Dense Real-time 3D Reconstruction from Multiple Images

A Dissertation submitted in fulfillment of the requirements for the Degree of Doctor of Philosophy

Li Ling
M.Eng.

School of Electrical and Computer Engineering
College of Science, Engineering and Health
RMIT University

Auguest 2013

Abstract

The rapid increase in computer graphics and acquisition technologies has led to the widespread use of 3D models. Techniques for 3D reconstruction from multiple views aim to recover the structure of a scene and the position and orientation (motion) of the camera using only the geometrical constraints in 2D images. This problem, known as Structure from Motion (SfM) has been the focus of a great deal of research effort in recent years; however, the automatic, dense, real-time and accurate reconstruction of a scene is still a major research challenge. This thesis presents work that targets the development of efficient algorithms to produce high quality and accurate reconstructions, introducing new computer vision techniques for camera motion calibration, dense SfM reconstruction and dense real-time 3D reconstruction.

Projective geometry and homogeneous coordinates are widely used to represent transformations and projections in computer vision. One key challenge is that projective geometry lacks the concept of orientation in the representation of lines, planes and space, and thus fails to distinguish the pixels in an image corresponding to points which lie in ‘front’ or ‘back’ of the camera. In computer vision, the camera projective reconstruction suffers from the projective ambiguity that the algebraic signs of the 3D points randomly swap from positive to negative during the reconstruction from multiple images under Euclidean coordinate frames, that is termed as cheirality of the point with respect to the camera. This thesis revisits the cheirality problem within projective geometry from a camera motion viewpoint, proposing and showing the root cause of cheirality to be a handedness problem in camera motion estimation. Eight possible solutions of camera motion from essential matrix are proposed in this thesis through mathematical derivation and geometrical analysis. Beyond the existing ‘twist pair’ of the rotation matrices R_1 and R_2, there is also the reversed sign solution pair of -R_1 and -R_2, which leads to the handedness of camera projection.

Geometrically, this thesis extends the representation of projective geometry into four dimensions, and proposes a space-time 4D visualization of cheirality from a motion viewpoint. When studying the movements of objects, the reference system must be set up first, and kept consistent in whole movements; thus, the cheirality problem can be resolved by confining all rotations to the right-hand rule (equivalent to applying the $det(R) = 1$ constraint). For a projective reconstruction, this cheirality problem exists in the camera motion estimation from the essential matrix, the linear estimation of projection matrix P, projective transformation H and epipolar constraint.

In SfM, a second challenge is to build an effective reconstruction framework that provides dense and high quality surface modelling. This thesis develops a complete, automatic and flexible system with a simple user-interface of 'raw images to 3D surface representation'. As part of the proposed image reconstruction approach, this thesis introduces an accurate and reliable region-growing algorithm to propagate the dense matching points from the sparse key points among all stereo pairs. This dense 3D reconstruction proposal addresses the deficiencies of existing SfM systems built on sparsely distributed 3D point clouds which are insufficient for reconstructing a complete 3D model of a scene.

The existing SfM reconstruction methods perform a bundle adjustment optimization of the global geometry in order to obtain an accurate model. Such an optimization is very computational expensive and cannot be implemented in a real-time application. Extended Kalman Filter (EKF) Simultaneous Localization and Mapping (SLAM) considers the problem of concurrently estimating in real-time the structure of the surrounding world, perceived by moving sensors (cameras), simultaneously localizing in it. However, standard EKF-SLAM techniques are susceptible to errors introduced during the state prediction and measurement prediction linearization. Taking the advantage of the known 3D depth data from RGB-D camera, this thesis improves upon existing EKF-SLAM techniques with the proposed approach of Geometric Modelling Iterated Extended Kalman Filter (GMIEKF) SLAM for a real-time 3D mapping. The measurement residual errors are minimized through a nonlinear least square optimization against the a priori state parameters, and the iterated linearization of the nonlinear measurement model is used to prevent linearized error.

Acknowledgements

This thesis would not have been possible without: Prof. Ian S. Burnett, Dr. Eva Cheng whose supervision and guidance were invaluable throughout my candidature, my loving family for their ever-present love, support and encouragement and my friends for taking care of the non-academic aspects of my life.

To my parents.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over the past two decades, the automatic recovery of a three dimensional structure from image or video sequences has become one of the central problems in computer vision. This problem, known as Structure from Motion (SfM), has great practical importance in image-based rendering, 3D mapping, scene visualization from online photo collections, augmented reality, human motion tracking, robot navigation, object recognition, 3D surveillance and 3D TV. There are two main groups of existing SfM reconstruction approaches: active and passive methods [3].

Active methods physically interact with the reconstructed object radiometrically or mechanically. In practice, a controlled source of light range, such as laser, moving light sources, coloured visible light, microwaves, ultrasound or a coded light, is used to recover the 3D information. Typically, an active reconstruction system utilizes a laser projector for scanning, retrieving high-precision dense correspondences with ease; the accuracy and density of 3D points are relatively high. For example, in the medical industry, 3D Positron Emission Tomography (PET) image reconstruction is used to provide information about how an organ or system in the body is functioning. The 3D PET scanners can create 3D models of a range of organs, helping to diagnose and assess the conditions of these organs. In contrast, manufacturing industries utilize laser range scanners to record 3D surfaces such as car bodies and mobile phone shells.

Passive methods of 3D reconstruction do not interact with the reconstructed object, and need only the information contained in recorded images of the scene. The information about the 3D structure of the scene can be obtained by integrating a number of visual cues that naturally exist in standard image observations, e.g., perspective transformation, epipolar constraint, texture, geometrical correspondences, motion parallax, stereo, focus and occluding contours. This lower equipment cost constitutes one competitive advantage of passive techniques compared to active techniques, which require specialized hardware such as 3D scanners. Such passive techniques that can efficiently and robustly create full 3D structures from only collections of images is of great interest to computer vision researchers; for example, researchers have reconstructed individual buildings or plazas from photo collections [4], [5], [6], [7], showing the potential of applying SfM algorithms on unstructured photo collections of up to a few thousand photographs. The 'Building Rome in One Day' project [8] reconstructs 3D scenes from extremely large collection of photographs such as those found by searching for a given city on internet photo sharing sites (e.g., flickr.com). Meanwhile, the Microsoft Photosynth software tool [5] analyzes digital photographs and generates a three-dimensional model of the photos and a point cloud of a photographed object. With the Google Earth '3D Buildings' project [9], a user can automatically construct realistic 3D models of existing street scenes such as buildings, monuments, fountains, bridges, towers, museums, homes etc.

The techniques for 3D reconstruction from multiple views proposed in this thesis belong to the passive SfM framework, which aims to recover the 3D structure of a scene and the position and orientation (pose) of the camera using only the geometri-

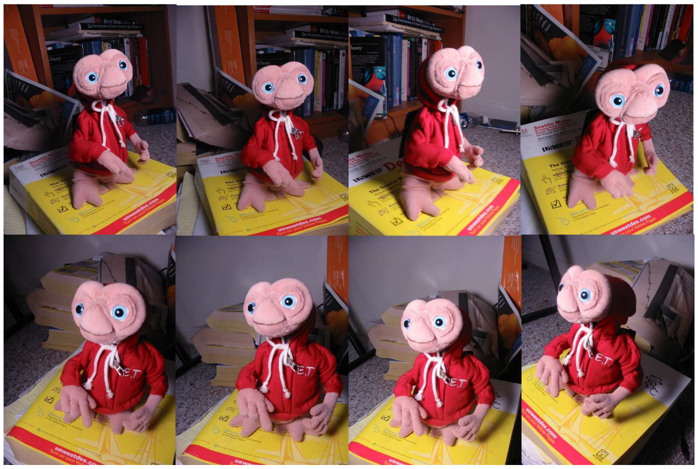


Figure 1.1: Eight different views of an object

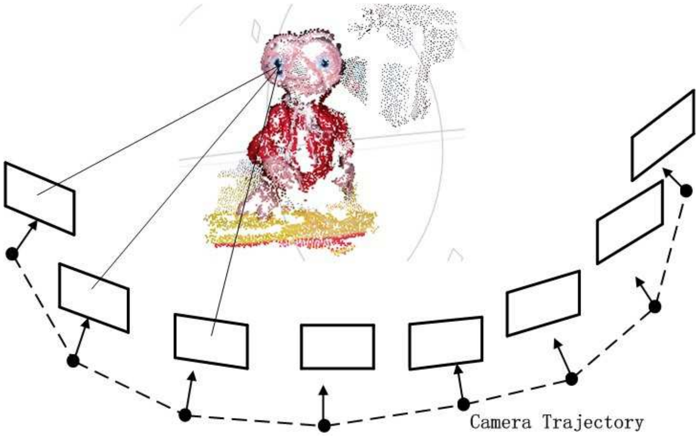


Figure 1.2: 3D reconstruction from multiple views

cal constraints in images. Given a set of images or video sequence of a scene taken from an uncalibrated camera with unknown movement in an unknown environment, this thesis creates the 3D models automatically. For example, in Figs. 1.1 and 1.2, given eight images of a toy object shot from different view angles, the 3D locations of points in the scene from multiple views can be reconstructed through geometrical constraints in the images.

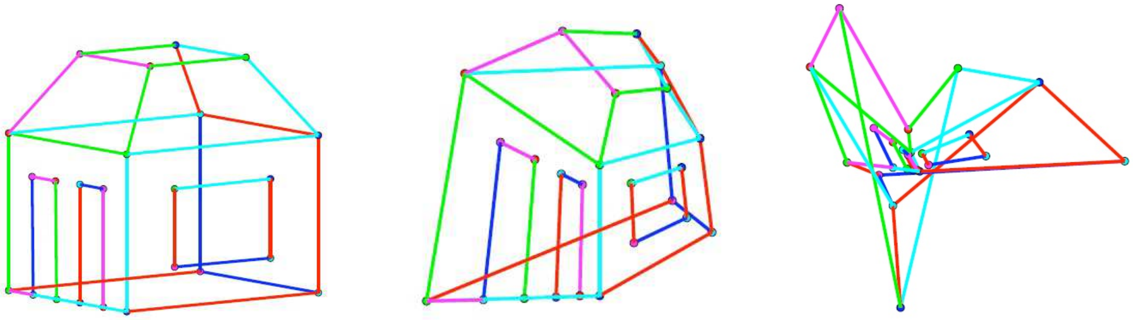
In this thesis, the proposed SfM approaches derive the passive structure from motion cues estimated between images. Two assumptions for SfM are required [10]: First, objects in the scene are moving rigidly or, equivalently, and only the camera is allowed to move in the environment. It is possible, by making certain assumptions about the intrinsic parameters of the cameras, to calibrate the cameras from the feature correspondences between images. Second, given the position of a scene feature in one image, it is possible to find the corresponding position of the same feature in successive images. Such features could include salient points in the image, corners of objects, lines along their edges or curves around their contours. However, the set of feature correspondences is likely to contain a significant number of mismatches. Thus, the use of robust estimation techniques such as RANSAC [11] is essential in matching feature correspondences to remove mismatches and achieve an accurate structure from the SfM algorithm.

1.1 Problem Statement

This thesis focuses on the process of reconstructing a 3D scene and the recovery of camera position and orientation information given a set of images. In particular, the following three problems are addressed: cheirality problem in camera motion calibration, dense 3D reconstruction and real-time dense 3D reconstruction.

1.1.1 Cheirality Problem in Camera Motion Calibration

Structure from motion came to the fore in computer vision following the technique proposed by Longuet-Higgins to reconstruct a scene from two views using eight point correspondences [12]. Later, many camera motion estimation systems [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], utilizing vision techniques have been developed to achieve accurate calibration of camera parameters to recover camera motion. Higgins [12] refers to relative orientation as the determination of the relative baseline of two perspective centres in two views (translation vector t)



(a)Euclidean Structure (b)A Quasi-affine Structure (c)A Projective Structure

Figure 1.3: Despite appearances to the contrary, (c) is as valid a projective reconstruction of the house as (b). While the quasi-affine reconstruction preserves the convex hull of the scene in (a), a general projective transformation might not. Note that incidence relations are still preserved in (c). This diagram is the courtesy of [1].

and rotation angles (rotation matrix R) of one image relative to the other. First, the geometric constraints of feature correspondences from two views are expressed by the fundamental and essential matrices, where the fundamental matrix relates corresponding points in the two images and the essential matrix contains the translation and rotation matrices up to an unknown scale factor. The translation and rotation matrices are derived from the essential matrix with linear decomposition techniques such as Singular Value Decomposition (SVD) [28]; such approaches have been intensively studied over the last two decades by [12], [16], [29], [30], [31], [32], [33]. Higgins [12] developed the first linear decomposition algorithm, which Tsai [16] applied to estimate the 3D motion of a rigid object from two perspective views and a more robust version of the algorithm was later developed by [29], [30] and [31], [32], [33].

Hartley [29] proposed four possible solutions for the camera motion from two perspective views. This widely accepted four solutions for the essential matrix are formed from combinations of the two solutions for the rotation matrix (R_1 and R_2), called the 'twist pair', and the two reversed sign solutions for the translation matrix ($T_\times$ and -$T_\times$). Thus, the camera projective matrix P has four possible solutions: $P \sim \left[R_1|t\right]$, $\left[R_1|-t\right]$, $\left[R_2|t\right]$ and $\left[R_2|-t\right]$, where translation matrix $T_\times$ is the cross product of the translation vector t ($T_{\times} = \begin{bmatrix} t \end{bmatrix}_{\times}$). The four solutions can correctly represent the static position and orientation information of the camera for two views; however, when interpreting the four solutions of camera motion estimation for multiple views, a projective reconstruction might not preserve the convex hull of the scene points when interpreted in Euclidean coordinate frames [34], [35], [30]. As shown in Fig. 1.3. Fig. 1.3 (a) is the Euclidean structure of a house, and (c) is the projective reconstruction of (a), which cannot preserve the convex hull of the object of interest. Hartley proposed that camera projective transforms possess the property of swapping points from the front to the back of the camera, where the problem of determining whether a 3D point lies in front of or behind a given camera is termed as the cheirality of the point with respect to the camera [34], [35], [30]. This problem is originated from non-orientation of projective geometry, that is, projective geometry lacks the concept of orientation, and fails to distinguish the pixels in an image corresponding to points which lie in front or back of the camera. To represent the orientation of the camera, Stolfi [36] developed the theory of oriented projective geometry by constructing a canonical two-side space with the front and back ranges divided by an infinite point. Laveau and Faugeras [37] extended the oriented projective geometry into computer vision, where the camera can only view the points on one side of the principal plane; those points are in front of the camera, points on the other side will not be visible. The following works [38], [39], [40] and [41], [42] and [43] are the applications of the above concepts in camera calibration, evalutation of epipoles and 3D reconstruction.

Existing attempts to resolve cheirality orient quasi-affine reconstructions whose projectivities preserve the convex hull of an object of interest, as shown in Fig. 1.3b. The quasi-affine reconstructions try to identify the point or plane at infinity to distinguish the front range of the camera; however, three main problems are encountered.

Firstly, locating the plane at infinity in a projective reconstruction is considered a difficult obstacle [38] in mathematics, especially as determining the orientation of the infinite plane is non-trivial; consequently it leads to difficulties in subsequent applications. Secondly, the geometrical model of oriented projective geometry is difficult to represent since the two sides of line and plane connected to the invisible infinite point are indistinguishable in 3D geometric space. For example, the concept of lying in 'front' or 'back' of the camera in oriented projective geometry is easily confused with the positive or negative depth in 3D geometric representation. Thirdly, the iterative searching of cheirality invariant constraints impacts the efficiency of the applications, as the computational complexity grows with the number of views.

This thesis revisits the cheirality problem with projective geometry [36] and presents a novel 4D geometric analysis of the antipodes of 3D model into a space-time framework to address the root cause of the cheirality problem. The goal of this work is to establish and pose the relationship between linear algebra and geometry to resolve the long-standing ambiguity inherent to the cheirality problem in camera projective reconstruction.

1.1.2 Dense Reconstruction from Multiple Images

Existing SfM approaches consist of three steps: detection of the feature points; calibration of the camera orientations image-by-image for the whole image sequence; and reconstruction of the surface with photo and visualization-consistent constraints for each image pair. The steps for camera calibration and surface reconstruction of multiple images are the most computationally complex. In most cases, the SfM system is built on sparsely distributed feature points, but the sparse approach is insufficient for the complete 3D model of a scene. For example, Fig. 1.4 shows the SfM reconstruc-

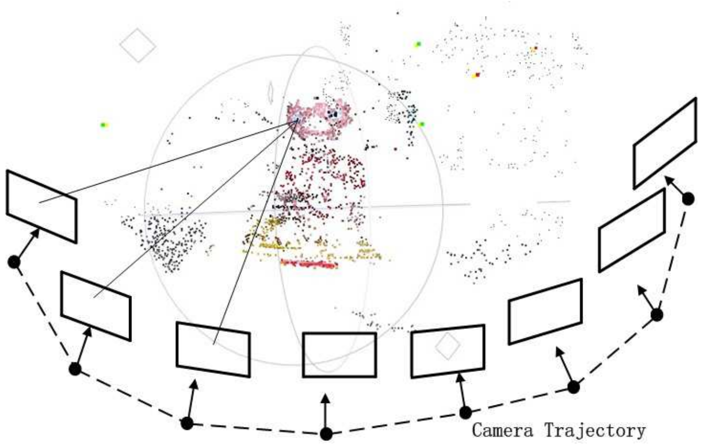


Figure 1.4: The drawback of sparse reconstruction is incomplete representation of the scene due to inadequate information.

tion result from the well-known Bundler [44] technique, however, the size, distance, shape and material of the measured object are incomplete. Consequently, there is a need to automatically create dense 3D point clouds from multiple views. Typically, the sparse approach can be used for calibration purposes to initialize a dense method, dense stereo methods [45] [46] are typical approaches to reconstruction. However, the main disadvantages of these dense stereo methods are computational expensive in terms of time and memory.

This thesis proposes a dense reconstruction method by taking advantage of an initial sparse approach for calibration purposes. With incremental bundle adjustment and dense propagation computed with a region growing algorithm combining the techniques of [47] and [48], to automatically extract high quality dense 3D geometry and surface properties of the object.

1.1.3 Real Time 3D Reconstruction from Video Sequences

Despite the accurate and flexible advantages of SfM, computational complexities in multiple-view reconstruction can be problematic. Typically, most SfM systems use bundle adjustment to refine the initial camera orientation and 3D point positions. Bundle adjustment performs a recursive global optimization over the whole image sequence; the core of bundle adjustment is the Levenberg-Marquardt algorithm [49], which combines the Gauss Newton algorithm [50] with the method of gradient descent to solve the non-linear criteria involved in all the point correspondences. However, such an optimization is very costly in terms of computing time: Bundle adjustment has a computational complexity of $O(3(m+n))$ per iteration and memory requirements of $O(mn(m+n))$, where m is the number of cameras and n is the number of structure points [51]. In recent years, there has been considerable work focused on increasing the efficiency and speed of SfM algorithms. Zhang et al. [52] proposed an incremental motion estimation algorithm based on a sliding window of triplets of images to process long image sequences. To overcome the severe memory and bandwidth limitations of current generation GPUs, Wang et al. [53] proposed a new inexact Newton-type bundle adjustment algorithm that exploited multi-core CPUs as well as multi-core GPUs for efficiently solving large scale 3D scene reconstruction problems. On the other hand, Liu et al. [54] proposed a new algorithm to simplify the computation of the re-projection error to speed up the intersection step in the interleaved bundle adjustment and the inner minimization in the layered bundle adjustment.

Concurrently, the estimation of the ego-motion of a moving camera and its surroundings has also been addressed by the robotics community from a slightly different point of view. Simultaneous Localization And Mapping (SLAM), or SfM under Bayesian filtering frameworks is the task of sequentially building a map of an

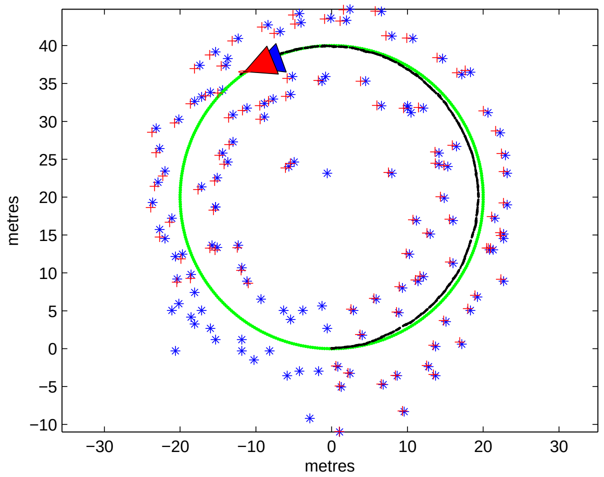


Figure 1.5: The SLAM problem

unknown environment whilst simultaneously localising the position of the camera. In SLAM, the image sequence grows at every step with the addition of new camera poses for each frame processed, where both the trajectory of the camera and the location of all landmarks are estimated in real time without the need for any *a priori* knowledge of location. As shown in Fig. 1.5, the green circles are the true camera trajectory and the blue stars are the true space points (landmarks) observed by the camera. The black circle and the red arrow show the estimated camera trajectory, and the red stars are the estimated positions of landmarks. SLAM processes a simultaneous estimate of both camera and landmark locations; the true locations are never known or measured directly, and observations are made between true camera and landmark locations.

Generally, SfM has addressed the 3D reconstruction problem in its most general form, for example, building 3D models from an un-ordered photo collection, whilst SLAM has focused on sequential approaches for the processing of video input. For example, Pollefeys et al. [44] proposed a large-scale, real-time 3D reconstruction system based on SLAM for large quantities of video data required to reconstruct

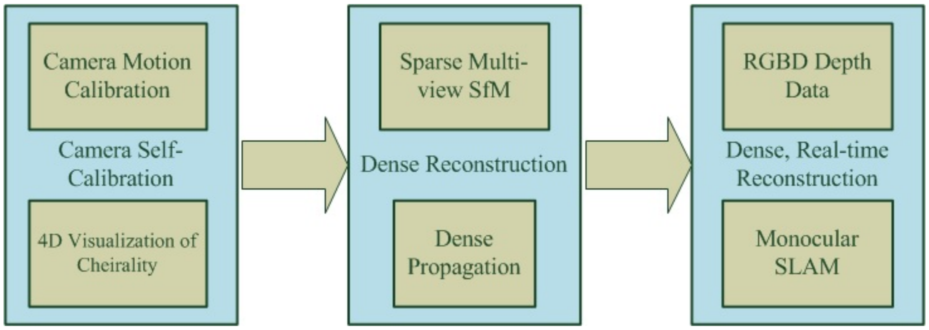


Figure 1.6: Thesis Architecture

entire cities, where the Extended Kalman Filter (EKF) was applied to real-time camera pose estimation and 3D mesh fusion. Monocular camera SLAM systems [55], [56], which naturally overlap with the highly developed SfM field in computer vision, offer a low-cost and real-time approach to 3D reconstruction in un-calibrated image sequences. Davison [55] first proposed real-time monocular tracking in a system which built sparse room-size maps. Clemente et al. [57] and Strasdat et al. [58] proposed a monocular SLAM method to build outdoor, closed-loop maps of several hundred metres in real-time. Motivated by monocular SLAM [55], [56], [44], this thesis aims to improve the efficiency and speed of SfM reconstruction and proposes a real-time 3D reconstruction technique from video sequences based on EKF-SLAM.

1.2 Organisation of the Thesis

1.2.1 The Architecture of the Thesis

Generally, the architecture of the work in this thesis involves three stages, as shown in Fig. 1.6. The first stage is camera calibration: Camera calibration is the estimation of the camera projective matrix including the intrinsic and extrinsic parameters of the camera. The intrinsic parameters include focal length, camera properties, and the extrinsic parameters are the camera motion (translation vector t) and rotation angles (rotation matrix R) of one image relative to the other. The existing four SVD solutions of the essential matrix form four possible solutions: $P \sim \Big[R_1| \pm t\Big]$ or $\Big[R_2| \pm t\Big]$. However, although reconstruction points X visible in the image are constrained to be in front of the camera, the algebraic signs of P and X may still swap randomly, which is termed as cheirality of the point with respect to the camera. This thesis analyses the oriented projective geometry from camera motion viewpoint and investigates the root cause of cheirality. The second stage of this thesis proposes a highly dense 3D reconstruction method from uncalibrated image sequences, where region growing algorithms are used to reconstruct a highly dense surface. Using the initial two images as a 'seed', the subsequent images are merged into the preliminary reconstruction image-by-image. Subsequently, the camera orientations and surface reconstruction are simultaneously computed from new dense point features with SfM techniques. In the third stage of this thesis, SLAM is used to concurrently estimate the structure of the surrounding world in real-time, perceived by moving the sensors (cameras), while simultaneously defining localization information. Initializing features from the environment by the output of a Kinect camera under a monocular SLAM framework, this thesis proposes a robust real-time 3D reconstruction method.

1.2.2 Chapter 2: Background

Chapter 2 discusses the techniques and literature related to the three main classes of research in this thesis: Firstly, camera calibration techniques in computer vision, including camera self-calibration, camera motion estimation and the cheirality problem are reviewed; secondly, 3D reconstruction approaches are summarized; Lastly, real-time 3D reconstruction techniques, especially the advantages and disadvantages of EKF-SLAM are discussed.

1.2.3 Chapter 3: Cheirality in Camera Projective Reconstruction

Starting from eight possible camera motion solutions, this chapter presents a graphical analysis of cheirality in 4D geometry. This chapter proposes and shows that the root cause of cheirality is due to the arbitrary coordiante system in the multiple solutions of the essential matrix for camera motion estimation between two perspective views. This chapter then proposes that the cheirality problem can be resolved by confining all the rotations in multiple views to the right-hand rule with $det(R) = 1$. In computer vision, the handedness problem exists in the camera motion estimation from essential matrix, the linear initialization of the projective matrix, projective transformation and the epipolar estimation. While existing literature proposes four solutions of camera motion from essential matrix [30], which encodes the epipolar geometry between two camera views, this chapter extends the four solutions to eight solutions of camera motion from mathematical computation and geometrical analysis. A camera motion simulation of the proposed eight camera motion solutions is proposed in Chapter 4, and application proposed in Chapter 5 models a 3D object from a 360° image sequence demonstrate and evaluate the reliability of the proposed solutions.

1.2.4 Chapter 4: Camera Motion Simulator

To directly visualize the eight camera motion solutions in a practical camera motion estimation experiment, this chapter proposes the design of an augmented reality camera motion simulator to demonstrate and evaluate continuous camera motion. A flexible, markerless registration method that addresses the problem of superposing a realistic virtual object placed at any position in a video sequence is proposed. The proposed registration method needs no reference fiducials, knowledge of camera parameters or the user environment, where the virtual object can be placed in any environment even without any distinct features. Experimental evaluations demonstrate low errors for several camera motion rotations around the X and Y axes for the proposed camera self-calibration algorithm, and virtual object rendering applications in different user environments are evaluated.

1.2.5 Chapter 5: Dense 3D Reconstruction

This chapter proposes a one-stop 3D reconstruction solution that reconstructs a highly dense surface from an uncalibrated video sequence; the camera orientations and surface reconstruction are simultaneously computed from new dense point features using an approach motivated by SfM techniques. Further, this chapter presents a flexible automatic reconstruction method with the simple interface of 'videos to 3D model'. The reliability of the proposed algorithm has been evaluated on various data sets and the accuracy and performance is compared with both sparse and dense reconstruction benchmark algorithms.

