 URL: http://opencv.org/ (visited on 01/02/2017) (see pp. 27, 46, 47, 55, 68, 73, 79, 97)

[106] A. Penate-Sanchez, J. Andrade-Cetto, and F. Moreno-Noguer. Exhaustive Linearization for Robust Camera Pose and Focal Length Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **35**: 2387–2400, 2013. DOI: 10.1109/TPAMI.2013.36 (see p. 68)

[107] Z. Peng. *Efficient matching of robust features for embedded SLAM*. PhD thesis. University of Stuttgart, 2012. (see p. 49)

[108] S. Perera, N. Barnes, X. He, S. Izadi, P. Kohli, and B. Glocker. "Motion Segmentation of Truncated Signed Distance Function Based Volumetric Surfaces" in: *IEEE Winter Conference on Applications of Computer Vision*. Waikoloa, HI, USA: IEEE, 2015. DOI: 10.1109/WACV.2015.144 (see pp. 16, 84)

[109] H. Pfister, M. Zwicker, J. VAN Baar, and M. Gross. "Surfels" in: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press, 2000. 335–342 DOI: 10.1145/344779.344936 (see p. 16)

[110] M. Pizzoli, C. Forster, and D. Scaramuzza. "REMODE: Probabilistic, monocular dense reconstruction in real time" in: *IEEE International Conference on Robotics and Automation*. Hong Kong, China: IEEE, 2014. DOI: 10.1109/ICRA.2014.6907233 (see pp. 13, 15)

[111] V. Pradeep, C. Rhemann, S. Izadi, C. Zach, M. Bleyer, and S. Bathiche. "MonoFusion: Real-time 3D reconstruction of small scenes with a single web camera" in: *IEEE International Symposium on Mixed and Augmented Reality*. Adelaide, SA, Australia: IEEE, 2013. DOI: 10.1109/ISMAR.2013.6671767 (see pp. 13–15)

[112] Prints and Photographs Division Washington *Library of Congress* 1901 URL: http://www.loc.gov/pictures/resource/ppmsca.08781/ (visited on 08/01/2017) (see p. 23)

[113] J. Richter-Gebert and T. Orendt. *Geometriekalküle*. Berlin, Heidelberg: Springer, 2009. DOI: 10.1007/978-3-642-02530-3 (see p. 81)

[114] E. Rosten and T. Drummond. "Machine learning for high-speed corner detection" in: *European Conference on Computer Vision*. 2006. DOI: 10.1007/11744023_34 (see pp. 47, 79)

[115] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. "ORB: An efficient alternative to SIFT or SURF" in: *Proceedings of the IEEE International Conference on Computer Vision*. Barcelona, Spain: IEEE, 2011. 2564–2571 DOI: 10.1109/ICCV.2011.6126544 (see pp. 14, 47)

[116] S. Rusinkiewicz and M. Levoy. "Efficient variants of the ICP algorithm" in: *Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3VVDIM*. 2001. DOI: 10.1109/IM.2001.924423 (see pp. 4, 77)

[117] R. B. Rusu and S. Cousins. "3D is here: Point Cloud Library (PCL)" in: *International Conference on Robotics and Automation ICRA*. Shanghai, China: IEEE, 2011. 1–4 DOI: 10.1109/ICRA.2011.5980567 URL: http://pointclouds.org/ (see p. 77)

[118] Sächsische Landesbibliothek - Staats- und Universitätsbibliothek Dresden (SLUB) *Deutsche Fotothek* 2007 URL: http://www.deutschefotothek.de/gallery/encoded/eJzjYBJS5OJIK0rNLEmtKBHiTE_Nz00tKcpMlWJ29HNRYi7JydZiEFJCUsJVllqUm1pcXJqXjqwGAO6hFMc* (visited on 01/19/2017) (see p. 39)

[119] D. Scharstein and R. Szeliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, **47**: 7–42, 2002. DOI: 10.1023/A:1014573219977 (see p. 72)

[120] S. S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. "A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms" in: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. New York, NY, USA, 2006. 519–528 DOI: 10.1109/CVPR.2006.19 (see pp. 1, 72, 101)

[121] K. Shoemake. Animating rotation with quaternion curves. *ACM SIGGRAPH Computer Graphics*, **19**: 245–254, 1985. DOI: 10.1145/325165.325242 (see p. 82)

[122] R. Siegel, D. Naishadham, and A. Jemal. Cancer statistics, 2013. *CA: A Cancer Journal for Clinicians*, **63**: 11–30, 2013. DOI: 10.3322/caac.21166 (see p. 5)