1.2.6 Chapter 6: Real-Time 3D Reconstruction

This chapter proposes a robust two-step Geometric Modelling Iterated Extended Kalman Filter (GMIEKF) SLAM algorithm to recover the 3D trajectory of a freely moving RGB-D camera for multi-view reconstruction applications. The first step of GMIEKF-SLAM takes advantage of the known 3D depth data from an RGB-D camera to geometrically model camera motion for dynamic state modelling, where the measurement residual errors are minimized through a non-linear least square optimization of the *a priori* state parameters. To prevent linearised error propagation and provide a running estimate of camera motion, the second step of GMIEKF-SLAM employs the Iterated Extended Kalman Filter (IEKF) to iteratively linearise the non-linear measurement model. To evaluate the proposed GMIEKF-SLAM technique, 360° trajectory recovery and 3D model reconstruction experiments were conducted in real indoor environments, where results demonstrate that the proposed GMIEKF-SLAM approach provides more robust and consistent estimations compared to the standard EKF algorithm.

1.2.7 Chapter 7: Conclusion and Future Work

The thesis concludes with a summary of the presented work and a discussion of its virtues and limitations. Suggestions are made as to where future research could be conducted in order to further enhance the proposed methods.

1.3 Original Contributions

This thesis presents novel 3D reconstruction techniques and practical camera motion estimation methods for the efficient computation of high quality 3D models from stereo images, multiple images and video sequences. There are six areas in which this thesis presents original contributions:

- Eight possible solutions of the essential matrix from two perspective views [59]: This thesis extends the existing four SVD solutions of the essential matrix that focus on geometrically static scenes to dynamic continuous camera motion. Eight possible camera motion solutions of essential matrix with geometrical analyses and theoretical derivation are proposed, where the eight possible solutions convey the complete camera orientation information under projective geometry.

- Cheirality revisited [60]: This thesis revisits the cheirality problem in computer vision within projective geometry using a camera motion viewpoint. A geometric proof and analysis of the cheirality of 3D points in 4D projective geometry is presented, where the visualisation of the cheirality of points in a

4D space-time framework has not been previously presented in literature. It is shown that the root cause of the cheirality problem is due to the handedness of camera motion, where the cheirality problem can be resolved by confining all rotations in multiple views to the right-hand rule (equivalent to applying the $det(R) = 1$ constraint).

- 4D camera motion simulator [60]: This thesis develops a 4D motion simulator to visualize cheirality points by treating time as movement and examining snapshots of the 4D model at various points in time. The movement of the cheirality of points can thus be easily and directly analysed under the 4D simulator.

- 3D augmented reality camera motion simulator [61]: This thesis develops an OpenGL augmented reality framework which addresses the problem of realistic and accurate placement of virtual objects superposed on an image sequence using the proposed camera motion estimation techniques.

- One-stop dense 3D reconstruction [62]: This thesis develops a flexible automatic system with the simple interface of 'videos to 3D model'. A high density approach to surface reconstruction from a sequence of un-calibrated images is proposed, motivated by the concepts of Structure from Motion (SfM). In particular, this thesis proposes a robust region-growing algorithm for dense matching propagation which addresses deficiencies in the surface integrity resulting from existing approaches. Experimental results indicate that the proposed algorithm performs comparably to existing benchmark sparse and dense reconstruction approaches, and works reliably on real-world objects and environments.

- A real-time 3D reconstruction system [63] [64]: This thesis proposes the Geometric Modelling Iterated EKF-SLAM (GMIEKF-SLAM) algorithm for accurate and robust localization and mapping with RGB-D data e.g., from a Microsoft Kinect camera. It is shown that the mechanism of geometrical camera pose estimation and iterative measurement linearization can be integrated into a novel GMIEKF-SLAM algorithm: Applying a geometric modelling method for dynamic camera motion modelling fundamentally avoids the linear assumption errors of the camera motion model, and is an ubiquitous solution for unknown camera motion in an unknown environment. Compared to the traditional EKF-SLAM approach, experimental results show that GMIEKF-SLAM can improve the camera motion estimation performance in the presence of *a priori* prediction statistics and alleviate the linearization error by iterating the measurement estimation around the update state.

1.4 Publications

- Li Ling, Eva Cheng, Ian Burnett "Analysis of Oriented Projective Geometry from a Camera Motion Viewpoint", *International Conference on Digital Image Computing 2013*, submitted [64]

- Li Ling, Eva Cheng, Ian Burnett "Cheriality Revisit in Camera Projective Reconstruction", *Eurasip Signal Processing Image Communications*, submitted [60].

- Li Ling, Eva Cheng, Ian Burnett "An Iterated Extended Kalman Filter for 3D mapping via Kinect Camera", *The 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP),Vancouver, Canada*, May 2013 [63].

- Li Ling, Ian Burnett and Eva Cheng "A Dense 3D Reconstruction Approach From Un-calibrated Image Sequences" *in Proc. IEEE Int. workshop on Multimedia and Expo. Melbourne, Australia*,9-13 July 2012 [62].

- Li Ling, Ian Burnett and Eva Cheng "A new flexible registration method for video augmented reality", *13th Int. Conf, on Multimedia Signal Processing, Hangzhou, China*, 15-18 Oct. 2011 [61].

- Li Ling, Eva Cheng and Ian Burnett "Eight solutions of the essential matrix for continuous camera motion tracking in video augmented reality", *in Proc. IEEE Int. Conf. on Multimedia and Expo, (Top 15 percentage), Barcelona, Spain*, 11-15 July 2011 [59].

Chapter 2

Background

In this chapter, a review of techniques and previous work that relates to camera calibration, camera motion estimation and structure from motion motivating this thesis work is presented. In addition, the key approaches that are used for comparative purposes or that this thesis work builds upon are outlined.

In the following, some preliminaries are given in Section 2.1. Section 2.2 briefly discusses key concepts and techniques in camera self-calibration. Section 2.3 reviews cheirality problem relevant to camera motion, and Section 2.4 discusses the 3D reconstruction techniques based on SfM. Section 2.5 gives a brief review of Simultaneous Localization And Mapping (SLAM) and discusses the advantages and disadvantages of the extended Kalman Filters typically used in SLAM implementations.

2.1 Preliminaries

2.1.1 Camera Model

In this thesis, the camera is described by a general projective pinhole camera model. A 3D point in space X is mapped to the point on the image plane where a line joining

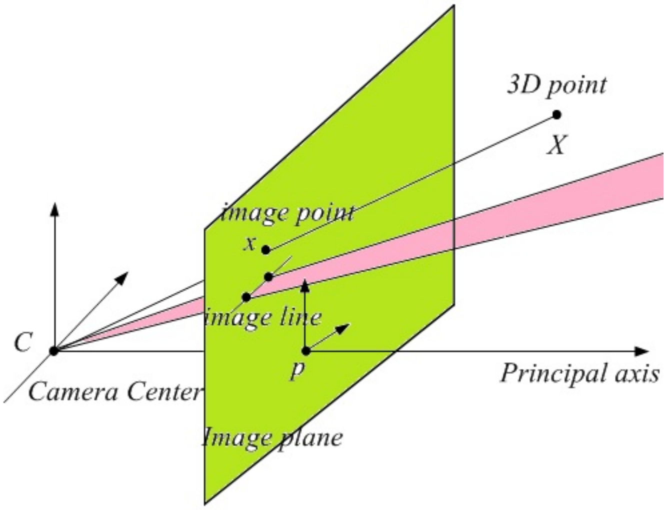


Figure 2.1: C is the camera centre and p the principal point. The camera centre is placed at the coordinate origin. Note that the image plane is placed in front of the camera centre.

the point X to the centre of projection meets the image plane, as shown in Fig. 2.1. The centre of projection is called the camera centre. The camera centre is placed at the coordinate origin and the image plane is placed in front of the camera. The line from the camera centre perpendicular to the image plane is called the principal axis or principal ray of the camera, and the point where the principal axis meets the image plane is called the principal point. As shown in Fig. 2.1, p is the principal point.

Perspective projection can be interpreted naturally using projective geometry. Identifying points along a ray through the projection centre means the 3D world can be interpreted as the projective space $\mathbf{P}^3$, where the image plane is interpreted as the projective plane $\mathbf{P}^2$. Suppose a 3D point $X = \begin{bmatrix} A_x & A_y & A_z \end{bmatrix}^T$ is observed at $x = \begin{bmatrix} u & v \end{bmatrix}^T$ on the image plane of a camera of focal length f. Then, assuming the imaging geometry as depicted in Figure 2.1, where the world coordinate system origin coincides with the camera′s centre of projection, the world axes are aligned with the image plane and the camera faces along the negative depth axis. The perspective projection equations for image formation are given by:

$$\frac{u}{A_x} = \frac{v}{A_y} = \frac{-f}{A_z} \tag{2.1}$$

2.1.2 Projective Geometry and Homogeneous Coordinates

Projective geometry is a fundamental tool for representing SfM problems in computer vision. In projective geometry, the image formation process is regarded as a projective transformation from a 3D to 2D projective space. Homogeneous coordinates provide a mathematical method for computations and theorem proofs in projective geometry. Points in n-dimensional projective space are represented by $n+1$ component column vectors; for example, $X = \begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$ is the homogeneous representation of an arbitrary point X in 3D space. A one-to-one correspondence exists between the points under Euclidean coordinates and the homogeneous coordinates of projective geometry. When $A_w \neq 0$, $\begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$ are the homogeneous coordinates for the Euclidean 3D point $\begin{bmatrix} A_{xE} & A_{yE} & A_{zE} \end{bmatrix}^T$, where the relationship between Euclidean coordinates and the homogeneous coordinates is:

$$\begin{bmatrix} A_{xE} & A_{yE} & A_{zE} \end{bmatrix}^T \sim \begin{bmatrix} \frac{A_x}{A_w} & \frac{A_y}{A_w} & \frac{A_z}{A_w} & 1 \end{bmatrix}^T ; A_{xE} = \frac{A_x}{A_w}, A_{yE} = \frac{A_y}{A_w}, A_{zE} = \frac{A_z}{A_w} \tag{2.2}$$

As shown in Fig. 2.2, the three axes of A_x, A_y and A_w are presented for brevity to illustrate the 4D vector $X = \begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$. The division by A_w indicates that the conversion of a homogeneous point to its Euclidean equivalent is inherently a projection of the homogenous point onto the $A_w = 1$ plane; a point at infinity can be represented by $A_w = 0$ in homogeneous coordinates. Furthermore, $\begin{bmatrix} A_x & A_y & A_z & 0 \end{bmatrix}^T$ and $\begin{bmatrix} -A_x & -A_y & -A_z & 0 \end{bmatrix}^T$ represent the same point at infinity

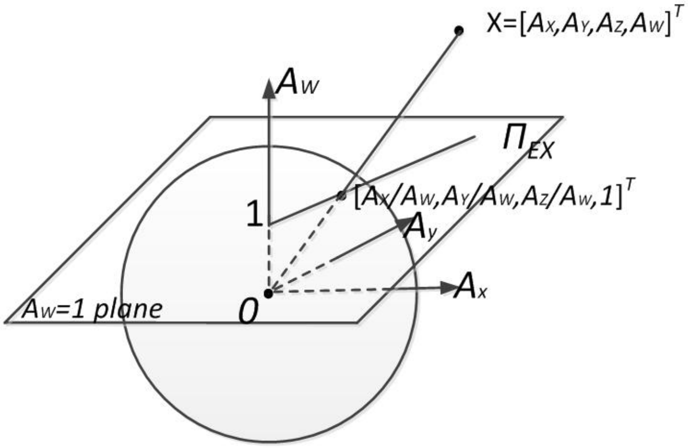


Figure 2.2: The conversion of a homogeneous point to its Euclidean equivalent is inherently a projection of the homogeneous point onto $A_w = 1$ plane, where $A_w = 0$ is the infinite point.

[65]. If two points are sign reversed and $A_w \neq 0$, the point $\begin{bmatrix} \frac{-A_x}{-A_w} & \frac{-A_y}{-A_w} & \frac{-A_z}{-A_w} \end{bmatrix}^T$ is equivalent to the point $\begin{bmatrix} \frac{A_x}{A_w} & \frac{A_y}{A_w} & \frac{A_z}{A_w} \end{bmatrix}^T$ wrapped around infinity returning from the opposite direction [66]. The point $\begin{bmatrix} -A_x & -A_y & -A_z & -A_w \end{bmatrix}^T$ does not present $\begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$, which is called the antipode of X. In this thesis, the sign $\neg$ is used to denote the antipode, thus the antipode of X is presented as $\neg X$.

2.1.3 Geometry of Camera Motion

Fig. 2.3 illustrates the geometry of camera motion, where the original point of the camera coordinate system is defined as the camera centre C_0. In motion, the camera moves to C with a rotation R and a translation t transform, where R is a 3×3 rotation matrix that represents camera orientation and t is a vector that represents camera translation. The extrinsic parameters R and t represent the rigid body transformation between C_0 and C, and the baseline t is the line joining the camera centres C_0C. $X = \begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$ is a homogeneous coordinate point in 3D space, and its

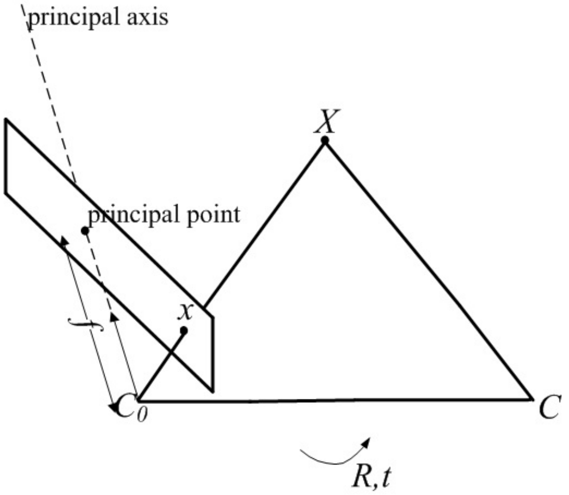


Figure 2.3: Geometry of the camera motion: the camera moves from C_0 to C, X is the 3D-space point, x and x' are the image points on the two image planes. R and t show the movement from C_0 to C, where C_0C is the camera baseline.

image point relative to C_0 is $x = \begin{bmatrix} u & v & w \end{bmatrix}^T$, and $x' = \begin{bmatrix} u' & v' & w' \end{bmatrix}^T$ relative to C. The relationship of the image points x and x' is:

$$\begin{bmatrix} u' & v' & w' \end{bmatrix} E \begin{bmatrix} u & v & w \end{bmatrix}^T = 0 \tag{2.3}$$

where $\begin{bmatrix} u & v & w \end{bmatrix}^T$ and $\begin{bmatrix} u' & v' & w' \end{bmatrix}^T$ are normalized and E is the essential matrix. The projection of X in 3D space to x on an image plane is described by:

$$x = PX \tag{2.4}$$

where P is the 3×4 camera projective matrix. Let $\begin{bmatrix} u_i & v_i & 1 \end{bmatrix}^T$ be the measured image position of 3D point $\begin{bmatrix} A_{xi} & A_{yi} & A_{zi} & 1 \end{bmatrix}^T$, each such correspondence generates two equations that the elements of the projection matrix P must satisfy:

$$u_i = \frac{p_{11}A_{xi} + p_{12}A_{yi} + p_{13}A_{zi} + p_{14}}{p_{31}A_{xi} + p_{32}A_{yi} + p_{33}A_{zi} + p_{34}} \tag{2.5}$$

$$v_i = \frac{p_{21}A_{xi} + p_{22}A_{yi} + p_{23}A_{zi} + p_{24}}{p_{31}A_{xi} + p_{32}A_{yi} + p_{33}A_{zi} + p_{34}} \tag{2.6}$$

For n pairs of corresponding points, $2n$ equations can be rearranged into the form:

$$\begin{bmatrix} A_{x1} & A_{y1} & A_{z1} & 1 & 0 & 0 & 0 & 0 & -u_1A_{x1} & -u_1A_{y1} & -u_1A_{z1} & -u_1 \\ 0 & 0 & 0 & 0 & A_{x1} & A_{y1} & A_{z1} & 1 & -v_1A_{x1} & -v_1A_{y1} & -v_1A_{z1} & -v_1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ A_{xn} & A_{yn} & A_{zn} & 1 & 0 & 0 & 0 & 0 & -u_nA_{xn} & -u_nA_{yn} & -u_nA_{zn} & -u_n \\ 0 & 0 & 0 & 0 & A_{xn} & A_{yn} & A_{zn} & 1 & -v_nA_{xn} & -v_nA_{yn} & -v_nA_{zn} & -v_n \end{bmatrix} \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ p_{34} \end{bmatrix} = 0 \tag{2.7}$$

Since there are 11 unknowns in the projection matrix (scale is arbitrary), at least six pairs of point correspondences are needed. Writing Eq. (2.7) into the form:

$$Bp = 0 \tag{2.8}$$

where p is the 12×1 vector, B is the $2n \times 12$ matrix of measurement, and n is the number of 3D points. The linear least squares solution that minimizes $\|Bp\|$ is given by the unit eigenvector corresponding to the smallest eigenvalue of B^TB; the equation can be solved using the SVD method:

$$B = U\Lambda V^T \tag{2.9}$$

where $\Lambda = diag(\sigma_1, \sigma_2, \ldots, \sigma_{12},)$ is the diagonal matrix of singular values and the matrices U and V are the orthonormal. The columns of V are the eigenvectors of $B^T B$ and the required solution is the column of V corresponding to the smallest singular value σ_{12}. However, the linear solution is only approximate and should be used as the starting point for non-linear optimizations. To find the elements of the projection matrix P that minimize the sum of squared errors between the measured and the back projected pixel positions u_i and $\hat{u}_i = PX_i$:

$$min_P \sum_i \|u_i, PX_i\|^2 \tag{2.10}$$

Once the projection matrix P has been estimated, the first 3×3 sub-matrix can be decomposed by QR decomposition into an upper triangular camera calibration matrix A and an orthonormal rotation matrix R.

2.2 Camera Self-Calibration

Camera calibration is the process of determining the camera intrinsic parameters, which includes the focal length and position of the optical centre and the 3D position and orientation of the camera frame relative to a certain world coordinate system (extrinsic parameters). Camera calibration is traditionally obtained off-line and using a 3D calibration object with a known structure. In the camera calibration approaches proposed by Zhang [67] and Tsai [68], known calibration patterns are utilized to determine the unknown camera parameters. However, most computer vision research does not assume any *a priori* information about the camera calibration. Thus, the approaches of [69], [70], [71], [72], [73], [74], determine internal camera parameters directly from multiple un-calibrated images of unstructured scenes; this approach is called self-calibration.

Self-calibration avoids the need for manual calibration by using a known pattern as a reference to calibrate a camera. This gives great flexibility since a camera can be calibrated directly from an image sequence despite unknown motions. The first approach for self-calibration was proposed by Maybank and Faugeras [75], where the geometric relation between views and both camera-intrinsic parameters and relative orientation between two frames were first determined in the form of fundamental matrices. Subsequently, the Kruppa's equations [75], which are constructed from the fundamental matrices, are used to compute intrinsic camera parameters from the epipolar geometry of two views. In contrast, affine stereo calibration [76] has been identified as the determination of the plane collineation induced by the plane at infinity, where affine calibration is possible with one less point at infinity due to the invariance of the intrinsic parameters for the two cameras. The affine calibration is then updated to a metric representation using the estimated camera intrinsic parameters determined by solving the general camera self-calibration equations. Pollefeys [69], [77] proposed a stratified approach that computes affine structure from the projective model and subsequently addresses self-calibration by upgrading the affine structure to a metric representation. Alternatively, Hartley [29] proposed a self-calibration method where two images are taken from the same point in space with different orientations of the camera and calibration is computed from an analysis of point matches between the two images. Triggs et al. [78] proposed the absolute quadric for camera self-calibration from three or more views taken by a moving camera with fixed but unknown intrinsic parameters.

In the work of this thesis, the camera self-calibration system employed is comprised of four stages, as shown in Fig. 2.4:

- Stage 1- Image Preparation: Feature detection and Matching

  Feature detection and matching is an initial stage of techniques for 3D recon-

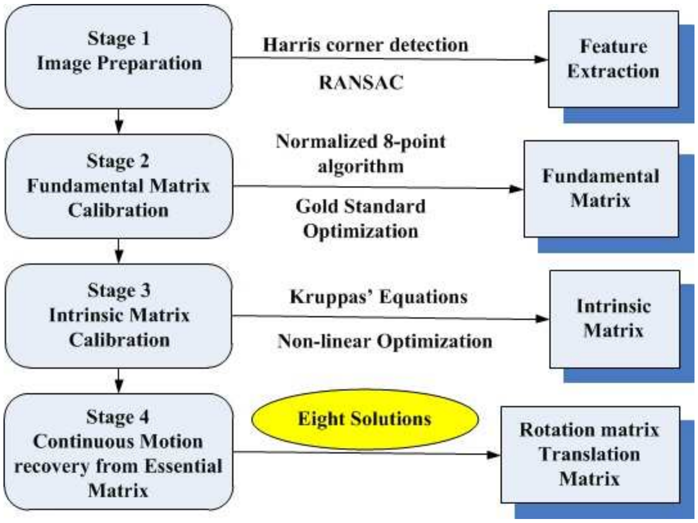


Figure 2.4: Camera Self-Calibration Framework

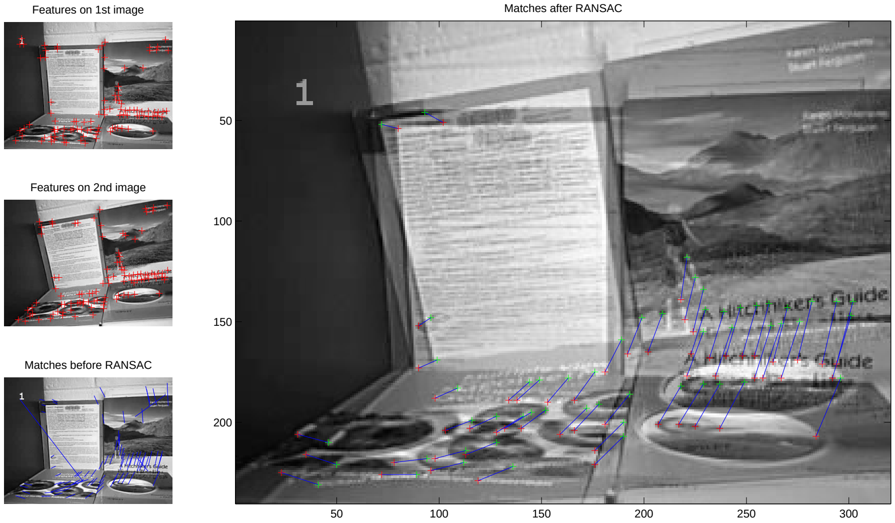


Figure 2.5: Feature Points Extraction and Matching

struction from multiple views. Given a pair of images, a set of correspondences need to be established such that a 3D structure can be constructed or an in-between view can be generated. Firstly, for any object in an image, feature points on the object can be extracted to provide a feature description of the object. Such feature points usually lie on high-contrast regions of the image, such as object edges, corners and blobs. Then, each region around detected feature locations is converted into a more compact and robust (invariant) descriptor that can be matched against other descriptors. Many feature detectors and descriptors have been proposed: Feature from Accelerated Segment Test (FAST) [79], [80], Scale Invariant Feature Transform (SIFT) [81], Speeded Up Robust Features (SURF) [82] and Oriented FAST and Rotated BRIEF (ORB) [83]. Augmented with pyramid schemes for scale, the FAST algorithm [79], [80] can efficiently find reasonable corner feature points. In SIFT, the features have been shown to be invariant to image rotation and scale, robust across a substantial range of affine distortion, addition of noise, and change in illumination. SURF obtains a large speed advantage over SIFT while retaining most of its desirable properties and comparable recognition rates. Binary Robust Independent Elementary Features (BRIEF) [84] makes use of random pixel intensity comparisons to efficiently create a binary descriptor, and ORB adds a fast and accurate orientation component to FAST and uses a learning method for de-correlating BRIEF features under rotational invariance.

Following descriptor extraction, the feature matching stage efficiently searches for likely matching features in other images. Matching metrics, such as the Sum of Squared Differences (SSD) [85] or Normalized Cross-Correlation (NCC) [85] can be used to directly compare the intensities in small patches around each feature point. Following the matching algorithm, the indexing search strategy devises efficient data structures and algorithms to perform the feature matching as quickly as possible. One efficient indexing structure algorithm is kd-trees [86], which divides the multi-dimensional feature space along alternating axis-aligned hyperplanes, choosing the threshold along each axis so as to maximize the search tree balance. However, there are contaminated outliers caused by noise, mismatching and inaccuracies in the feature matches. Apart

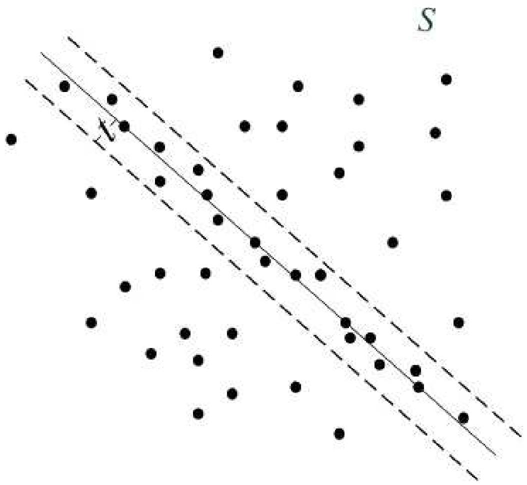


Figure 2.6: In RANSAC, the support for lines through randomly selected point pairs is measured by the number of points within a threshold distance from the solid line. The dash-lines indicate the threshold distance.

from errors in the computation of the point-correspondences, the RANdom SAmple Consensus (RANSAC) [11] method is typically used to remove the outliers from the feature points. Once an initial set of feature correspondences has been established, the camera calibration may be performed.

RANSAC starts from a set of data, iteratively estimating parameters of a mathematical model from a set of observed data which contains outliers, as shown in Fig. 2.6. The RANSAC algorithm consists of five steps:

1. Randomly select a point from the set of points S and instantiate the model from this subset.
2. Determine the set of data points S_i that are within a distance threshold t of the model. The set S_i is the consensus set of the sample and defines the inliers of S.
3. If the size of S_i (the number of inliers) is greater than some threshold T, re-estimate the model using all the points in S_i then terminate.
4. If the size of S_i is less than T, select a new subset and repeat the above.

5. After N trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i.

The distance threshold is defined as $t^2 = F_m^{-1}(\alpha)\sigma^2$ [30] for a probability, in practice, $\alpha = 0.95$ that the point is an inlier. The acceptable consensus set threshold is defined by assuming the proportion of outliers for n data points $T = (1 - \epsilon)n$, where typically $\epsilon = 0.2$. The RANSAC algorithm has two parameters initialization: the number of iterations N and the inlier threshold t. A good value for the inlier threshold can be obtained from the evaluation of the feature point detector. The more accurately the detector can locate the features, the smaller t can be. The RANSAC results are shown in the right image of Fig. 2.5.

- Stage 2- Fundamental Matrix Calibration

  With two views of a scene taken from different view angles, the geometrical relationship between these views is given by epipolar geometry. In epipolar geometry, the geometric relation is expressed with the fundamental matrix. Significant research efforts have been invested into estimating the fundamental matrix from a set of feature correspondences [87], [88]. Hartley [87] formulates the normalized eight-point algorithm as the algebraic relation between pixel locations and camera orientation and position in a linear form, where the nonlinear methods strictly impose the constraints of the fundamental matrix. Additionally, Li et al. [89] and Nister et al. [90] subsequently proved that for two views with five point correspondences, the camera poses and 3D point locations can be determined, proposing five-point algorithms for estimating two-view geometry. The goodness of the estimation can be formulated in a cost function that minimizes the distance of the points to their corresponding epipolar line: for example, minimization of the re-projection error (Gold

Standard method) [30] or the Sampson cost function [30]. Commonly, the non-linear minimization is performed using the Levenberg-Marquardt algorithm [91].

In this thesis, the fundamental matrix is first initialized with the linear normalized eight-point method [87]. The fundamental matrix F defines the epipolar geometry between two images, and can be calculated directly from image correspondences. For any pair of corresponding points $x_i \leftrightarrow x_i^{'}$ in the two images, F satisfies:

$$x_i^{'} F x_i = 0 \tag{2.11}$$

Defining point correspondences $x_i \sim \begin{bmatrix} u_i & v_i & 1 \end{bmatrix}^T$ and $x_i' \sim \begin{bmatrix} u_i' & v_i' & 1 \end{bmatrix}^T$, the constraint on the elements of the fundamental matrix F has the form:

$$\begin{bmatrix} u_i' & v_i' & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = 0 \tag{2.12}$$

For n pairs of correspondences, the constraints can be written as:

$$\begin{bmatrix} u_1'u_1 & u_1'v_1 & v_1'u_1 & v_1'v_1 & v_1' & u_1' & v_1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ u_n'u_n & u_n'v_n & v_n'u_n & v_n'v_n & v_n' & u_n' & v_n & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = 0 \tag{2.13}$$

This equation has the form:

$$Kf = 0 \tag{2.14}$$

where K is a $n \times 9$ measurement matrix, and the Gold Standard method [30] is used to optimize F in the set of corresponding points for each image pair that minimizes the geometric distance d:

$$d = \sum_i d(x_i, \hat{x}_i)^2 + d(x_i^{'}, \hat{x}_i^{'})^2 \tag{2.15}$$

where x_i and $x_i^{'}$ are the measured correspondences, and $\hat{x}_i$ and $\hat{x}_i^{'}$ are the estimated correspondences. Hartley [30] suggested normalising the point correspondences by translating the centroid to the origin and then performing an isotropic scaling such that the RMS distance from the origin is $\sqrt{2}$. This process dramatically increases the stability of the fundamental matrix and other parameter calculations.

- Stage 3- Intrinsic Matrix Calibration

  The intrinsic parameters can be represented by the matrix A:

$$A = \begin{bmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.16}$$

  such that A maps the camera coordinate system into the image coordinate system, where f_u is the magnification in the u coordinate direction, f_v is the magnification in the v coordinate direction, u_0 and v_0 are the coordinates of the principal point, and s is the skew of the coordinate axes. Pixels are usually assumed to be square, in which case $f_u = f_v = f$ and $s = 0$. Hence, f can be considered to be the focal length of the lens expressed in units of the pixel dimension.

Point correspondences between three images, and the fundamental matrices computed from these point correspondences, are typically sufficient for recovering the intrinsic parameters of the camera, the motion parameters, and to compute coherent perspective projection matrices that enable reconstruction of a 3D structure up to a scale factor. This solution utilizes invariant properties of the image of the so-called absolute conic, which is invariant under Euclidean transformations and depends only on the camera intrinsic parameters. The recovery of the image of the absolute conic is equivalent to the recovery of the camera intrinsic parameter matrix, where the constraints on the absolute conic are captured by the Kruppa's equations [92].

The basic assumption is that the intrinsic parameters remain constant throughout the image sequences. Consider the Singular Value Decomposition (SVD) of fundamental matrix F to be UDV^T and the elements of the column vectors of U and V to be u_1, u_2, u_3 and v_1, v_2, v_3; one form of Kruppa's equations is thus given by [30]:

$$\frac{u_2^T w^* u_2}{\sigma_1^2 v_1^T w^* v_1} = -\frac{u_1^T w^* u_2}{\sigma_1 \sigma_2 v_1^T w^* v_2} = \frac{u_1^T w^* u_1}{\sigma_2^2 v_2^T w^* v_2} \tag{2.17}$$

where u_k, v_k and σ_k are the column elements of the SVD of F, and $w^* = AA^T$. Thus, w^* has the form:

$$w^* = \begin{bmatrix} f_x^2 + s^2 + u_0^2 & sf_y + u_0 v_0 & u_0 \\ sf_y + u_0 v_0 & f_y^2 + v_0^2 & v_0 \\ u_0 & v_0 & 1 \end{bmatrix} \tag{2.18}$$

For improved numerical stability and robustness, $n-1$ pairs are extracted from the video sequence to provide more constraints. Then, the non-linear optimization is conducted using the Levenberg-Marquardt algorithm [49], where the cost function has the form:

$$\sum_{i}^{n-1}\left(\frac{u_2^T w^* u_2}{\sigma_1^2 v_1^T w^* v_1}+\frac{u_1^T w^* u_2}{\sigma_1\sigma_2 v_1^T w^* v_2}\right)^2-\left(\frac{u_2^T w^* u_2}{\sigma_1^2 v_1^T w^* v_1}-\frac{u_1^T w^* u_1}{\sigma_2^2 v_2^T w^* v_2}\right)^2 \tag{2.19}$$

- Stage 4- Camera Motion Estimation from the Essential Matrix

  Camera motion estimation is one of the central problems in computer vision, and the camera motion tracking methods are divided into marker based or markerless-based methods. In a marker-based system [22], [23], [24], one or more markers or fiducial points or a known pattern or reference object (denoted by a fiducial mark) are placed or specified beforehand in the region of interest. The fiducial marks are detected and identified to determine the 3D motion of the camera. For example, in ARToolKit applications [24], a square marker of known dimensions is used to define the world coordinate system, and the camera motion is estimated using the four vertices of the marker. Although fiducial marker methods work well in many applications, there are some limitations. First, the environment can only work in a relatively fixed and small sized scene, if the markers are occluded or partially occluded while the camera moves, there will be errors. Second, the projection reconstruction is built by several fiducial marker vertices, this method is not robust especially when noise is added to the scene. In markerless systems [13], [14], [25], [26], [27], the camera motion is tracked using natural scene features without any fiducial (reference) marks in unprepared environments. The full 3D camera motion is estimated based on geometric constraints between feature correspondence points in multiple

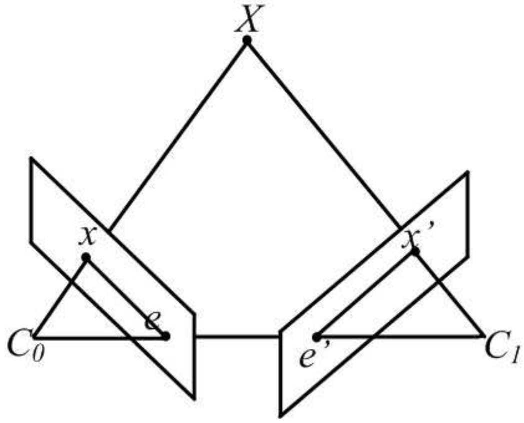


Figure 2.7: The two camera are indicated by their centres C_0 and C_1. The camera baseline intersects each image plane at the epipoles e and e'

images [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27].

As shown in Fig.2.7, the two cameras are indicated by their camera centres C_0 and C_1. The camera centres, 3D point X and its image points x and x' lie in the common epipolar plane. The epipolar geometry between two views is the geometry of the intersection of the image planes with the pencil of planes on a baseline axis. Epipoles are the points of intersection between the line joining the camera centres (the baseline) and the image plane. The epipolar line is the intersection of an epipolar plane with the image plane. The epipolar constraint relating corresponding image points in two views can be formulated using the essential matrix E. Consider two cameras with projection matrices P and P', a pair of corresponding points X' and X in two images is given by:

$$X = RX' + t \tag{2.20}$$

Pre-multiplying both sides by $X^T \Big[t\Big]_\times$ gives:

$$X^T \Big[t\Big]_\times RX' = X^T EX' = 0 \tag{2.21}$$

where the essential matrix E is defined as the cross product of the translation vector t and rotation matrix R:

$$E = \Big[t\Big]_{\times} R \tag{2.22}$$

The essential matrix encodes the epipolar geometry between two camera views, and the normalized essential matrix is given by:

$$\tilde{x}^T E \tilde{x} = 0 \tag{2.23}$$

where $\tilde{x}$ is the image point expressed in the normalized form of x, $\tilde{x} = A^{-1}x$ The relationship between the fundamental and essential matrices is thus:

$$E = A^{'T} F A \tag{2.24}$$

With the fundamental and intrinsic matrices known, two corresponding views are sufficient to compute the motion from the essential matrix. Substituting Eq. (2.24) into Eq. (2.23), the error function ε of the essential matrix is given by:

$$\varepsilon = \sum_i (x_i^{'} A^{-T} E A^{-1} x_i)^2 \tag{2.25}$$

Once the essential matrix is known, the camera motion can be reconstructed, and the camera pose and orientation are also thus known. Due to the sign ambiguity of the essential matrix, a novel algorithm composed of three constraints for choosing the correct solution from eight possible essential matrix solutions is proposed in Chapter 3.

2.3 Projective Geometry and Cheirality

Projective geometry and homogeneous coordinates are widely used in computer vision to provide a mathematical representation for computations and theorem proofs for pinhole camera projection instead of Euclidean geometry. However, the orientation of the lines, planes and intersections in projective geometry can be wrapped back from infinity under four dimensional homogeneous coordinates, which is called the non-orientation of projective geometry. This non-orientation degrades the ability of projective geometry to present the orientation of the camera (the 'front' and the 'back' range), and loses the constraints that a point lying in an image must lie in front of the camera producing that image [93].

The problem of enforcing the orientations in camera motion calibration, the projective matrix, projective transformation and epipoles estimation has great practical importance in object tracking, structure from motion, augmented reality and 3D shape recognition. There are a large number of works reported in this domain: To distinguish that points are in front of the camera, Stolfi [94] developed the oriented projective geometry framework. For an $n+1$ dimensional vector space, Stolfi constructed a canonical two-side space T^n with the 'front' and 'back' ranges divided by an infinite point, where the front range is considered as the set of 'real' points X and the back range as the set of 'phantom' points, the antipodal points $\neg X$. Laveau et al., [37] extended the oriented projective geometry into computer vision, applied to solve problems in stereo reconstruction, epipolar constraints and the convex hull of an object from multiple images.

Oriented projective geometry is implicitly expressed by the cheirality problem proposed by Hartley [34], [35], [30]: A projective reconstruction may manifest as a violation of the convex hull of scene points when interpreted in Euclidean coordinate frames [35]. This problem is due to the projective ambiguity represented by algebraic signs of the 3D points randomly swapping from positive to negative during the reconstruction from multiple images under Euclidean coordinate frames; the correct camera projection matrices and 3D reconstruction points thus require multiplication by -1 as necessary to ensure that the projections and reconstructions are in front of the camera producing the image [34]. To determine the visible points lying in 'front' of the camera, Hartley [34],[35] proposed a quasi-affine reconstruction that preserves the convex hull of the set of points. A quasi-affine reconstruction is a projective reconstruction where the reconstruction scene is not split across the infinite plane. The quasi-affine reconstruction can be computed from a projective reconstruction by solving the linear cheiral inequalities which are proposed to ensure that any 3D point visible in an image must lie in front of the camera producing that image. Although the cheiral inequalities can be solved by linear programming, the solutions are not unique: One or possibly two differently oriented quasi-affine reconstructions of the scene are obtained. Cheirality invariant constraints then were proposed to determine a set of orientation preserving points in the two consecutive images: The sum of the algebraic sign values of 3D points X_i to the plane at infinity for all projective possibilities are computed (denoted as cheiral sequences, where the positive or negative sign of projections of X_i is treated as 1 or -1), and the two images are considered to be projectively equivalent if they have the same cheiral sequences. Subsequent applications based on Hartley's cheirality theories were proposed [38], [40], [95], [96], [97], [98].

2.4 Structure from Motion

Structure from Motion (SfM) approaches that reconstruct a 3D model from multiple views taken from one object or one scene can be catalogued into three classes:

depth-map merging, volumetric-based, and feature-point based.

The depth-map merging SfM reconstruction method can typically be divided into two separate processes: The independent depth of each image is estimated from disparities between adjacent stereo image pairs; all the independent depth-maps are then merged into a common 3D model. This two-stage strategy enables different applications, such as [99], [100], [101], [4], [102], [103] and [104]. Goesele et al. [101], [4] and Bradley et al. [99], [100] used a scaled window matching scheme to assign one depth candidate for each pixel: the Normalized Cross Correlation (NCC) applied to the intensities of the square pixel regions was used as a metric for the best match. Further, Liu et al. [105], [106] introduced the Multiple Starting Scales (MSS) to generate the multiple depth candidates for the final depth map synthesis. Zach et al. [107] used total variation regulation and the L1 norm to generate a 3D model from the depth map. Deng et al. [108] proposed log-sum penalty completion for matrix completion to fuse the noisy point clouds from multiple views. Mordohai et al. [109] proposed a real time depth-based reconstruction of urban environments via GPU computation and Merrell et al. [104] presented a viewpoint-based approach for the quick fusion of multiple stereo depth maps. Among all the techniques in multi-view stereoscopic depth-map merging, the photo-consistency of visual correspondence among images is crucial to the 3D reconstruction performance. Recently, optimization techniques based on graph cuts have been proposed [110],[111], [112], [113] in addition to minimizing energy functions corresponding to pairwise Markov Random Fields (MRFs) to obtain the depth map [114].

However, merged depth maps often result in incomplete models due to occlusions or the inaccurate estimation of the discontinuities along the object borders at different depths. Volumetric methods [112], [113], [115] and [46] thus directly work on the 3D object space and do not require a matching process between images. In this class of approaches are well known techniques of volumetric graph-cuts [112], [113] and space carving [46]. Incorporating the surface regularization into the volumetric fusion framework, volumetric graph cuts [112], [113] formulate a photo-consistent cost function for 3D volume and extract the complete surface from the discrete voxels. Since graph cuts typically require a lot of memory for high resolution volumes, Lempitsky and Boykov [116] proposed a memory-efficient touch-expand algorithm without performing computations on a full grid. In contrast, in the space carving [46] framework, the non-photo-consistent voxels in the space are greedily carved until all visible surface voxels are photo-consistent. The probabilistic space carving algorithm [117] uses the probability of the voxel existence to avoid reliance on the global threshold parameter. In these volumetric methods, the convergence properties in the presence of noise are not well-understood and are susceptible to local minima convergence. Further, the scene in the volumetric based approaches is first represented as a set of 3D voxels, before the use of energy minimization to determine whether or not voxels should be filled. The accuracy of volumetric based approaches is also limited by the resolution of the voxel grid.

In contrast, feature-point based algorithms [118], [4], [119], [120], [121] firstly extract and match a set of feature points and then reconstruct the surface with geometric, photometric or visualization constraints. Feature extraction is followed by feature matching, and the resulting set of correspondences is used to compute the camera parameters and pose. Tsai [68] and Zhang [122], [68] first proposed camera motion estimation and reconstruction from correspondent feature points. There are two classes of surface reconstruction approaches: Sparse [4], [120], [121] and dense [118], [47]. Generally, the sparse SfM [4], [120] technique focuses on robust, efficient and automatical recovery of camera motions and sparse 3D scene geometry from a large set of unorganized images. In contrast, the surface growing approaches [118][121] propagate a dense set of small surfaces with global visibility constraints by minimizing the effects of outliers repeatedly; this feature-point based dense SfM is the approach adopted in this thesis and is further detailed in the Chapter 5.

2.4.1 Stereo Reconstruction

The simplest approach to feature-point SfM involves just two images. Hesse [123] and Sturm [124] proposed a method to compute the epipolar geometry relating two images from seven point correspondences. Longuet-Higgins proposed the reconstruction of a scene from two views using eight point correspondences [12]. Nister [90] subsequently proposed a five-point algorithm to estimate the camera pose from two views. After the camera position and orientation in each view is estimated from the five-point algorithm, the 3D points can be solved using triangulation.

Triangulation applies projective geometry to determine an unknown point or location by using the position of two fixed points a known distance apart. The camera projective matrix can be expressed by the camera intrinsic matrix and the camera pose. Each image has a measurement $x = PX$, $x' = P'X$, and these equations can be combined into a form $AX = 0$, which is an linear equation in X. The homogeneous scale factor is eliminated by a cross product:

$$x \times (PX) = 0 \tag{2.26}$$

where an equation of the form $AX = 0$ can be computed with SVD.

Whilst it is possible to recover 3D coordinates given only two observations or two image coordinates, the accuracy is highly dependent upon the exact match between

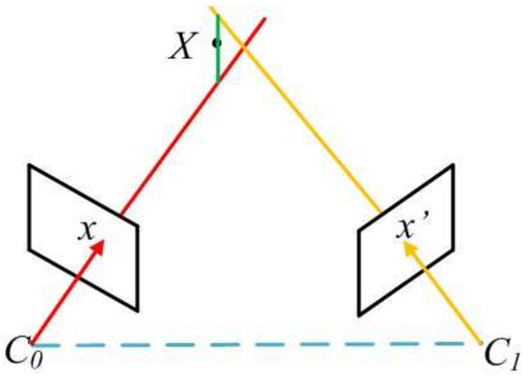


Figure 2.8: Two rays will not actually intersect in space due to errors in calibration and correspondences

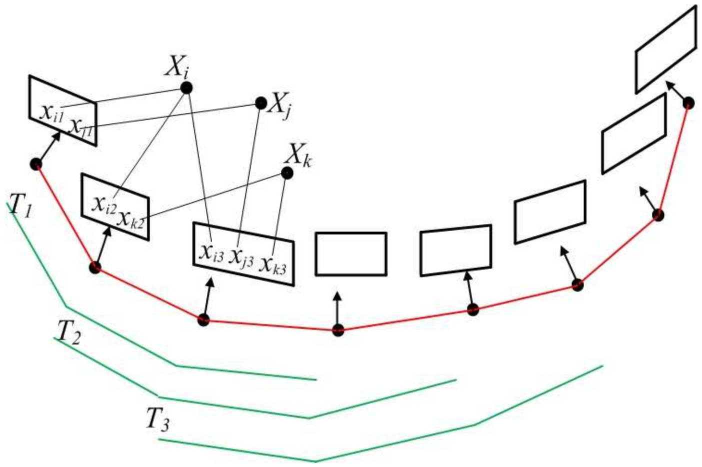


Figure 2.9: The merging of image triplets using trifocal tensors

the two image points. Since there are generally errors involved, a set of points is usually used and an over-determined system is solved, as shown in Fig. 2.8. More accurate 3D triangulated points can be obtained by optimizing the minimization of an appropriate error metric: For example, the Gold Standard reconstruction algorithm [30] or Sampson cost function [30] minimizes the sum of squared errors between the measured and predicted image positions of the 3D point in all views in which it is visible, and the non-linear optimisation is performed by the Levenberg-Marquardt algorithm [125].

2.4.2 Multiple View Reconstrution

The key problem in multiple view structure from motion is the determination of the 3D location of a point in a scene from multiple images taken from different view points. The mathematical and algorithmic aspects of the three-view problem have also received a great deal of attention. The trifocal tensor [126], [124], [127], [128], [129] plays an analogous role in three views to that played by the fundamental matrix in two. The projective geometry relating three views of a scene is encapsulated by the trifocal tensor and is independent of scene structure. One of the most important consequences of the move from two to three-view geometry is that, although a point in one view still only constrains corresponding points in the other two views to lie on the appropriate epipolar line, correspondences between two of the views uniquely defines the position of the point in the third view. In multiple view reconstruction, trifocal tensors are calculated separately for overlapping triplets of all the images in the sequence; this overlap allows correspondences to be carried through the sequence. Fig. 2.9 illustrates the merging of image triplets using trifocal tensors. Camera projection matrices are extracted from the overlapping tensors and each subsequent tensor can be added into the same projective frame as the first.

Motivated by image triplets and trifocal tensors, Pollefeys [69] proposed structure from motion reconstruction from uncalibrated image sequences. The surface reconstruction starts with two 'initial' images and updates when each subsequent image is merged into the projective frame defined by the first two images. The subsequent images are merged into the preliminary reconstruction image-by-image with two steps: First, the matches that correspond to an already reconstructed point are used to compute the new projective matrix; second, the reconstruction is updated by initializing new points for new matches, refining these points and deleting incorrect points. For example, in Fig. 2.9, image 3 firstly matches points corresponding to tracks from images 1 and 2, where x_{i3} is in the same track as x_{i1} and x_{i2} and are hence connected to the same 3D point X_i. The 3D point X_i was initialized by images 1 and 2, thus P_3 can be calculated with the Direct Linear Transform (DLT) [61]. The new tracks on image 3 may then be added: as illustrated in Fig. 2.9, x_{j1} and x_{j3} form a new track, while x_{k2} and x_{k3} consist of another new track. The new 3D points X_j and X_k can thus be calculated from the linear triangulation method [59].

2.4.3 Dense Propagation

The basic idea of dense propagation algorithms is to start from a set of sparse matches as seed points, then propagate the seed points to neighbouring pixels using a region growing technique. Patch-based Multi-View Stereo (PMVS) [118] propagates a dense set of small patches covering the surfaces, based on the calibrated information of Bundler [120], [5]. Lhuillier et al. [47], [119] proposed a dense pixel matching approach that simultaneously expanded the initialized sparse matching to immediate neighbouring areas in two images. Avoiding mismatches at small noise points or nearly repetitive patterns, Tang et al. [48] proposed a two-window matching procedure: a larger window is used to contain enough intensity variation to achieve reliable matching, whilst a smaller window is used to obtain accurate matches. Xing et al. [130] improved the accuracy and speed of the region growing algorithm between two 2D images by using the best-first strategy to select the seed points. The epipolar line constraint and continuity constraint then reduces the double phase matching course into single phase matching before a dynamic and adaptive window is adopted instead of the large window for the region propagation.

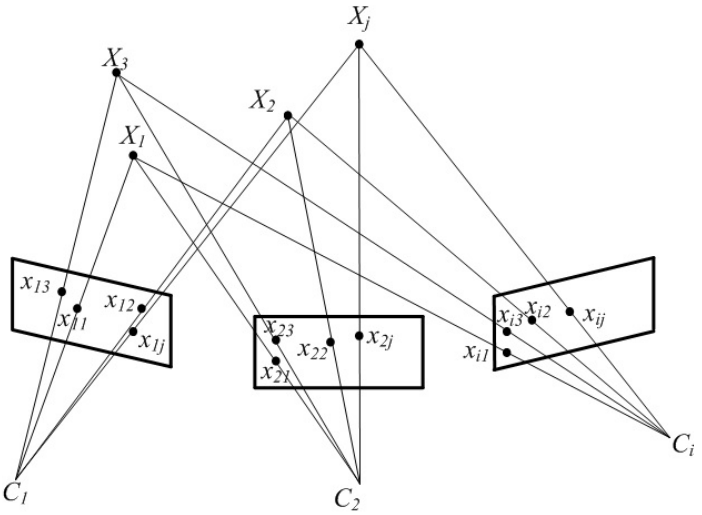


Figure 2.10: Bundle adjustment problem of n 3D points in m images

2.4.4 Bundle Adjustment

Bundle adjustment [131], [132] is used iteratively to refine structure and motion parameters by the minimisation of a cost function. Bundle adjustment aims to refine a visual reconstruction to simultanously produce optimal 3D structure and camera pose estimates, which are then widely applied to many similar estimation problems in computer vision, geodesy, photogrammetry, industrial metrology, and 3D recognition. The bundle adjustment optimization problem is usually formulated as a non-linear least squares problem, where the error is the squared L_2 norm of the difference between the observed feature location and the projection of the corresponding 3D point on the image plane of the camera. Suppose a set of n 3D points are visible in m perspective images. As shown in Fig. 2.10, P_i is the camera projective matrix of the i-th image (where $i = 1 \dots m$), and x_{ij} are the homogeneous coordinate vectors of the image points (where $j = 1 \dots n$). The global solutions of 3D points X_j and P_i are resolved by bundle adjustment according to the minimization:

$$E = min_{P_i, X_{ij}} \sum_{i=1}^{m} \sum_{j=1}^{n} d(P_i X_{ij}, x_{ij})^2 \tag{2.27}$$

However, two main shortcomings of bundle adjustment are: Firstly, bundle adjustment requires a good initialization to be provided; secondly, bundle adjustment can be an extremely large minimization problem because of the number of parameters involved. For example, since each camera has 11 degrees of freedom and each 3D point has 3 degrees of freedom, a reconstruction involving n points over m views thus requires minimization over $3n + 11m$ parameters. If the Levenberg-Marquardt algorithm is used, then matrices of dimension $(3n + 11m) \times (3n + 11m)$ must be factorized. As m and n increase, this becomes extremely costly; even approaches that take advantage of sparsity can become very slow when the number of cameras is large. Thus, sparse bundle adjustment methods [131] were proposed, with approximately cubic complexity in the number of cameras.

2.5 Real-Time 3D Reconstruction: EKF-SLAM

In the robotics community, SLAM has been used to estimate the motion of a moving camera and 3D localization of the camera surroundings. The first SLAM algorithm [133] was comprised of an explicit and consistent representation of uncertainty, and therefore provided the qualified map convergence, which built the foundation of all subsequent SLAM methods using landmark-based map frameworks. Almost concurrently, Thrun et al. [134] introduced the degree of convergence between Kalman filter-based methods and probabilistic localisation and mapping-based SLAM methods. The most common representation in SLAM is the Extended Kalman Filter (EKF) comprised of a state-space model with additive Gaussian noise.

The Extended Kalman Filter (EKF) is an algorithm that operates recursively on streams of noisy measurements observed over time and produces statistically optimal estimates of unknown variables. Motivated by several advantages, the EKF algorithm for real-time camera motion estimation has become one of the key approaches to SLAM. Firstly, as the EKF is a dynamic and recursive implementation, the use of the EKF for the estimation of the state vector to track continuous camera motion is efficient. Pollefeys et al. [44] proposed automatic, geo-registered, real-time 3D reconstruction from videos of urban scenes using an acquisition system consisting of eight cameras mounted on a vehicle. To perform the camera pose estimation and fusion with GPS and inertial measurements, the EKF is modelled as smooth motion with constant velocity in translation and rotation. The dense 3D models are then formed as textured polygonal meshes, where a large-scale urban area can be reconstructed from millions of video frames at approximately 30 frames/s.

Secondly, the EKF can be used with a reduced number of features that enables the estimation in real-time. Davison et al. [55] proposed an EKF-based approach for real-time estimation of combined target model and pose for uncertain environments with an unknown object model. Moreover, adding or removing the features from the EKF can be arbitrary, which simplifies the handling of occluded features. The features management proposed in [55] simply adds or deletes the rows and columns of the state vector and covariance matrix. In contrast, for other motion tracking techniques, such as SfM [120], [5], the procedures for motion-tracking methods are much more complicated [62].