[123] M Slavcheva, M Baust, D Cremers, and S Ilic. "KillingFusion: Non-rigid 3D Reconstruction without Correspondences" in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017. URL: http://campar.in.tum.de/pub/slavcheva2017cvpr/slavcheva2017cvpr.pdf (see pp. 19, 120)

[124] M. Slavcheva, W. Kehl, N. Navab, and S. Ilic. "SDF-2-SDF: Highly Accurate 3D Object Reconstruction" in: *ECCV*. 2016. 680–696 DOI: 10.1007/978-3-319-46448-0_41 (see p. 19)

[125] O. Sorkine and M. Alexa. "As-Rigid-As-Possible Surface Modeling" in: *Proceedings of the fifth Eurographics symposium on Geometry processing*. ed. by A. Belyaev and M. Garland The Eurographics Association, 2007. 109–116 DOI: 10.1007/s11390-011-1154-3 (see pp. 4, 18, 20, 83, 85, 86)

[126] D. Stoyanov, M. V. Scarzanella, P. Pratt, and G. Z. Yang. "Real-time stereo reconstruction in robotically assisted minimally invasive surgery" in: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. 2010. DOI: 10.1007/978-3-642-15705-9_34 (see p. 5)

[127] H. Strasdat, J. M. M. Montiel, and A. J. Davison. Scale Drift-Aware Large Scale Monocular SLAM. *Robotics: Science and Systems*, **2**: 2010. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.167.287(see p. 76)

[128] B. Stroustrup *Bjarne Stroustrup Homepage* URL: http://www.stroustrup.com (visited on 07/08/2017) (see p. 45)

[129] J. Stückler and S. Behnke. Multi-resolution surfel maps for efficient dense 3D modeling and tracking. *Journal of Visual Communication and Image Representation*, **25**: 137–147, 2014. DOI: 10.1016/j.jvcir.2013.02.008 (see p. 16)

[130] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. "A benchmark for the evaluation of RGB-D SLAM systems" in: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vilamoura, Portugal: IEEE, 2012. 573–580 DOI: 10.1109/IROS.2012.6385773 (see pp. 53, 63, 74)

[131] R. W. Sumner, J. Schmid, and M. Pauly. Embedded deformation for shape manipulation. *ACM Transactions on Graphics*, **26**: 80, 2007. DOI: 10.1145/1276377.1276478 (see pp. 4, 19, 20, 80, 85)

[132] R. Szeliski. *Computer Vision*. vol. 5 Texts in Computer Science London: Springer London, 2011. 832 DOI: 10.1007/978-1-84882-935-0 (see pp. 33, 50)

[133] R. Szeliski and D. Scharstein. Sampling the Disparity Space Image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **26**: 419–425, 2004. DOI: 10.1109/TPAMI.2004.1262341 (see p. 63)

[134] E. F. Tedesco *Ceres* 2015 URL: https://www.britannica.com/place/Ceres-dwarf-planet (visited on 02/17/2017) (see p. 104)

[135] L Torresani, D. B. Yang, E. J. Alexander, and C Bregler. "Tracking and modeling non-rigid objects with rank constraints" in: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2001. DOI: 10.1109/CVPR.2001.990515 (see pp. 12, 17)

[136] United States Patent and Traddemark Office *Biographical Sketches of the Commissioners of Patents: Charles Holland Duell* 2017 URL: https://www.uspto.gov/about-us/charles-holland-duell (visited on 07/08/2017) (see p. 10)

[137] R. Vaillant *Rodolphe Vaillant's Homepage - Research, teaching and more...* URL: http://rodolphe-vaillant.fr (visited on 07/07/2017) (see p. 83)

[138] Visicom Media Inc. *manycam.com* 2017 URL: https://download.manycam.com/effects/get?f=l&i=20806&v=3.0 (visited on 07/01/2017) (see p. 25)

[139] G. Vogiatzis, C. Hernández, P. H. Torr, and R. Cipolla. Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**: 2241–2246, 2007. DOI: 10.1109/TPAMI.2007.70712 (see p. 19)

[140] S. Wang, S. R. Fanello, C. Rhemann, S. Izadi, and P. Kohli. "The Global Patch Collider" in: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016. 127–135 DOI: 10.1109/CVPR.2016.21 (see p. 80)

[141] X. Wei and J. Mulligan. "Robust relative pose estimation with integrated cheirality constraint" in: *19th International Conference on Pattern Recognition*. IEEE, 2008. DOI: 10.1109/ICPR.2008.4761509 (see p. 57)

[142] E. W. Weisstein *Dual Number* URL: http://mathworld.wolfram.com/DualNumber.html (visited on 07/07/2017) (see p. 82)