Thirdly, the EKF is based on two fundamental assumptions: The processing and measurement noise should be white/uncorrelated or Gaussian noise with zero mean, and the motion model should be known or almost linear on the time scale of the updates. If both assumptions are satisfied, it is possible for the EKF to make a very accurate and reliable state prediction using just the previous position and the estimated motion. Yun et al. [135] presented an EKF designed for accurate real-time estimation of the movement of human limb segments, reducing the dimension of the state vector and linearizing the measurement equations using a Quest algorithm to pre-process the acceleration and magnetometer measurements. Based on a predefined transformation model, the work in [136] uses the EKF to recursively find the pose of an object in servo robot manipulators, where the transformation model was obtained by fixing a reference object at a known location with respect to the test system.

Despite many applications utilizing the EKF, there are a few shortcomings of the EKF algorithm when EKF is applied to free-moving camera motion tracking in SLAM:

Issue 1: A well known limitation in the application of EKF is the assumption of *a priori* knowledge of the camera motion model and the noise statistics in state and measurement processing. Typically, the *a priori* information is tuned to the experiments before hand. In most practical situations, such *a priori* information is unknown. The poor use of *a priori* information in the design of an EKF can lead to estimation errors or even to a divergence of estimate results. Bailey [137] claimed that the main source of inconsistencies is camera motion model variance which, if large, can lead to failure in just a few updates. Only when the camera motion model uncertainty is small, can dynamically adapting the process and measurement noise improve the result. MonoSLAM [55] [138] is an EKF that performs real-time motion and structure estimation from a single free-moving camera by smoothing the camera motion accelerations with a constant profile. MonoSLAM assumes that the statistics of measurement and dynamic noise are known and remain constant. However, in most practical situations, the statistics of process and measurement noise are unknown. When such unknown noise enters the measurement models in a nonlinear manner, the poor estimates degrade system performance and may lead to biased estimation, where the expected value of the estimator is not the true value of the states. Traditional solutions to resolve this problem tune the covariance matrices by experiments, however, the dynamic process noise covariance matrix is difficult to tune. Applied to a target tracking simulation, Alcantarilla et al. [139] presented the mathematical and empirical evidence of correlations between the noise statistics and innovation sequences causing an estimation bias in the EKF. Foo et al. [140] proposed the combination of the Interacting Multiple Model (IMM) method and variants of Particle Filters (PFs) for manaeuvring target tracking, where particle filters are adopted to account for the non-linear or non-Gaussian characteristics of the target motion model.

Issue 2: The feature initialization uncertainty is a problem critical to the EKF that precipitates immediate and substantial estimation inconsistency. As feature depths cannot be initialized from a single observation, the feature initialization uncertainty is one of the main difficulties of monocular visual SLAM. Civera et al. [141] and Montiel et al. [142] proposed an inverse depth parametrization for single camera EKF-SLAM that permits efficient representation of the Gaussian linearity of the measurement model. Assuming Gaussian uncertainty of the parameters, inverse depth parametrization can process feature points from nearby to infinity without delay. However, this linear parametrization method causes a non-linearity reduction of the measurement model; in realistic applications, the measurement model under large uncertainties from the camera and environment cannot always maintain linearity.

Issue 3: The EKF linearizes the measurement prediction and all unknown transformations in the state prediction using a first order series expansion, substituting Jacobian matrices for linear transformations in the Kalman filter. In practice, the EKF is often used for nonlinear systems by linearizing the process and measurement function of the system. Such linearizations assume that the prediction errors can be well-approximated by a linear function, thus the higher order Taylor series expansions are ignored. If this condition cannot be satisfied, the errors will propagate and result in divergence of the estimations; this problem is well documented in many applications [143], [144], [145], [146]. Taking the linearization errors into account, Lefebvre et al. [147] compared the performance of the EKF, Iterated EKF (IEKF), Central Difference Filter (CDF), first order Divided Difference Filter (DD1) and Unscented Kalman Filter (UKF), concluding that the performance of the process and measurement updates is due to the linearization of the process function, measurement function and state estimate and its uncertainty. A key result from [147] is that the IEFK is the most accurate way to process a nonlinear measurement model since it uses the re-linearization of the measurement function, yet requires careful tuning. IEKF has been precisely applied to alleviate the linearization error in EKF-SLAM applications [148], as the IEKF re-linearizes the measurement equation by iterating an approximate maximum a posteriori (MAP) estimate around the updated state, rather than relying on the predicted state. Bell et al. [149] proposed a Gauss-Newton iterated method for approximating a maximum likelihood estimation on nonlinear update for EKF. In the sensitivity analysis of EKF and IEKF for camera pose estimation, Shademan et al. [150] proposed that the performance and convergence of the EKF were highly sensitive to feature outliers and occlusion, 3D feature initialization and tuning of noise parameters. Shojaei et al. [151] investigated the effects of iteration on EKF and Sigma Point Kalman Filter (SPKF), where iterated versions of Kalman filters were found to increase consistency and robustness against linear error propagation.

Thus, the work of this thesis targets the improvement of the SLAM algorithm to provide more robust and consistent estimations to recover the 3D trajectory of a free moving RGB-D camera in real-time for multi-view reconstruction applications.

Part I

Camera Motion Calibration

Chapter 3

Review of Cheirality in Camera Motion Calibration

Chapter 2 reviewed the key concepts and techniques in cheirality. Existing attempts of cheirality by orienting quasi-affine reconstructions whose projectivities preserve the convex hull of an object of interest; however, the main difficulty lies in obtaining the plane at infinity in the 3D projective coordinate frame, especially as the determination of the orientation of the quasi-affine reconstruction with respect to the infinite plane is mathematically and geometrically non-trivial. This chapter revisits the cheirality problem from first principles and proposes the root cause of cheirality to be due to the handedness of the cross product of two variables about rotation. Starting from two discrepancies between the definition of cheirality and 3D projective geometry between Euclidean coordinates and homogeneous coordinates, a 4D geometric visualization and analysis of cheirality in homogenous coordinates is presented to show the handedness cause of cheirality. And the cheirality problem can be resolved by confining all rotations in multiple views to the right-hand rule, equivalent to applying the $det(R) = 1$ constraint. For a projective reconstruction, the same cheirality problem exists in the Singular Value Decomposition(SVD) of essential matrix for camera motion estimation, and the Direct Linear Tansform (DLT) of projection matrix, and projective transformation.

3.1 Introduction

The problem of projective reconstruction from image sequence is one of the central problems in computer vision. Typically, the camera projection matrices including the camera motion parameters (rotation and translation) are firstly evaluated image-by-image; these parameters are then used to reconstruct the image points into 3D space points. However, this problem suffers from the projective ambiguity that the algebra signs of the 3D points randomly swap from positive to negative during the reconstruction from multiple images under Euclidean coordinate frame; the correct camera projection matrices and 3D reconstruction points require multiplication by -1 if necessary to ensure that the projections and reconstructions are in front of the camera producing that image [35]. The camera projective transforms possess the property of swapping points from the front to the back of the camera, and the problem of determining whether a 3D point lies in front of or behind a given camera is termed as the cheirality of the point with respect to the camera [35]. The camera can only view the points on one side of the principal plane; those points are in front of the camera. Points on the other side will not be visible.

To address the cheirality problem, [34] and [35] proposed a quasi-affine reconstruction that preserves the convex hull of the set of points. A quasi-affine reconstruction is a projective reconstruction where the reconstruction scene is not split across the infinite plane. The quasi-affine reconstruction can be computed from a projective reconstruction by solving the linear cheiral inequalities which are proposed to ensure that any 3D point visible in an image must lie in front of the camera producing that image. Although the cheiral inequalities can be solved by linear programming, the solutions are not unique: One or possibly two differently oriented quasi-affine reconstruction of the scene are obtained. [34], [35], [30] claimed that 3D points X_i and camera projection matrices P_j may be normalized by multiplying by -1 if necessary. Cheirality invariant constraints then were proposed to determine a set of points orientation preserving in the consecutive two images: The sum of the algebraic sign values of 3D points X_i to the plane at infinity for all projective possibilities are computed (denoted as cheiral sequences, where the positive or negative sign of projections of X_i is treated as 1 or -1), and the two images are considered to be projectively equivalent if they have the same cheiral sequences. Subsequent applications based on the cheirality theorys were proposed [38], [40], [95], [96], [97], [1]. However, these applications suffer from main difficulties: Estimating the position of the plane at infinity in a projective reconstruction is considered the most difficult obstacle to obtain an affine reconstruction, especially as the determination of the orientation of the quasi-affine reconstruction with respect to the infinite plane is mathematically and geometrically non-trivial.

In contrast, many other projective reconstruction approaches have **not** encountered the cheirality problem. [12] first proposed four distinct solutions to the two prospective views with the possibility of two algebraic signs for the essential matrix $\pm E$ and translation vector $\pm t$, under the constraint $det(R) = 1$. Utilising this $det(R) = 1$ constraint, [31], [32], [33] proposed a multi-stage approach to camera motion and structure estimation to correctly recover 3D projections from multiple views. Almost concurrently, [152], [153], [154] proposed the technique of applying SVD to the essential matrix for camera motion estimation; subsequently, [16] developed the famous two-stage 3D camera calibration method, which uniquely determined the camera position t and orientation R relative to object reference coordinate systems for multiple images. The camera calibration method of [16] was successfully used in many applications [155], [156], [157], [158], [159] etc.

This chapter targets to understand the root cause of cheirality from first principles, geometrically and through mathematical derivation, to address why some researchers have encountered the cheirality problem whilst others have not. This chapter starts from the theories of the camera motion estimation, and two discrepancies between the definition of cheirality are addressed using Euclidean versus homogeneous coordinates. A geometric proof and analysis of the cheirality of 3D points in 4D projective geometry shows the handedness (right-handed and left-handed coordinate system) essence of cheirality. The root cause of cheirality is the handedness problem due to the cross product of two variables about rotation. For a projective reconstruction, this cheirality problem exists in the SVD estimation of the camera motion estimation from the essential matrix, the DLT linear estimation of projection matrix P and projective transformation H. The cheirality problem can be resolved by confining all rotations to the right-hand rule (equivalent to applying the $det(R) = 1$ constraint). To validate the theoretical derivation and proposed solution, results from a 3D reconstruction application conducted using a 360° image sequence (multi-view projections) are presented and discussed. To demonstrate and evaluate the proposed handedness and constraint, two experiments are presented in this thesis in Chapter 4 and 5, respectively: a 3D camera motion simulator presents the eight possible camera motions recovered from two perspective views in a camera self-calibration approach; and, a model reconstruction experiment that is directly resolved from a 360° image sequence.

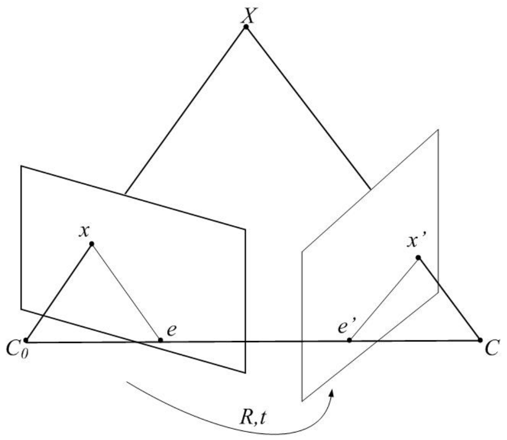


Figure 3.1: Geometry of the camera motion: the camera moves from C_0 to C, X is the 3D-space point, x and x' the image points on the two image planes. R and t show the movement from C_0 to C, where C_0C is the camera baseline. The camera baseline intersects each image plane at the epipoles e and e'

3.2 Preliminaries

The camera is described by a general projective pinhole camera model. Fig. 3.1 illustrates the camera motion geometry. The original point of the camera coordinate system is defined as the camera centre C_0. In motion, the camera centre moves from C_0 to C with a rotation R and a translation t transform. The baseline t is the line joining the camera centres C_0C. $X = \begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$ is a homogeneous coordinate point in 3D space, and its image point relative to C_0 is $x = \begin{bmatrix} u & v & w \end{bmatrix}^T$, and $x' = \begin{bmatrix} u' & v' & w' \end{bmatrix}^T$ relative to C. The relationship of the image points x and x' is:

$$\begin{bmatrix} u' & v' & w' \end{bmatrix} E \begin{bmatrix} u & v & w \end{bmatrix}^T = 0 \tag{3.1}$$

where $\begin{bmatrix} u & v & w \end{bmatrix}^T$ and $\begin{bmatrix} u' & v' & w' \end{bmatrix}^T$ are normalized and E is the essential matrix. [12] first solved the problem of camera motion estimation from two perspective views as:

$$E = T_\times R \tag{3.2}$$

where R is the 3×3 orthonormal matrix with $det(R) = 1$ and $T_\times$ is a skew-symmetric matrix composed of the translation variables:

$$T_\times = \begin{bmatrix} 0 & t_z & -t_y \\ -t_z & 0 & t_x \\ t_y & -t_x & 0 \end{bmatrix} \tag{3.3}$$

The translation vector t is represented as:

$$t = \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^T \tag{3.4}$$

The projection of X in 3D space to x on an image plane is described by:

$$x = PX \tag{3.5}$$

where P is the 3×4 camera projection matrix. Without loss of generality, the reference image is defined at the image with the camera centre C_0, and the camera projection matrix can be represented as:

$$P_0 = A \begin{bmatrix} I|0 \end{bmatrix} \tag{3.6}$$

where I is the identity matrix, A is the 3×3 intrinsic matrix that represents the camera internal parameters, including the focus length and the image centre. The projection matrix of the corresponding image at camera centre C thus has the form:

$$P = A \begin{bmatrix} R|t \end{bmatrix} \tag{3.7}$$

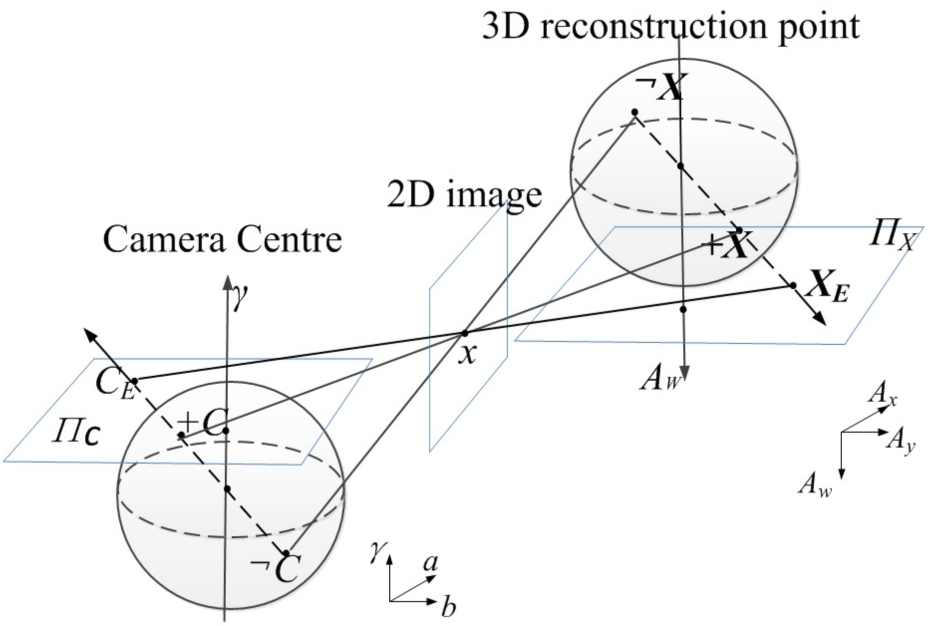


Figure 3.2: In (R_1, t) and $(-R_1, -t)$, the $+C$ and $\neg C$ camera centres project the $+X$ and $\neg X$ reconstruction points in projective geometry. But in the Euclidean plane, $+C$ and $\neg C$ project to the same point C_E and $+X$ and $\neg X$ project to the same point X_E.

3.3 Geometric Analysis

3.3.1 Discrepancies on Cheirality Definition

According to [34], [35], the camera projective transforms have the property of swapping points from the front to the back of the camera; thus, the camera matrix should be normalized by multiplying it by ±1 if necessary. Thus, there are two sign-reversed camera matrices $\pm P$ in the projective reconstruction, where the camera centres and reconstruction points corresponding to $\pm P$ are subsequently computed. The camera centre is a column vector $C = \begin{bmatrix} a & b & c & \gamma \end{bmatrix}^T$, defined by $PC = 0$; thus, the camera centres have two sign-reversed possibilities of $+C$ and $\neg C$:

$$+C = \begin{bmatrix} a & b & c & \gamma \end{bmatrix}^T, \neg C = \begin{bmatrix} -a & -b & -c & -\gamma \end{bmatrix}^T \tag{3.8}$$

The reconstructed point X is calculated using the linear triangulation method [160]. There are two possible positions for the reconstructed point:

$$+X = \begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T, \neg X = \begin{bmatrix} -A_x & -A_y & -A_z & -A_w \end{bmatrix}^T \tag{3.9}$$

Fig. 3.2 illustrates the 4D vector $C = \begin{bmatrix} a & b & c & \gamma \end{bmatrix}^T$ in 3D space, only the three axes of a, b, γ are presented for brevity, and Figs. 3.3 and 3.4 follow the same convention. Similarly, the three axes of A_x, A_y and A_w are presented for brevity to illustrate the 4D vector $X = \begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$. As shown in Fig. 3.2, the two sign-reversed camera centres correspond to the same point C_E in the Euclidean plane Π_{EC} ($\gamma = 1$ plane). $+X$ is projected by camera centre $+C$ while $\neg X$ is projected by camera centre $\neg C$; however, the two points $+X$ and $\neg X$ correspond to the same point X_E in Euclidean plane Π_{EX} ($A_w = 1$ plane). Two discrepancies thus arise with the definition of cheirality: Firstly, in Euclidean space, the point X_E is uniquely determined without camera front/back ambiguity. The depth of the 3D point is measured relative from its location in front of or behind the camera; the depth of $\pm X$ is same in Π_{EX}. Secondly, in homogeneous coordinates, these two points $\pm X$ are projected by two camera centres $\pm C$, respectively. However, if the cheirality of point $\pm X$ is defined as the swapping of points in front of and behind the camera, the cheirality must be defined with respect to the one camera centre.

This chapter proposes the definition of the cheirality of the point with respect to the camera should be its originial meaning **handedness**.

3.3.2 Camera Motion Between $+C$, $\neg C$ and C_0

To derive the geometric meaning of the two possible camera projections P and $-P$, the reference camera C_0 is assumed as known $C_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$. The camera

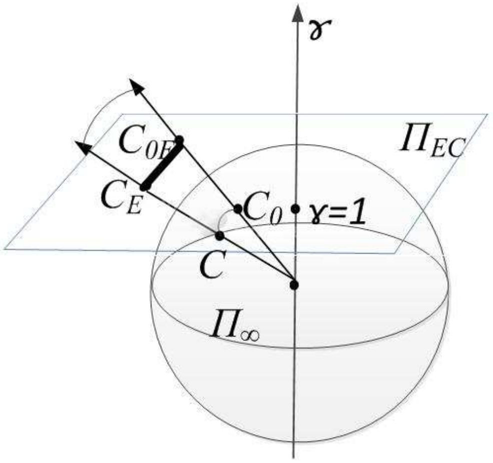


Figure 3.3: $C_{0E}C_E$ is an internal segment

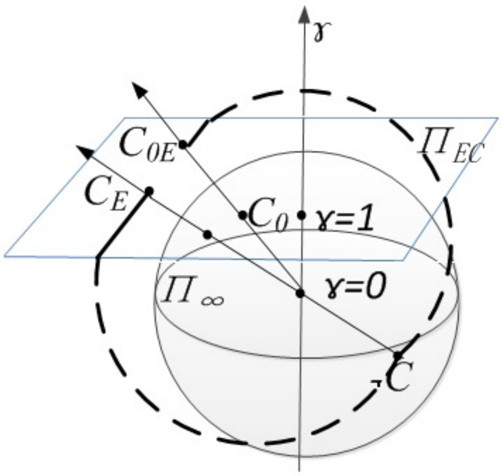


Figure 3.4: $C_{0E}C_E$ is an external segment

centre $+C = \begin{bmatrix} a & b & c & \gamma \end{bmatrix}^T$ is calculated from the projective matrix $+P$, and the camera centre $\neg C = \begin{bmatrix} -a & -b & -c & -\gamma \end{bmatrix}^T$ is calculated from the projective matrix $-P$. Fig. 3.3 shows the camera movement between C_0 and $+C$: C_0 is projected on the Euclidean plane (Π_{EC}) as C_{0E}, and the projection of $+C$ on the Euclidean plane is C_E. Since $+C$ has a positive value in the γ axis, $C_{0E}C_E$ is thus denoted as an internal segment [66], [161]. Fig. 3.4 shows the relationship between C_0 and $\neg C$: the projection of $\neg C$ on the Euclidean plane is C_E. Since $\neg C = \begin{bmatrix} -a & -b & -c & -\gamma \end{bmatrix}^T$ has a negative value of γ, the line segment passes through infinity $\gamma = 0$. That is, the projection line begins at C_{0E}, traverses away from C_{0E} to the inner, passes through infinity, and returns to meet C_E. The wrapped line segment $C_{0E}C_E$ in Fig. 3.4 is denoted as an external segment [66], [161]. Comparing the line segments $C_{0E}C_E$ for the two solutions of $+P$ and P in Fig. 3.3, the direction of $C_{0E}C_E$ is outer-pointing denoted as $(\overrightarrow{C_{0E}C_{Einternal}})$. In contrast, in Fig. 3.4 the direction of $C_{0E}C_E$ is inward-pointing passing through infinity to return to C_E, denoted as $(\overrightarrow{C_{0E}C_{Eexternal}})$. It can thus be seen that:

$$\overrightarrow{C_{0E}C_{Einternal}} + \overrightarrow{C_{0E}C_{Eexternal}} = 0 \tag{3.10}$$

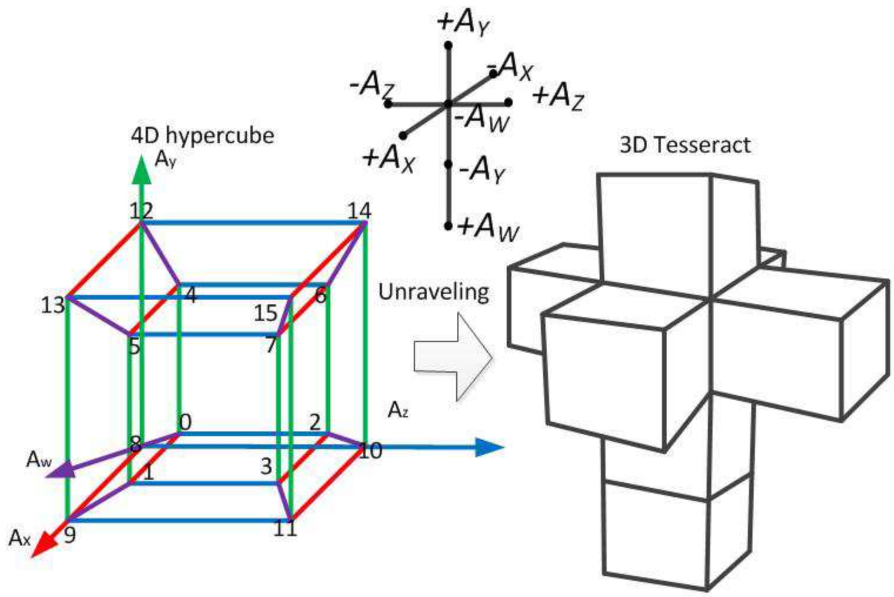


Figure 3.5: A 4D-hypercube can be unravelled to a 3D tesseract

Eq. (3.3) shows that in the solution of $\neg C$ the camera wraps around infinity to return from the opposite direction. When studying the movements of objects, the reference system must therefore be set up first: If the reference frame of the movement from C_{0E} to C_E has not been pre-defined, directly moving from C_{0E} to C_E or moving oppositely from C_{0E} to C_E after wrapping from infinity are both possible.

3.3.3 4D Geometric Analysis for the Cheirality of Points

Since the homogeneous representation of point $+X = \begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$ and $\neg X = \begin{bmatrix} -A_x & -A_y & -A_z & -A_w \end{bmatrix}^T$ is in four dimensions, this chapter presents a 4D geometric visualization of cheirality. According to [162], the $4th$ dimension can be treated as a temporal dimension. Thus, it is possible to visualize the 4D model simply by treating time as movement and examining 'snapshots' of the 4D model at various points in time. A 4D-hypercube [163] shown in Fig. 3.5 is used to represent 4D space: The red lines indicate the A_x axis, green lines indicate the A_y axis, blue lines indicate the A_z axis and purple lines indicate the A_w axis. The 4D hypercube

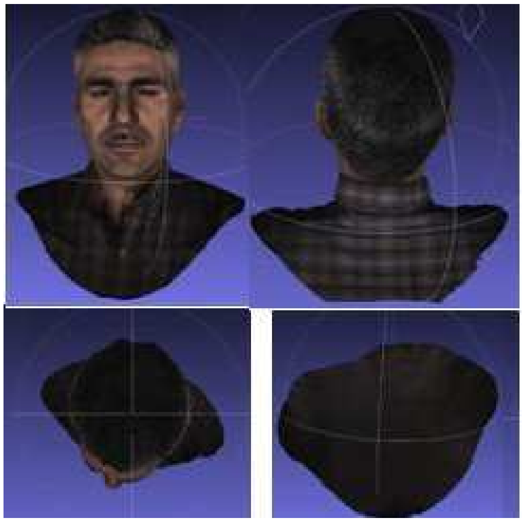
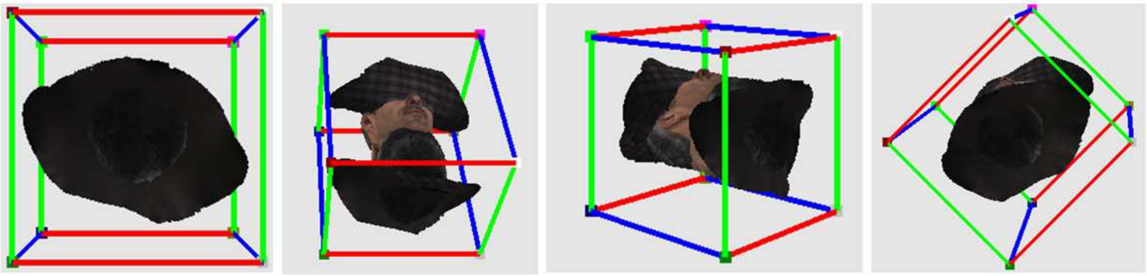


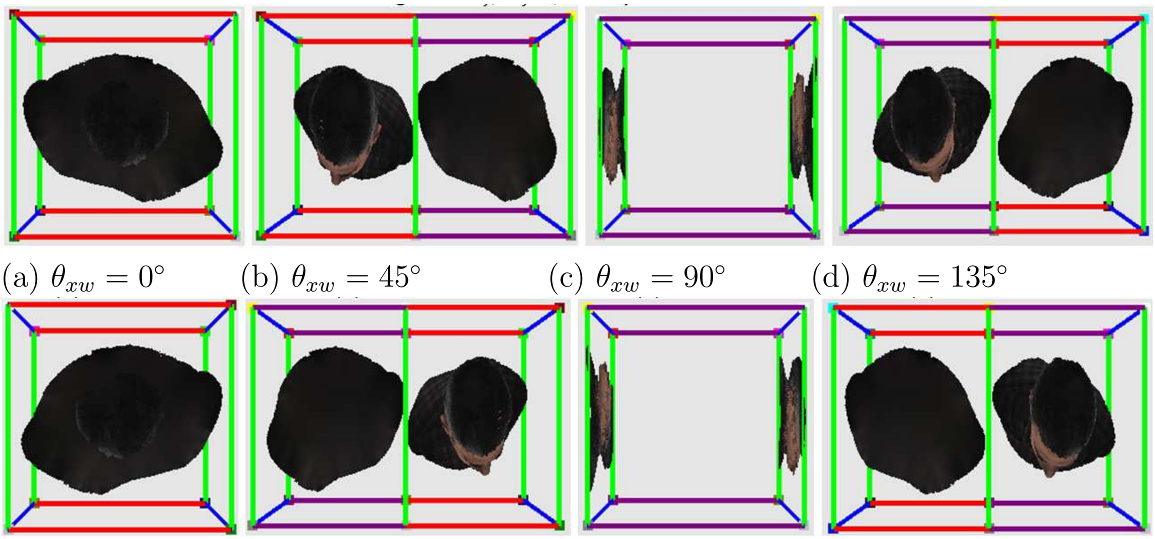
Figure 3.6: 3D head model



(a) $\theta_{xy}, \theta_{yz}, \theta_{xz} = 0°$ (b) $\theta_{xy} = 45°, \theta_{yz}, \theta_{xz} = 0°$ (c) $\theta_{yz} = 45°, \theta_{xy}, \theta_{xz} = 0°$ (d) $\theta_{xz} = 45°, \theta_{xy}, \theta_{yz} = 0°$

Figure 3.7: Plane rotations of A_xA_y, A_yA_z, A_xA_z

appears as a cube within cube, where the inner cube is deeper in the fourth direction (A_w axis) than the outer cube. A 4D hypercube can be unravelled into a 3D tesseract consisting of eight 3D cubes, as shown in Fig. 3.5. In Fig. 3.5, the middle 2D cross indicates the locations of the eight cubes in the tesseract. The four 3D cubes located in the positive directions of the A_x, A_y, A_z and A_w axes are denoted as positive cubes, and the four 3D cubes located at the negative directions of A_x, A_y, A_z and A_w axes are denoted as negative cubes; positive and negative cubes for each axis have opposite handedness [162].



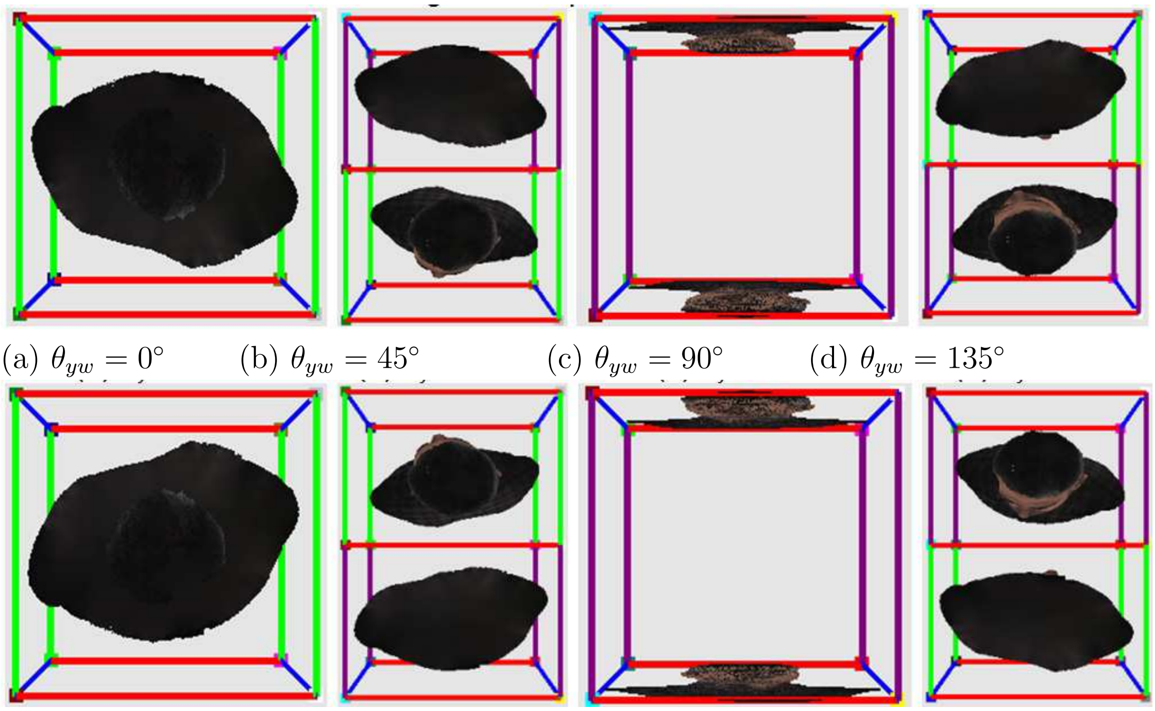
(a) $\theta_{xw} = 0°$ (b) $\theta_{xw} = 45°$ (c) $\theta_{xw} = 90°$ (d) $\theta_{xw} = 135°$

(e) $\theta_{xw} = 180°$ (f) $\theta_{xw} = 225°$ (g) $\theta_{xw} = 270°$ (h) $\theta_{xw} = 315°$

Figure 3.8: A_xA_w plane rotation

As an illustrative example, a 3D head model [1] shown in Fig. 3.6 is presented in four orientations (face, back, top and bottom), and the cheirality of points $+X_i$ and $\neg X_i$ are drawn onto the hypercube. In four dimensions with four axes A_x, A_y, A_z, A_w, six principal rotations are possible: $\theta_{xy}, \theta_{yz}, \theta_{xz}, \theta_{xw}, \theta_{yw}, \theta_{zw}$. First, let $\theta_{xw} = \theta_{yw} = \theta_{xw} = 0°$, the inner and outer cubes are overlapped and X_i and $\neg X_i$ are superimposed in two opposite directions, as shown in Fig. 3.7. Without the w-axis rotation, the 4D hypercube rotates in the A_xA_y, A_yA_z, A_xA_z planes which are analogous to the three principal rotations around A_x, A_y, A_z axes in 3D. For example, let $\theta_{yz} = \theta_{xz} = \theta_{xw} = \theta_{yw} = \theta_{xw} = 0°$, and only change θ_{xy}. As shown in Fig.3.7a the hypercube rotating in the A_xA_y plane is analogous to the rotation around the A_x axis in 3D. Fig.3.7b shows the rotation of $\theta_{xy} = 45°$. Similiarly, Fig.3.7c and Fig.3.7d show the rotation in the A_yA_z and A_xA_z planes with θ_{yz} and θ_{xz} both equal to 45°.

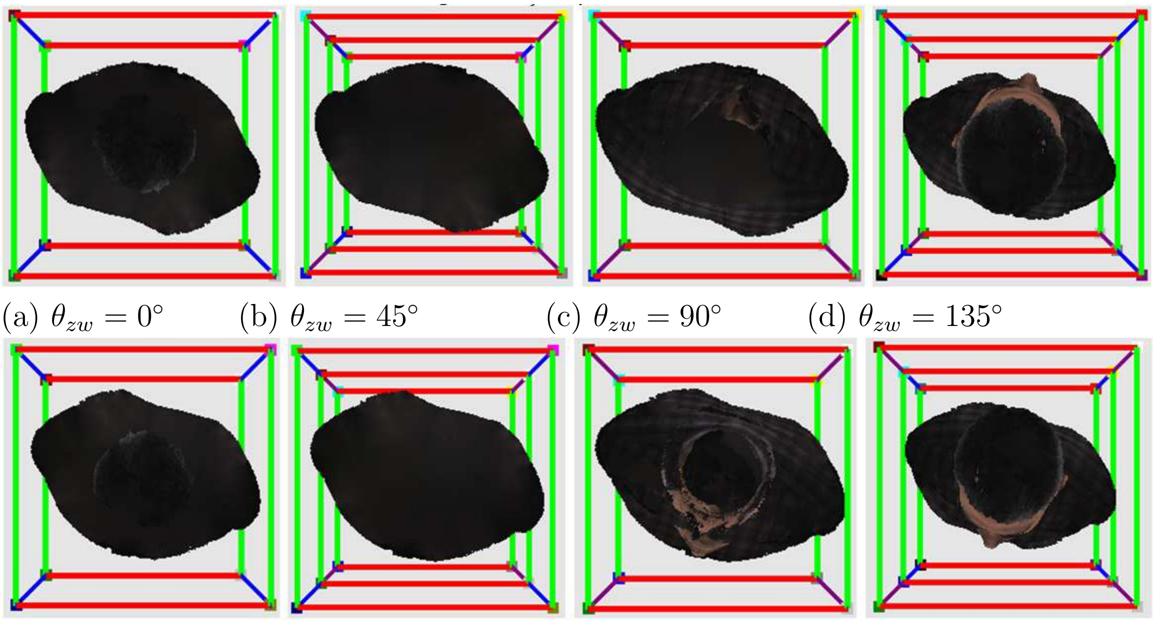
[1] http://viewshape.com/downloads/



(a) $\theta_{yw} = 0°$ (b) $\theta_{yw} = 45°$ (c) $\theta_{yw} = 90°$ (d) $\theta_{yw} = 135°$

(e) $\theta_{yw} = 180°$ (f) $\theta_{yw} = 225°$ (g) $\theta_{yw} = 270°$ (h) $\theta_{yw} = 315°$

Figure 3.9: A_yA_w plane rotation



(a) $\theta_{zw} = 0°$ (b) $\theta_{zw} = 45°$ (c) $\theta_{zw} = 90°$ (d) $\theta_{zw} = 135°$

(e) $\theta_{zw} = 180°$ (f) $\theta_{zw} = 225°$ (g) $\theta_{zw} = 270°$ (h) $\theta_{zw} = 315°$

Figure 3.10: A_zA_w plane rotation

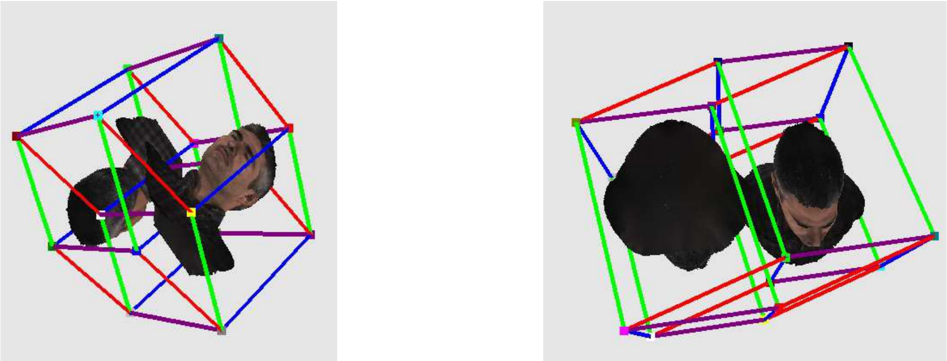


Figure 3.11: Arbitrary movements

Observing the movements about the A_w axis, the rotations about the A_xA_w, A_yA_w, A_zA_w planes are shown in Figs. 3.8 $\sim$ 3.10. In Fig. 3.8, the angles θ_{xy}, θ_{yz}, θ_{xz}, θ_{yw}, θ_{zw} are fixed to zero, and the 4D models are free to rotate around the A_xA_w plane. As shown in Fig. 3.8, the cheirality models have an appearance of 'turning inside-out' along the A_x axis (red axis). First, the two models overlap in the middle of the hypercube in Fig. 3.8a. In the rotation around the A_xA_w plane, the two head models separate and move along the $+A_x$ and $-A_x$ axes. When $\theta_{xw} = 90°$, the two models reach the leftmost and rightmost extremes of the hypercube, as shown in Fig. 3.8c. Then, the two models change movement orientation and return to the middle (Fig. 3.8d). When $\theta_{xw} = 180°$, the two models overlap in the middle, as shown in Fig. 3.8e. In continuous rotation, the two models move to the rightmost and leftmost extremes of the hypercube. At $\theta_{xw} = 270°$ (Fig. 3.8g), the two models change directions and again return to the middle. The movement repeats in an oscillatory nature along the A_x axis. When the 4D models rotate separately in the A_yA_w and A_zA_w planes, the models appear to oscillate along the A_y axis (green axis) and A_z axis (blue axis), as shown in Figs. 3.9 and 3.10. Similarly, at arbitrary rotation angles, the cheiral models $+X$ and $\neg X$ are a pair of reflections that move along two opposite axes: If one model rotates around the $+l$ axis, the other model rotates around $-l$ axis, as shown in Fig. 3.11.

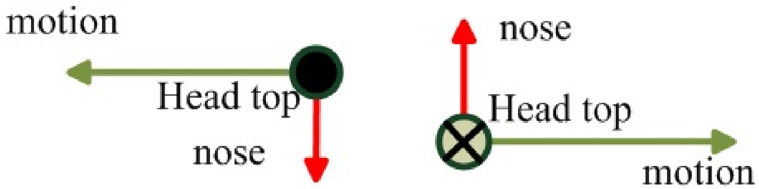


Figure 3.12: Handedness in Fig. 3.8b

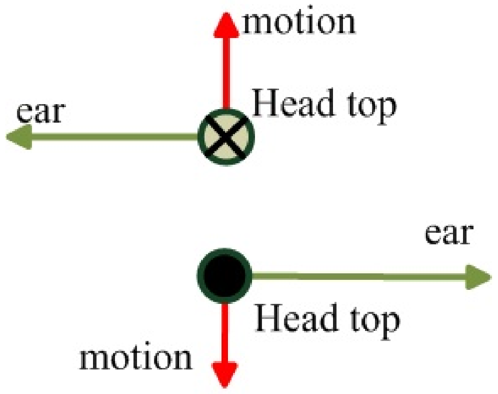


Figure 3.13: Handedness in Fig. 3.9b

This phenomenon is called handedness in geometry. Firstly, the head model $+X$ is not identical to its mirror reflection $\neg X$; that is, $+X$ cannot be superposed onto $\neg X$, as shown in Fig. 3.7 and any overlay snapshot in Fig. 3.8 $\sim$ Fig. 3.10. Secondly, when one head model movement follows the right-hand rule, the other sign-reversed model will follow left-hand rule. For example, Fig. 3.12 is the movement orientation diagram of Fig. 3.8b. From Fig. 3.8b, the left head model moves to the left, and the right head model moves to the right. In Fig. 3.12, the movement orientation is represented by the green arrow and the orientation of the nose is represented by the red arrow. In the left image of Fig. 3.12 (the left head model of Fig. 3.8b), the top of the head is outer perpendicular to the page. Thus, the green arrow, red arrow and the top of the head in the left image follow the right-hand rule. Similarly, in the right image of Fig. 3.12 (the right head model of Fig. 3.8b), the top of the head is inner perpendicular to the page, and the green arrow, red arrow and the top of head in the right image follow the left-hand rule. Such a relationship in handedness can be found in the snapshots of Fig. 3.8 $\sim$ Fig. 3.10, where Fig. 3.13 is the handedness representation of Fig. 3.9b. Cheirality is derived from the Greek word meaning *hand*. This chapter thus relates that the definition of cheirality of a point with respect to the camera back to its originial meaning of **handedness**.

3.4 Mathematical Derivation

3.4.1 Camera Motion Between Image Correspondence

In computer vision, the orientation of the camera projective matrix P, camera transformation matrix H and epipoles originate from the handedness in camera motion estimation. In the field of camera motion recovery, Hartley et al. [29] presented the widely accepted four Singular Value Decomposition (SVD) solutions of the essential matrix. However, Wang et al. [164] later reported eight solutions by adding an arbitrary sign to the rotation matrix. Wang et al. [29], [164] focused on just two projections of a scene and the four solutions only represent the static position and relative orientation information of the camera; the dynamic relative orientation of the camera motion in a continuous video sequence cannot be transformed. Although Wang et al. [164] introduced the possibility of eight essential matrix solutions, the relative geometrical meaning of the eight possible solutions and the application environments of these solutions were not explored.

In this chapter, eight SVD solutions of camera motion for the essential matrix is proposed and derived. Beyond the 'twist pair' of the rotation matrices R_1 and R_2 representing the relative orientation of two perspective views [29], there is also the reversed sign solution pair of $-R_1$ and $-R_2$. Thus, the camera projection matrix P

has eight possible solutions with two sign-reversed sets of projection matrices [59]:

$$P \sim [R_1|t], [R_1|-t], [R_2|t], [R_2|-t] \tag{3.11}$$

$$-P \sim [-R_1|t], [-R_1|-t], [-R_2|t], [-R_2|-t] \tag{3.12}$$

This chapter presents eight possible solutions derived from mathematical computation and geometrical analysis. The eight solutions not only reflect the position and orientation of the camera in static displacement but also the dynamic orientation between the camera and an object in continuous motion (multiple views). The positive and negative projective matrix form a pair of handedness projections due to the sign-reversed rotation matrices. A three geometric constraints is then proposed to determine the unique camera motion from the eight possible essential matrix solutions.

3.4.2 Traditional Four Possible Solutions of Camera Motion Estimation

The essential matrix E is decomposable if and only if one of its singular values is zero and the other two singular values are equal [165]. The Singular Value Decomposition (SVD) of E is:

$$E = UDV^T \tag{3.13}$$

where D is diagonal matrix with the form of:

$$D = \begin{bmatrix} d & 0 & 0 \\ 0 & d & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.14}$$

The non-zero singular values of D are the square roots of the non-zero eigenvalues of $E^T * E$ or $E * E^T$. U is the eigenvector of $E * E^T$ while V is the eigenvector of

$E^T * E$; U and V are called the left and the right singular vectors of E and are 3×3 orthogonal matrices. The four SVD solutions derived from two projections are given by [29]:

$$T_{\times} = UZU^T; \tag{3.15}$$

$$R_1 = UWV^T, R_2 = UW^TV^T \tag{3.16}$$

where $W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

In the camera motion solutions from the essential matrix, the translation vector t has two reversed sign values $\pm t$, corresponding the two reversed sign solutions of Z:

$$Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}; or \quad Z = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.17}$$

and the rotation matrix R has two different values R_1 and R_2. The camera projection matrix P can then be written as:

$$P \sim \begin{bmatrix} R_1 | t \end{bmatrix}, \begin{bmatrix} R_1 | -t \end{bmatrix}, \begin{bmatrix} R_2 | t \end{bmatrix}, \begin{bmatrix} R_2 | -t \end{bmatrix} \tag{3.18}$$

$$R_1 = U \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} V^T \tag{3.19}$$

$$R_2 = U \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} V^T \tag{3.20}$$

3.4.3 Eight Camera Motion Solutions from SVD of the Essential Matrix for Two Perspective Views

This section proposes and derives eight solutions of camera motion from two perspective views. Beyond the traditional four possible solutions, there is also the reversed sign solution pair of

$$P = \begin{bmatrix} -R_1|t \end{bmatrix}, \begin{bmatrix} -R_1|-t \end{bmatrix} \; or \; \begin{bmatrix} -R_2|t \end{bmatrix}, \begin{bmatrix} -R_2|-t \end{bmatrix} \tag{3.21}$$

The traditional four solutions are the positive subset $P^+ \sim \begin{bmatrix} R_1|\pm t \end{bmatrix}$ or $\begin{bmatrix} R_2|\pm t \end{bmatrix}$ of the eight possible solutions. There also exist four possible negative solutions $P^- \sim -\begin{bmatrix} R_1|\pm t \end{bmatrix}$ or $-\begin{bmatrix} R_2|\pm t \end{bmatrix}$, thus the eight possible solutions are $P = P^+ \cup P^-$. In the following, Sections 3.3.3 and 3.3.4 present the mathematical derivation and geometrical analysis of the proposed eight solutions, respectively.

3.4.4 Mathematical Derivation

Inspired by [12] and [16],[152], this thesis revisits the SVD of camera motion for two perspective views from the essential matrix to mathematically derive the proposed eight solutions.

Two Solutions of Translation Matrix $T_\times$ From [166], any skew-symmetric $n \times n$ matrix L $(n \geq 2)$ can be written as:

$$L = KJK^T \tag{3.22}$$

where K is the orthogonal matrix and J is the block diagonal matrix of the form:

$$J = \begin{bmatrix} J_1 & 0 & 0 & \dots & \dots & \dots & 0 \\ 0 & J_2 & 0 & \dots & \dots & \dots & 0 \\ & & \ddots & & & & \\ 0 & 0 & \dots & J_m & \dots & \dots & 0 \\ & & & & \ddots & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.23}$$

Each block J_i is a real two-dimensional matrix of the form:

$$J_i = \begin{bmatrix} 0 & -\theta_i \\ \theta_i & 0 \end{bmatrix} = \theta_i \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} (\theta_i > 0) \tag{3.24}$$

Since $T_\times$ is a 3×3 skew-symmetric matrix, $T_\times$ can be written as:

$$T_\times = KQK^T \tag{3.25}$$

where Q has the form:

$$Q = \begin{bmatrix} 0 & -\theta & 0 \\ \theta & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.26}$$

Following the properties of E in Eq. (3.6) and Eq. (3.7) , EE^T has the form:

$$EE^T = U \begin{bmatrix} d^2 & 0 & 0 \\ 0 & d^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T \tag{3.27}$$

Since R is an orthogonal matrix, EE^T can also be written as:

$$EE^T = T_\times T_\times^T = KQQ^TK^T = K \begin{bmatrix} \theta^2 & 0 & 0 \\ 0 & \theta^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} K^T \tag{3.28}$$

From Eq. (3.20) and Eq. (3.21), K is one of the singular vector matrices of E, and $\pm\theta$ are the eigenvalues of E. Thus, $T_\times$ can be written as:

$$T_\times = U \begin{bmatrix} 0 & \theta & 0 \\ -\theta & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T \tag{3.29}$$

Or

$$T_\times = U \begin{bmatrix} 0 & -\theta & 0 \\ \theta & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T \tag{3.30}$$

Dividing $T_\times$ by the common scalar θ, Eqs. (3.22) and (3.23) simplify to:

$$T_\times = U \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T \tag{3.31}$$

Or

$$T_\times = U \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U^T \tag{3.32}$$

Eqs. (3.24) and (3.25) are two reversed sign solutions for the translation matrix, and the two reversed sign solutions of Z in Eq. (3.10) have thus been proved.

Four Possible Solutions of Rotation Matrix R Substitute Eq. (3.8) into Eq. (3.6):

$$UDV^T = UZU^TR \tag{3.33}$$

Pre-multiplying U^T on both sides of Eq. (3.26):

$$R = UZ^{-1}DV^T \tag{3.34}$$

Since Z is singular, let:

$$B = Z^{-1}D, \tag{3.35}$$

Eq. (3.28) can be transformed into the function:

$$BZ - D = 0 \tag{3.36}$$

Using the least squares method [167], B is the solution that minimises the equation:

$$\Pi_{min} = min(||BZ - D||_f^2) \tag{3.37}$$

where $||BZ - D||_f^2$ denotes the Frobenius norm. B can be defined as:

$$B = \begin{bmatrix} b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 \\ b_7 & b_8 & b_9 \end{bmatrix} \tag{3.38}$$

Up to an unknown scale factor d, and with the two possible solutions of Z in Eq. (3.10), Eq. (3.30) can be extended to:

$$\Pi = (b_4 + 1)^2 + b_1^2 + b_5^2 + (b_2 - 1)^2 + b_6^2 + b_3^2 \tag{3.39}$$

$$\Pi = (b_4 - 1)^2 + b_1^2 + b_5^2 + (b_2 + 1)^2 + b_6^2 + b_3^2 \tag{3.40}$$

The minimum value of Π exists when $b_4 = -1$, $b_2 = 1$ or $b_4 = 1$, $b_2 = -1$. B can be written as:

$$B = \begin{bmatrix} b_1 & 1 & b_3 \\ -1 & b_5 & b_6 \\ b_7 & b_8 & b_9 \end{bmatrix} \tag{3.41}$$

$$B = \begin{bmatrix} b_1 & -1 & b_3 \\ 1 & b_5 & b_6 \\ b_7 & b_8 & b_9 \end{bmatrix} \tag{3.42}$$

Substituting B into Eq. (3.27):

$$R = UBV^T \tag{3.43}$$

Since R is an orthonormal matrix, $RR^T = 1$:

$$UBB^TU^T = I \tag{3.44}$$

Pre-multiplying U^T and post-multiplying U on both sides of Eq. (3.37):

$$BB^T = I \tag{3.45}$$

And,

$$b_1^2 + b_2^2 + b_3^2 = 1 \tag{3.46}$$

$$b_1b_4 + b_2b_5 + b_3b_6 = 0 \tag{3.47}$$

$$b_1b_7 + b_2b_8 + b_3b_9 = 0 \tag{3.48}$$

$$b_4^2 + b_5^2 + b_6^2 = 1 \tag{3.49}$$

$$b_4b_7 + b_5b_8 + b_6b_9 = 0 \tag{3.50}$$

$$b_7^2 + b_8^2 + b_9^2 = 1 \tag{3.51}$$

Substituting Eq. (3.34) into Eqs. (3.39) to (3.44):

$$B = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & \pm 1 \end{bmatrix} \tag{3.52}$$

Substituting Eq. (3.35) into Eqs. (3.39) to (3.44):

$$B = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \pm 1 \end{bmatrix} \tag{3.53}$$

Finally, the four solutions of R are achieved:

$$R = U \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & \pm 1 \end{bmatrix} V^T \tag{3.54}$$

or

$$R = U \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & \pm 1 \end{bmatrix} V^T \tag{3.55}$$

Since $T_\times$ or t has two solutions and R has four solutions, there are eight conbinations of camera motion. The proposed eight solutions of essential matrix have thus been proved.