[143] E. W. Weisstein *Euler Parameters* 2017 URL: http://mathworld.wolfram.com/EulerParameters.html (visited on 02/11/2017) (see p. 81)

[144] E. W. Weisstein *Frobenius Norm* 2017 URL: http://mathworld.wolfram.com/FrobeniusNorm.html (visited on 07/02/2017) (see p. 57)

[145] E. W. Weisstein *Rotation Matrix* 2017 URL: http://mathworld.wolfram.com/RotationMatrix.html (visited on 02/11/2017) (see pp. 34, 36–38)

[146] A. Wendel, M. Maurer, G. Graber, T. Pock, and H. Bischof. "Dense reconstruction on-the-fly" in: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012. 1450–1457 DOI: 10.1109/CVPR.2012.6247833 (see pp. 84, 118)

[147] T. Whelan, S. Leutenegger, R. F. Salas-moreno, B. Glocker, and A. J. Davison. "ElasticFusion : Dense SLAM Without A Pose Graph" in: *Robotics: Science and Systems*. vol. 2015 Robotics: Science and Systems Foundation, 2015. DOI: 10.15607/RSS.2015.XI.001 (see p. 15)

[148] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger. ElasticFusion: Real-time dense SLAM and light source estimation. *The International Journal of Robotics Research*, **35**: 2016. DOI: 10.1177/0278364916669237 (see p. 16)

[149] WikiArt *The Cyclops by Odilon Redon* 2016 URL: https://www.wikiart.org/en/odilon-redon/the-cyclops (visited on 12/29/2016) (see p. 22)

[150] C. Wu. "Towards linear-time incremental structure from motion" in: *International Conference on 3D Vision*. Seattle, WA, USA: IEEE, 2013. 127–134 DOI: 10.1109/3DV.2013.25 (see pp. 1, 18)

[151] G. Xiao-Shan, H. Xiao-Rong, T. Jianliang, and C. Hang-Fei. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**: 930–943, 2003. DOI: 10.1109/TPAMI.2003.1217599 (see p. 68)

[152] S. Yin, M. Badr, and D. Vodislav. Dynamic Multi-probe LSH: An I/O Efficient Index Structure for Approximate Nearest Neighbor Search. *Proceedings of the 33rd International Conference on Very Large Data Bases*, 48–62, 2013. DOI: 10.1007/978-3-642-40285-2_7 (see p. 48)

[153] R. Yu, C. Russell, N. D. F. Campbell, and L. Agapito. "Direct, Dense, and Deformable: Template-Based Non-rigid 3D Reconstruction from RGB Video" in: *IEEE International Conference on Computer Vision (ICCV)*. Santiago, Chile: IEEE, 2015. DOI: 10.1109/ICCV.2015.111 (see pp. 1, 4, 12, 13, 17–20, 85, 86, 119)

[154] C. Zach, T. Pock, and H. Bischof. "A Globally Optimal Algorithm for Robust TV-L1 Range Image Integration" in: *IEEE International Conference on Computer Vision*. vol. 1 Rio de Janeiro, Brazil: IEEE, 2007. 1–8 DOI: 10.1109/ICCV.2007.4408983 (see p. 15)

[155] R. D. Zakia. *Perception and Imaging: Photography-A Way of Seeing*. Fourth Focal Press, 2014. URL: https://books.google.de/books?id=tlYmF9nwVlYC&printsec=frontcover&hl=de&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false (see p. 33)

[156] Z. Zhang. Determining the Epipolar Geometry and its Uncertainty: A Review. *International Journal of Computer Vision*, **27**: 161–195, 1998. DOI: 10.1023/a:1007941100561 (see p. 51)

[157] H. Zhou and G. Schaefer. "Robust estimation of the fundamental matrix" in: *2010 IEEE International Conference on Image Processing*. IEEE, 2010. 4233–4236 DOI: 10.1109/ICIP.2010.5651940 (see p. 94)

[158] H. Zhu, Y. Yu, Y. Zhou, and S. Du. Dynamic human body modeling using a single RGB camera. *Sensors*, **16**: 2016. DOI: 10.3390/s16030402 (see p. 1)

[159] M. Zollhoefer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, A. Fitzgibbon, C. Loop, C. Theobalt, and M. Stamminger. Real-time Non-rigid Reconstruction using an RGB-D Camera. *ACM Trans. Graph.*, **33**: 2014. DOI: 10.1145/2601097.2601165 (see p. 18)

[160] R. Zone. *Stereoscopic Cinema and the Origins of 3-D Film*. The University Press of Kentucky, 2007. (see p. 34)

[161] *evoganeo* URL: https://www.evogeneao.com (visited on 08/02/2017) (see p. 10)