3.5 Unique Solution From Eight Possible Camera Motion Solutions

3.5.1 Proper Rotation and Improper Rotation

In the eight possible solutions of camera motion estimation,the sign-reversed rotation matrices directly lead to two camera projective matrices P and -P. This section explains the mathematical relationship between R and -R. The rotation matrix R is an orthogonal matrix that satisfies:

$$RR^T = I \tag{3.56}$$

where I is the 3×3 identity matrix. Using the properties of the matrix determinant where $det(R^T) = det(R)$, it follows that:

$$(det(R))^2 = 1 \tag{3.57}$$

which implies that $det(R) = \pm 1$. The rotation with $det(R) = 1$ is known as proper rotation, and the rotation with $det(R) = -1$ is known as improper rotation [168]. Denoting λ as the eigenvalue, the characteristic polynomial of R is:

$$P_R(\lambda) = det(R - \lambda I) = -\lambda^3 + tr(R)\lambda^2 + \cdots + det(R) \tag{3.58}$$

where $tr(R)$ is the trace of R. Assuming that l is an eigenvector accompanying λ, $lR = \lambda l$:

$$(lR)(lR)^T = (l)(l)^T \tag{3.59}$$

And $RR^T = I$, thus:

$$\lambda^2 = 1 \tag{3.60}$$

If R is proper orthogonal, the polynomial Eq. (3.66) has at least one positive real eigenvalue $\lambda > 0$, such that $\lambda = 1$ and:

$$lR = l \tag{3.61}$$

Thus, l is the rotation axis of R. Assuming that θ is the rotation angle, the proper rotation is denoted as $R(l, \theta)$. Conventionally, the direction of the axis is determined by the right-hand rule [168]. Defining l in Cartesian coordinates $l = \begin{bmatrix} l_x & l_y & l_z \end{bmatrix}^T$, the proper rotation matrix can be written explicitly as a 3×3 matrix:

$$R(l,\theta) = \begin{bmatrix} cos\theta + l_x^2(1 - cos\theta) & l_xl_y(1 - cos\theta) - l_zsin\theta & l_xl_z(1 - cos\theta) + l_ysin\theta \\ l_xl_y(1 - cos\theta) + l_zsin\theta & cos\theta + l_y^2(1 - cos\theta) & l_yl_z(1 - cos\theta) - l_xsin\theta \\ l_xl_z(1 - cos\theta) - l_ysin\theta & l_yl_z(1 - cos\theta) + l_xsin\theta & cos\theta + l_z^2(1 - cos\theta) \end{bmatrix} \tag{3.62}$$

If the rotation angle is zero, the proper rotation matrix is:

$$R(l, 0) = I \tag{3.63}$$

An improper rotation matrix is an orthogonal matrix with $det(R) = -1$, denoted as $\bar{R}$. The characteristic polynomial Eq. (3.66) has at least one negative root, thus the eigenvalue $\lambda = -1$. Then:

$$l\bar{R} = -l \tag{3.64}$$

The improper rotation is a reflection of the proper rotation through a plane passing through the origin perpendicular to l, denoted as:

$$\bar{R}(l, \theta) = R(-l, \theta) \tag{3.65}$$

If the angle equals π, the improper matrix can be computed as:

$$\bar{R}(l, \pi) = -I \tag{3.66}$$

Comparing Eqs. (3.71) and (3.74), the relationship between proper and improper rotation can thus be written as:

$$R(l,0) = -\bar{R}(l,\pi) \tag{3.67}$$

Eq. (3.75) indicates that the improper rotation axis direction is a reflection through a plane that passes through the origin perpendicular to a proper rotation of 180° around the axis l. Conventionally, improper rotation is described by the left-hand rule.

3.5.2 Three Constraints for the Eight Possible Solutions

In mathematics, an orientation is the choice of an equivalent class of coordinate systems, where two coordinate systems belong to the same class (e.g., right-hand coordinate system). That is, when studying the movements of objects, the reference system must be set up first, and kept consistent during whole movements. In the eight solutions of camera motion from the essential matrix, if the reference frame of R is not defined, there will be two possible algebraic signs of the rotation axis l, and the cheirality problem can be resolved by defining the right-hand rule of rotation: $det(R) = 1$ selects the $+P$ solution set from the $+P \sim [R_1| \pm t]$ or $[R_2| \pm t]$ and $-P \sim -[R_1| \pm t]$ or $-[R_2| \pm t]$.

A three-constraint algorithm is proposed in this thesis to select the correct essential matrix solution from the proposed eight solutions. The core principles of the algorithm are to ensure that the 3D point visible within the image lies in front of the camera producing the image and that all the camera motions are recovered with the same handedness. First, the cheirality problem is resolved by defining the right-hand rule of rotation: $det(R) = 1$. Second, the points in front of or behind the camera are selected by positive depths in both images. Mathematically, if the point X lies in front of the two corresponding images with camera projection matrices P and P', X must have positive depth with respect to these images. The depth of a point in front of the principal plane of a camera is given by [30]:

$$depth(X, P) \approx \omega W det(M) \tag{3.68}$$

where M is the leftmost 3×3 block of P, m^{3T} is the third row of M, and $\omega = m^3 X$. The proposed three-constraint algorithm is summarized as follows:

1. Select right-hand rule of rotation:

$$det(R) = 1 \tag{3.69}$$

2. Points must lie in front of the first image:

$$sign(depth(X; P)) > 0 \tag{3.70}$$

3. Points must lie in front of the second image:

$$sign(depth(X; P')) > 0 \tag{3.71}$$

For a projective reconstruction, this cheirality problem exists in the DLT linear estimation of the projection matrix P and projective transformation H, and epipoles E.

3.5.3 Cheirality of the Camera Projection Matrix

The camera projective matrix can be obtained in two methods. Firstly, if the camera motion is correctly estimated, the camera projective matrix can be obtained from

$P = A \begin{bmatrix} R|t \end{bmatrix}$. Secondly, if the camera motion is unknown but the 3D point X is known, for example, in multiple-view reconstruction, the Direct Linear Transform (DLT) linear initialization is used to estimate the camera projection matrix. By applying the cross product of x to both sides of $x = PX$:

$$x \times PX = 0 \tag{3.72}$$

P can then be resolved by the Singular Value Decomposition (SVD) method. Due to the handedness of the cross product [169], P has possible two sign-reversed solutions $\pm P$. To resolve the orientation of camera projection in multiple views, the constraint $det(P) > 0$ should be applied during the linear initialization of P for each view. Since $P \sim [R|t]$, it can be proved that $det(P) > 0$ equals $det(R) = 1$. To resolve the orientation of camera projection in multiple views, the constraint $det(R) = 1$ should be applied during the linear initialization of P for each view. That is, after obtaining the DLT result of P, the rotation matrix R can be computed from QR factorization of P. Then, the projective matrix with a positive sign for $det(R)$ is selected. After all the views are thus checked, all the camera rotations are confined to the same right-hand rule, and the visibility problem of camera projection is thus resolved.

3.5.4 Cheirality of Projective Transformation

In computer vision, an invertible 4×4 matrix H is used to represent projective transformation. The corresponding point in the two views of the same point X can be written as:

$$X' = HX \tag{3.73}$$

The linear method of H can be described as:

$$X' \times HX = 0 \tag{3.74}$$

Similarly, in the DLT of projective transformation H, the angle between vectors X' and HX are also determined by the rotation matrix R and $\hat{n}$ is determined by the rotation axis l. Thus, to resolve the cheirality of the projective transformation, $det(R) = 1$ must be defined to ensure right-hand coordinates in the results from the DLT algorithm applied to H for each view.

3.5.5 Cheirality of Epipoles

The relationship between the fundamental matrix F and essential matrix E can be written as:

$$F = A^{-T} E A^{-1} \tag{3.75}$$

$$F = A^{-T} t_{\times} R A^{-1} \tag{3.76}$$

The epipolar points have the following relationship with the fundamental matrix:

$$Fe = 0; \qquad F^T e' = 0 \tag{3.77}$$

Then, the epipolar lines can be obtained from:

$$l = F^T x'; \qquad l' = Fx \tag{3.78}$$

If the camera motion is estimated correctly, the epipoles can be resolved from the SVD method and epipolar lines can also be estimated correctly.

3.6 Experiments

In this thesis, two experiments were conducted to validate the proposed eight possible camera motion solutions and handedness constraint. In Chapter 4, a camera motion simulator is presented to visualize the cheirality of the eight possible solutions of camera motion. In Chapter 5, a 3D reconstruction application is used to validate the $det(R) = 1$ constraint for continuous camera motion estimation from multiple views.

3.7 Summary

This chapter revisited the cheirality problem in computer vision from a camera motion viewpoint, proposing and showing the root cause of cheirality to be a handedness problem in camera motion estimation. Through theoretical derivation and 4D geometric proof using homogeneous coordinates, it was shown that the $det(R) = 1$ constraint is fundamental to resolving the cheirality problem, as the constraint confines all the rotations in multiple views to the right-handed reference frame.

Chapter 4

Camera Motion Simulator

Chapter 3 proposed techniques to uniquely determine the camera motion for each frame in an image or video sequence, and derived mathematically and geometrically. To dynamically visualize the camera motion in the world coordinate system, this chapter presents the design of an augmented reality camera motion simulator to validate the proposed camera motion estimation approaches and demonstrate the camera motion results frame-by-frame. A flexible, markerless registration method that addresses the problem of realistic virtual object placement at any position in a video sequence is proposed in this chapter.

4.1 Introduction

Augmented reality enhances the user′s perception of the real world by rendering virtual objects on top of an image sequence. Fundamental to creating a high quality augmented reality system is an accurate registration technique, where registration consists of virtual object rendering and camera motion tracking. Virtual object rendering includes initializing virtual object locations and precise alignment of the virtual object coordinate system and real user environment. Once the coordinate systems are aligned, virtual objects can be rendered dynamically and correctly according to the camera′s motion, position and orientation, which are continuously tracked throughout the image sequences.

There are two approaches to registration: marker-based and markerless. Marker-based methods track reference fiducial markers, a pre-defined geometrical pattern, to estimate the camera viewpoint and superimpose the virtual objects on the markers. Although fiducial marker methods work well in many applications [23] [22], these approaches are limited to relatively fixed environments i.e., the markers must be placed in advance in the user environment, and if the markers are partially occluded the virtual object cannot be accurately placed. In markerless registration methods, natural features in the video scene are tracked to estimate the camera intrinsic and extrinsic (camera motion) parameters. The camera self-calibration approaches of [26], [25] assume the camera intrinsic parameters to be known in advance; however, videos filmed with different cameras exhibit dissimilar intrinsic parameters, and even for the same camera, parameters such as focal length can change. Optical flow methods are also used to track natural features i.e., using a Kanade-Lucas-Tomasi (KLT) tracker that establishes corresponding features between consecutive video frames [27][170][171]. Since the KLT tracker heavily depends on the image illumination gradient, only distinct feature points can be tracked. Yuan et al. [27] proposed four feature point locations to register a virtual object; however, the KLT tracker cannot augment a virtual object onto an environment which lacks distinctive features e.g., a smooth tabletop. Ong et al. [25] proposed a four-point registration method to render virtual objects on a smooth surface by specifying an approximate square to calculate the projective matrix without computing the related fundamental matrix. However, the camera scenes must be fixed, as the translation and rotation information of the camera motion cannot be tracked.

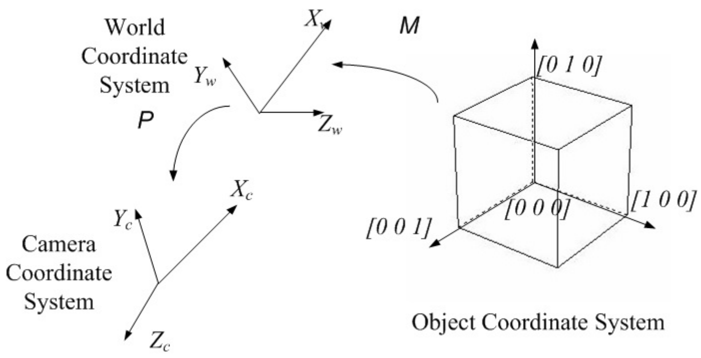


Figure 4.1: Transformation of the OpenGL Camera

This chapter proposes an augmented reality camera motion simulator to demonstrate and evaluate continuous camera motion. A novel, flexible, markerless registration method that remains effective in continuous camera movement is proposed. The rotation and translation relationship between virtual objects and the world coordinate system is computed by 3D reconstruction of four specified points. The camera position and orientation are estimated by the camera motion calibration algorithms proposed in Chapter 3, which automatically estimate the intrinsic and extrinsic parameters of the camera from an unknown video sequence. Distinct to the approaches based on affine object representation [26], the camera model is generalized into a perspective projection camera. Variation of the distance and angle between the object and user environment caused by the camera movement can be easily recovered by camera motion calibration. In the proposed markerless registration method, the registration of a virtual object on a scene is analogous to photography using a virtual camera (denoted here as the OpenGL camera). The OpenGL camera projective matrix conveys the rendering transformation, and camera position and orientation information; hence, virtual objects move according to the camera motion frame by frame.

4.2 Coordinate System Transformation

To establish 3D geometric relationships between the user environment and virtual objects, the key issue is the registration of three coordinate systems in the one frame of reference: the virtual object, the user environment and the camera orientation. As shown in Fig. 4.1, the object registration matrix M and camera projective matrix P of the real camera connect these three coordinate spaces. The virtual object has its own coordinate space, different from the user environment (hereafter denoted as object coordinates); the object registration matrix M transforms the object coordinates to the user environment in the world coordinate system. A real camera provides a mapping between a 3D world (user environment) and a 2D image. This is represented by a 3×4 matrix P which maps a 3D point in world coordinates to a 2D image point on the image plane (camera coordinates). P thus describes the camera's intrinsic parameters (focal length and principal point) as well as the extrinsic parameters (rotation and translation). Overall, the OpenGL camera projective matrix consists of both the object registration matrix M and the projective matrix of the real camera P.

The OpenGL camera projective matrix: A 3D space point is represented by a 4D vector in a world coordinate system as $X = \begin{bmatrix} A_x & A_y & A_z & A_w \end{bmatrix}^T$, and the projection of this to an image point $x = \begin{bmatrix} u & v & w \end{bmatrix}^T$ on an image plane is described by:

$$x = PX \tag{4.1}$$

where P is the camera projective matrix, and the k-th image matrix P_k can be decomposed such that:

$$P_k = A \begin{bmatrix} R_k | t_k \end{bmatrix} \tag{4.2}$$

where A is the 3×3 intrinsic matrix, which can be evaluated with the method proposed in Chapter 3. The translation t_k and rotation R_k in Eq. (4.2) are extrinsic parameters of the k-th image that transform the 3D displacement in a world coordinate system to the camera coordinate system. The registration transformation from the object coordinates to the world coordinate system can then be represented by a 4×4 matrix M, such that

$$M = \begin{bmatrix} R_m & t_m \\ 0 & 1 \end{bmatrix} \tag{4.3}$$

where R_m is the rotation matrix from object coordinates to the user world coordinates, and t_m is the translation of the origin of the object coordinates to the origin of the world coordinates. Hence, the overall OpenGL camera projective matrix O_k of the k-th video image is given by:

$$O_k = P_k M = A \begin{bmatrix} R_k | t_k \end{bmatrix} \begin{bmatrix} R_m & t_m \\ 0 & 1 \end{bmatrix} \tag{4.4}$$

4.3 Registration Method

4.3.1 Object Registration in the User Environment

The object registration is accomplished by first specifying the graphic world coordinate system into the control images denoted by I_1 and I_2. For example, Fig. 4.2 shows two control images, frame 0 and frame 140, taken from 192 video images. The model registration procedure consists of four steps:

- The world coordinates are first inserted into the image coordinates by specifying the four points $x_i(\begin{bmatrix} u_i & v_i & w_i \end{bmatrix}^T)(i = 1, 2, 3, 4)$, in the first control image I_1; an example is shown in the top image of Fig. 4.2.

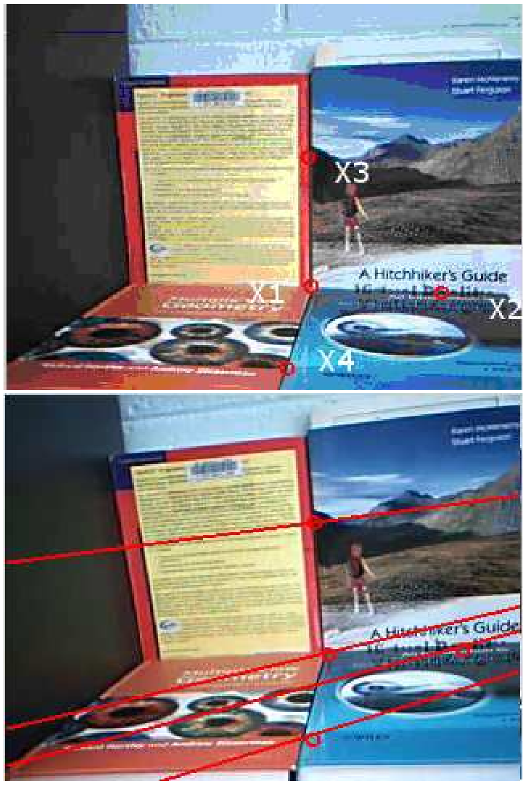


Figure 4.2: Four points in control image I_1 (the top image), and four corresponding points on the epipolar lines in I_2 (the bottom image).

- Compute the corresponding points ($\begin{bmatrix} u_i' & v_i' & w_i' \end{bmatrix}^T$) in the second control image I_2 using epipolar geometry constraints: To compute the corresponding points ($\begin{bmatrix} u_i' & v_i' & w_i' \end{bmatrix}^T$) in the second control image (I_2), homography is used to relate the pixel coordinates in the two images (I_1 and I_2) by:

  $$x' = Hx \tag{4.5}$$

  where H is the homography matrix estimated by the Direct Linear Transform (DLT) [30]; Fig. 4.2 (bottom image) shows ($x_i' = \begin{bmatrix} u_i' & v_i' & w_i' \end{bmatrix}^T$) located on the corresponding epipolar lines, given by $l_i = Fx_i$.

- Reconstruct the 3D points X_i of the specified points in the world coordinate system: The world coordinates of X_i $(i = 1, 2..4)$ are computed using linear triangulation methods [160] from the points pair $(x_i\ ,x'_i\ )$.

- Compute the registration matrix M with the rotation matrix R_m and translation vector t_m between the world coordinates and object coordinates: Without loss of generality, a unit cube is used as an example to indicate the object coordinate system. In the cube, M_i $(i = 1, 2..4) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$, $\begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}$, $\begin{bmatrix} 0 & 1 & 0 & 1 \end{bmatrix}$, $\begin{bmatrix} 0 & 0 & 1 & 1 \end{bmatrix}$, where the point M_1=$\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$ is the origin point. The remaining basic points indicate the XYZ directions, respectively. From the control image I_1 of Fig. 4.2, X_1 is the origin point of the user′s coordinates, $\vec{X}_{21}$ indicates the direction of X axis, while $\vec{X}_{31}$ and $\vec{X}_{41}$ indicate the directions of the Y and Z axes, respectively. Similarly, $\vec{M}_{21}$, $\vec{M}_{31}$, $\vec{M}_{41}$ indicate the directions of XYZ in the object coordinates. Thus, the transformation from the object coordinates to the user′s coordinate system can be computed using the following manipulation representing the rotation matrix R_m and translation vector t_m as:

$$R_m = \begin{bmatrix} \theta_x & \theta_y & \theta_z \end{bmatrix}, t_m = \begin{bmatrix} t_{mx} & t_{my} & t_{mz} \end{bmatrix} \tag{4.6}$$

where θ_x, θ_y, θ_z are the rotation angles around the X, Y, Z axes, respectively, and t_{mx}, t_{my}, t_{mz} are the translation values for each axis. The rotation angle of two vectors can be resolved as the inverse cosine of the dot product of the vectors, where the vectors in world coordinates $\vec{X}_{i1}(i = 2, 3, 4)$ are normalized. The translation vector is computed as the translation of the two origin points:

$$\theta_x = arccos(dot(\frac{\vec{X}_{21}}{\parallel \vec{X}_{21} \parallel}, \vec{M}_{21})) \tag{4.7}$$

$$\theta_y = arccos(dot(\frac{\vec{X}_{31}}{\parallel \vec{X}_{31} \parallel}, \vec{M}_{31})) \tag{4.8}$$

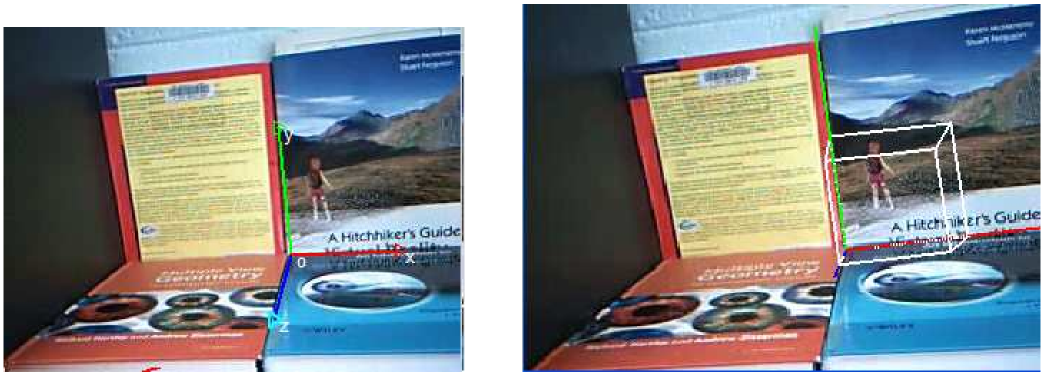


Figure 4.3: A unit cube registered to the user environment

$$\theta_z = arccos(dot(\frac{\vec{X}_{41}}{\| \vec{X}_{41} \|}, \vec{M}_{41})) \tag{4.9}$$

$$t_m = \frac{X_1 - M_1}{\| X_1 - M_1 \|} \tag{4.10}$$

where '$\|\|$' denotes the Euclidean norm. The sign of θ_x is then chosen such that the product $\theta_x \cdot t_{mx}$ is positive, and similarly, the signs of θ_y and θ_z are chosen for their respective products. The object registration matrix M can then be derived by substituting the rotation matrix R_m and the translation vector t_m into Eq. (4.6). The virtual coordinates in the left image of Fig. 4.3 show the object coordinate space transformed to world coordinates, and the virtual cube is then overlaid in the right image of Fig. 4.3.

4.3.2 Rendering with the OpenGL Camera Projective Matrix

As detailed in Section 4.2, the OpenGL camera projective matrix is generated such that the geometrical relationship between the virtual object and the environment can be represented by:

$$x_{ki} = O_k X_i = A \left[R_k | t_k \right] M X_i \tag{4.11}$$

For rendering, the parameters A and R_k , t_k and M are set as the view and modeling transformations using OpenGL [59]. This maps the virtual objects into the image locations, where the virtual object is then overlaid on the video images frame by frame.

4.4 Experimental Results

To evaluate the proposed markerless registration method, two experiments were conducted: to test the precision of the proposed camera motion calibration algorithm; and, to test the validity of the proposed virtual object rendering method.

4.4.1 Camera Motion Tracking Precision Test

The precision of the proposed self-calibration algorithm was tested for rotational motion by fixing the camera and protractor to a tripod. The rotation angle varied from $5°$ to $20°$ (in $5°$ intervals) around both the X and Y axes. The video sequence used a frame size of 320×240, a frame rate of 25 frames/s and approximately 80 feature points were extracted for each frame.

$10°$ **rotation test for the camera self-calibration algorithm**

Fig. 4.4 shows the results for a $10°$ rotation around the X axis, where R_x, R_y, R_z are the radial angles of rotation. The camera moves from $0°$ to $10°$ around the X axis, and then held at $10°$ for more than 5s to ensure camera stability. R_x starts at the radial angle 0.000 in radians (approximately $0°$), then with the camera moving the R_x value increases to a mean value around 0.1619 rad ($9.282°$) for the last 50 frames. The mean values of R_y and R_z are 0.0028 rad and 0.0005 rad; since the camera only rotates around X axis, R_y, R_z remain approximately constant throughout. As can

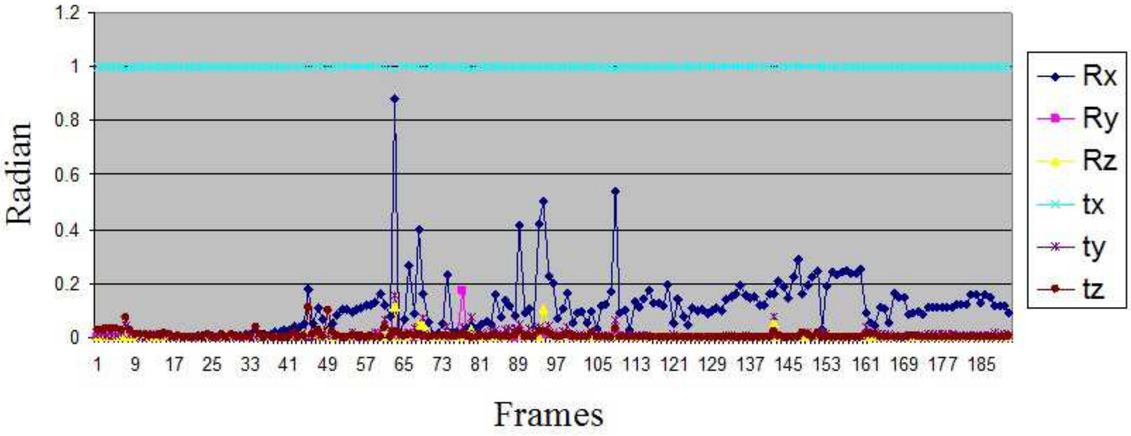


Figure 4.4: Results of 10° rotation around X axis

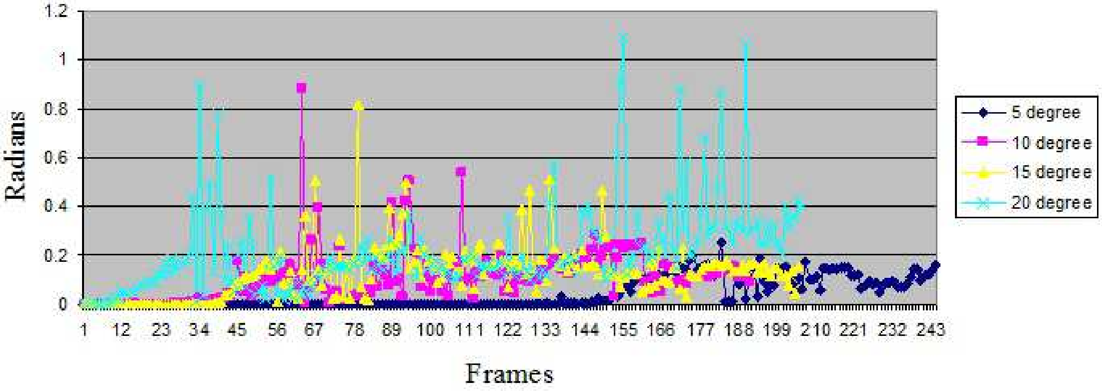


Figure 4.5: R_x value of the rotation around X axis

Table 4.1: R_x error of the rotation around X axis

Rotation Angle	5°	10°	15°	20°
Ideal (rad)	0.0872	0.1744	0.2617	0.3491
Evaluated (rad)	0.0967	0.1619	0.2231	0.2251
Error	0.1086	0.07165	0.1475	0.3551

Table 4.2: R_y value of the rotation around Y axis

Rotation Angle	5°	10°	15°	20°
Ideal (rad)	0.0872	0.1744	0.2617	0.3491
Evaluated (rad)	0.0892	0.1811	0.2335	0.171
Error	0.0229	0.0384	0.1077	0.5104

be observed from Fig. 4.4, t_x, t_y, t_z also remain approximately constant. Since the test is a pure rotation and points in the image plane are perpendicular to the axis of rotation, the ideal value of the translation vector would be $\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$. In practice, the mean values of t_x ,t_y ,t_z are 0.9997, 0.0113 and 0.0068 in unit directions, respectively, with average measurement error under 0.01.

Further Rotation Tests for the Camera Tracking Algorithms

Fig. 4.5 and Tabs 4.1 and 4.2 show results for a broader set of rotational experiments which followed the same experimental process as for the rotational test. Tab. 4.1 shows the results of R_x for $5°, 10°, 15°, 20°$ pure rotation around the X axis whilst Tab. 4.2 shows results of R_x in $5°, 10°, 15°, 20°$ pure rotation around the Y axis. While the ideal and 'evaluated' rotations are given in radians, the error is computed as the (ideal-evaluated)/ideal following [172]. Luong et al. [172] tested an unknown angle for three static images with two methods and generated an error around 0.15. From Tabs 4.1 and 4.2, the proposed algorithm generates comparable results for 5°, 10° and 15° rotation angles. From Fig. 4.5, the values of R_x at a 20° rotation angle vary more widely than for $5°, 10°, 15°$ rotation angles. Further, the values of R_x at a

20° rotation still fluctuate when the camera is held steady while the remaining three curves enter stability during the last 5 seconds of the sequence.

Analysis and Discussion

There are three possible issues affecting the precision of measurements from the proposed camera motion calibration framework:

- When the rotation angle increases, matching feature points are lost from consecutive frames. For example, for the 20° rotation experiment, the corner feature points do not appear over all images; the missing correspondences then affect the accuracy of motion recovery. One solution to this problem would be to cut the long video sequence into sub-sequences, and then compute the 3D structure points, X_i, and camera motions for each sub-sequence individually for all views before registering these sub-sequences back into the long sequence, this approach is discussed as future work in Chapter 7.

- As with most non-linear optimization algorithms, the Levenberg-Marquardt algorithm only converges if the initial value is close to the true solution. Further, since Kruppa's equations are non-linear, the initial value of the intrinsic parameter matrix A cannot be directly obtained. Some authors assume that the initial value of A is known or pre-calibrated [32], but A is necessarily variable since the focal length generally changes when shooting different scenes. This is a further source of possible calibration error. To address this issue, Kruppa's equations are firstly transformed into two quadratic equations in two variables [33]: This allows flexible resolution of the initial value A for the same image sequence such that non-linear refinement can be performed.

- Since modern cameras are manufactured very accurately [30], the camera model is assumed to be linear with central projection and radial distortion of the

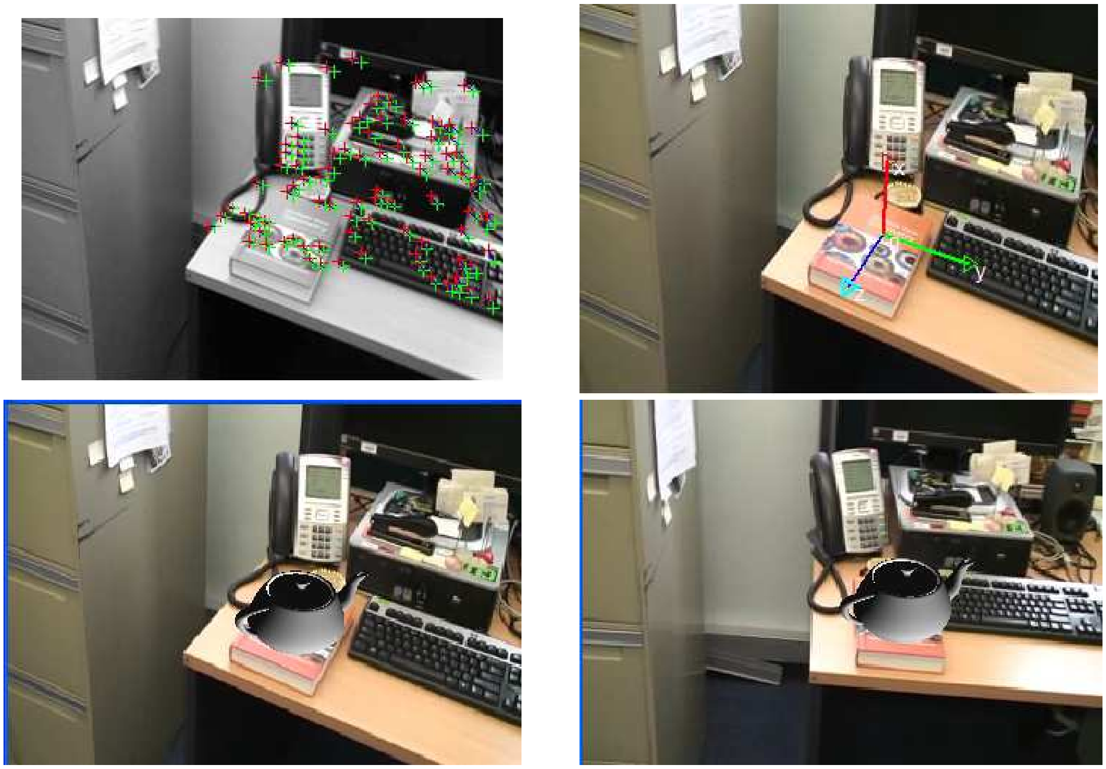


Figure 4.6: Indoor environment

camera lens ignored. In reality, the image magnification increases/decreases with distance from the optical axis and this will lead to calibration errors.

4.4.2 Different Environments for Virtual Object Rendering

Indoor and Outdoor Environment Rendering The proposed markerless registration method for video augmented reality was tested using two further representative video sequences taken in realistic indoor and outdoor environments. Fig. 4.6 shows a virtual teapot rendered on a book in an indoor office environment at a distance of 1m from the camera. Two control images were first selected with different camera positions. The four points were specified as shown in Fig. 4.6a, and the corresponding points were estimated by the homography matrices for each frame. Then, the rotation and translation relationship between the virtual object coordinates and world coordinates was setup in the scene, as shown in Fig. 4.6b. The proposed cam-

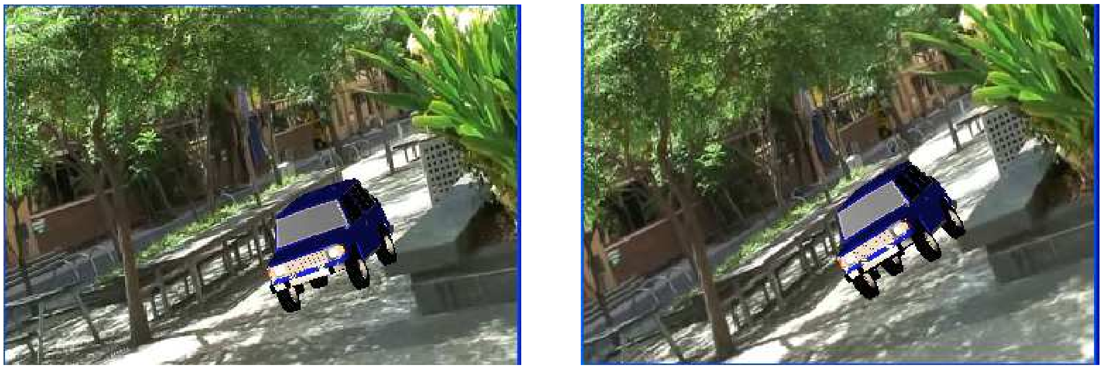


Figure 4.7: Outdoor environment

era motion calibration algorithm can be seen to accurately track the camera motion, where the virtual teapot coordinates are registered into the scene following Eq. (4.12). In Figs. 4.6c and 4.6d, the teapot orientation is consistent with the camera viewpoint. In the outdoor experiment shown in Fig. 4.7, the extraction of natural features is more complex than an indoor environment due to varying lighting conditions. There are more mismatched points (outliers) in outdoor compared to indoor environments e.g., in the leaves of Fig. 4.7. The robust estimation algorithm RANSAC and the nonlinear optimization minimize the effects of residual errors in the scene, where Fig. 4.7 shows a virtual car rendered following camera motion in an outdoor environment at 6m.

Virtual Object Placement on Non-Feature Points In this experiment, the four points are specified on non-feature points. As shown in Fig. 4.8a, all the four points are non-feature points: Three points are located on the table and one point on the back of a book. The correspondences of the specified points can be solved by Eq. (4.6). Fig. 4.8b shows the alignment of the virtual coordinates and the user environment, and Figs. 4.8c and 4.8d show the virtual cube rendered onto the smooth table consistent with the camera motion.

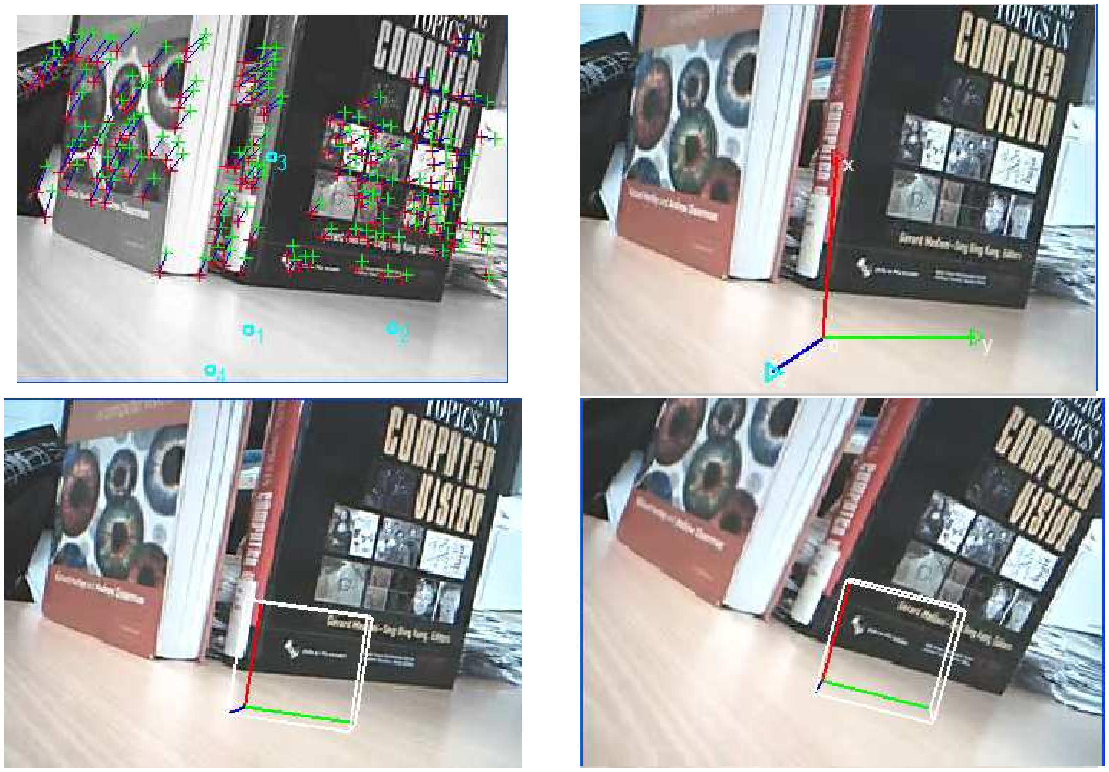


Figure 4.8: Placing the virtual object at non-feature points

4.4.3 Camera Motion Simulation for Eight Possible Camera Motion Solutions

A camera motion simulator was developed to demonstrate the eight possible camera motion solutions, as presented in chapter 3. Motivated by [172], the orientation of a moving camera is computed from the essential matrices obtained from the point correspondences between images. Assuming that the projection matrix of image 1 is the reference frame with $P_0 = A\begin{bmatrix} I|0 \end{bmatrix}$, the natural feature points are extracted from every image using SIFT [81]. To find corresponding points, feature points are then matched between images using a normalized correlation algorithm. To remove the effect of mismatched points (outliers), the robust estimation algorithm RANSAC [11] is then employed to result in a refined set of essential feature points.

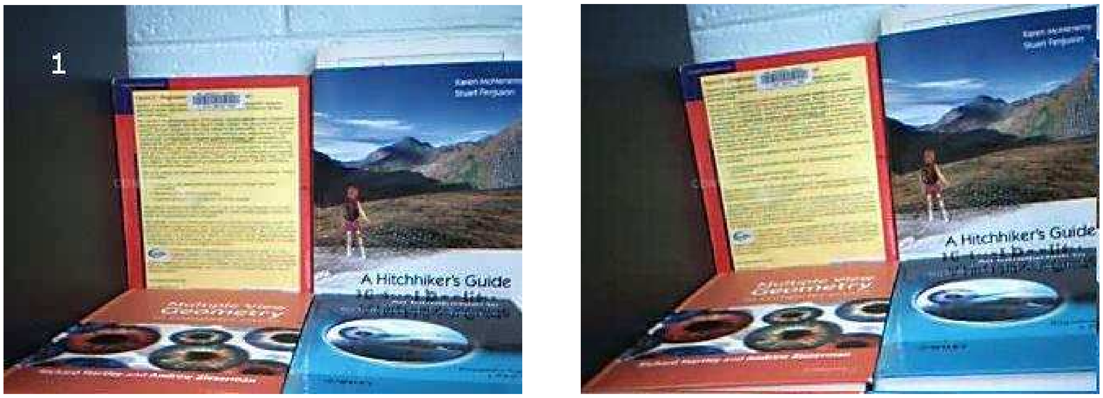


Figure 4.9: The two perspective views with different view angles

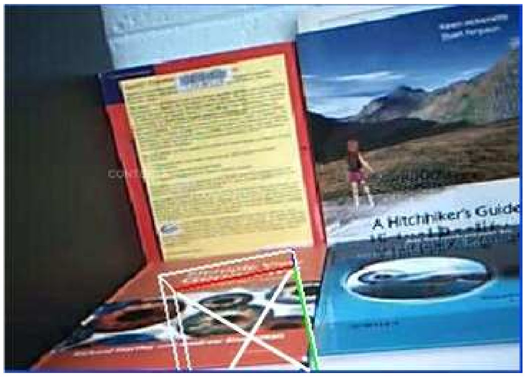


Figure 4.10: (R_1, t) camera position

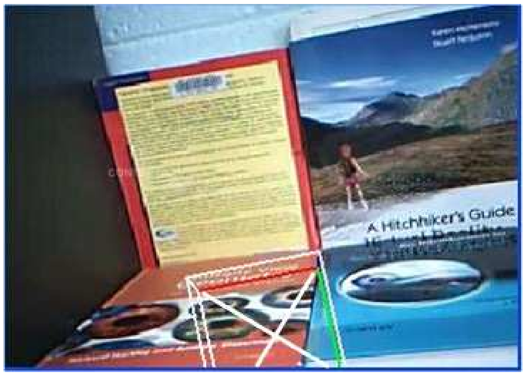


Figure 4.11: $(R_1, -t)$ camera position

The computation of the camera projective matrix P is based on a self-calibration framework: The fundamental matrix is first computed using the linear normalized eight-point method and then optimized using the Gold Standard method [30]; the intrinsic parameters A are assumed as known in this chapter. In turn, the camera motion (rotation and translation) can then be recovered from the essential matrix with SVD decomposition. P is then obtained by:

$$P = A \begin{bmatrix} R|t \end{bmatrix} \tag{4.12}$$

Fig. 4.9 shows two perspective views with different view angles. The eight absolute orientations are shown in Figs. 4.10 $\sim$ 4.17, where the cube indicates camera motion and the three coloured lines represent the camera coordinates. The solid line

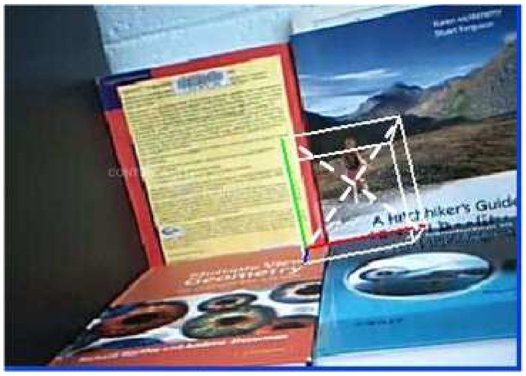


Figure 4.12: $(-R_1, t)$ camera position

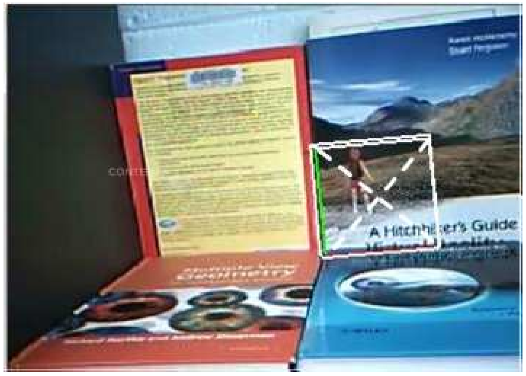


Figure 4.13: $(-R_1, -t)$ camera position

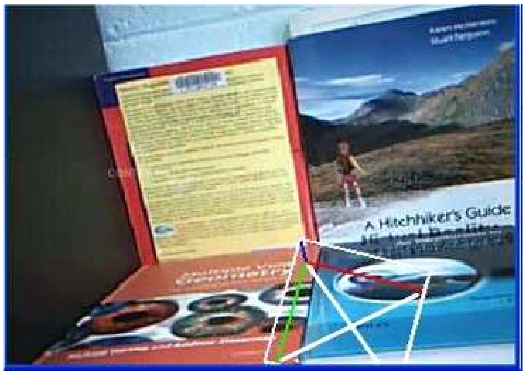


Figure 4.14: (R_2, t) camera position

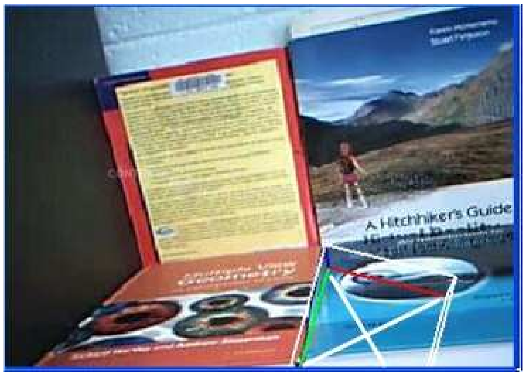


Figure 4.15: $(R_2, -t)$ camera position

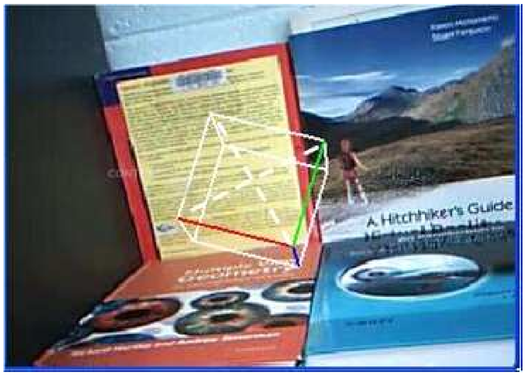


Figure 4.16: $(-R_2, t)$ camera position

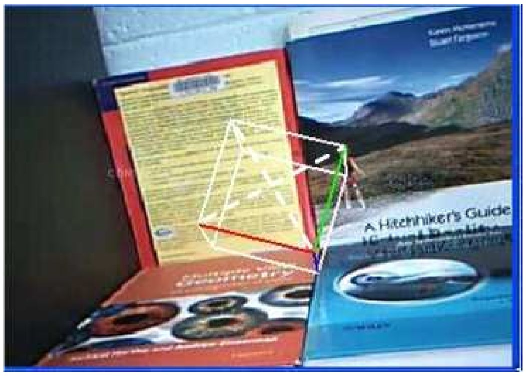


Figure 4.17: $(-R_2, -t)$ camera position

cross indicates that the face is outside of the cube, and the dashed line cross indicates that the face is inside the cube. The red line denotes the A_x axis, and the green and blue lines denote the A_y and A_z axes, respectively. In the eight possible solutions, only one solution is correct. In this experiment, the correct position is $(-R_1, -t)$ of Fig. 4.13. Fig. 4.12 is the translation-reversed position of Fig. 4.11. In Fig. 4.12, the sign of the rotation matrix is reversed, and the principal axes of the camera are also reversed. Fig. 4.13 shows the position of $(-R_1, -t)$, where the rotation and translation are both reversed compared to Fig. 4.10. Similarly, Figs. 4.14 $\sim$ 4.17 are the four positions corresponding to R_2. The experimental results are thus consistent with the handedness problem discussed in Chapter 3: First, the P^- set of four sign reversed cubes do not superpose with the P^+ set of cubes. For example, the camera orientations of $(-R_1, t)$ and $(-R_1, -t)$ differ to $(R_1, -t)$ and (R_1, t), as shown in Figs. 4.10$\sim$ 4.13. Second, from the sign-reversed solution pair, it is easy to find the handedness of the coordinate system of the camera. For example, in Fig. 4.13 the red, green and blue three axes are right-handed in solution $(-R_1, -t)$, but in solution (R_1, t) illustrated in Fig. 4.10, these three axes follow the left-hand rule.

4.5 Summary

This chapter described a framework for the generation of video augmented reality using a virtual OpenGL camera that is defined by real camera projective and object registration matrices. The proposed self-calibration algorithm is improved by the combination of recursive refinement and epipolar constraints ensuring calibration accuracy with average mean errors of less than 0.14 for 5°, 10°, and 15° pure rotation experiments. The framework has also been demonstrated to work acceptably in a number of different user environments, both indoor and outdoor. Future work, beyond the scope of this thesis, includes self-calibration of lens distortion parameters, and automatic motion recovery from long image sequences.

Part II

Dense 3D Reconstruction from Multiple Images

Chapter 5

Dense 3D Reconstruction

5.1 Introduction

Chapters 3 and 4 in Part I proposed techniques for camera motion estimation from multiple views. This chapter builds upon this work to integrate the camera motion and 3D surface reconstruction into a SfM framework to propose a flexible, high quality dense reconstruction method. In recent years, multi-view 3D reconstruction of rigid scenes has made significant progress, and applications have been developed e.g., the reconstruction of objects from image or video sequences, image-based modelling from large photo collections, 3D shape recognition and 3D obstacle detection for mobile robotics etc. Generally, SfM-based 3D reconstruction techniques [118],[4],[47], [120], [121] firstly extract and match a set of feature points and then reconstruct the surface with geometric, photometric or visualization constraints, where there are two classes of surface reconstruction approaches: sparse [4], [120], [121] and dense [118], [47],[173],[174].

The Bundler sparse approach [4], [120] orients the camera from thousands of unstructured photographs and deforms a sparse 3D reconstruction of the scene.

While the sparse approach is sufficient for calibrating camera motion, it is insufficient for reconstruction of a scene since only sparsely distributed feature points are represented. Hence, for scene reconstruction, a sparse approach is generally only used for calibration purposes to initiate a dense approach e.g., Patch-based Multi-View Stereo (PMVS) [118]. PMVS propagates a dense set of small patches covering surfaces based on the calibrated information of Bundler [4], [120]. However, although the PMVS approach can reconstruct the surface of the scene accurately and completely, duplicated procedures primarily in the pre-calibration procedure are computationally costly due to key point detection and matching, and removal of bad matches. Further, the dense approach must be combined with calibration software, which limits the applicability and flexibility of the dense approach. For example, the EXIF tags of images are needed to initialize the focal length for Bundler, thus video image sequences that lack these tags cannot be directly used with PMVS.

After initialization using a SfM approach, the dense reconstruction region growing is based on merging the computed depth maps [175], [5], [4]. Hiep et al. [175] proposed a dense reconstruction pipeline Dense Tracking and Mapping (DTAM) for efficiently handling large scenes. DTAM [175] uses the hundreds of images available in a video stream to improve the quality of a simple photometric data term, and minimise a global spatially regularised energy function in a novel non-convex optimisation framework. First, a point cloud is created with millions of points, converted to visibility consistent triangle mesh. Then, a variational method refines the photo consistency of the mesh. The multi-view stereo for community photo collections proposed by Goesele et al. [4] computes depth maps from internet photo collections, whilst Newcombe et al. [5] estimate detailed textured depth maps at selected key frames to produce a surface patchwork with millions of vertices. However, these depth map-based approaches [175], [5], [4] share problems known to depth map fusion: the holes due to the occlusions on the individual depth-maps which may impact the subsequent multi-view stereo global optimisation.

To overcome these disadvantages of existing sparse and dense algorithms, this chapter proposes a one-stop solution for dense surface reconstruction from uncalibrated videos. The proposed approach develops a complete, automatic and flexible system with a simple user-interface of 'raw images to 3D surface representation'. In contrast to approaches based on depth map fusion, the proposed system obtains a fully consistent 3D object reconstruction without holes in the surface. The standard procedure of dense approaches consists of four steps:

1. Detection of sparse feature points;
2. Calibration of the camera orientations image-by-image for the whole image sequence;
3. Expansion of the dense points;
4. Reconstruction of the surface with photo and visualization-consistent constraints for each image pair.

However, the steps for camera calibration and surface reconstruction of multiple images are the most computationally complex. Motivated by SfM approaches [69], [120], [102], this chapter proposes an iterative image-by-image procedure for dense surface reconstruction, alternating between camera self-calibration for good initial value camera parameters/3D points and bundle adjustment optimization. However, existing SfM techniques extract a set of sparse feature points from each image, and deform the sparse 3D shape obtained from photometric stereo. To resolve this surface insufficiency in existing SfM approaches, as part of the proposed image reconstruction approach, this chapter introduces an accurate and reliable region-growing algorithm to propagate the dense matching points from sparse key points among all stereo pairs.

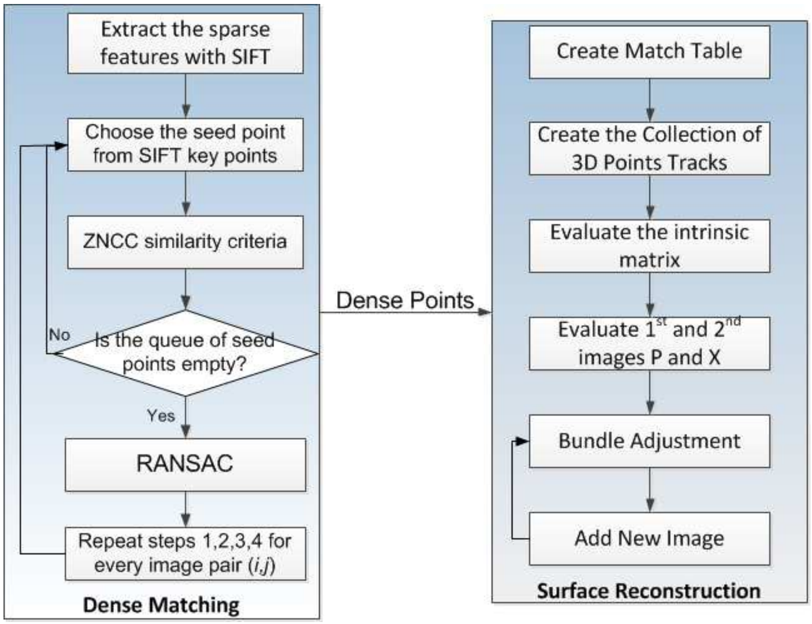

Extract the sparse features with SIFT
Choose the seed point from SIFT key points
ZNCC similarity criteria
Is the queue of seed points empty?
No
Yes
RANSAC
Repeat steps 1,2,3,4 for every image pair (i,j)
Dense Matching
Dense Points
Create Match Table
Create the Collection of 3D Points Tracks
Evaluate the intrinsic matrix
Evaluate 1st and 2nd images P and X
Bundle Adjustment
Add New Image
Surface Reconstruction

Figure 5.1: Dense 3D Reconstruction Architecture

5.2 Proposed System Architecture

As shown in Fig. 5.1, this chapter proposes a dense surface reconstruction approach from multiple uncalibrated images, comprised of two stages:

- Stage 1 Dense matching: The Scale Invariant Feature Transform (SIFT) [81] is firstly used to detect the key points from images, where this work utilised the SIFTGPU package [176]. To create dense matching points from SIFT key points, this chapter proposes an accurate two-window region growing algorithm based on the Zero-mean Normalized Cross-Correlation (ZNCC) similarity metric [47]. The RANdom SAmple Consensus (RANSAC) method [11] is then applied to remove outliers from corresponding points.

- Stage 2 Surface reconstruction from multiple views: The camera intrinsic parameters are assumed constant during the entire video sequence and evaluated with Kruppa's equations from fundamental matrices. The camera projective matrix and 3D points are then self-calibrated for each image, and Bundle Adjustment [132], [131] is employed to minimize the re-projection error of the 3D points and camera projective matrices.

5.3 Dense Matching

Lhuillier et al. [47] proposed a dense pixel matching approach by simultaneously expanding the initialized sparse matching to immediate neighbouring areas in two images. Avoiding mismatches at the small noise points or nearly repetitive patterns, Tang et al. [48] proposed a two window matching procedure: a larger window is used to contain enough intensity variation to achieve reliable matching, while a smaller window is used to obtain more accurate matches. The approach proposed in this chapter combines these two algorithms: First, a standard sparse matching algorithm based on SIFT features [81] is used to detect the points of interest for each stereo image pair; second, a novel window-based algorithm is proposed. Rather than performing feature matching, window-based region growing methods compare intensity similarity of neighboring pixels within a window between images to determine whether the centre points of the windows are a pair of corresponding points. The proposed approach is based on an assumption that surfaces of objects are smooth; that is, disparity varies continuously. In this chapter, the Zero-mean Normalized Cross-Correlation (ZNCC) score of the large window is calculated, moving the large window pixel-by-pixel in the area of the smaller window to search for the best match point.

5.3.1 Region Growing

The proposed region growing algorithm exploits two facts: First, the pixels that are matched exhibit similar intensities; second, if two points (x, x') are matched in an

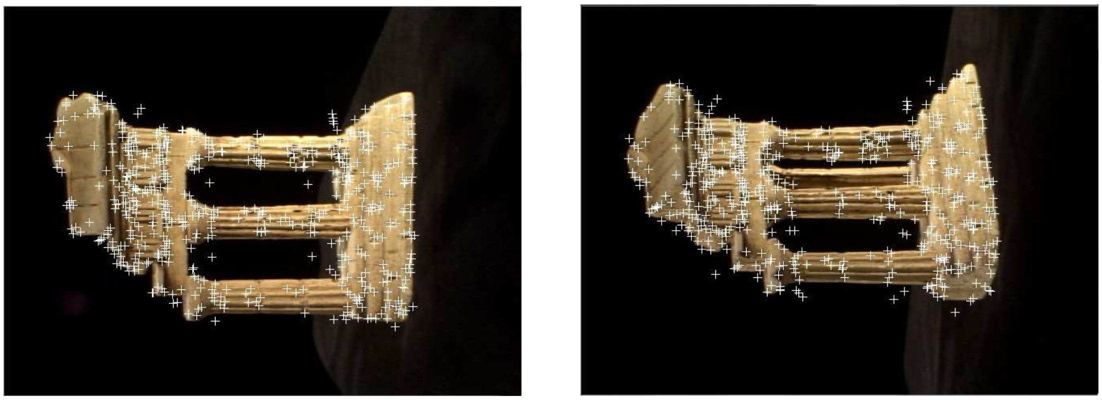


Figure 5.2: Sparse key point matches on an image pair

image pair, the matching points of the neighbours of x must be located close to x'. Further, the proposed use of two windows increases the matching reliability and accuracy: a larger area (correlation window) is used to guarantee a reliable result for the positions of the corresponding points and avoid errors of noise and repetitive patterns. Conversely, a smaller area (neighbor window) is used to accurately localize the position of the corresponding point. The proposed region growing dense matching algorithm thus consists of five steps for each image pair:

- Step 1 - Choose the seed point from SIFT key points.

- Step 2 - Check the neighboring pixels and add these pixels to the region if they conform to the ZNCC similarity criteria.

- Step 3 - Repeat step 2 for each of the seed points; stop if no more seed points can be found.

- Step 4 - To remove the effect of mismatched points (outliers), the robust estimation algorithm RANSAC [11] is employed to result in a refined set of essential feature points.

- Step 5 - Repeat steps 1 to 4 for every image pair (i, j).

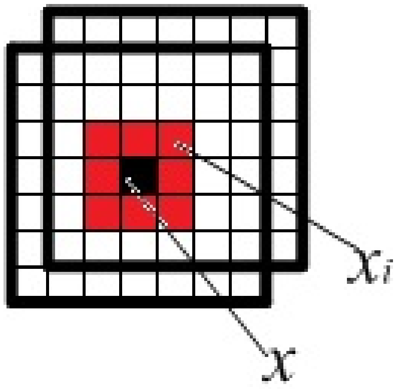


Figure 5.3: Two windows are used for region growing

In Step 1, the region growing algorithm extracts and matches a sparse set of seed points, which determines the performance of the algorithm. The SIFT-GPU software package is used for the first-level matching, and the main intention of the matching is to obtain a reliable result for the positions of the corresponding points, avoiding errors between repetitive patterns with a large variation in position. Fig. 5.2 shows two images of the temple data set [177] with different viewpoints matched. In Fig. 5.2, the superimposed white crosses are the sparse key points returned by SIFTGPU and used as initial seed points. However, in practice, if the total number of the matched key points is smaller than 50, these two images are considered as weakly matched; in this case, the dense matching will not proceed. Key points located at the boundaries of the image are removed from the seeds, and each point is only matched once. For example, if two points are located in the same region, the second point will be skipped in the selection of seed points.

Step 2 then examines the neighbouring pixels of seed points to determine the best match and whether the pixel neighbours should be merged into the region. The neighbouring window of an image point $x = \begin{bmatrix} u, v \end{bmatrix}^T$ is defined as $x_i = \begin{bmatrix} u_i, v_i \end{bmatrix}^T$,

$(i = 1..8)$ where x_i satisfies:

$$|u - u_i| + |v - v_i| = 1 \tag{5.1}$$

As shown in Fig. 5.3, the centre black point is the location of x, and the red area is the neighbour window of x. Areas with small variations in intensity can offer little information to the computation. Thus, a larger window of $n \times n$ pixels is used to compute the correlation, denoted as the correlation window in a bold black frame in Fig. 5.3. In the proposed approach, the selection of an appropriate window size is critical to achieve a smooth and detailed disparity map; the optimal choice of window size depends on the local amount of variation in texture and disparity. The size of the correlation window is a trade-off between computation speed and matching reliability and following Li et al. [48], n equals 7 in this chapter.

Once a pair of matching points has been found, the position of the search window with a very small size is determined accordingly and cost functions are only computed within the 3×3 (pixel) search window to find the maximum match value, which can be considered to be the correct correspondence of the point under consideration. If the confidence coefficient of the new corresponding points is high enough, the points are added to the set of seed points to produce new matches. The set of correspondence relationships between stereo pairs thus propagates from the seeds towards other image regions.

The ZNCC score of the correlation window is computed by:

$$r = \frac{\sum_{i=1}^{n}(I_i - \bar{I})(I'_i - \bar{I}')}{\sqrt{\sum_{i=1}^{n}(I_i - \bar{I})^2 \sum_{i=1}^{n}(I'_i - \bar{I}')^2}} \tag{5.2}$$

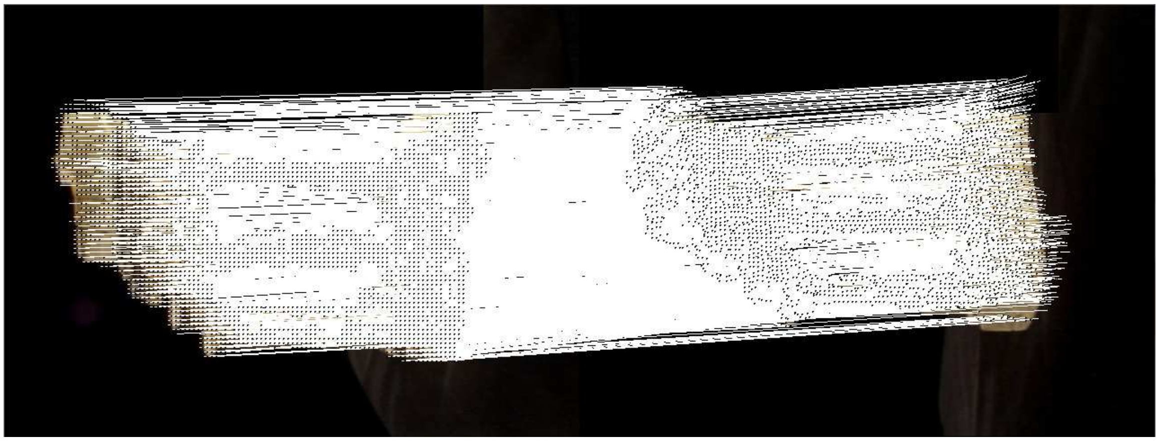


Figure 5.4: Dense matching results for Fig. 5.2

where I_i $(i = 1...n)$ are the intensity values of each pixel in the correlation window of the image, and I'_i $(i = 1...n)$ are the intensity values in the corresponding correlation window of the matched image. The absolute value of the correlation coefficient γ ranges from 0 to 1. Following [47], if γ is larger than 0.8, the two windows are considered as correlated.

The ZNCC score is then iteratively computed by moving the correlation window by one pixel in the neighboring area, where the location of the largest γ indicates the best (highest score) matching point. In Fig. 5.3, the correlation window centred at x_i moves pixel-by-pixel in the red window to search for the best matches. Fig. 5.4 shows the dense matching result from the two images of Fig. 5.2, where the white dense lines connect the matching points in the two images and the black dots are the positions of the matched dense points.

5.4 Surface Reconstruction

The geometric constraints associated with different views can be used to conduct matching and reconstruction of a 3D surface. The surface reconstruction approach proposed in this chapter starts with two 'initial' images and updates when each sub-

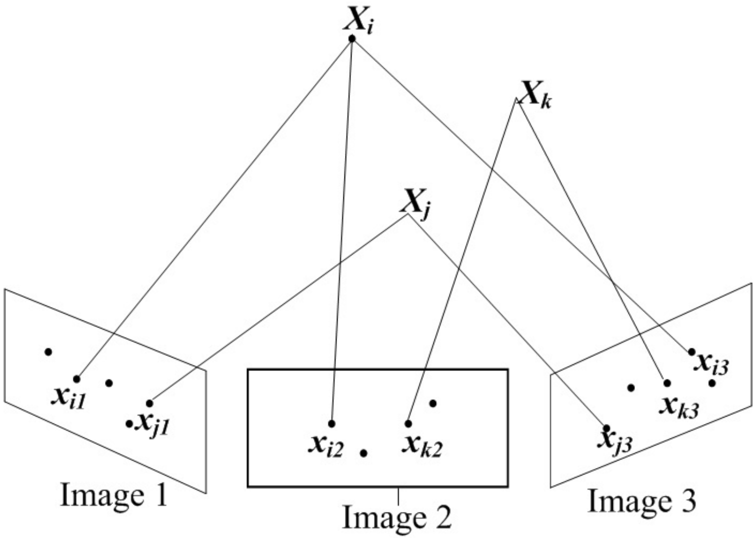


Figure 5.5: Multi-view self-calibration

sequent image is merged into the projective frame defined by the first two images. Thus, the self-calibration based surface reconstruction method is an image-by-image iteration of two stages: initialization with self-calibration and then optimization with bundle adjustment.

5.4.1 Initialization with Self-Calibration

As illustrated in Fig. 5.1, the proposed self-calibration initialization consists of five steps:

- Step 1 - Create Match Table: The match table is built from the results of the dense matching algorithm for each image pair in the whole image sequence.

- Step 2 - Create the Collection of Tracks: One track is a set of matching points and connects a certain physical surface point across the views. For example, in Fig. 5.5, X_i is a space point and the image points x_{i1}, x_{i2} and x_{i3} form the track of X_i. The image sequences are densely connected by a number of tracks, where a track list is built up from the match table.

- Step 3 - Calibrate the Camera Intrinsic Parameters: The calibration of the camera intrinsic parameters is based on [61]: for each consecutive image pair, the fundamental matrix is first computed using the linear normalized eight-point method and then optimized using the Gold Standard method. Kruppa$'$s equations are then used to calculate the intrinsic matrix A which is refined using the Levenberg-Marquardt algorithm [49].

- Step 4 - Preliminary Reconstruction: The initial surface reconstruction starts from the first two images (denoted as image 1 and 2, without loss of generality). The first image is used as the reference image, and the camera projective matrix is defined according to $P_1 = A \begin{bmatrix} I|0 \end{bmatrix}$. To evaluate the camera projective matrix, the camera motion parameters R and t are uniquely determined from the eight solutions of the essential matrix, as proposed in Chapter 3. With the rotation and translation matrix, the projective matrix P_2 can be obtained from $P_2 = A \begin{bmatrix} R|t \end{bmatrix}$. The projection can be geometrically modelled by a ray through the camera centre and the point in space that is being projected onto the image plane. Assuming that the matching points are (x_{i1}, x_{i2}), where $i = 1\dot{n}$, the 3D coordinates of X_i in images 1 and 2 are computed using linear triangulation from the matching points [59]. P_2 is then refined by the Levenberg-Marquardt algorithm [49], minimizing the function:

$$min \sum_{i=1}^{n} d(PX_i - x_i)^2 \tag{5.3}$$

- Step 5 - Multi-view Reconstruction: Motivated by [69], the subsequent images are merged into the preliminary reconstruction image-by-image in two steps: First, the matches that correspond to an already reconstructed point are used to compute the new projective matrix; second, the reconstruction is updated by initializing new points for new matches, refining these points and deleting incorrect points. This procedure is illustrated in Fig. 5.5 by the example of merging a third image into the surface reconstruction. In Fig. 5.5, image 3 firstly matches points corresponding to tracks from images 1 and 2. In Fig. 5.5, x_{i3} is in the same track as x_{i1} and x_{i2} and hence all three points are connected to the same 3D point X_i. The 3D point X_i was initialized by images 1 and 2, thus P_3 can be calculated as:

$$x_{i3} = P_3 X_i \tag{5.4}$$

The Direct Linear Transform (DLT) algorithm [61] can then be used to evaluate P_3 such that:

$$\begin{bmatrix} 0 & -X_i^T & v_i X_i^T \\ X_i^T & 0 & u_i X_i^T \\ -v_i X_i^T & u_i X_i^T & 0 \end{bmatrix} P^T = 0 \tag{5.5}$$

where x_i is denoted by (u_i, v_i). Since P is a 3×4 matrix, if the points number more than six, then P can be computed and P_3 is refined by minimizing Eq. (5.3). The new tracks on image 3 are subsequently added. As shown in Fig. 5.5, x_{j1} and x_{j3} consist of a new track, while x_{k2} and x_{k3} consist of another new track. The new 3D points X_j and X_k can be calculated from the linear triangulation method [59]. In practice, triangulating points at infinity can result in erroneous points; thus, the proposed approach rejects the points at infinity with a small angle threshold ($\theta = 2°$, following [120]). To increase the algorithm robustness and speed, two modifications are made: Firstly, it is important to add the visible image instead of the whole set. In this chapter, only the image projections of reconstructed points in visible images are tracked. If the angle between two rays of the matching point pair is less than 60°, the 3D point is considered as reconstructed by its visible images. Thus, the eligible matching point pair is constrained by the angle between two rays:

$$2^\circ < \theta_i < 60^\circ \tag{5.6}$$

The matches for which the angle falls outside this constraint are removed. This angle can be computed from the projective matrix and the image point, and the camera projective matrix is thus re-written as:

$$P = \begin{bmatrix} M | p_4 \end{bmatrix} \tag{5.7}$$

where M is the leftmost (3×3) block of P, and p_4 is a (3×1) column vector. The orientation of the ray that passes through the point $(x_{i1},\ x_{i2})$ is denoted as the vector $(v_{i1},\ v_{i2})$, which can be obtained from:

$$v_i = \frac{M^{-1}x_i}{\|M^{-1}x_i\|} \tag{5.8}$$

And θ_i is obtained from:

$$\theta_i = arccos(\frac{v_{i1} \bullet v_{i2}}{\|v_{i1}\|\|v_{i2}\|}) \tag{5.9}$$

Secondly, the outlier tracks that contain at least one key point with a high re-projection error are removed from the optimization. In the proposed approach, if the re-projection error of the point is larger than the threshold (eight pixels in practice), the track and the 3D point will be removed. The projection distance error, d, is computed as:

$$d = PX_i - x_i \tag{5.10}$$

Using the initial values of P_1, P_2 and P_3 and all the image points in the tracks x_{i1},x_{i2},x_{i3},x_{j1},x_{j2},x_{k2},x_{k3} and the 3D points X_i, X_j, X_k as input, the bundle adjustment method is used to refine the camera motion P_1, P_2, P_3 and 3D structure X_i, X_j, X_k. Finally, this entire multi-view reconstruction procedure is repeated until there are no images remaining in the sequence.

5.4.2 Bundle Adjustment

Bundle adjustment aims to refine a visual reconstruction to jointly estimate 3D structure and camera motion parameters, where the Sparse Bundle Adjustment software package was utilised in this thesis [131]. Suppose a set of n 3D points are visible in m perspective images, P_i is the camera projective matrix of the i-th image (where $i = 1...m$), and x_{ij} are the homogeneous coordinate vectors of the image points (where $j = 1...n$). The global solutions of 3D points X_j and P_i are then resolved by bundle adjustment according to the minimization:

$$E = min_{P_i, X_{ij}} \sum_{i=1}^{m} \sum_{j=1}^{n} d(P_i X_{ij}, x_{ij})^2 \tag{5.11}$$

Levenberg-Marquardt Algorithm

The Levenberg-Marquardt algorithm [49] [125] is the most popular algorithm for solving non-linear least squares problems, and the algorithm of choice for bundle adjustment. Let f be an assumed functional relation that maps a parameter vector $x \in R^n$, for a small $\|\delta_x\|$; f is approximated by:

$$f(x + \delta_x) = f(x) + J_{f(x)} \delta_x \tag{5.12}$$

where J is the Jacobian of f. The cost function with a quadratic Taylor expansion can be written:

$$c(x + \delta_x) = c(x) + \bigtriangledown c(x)^T \delta_x + \frac{1}{2} \delta_x^T H_{c(x)} \delta_x \tag{5.13}$$

where $\bigtriangledown c(x)$ is the gradient:

$$\bigtriangledown c(x) = \begin{bmatrix} \frac{\partial c(x)}{\partial x_1} & \dots & \frac{\partial c(x)}{\partial x_m} \end{bmatrix}^T \tag{5.14}$$

and $H_{c(x)}$ is the Hessian:

$$H_{c(x)} = \begin{bmatrix} \frac{\partial^2 c(x)}{\partial_{x_1} \partial_{x_1}} & \dots & \frac{\partial^2 c(x)}{\partial_{x_1} \partial_{x_m}} \\ \vdots & \vdots & \vdots \\ \frac{\partial^2 c(x)}{\partial_{x_n} \partial_{x_1}} & \dots & \frac{\partial^2 c(x)}{\partial_{x_n} \partial_{x_m}} \end{bmatrix} \tag{5.15}$$

Typically, the cost function is the square sum of all the dimensions of an error vector function $f(x)$:

$$c(x) = f(x)^T f(x) \tag{5.16}$$

Substituting Eq. (5.16) to Eq. (5.13),

$$c(x + \delta_x) = f^T f(x) + 2f^T J_{f(x)} \delta_x + \delta_x^T J_f^T J_{f(x)} \delta_x \tag{5.17}$$

which by equating the derivative to zero results in the update equation:

$$J_{f(x)}^T J_{f(x)} \delta_x = -J_{f(x)}^T f(x) \tag{5.18}$$

Multiplying the diagonal of $J_{f(x)}^T J_{f(x)}$ by the scalar $(1 + \lambda)$ leads to the Levenberg-Marquardt algorithm. It is guaranteed that an improvement will eventually be found: when an update δ_x with a sufficiently small magnitude and a negative scalar product, or when λ increases, the update tends to:

$$\delta_x = -\frac{1}{\lambda} (J_{f(x)}^T J_{f(x)})^{-1} J_{f(x)}^T f(x) \tag{5.19}$$

With this strategy, only the cost function needs to be re-evaluated when the step λ is increased upon failure to improve.

The core feature of bundle adjustment is to take advantage of the sparsity, which arises because the parameters for scene features and camera orientations jointly predict the measurements. More precisely, the reprojection error can be robust by applying a nonlinear mapping that decreases large errors, where the Jacobian J_f has the structure:

$$J_f = \begin{bmatrix} J_P & J_X \end{bmatrix} \tag{5.20}$$

where J_P is the Jacobian of the error vector f with respect to the camera orientation, and J_X is the Jacobian of the error vector f with respect to the 3D point positions. The Hessian approximation is given by:

$$H = \begin{bmatrix} J_P^T J_P & J_P^T J_X \\ J_X^T J_P & J_X^T J_X \end{bmatrix} \tag{5.21}$$

Thus, it has the form:

$$\begin{bmatrix} H_{PP} & H_{PX} \\ H_{PX}^T & H_{XX} \end{bmatrix} \begin{bmatrix} \delta_P \\ \delta_X \end{bmatrix} = \begin{bmatrix} b_P \\ b_X \end{bmatrix} \tag{5.22}$$

where $H_{PP} = J_P^T J_P$, $H_{PX} = J_P^T J_X$, $H_{XX} = J_X^T J_X$,$b_P = -J_P^T f$, $b_X = -J_X^T f$. Multiplying Eq. (5.22) by

$$\begin{bmatrix} H_{PP}^{-1} & 0 \\ 0 & I \end{bmatrix} \tag{5.23}$$

to reduce the upper left block to the identity matrix results in

$$\begin{bmatrix} I & H_{PP}^{-1} H_{PX} \\ H_{PX}^T & H_{XX} \end{bmatrix} \begin{bmatrix} \delta_P \\ \delta_X \end{bmatrix} = \begin{bmatrix} H_{PP}^{-1} b_P \\ b_X \end{bmatrix} \tag{5.24}$$

Then, multiplying both sides by:

$$\begin{bmatrix} I & 0 \\ -H_{PC}^T & I \end{bmatrix} \tag{5.25}$$

, results in the smaller equation system:

$$(H_{XX} - H_{PX}^T H_{PP}^{-1} H_{PC})\delta_X = b_X - H_{PX}^T H_{PP}^{-1} b_P \tag{5.26}$$

Eq. (5.26) is still a sparse system due to the fact that not all scene features appear in all cameras. Rearranging Eq. (5.26), the relationship between δ_P and δ_X can thus be written as:

$$\delta_P = H_{PP}^{-1} b_P - H_{PP}^{-1} H_{PX} \delta_X \tag{5.27}$$

For the problem shown in Fig. 2.10, the Bundle Adjustment(BA) algorithm consists of six steps [131]:

- Step 1 - Compute the derivative matrices $A_{ij} = \frac{\partial f(P_j, X_i)}{\partial P_j}$, $B_{ij} = \frac{\partial f(P_j, X_i)}{\partial X_i}$ and the error vectors $\epsilon_{ij} = x_{ij} - f(P_j, X_i)$, where $i = 1\dot{n}$ and $j = 1\dot{m}$.

- Step 2 - Compute the following auxiliary variables and augment the diagonal elements of U_j and V_i to yield $U_j^\star$ and $V_i^\star$:
  $U_j = \sum_i A_{ij}^T \sum_{x_{ij}}^{-1} A_{ij}$, $V_i = \sum_j B_{ij}^T \sum_{x_{ij}}^{-1} B_{ij}$, $W_{ij} = A_{ij}^T \sum_{x_{ij}}^{-1} B_{ij}$,
  $\epsilon_{P_j} = \sum_i A_{ij}^T \sum_{x_{ij}}^{-1} \epsilon_{ij}$, $\epsilon_{X_i} = \sum_j B_{ij}^T \sum_{x_{ij}}^{-1} \epsilon_{ij}$

- Step 3 - Compute $Y_{ij} = W_{ij} V_i^{\star -1}$.

- Step 4 - Compute δ_P from $S(\delta_{P_1}^T, \delta_{P_2}^T, \ldots, \delta_{P_m}^T)^T = (e_1^T, e_2^T, \ldots, e_m^T)^T$, where S is a matrix consisting of $(m \times m)$ blocks; block jk is defined by:
  $S_{jk} = \delta_{jk} U_j^\star - \sum_i Y_{ij} W_{ik}^T$, where $e_j = \epsilon_{P_j} - \sum_i Y_{ij} \epsilon_{X_i}$

- Step 5 - Compute each δ_{X_i} from the equation $\delta_{X_i} = V_i^{\star -1}(\epsilon_{X_i} - \sum_j W_{ij}^T \delta_{P_j})$

- Step 6 - Form δ as $(\delta_P^T, \delta_X^T)^T$.

This procedure can be embedded in the Levenberg-Marquardt algorithm for solving sparse normal equations.

5.5 Experimental Results

Two experiments were conducted to evaluate the proposed image reconstruction approach. Firstly, this section presents evaluations of the performance and accuracy of the proposed method with comparisons against two benchmark approaches: Bundler [120] (sparse reconstruction) and PMVS [118] (dense reconstruction). Secondly, this section presents results obtained from real world image sequences including Lambertian and non-Lambertian surfaces, to test the robustness and reconstruction accuracy of the proposed method.

5.5.1 Evaluation Experiments

The evaluation data sets were temple-sparse-ring (16 images) and temple-ring (47 images) [1],where all images were of size 640×480 pixels. Two snapshots with different view angles in the image sequence are shown in Fig. 5.2. The accuracy of the proposed algorithm is measured by back projecting the reconstructed points to 2D images, and the mean re-projection errors present the disparity between the back projection points and corresponding image points. The mean projection error is computed from:

$$e = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} d(P_i X_j - x_{ij})^2}{mn} \tag{5.28}$$

[1] http://vision.middlebury.edu/mview/

Table 5.1: Accuracy evaluation

Number of Images	Mean Re-projection error (pixels)
16	0.954
47	0.832

Table 5.2: Comparative results

Method	Image Number	3D points Number
Bundler	16	1256
PMVS	16	5342
Proposed Algorithm	16	16171
Bundler	47	11257
PMVS	47	12946
Proposed Algorithm	47	52289

As shown in Tab. 5.1, the proposed method reaches sub-pixel accuracy. In the experiments with two different data sets, the mean re-projection errors are both less than one pixel.

Tab. 5.2 shows the reconstruction results of Bundler, PMVS and the proposed algorithm on the evaluation data sets. From Tab. 5.2, for the 16 demo image temple sparse-ring sequence, the number of the reconstructed 3D points generated by the proposed algorithm is $\sim$ 12.8 times greater than the sparse method, while the number of reconstruction points from the proposed method is $\sim$ 3.0 times more than PMVS. The left-hand side images in Figs. 5.6$\sim$5.8 show the results of the Bundler, PMVS and proposed methods with 16 images. In Figs. 5.6 and 5.7, the results of Bundler and PMVS are perceptually clean and nearly noise free; however, the sparse results only show a rough outline of the temple. In comparison, the results in Fig. 5.8 obtained using the proposed method demonstrates more temple detail but accompanied by a number of diffuse points.

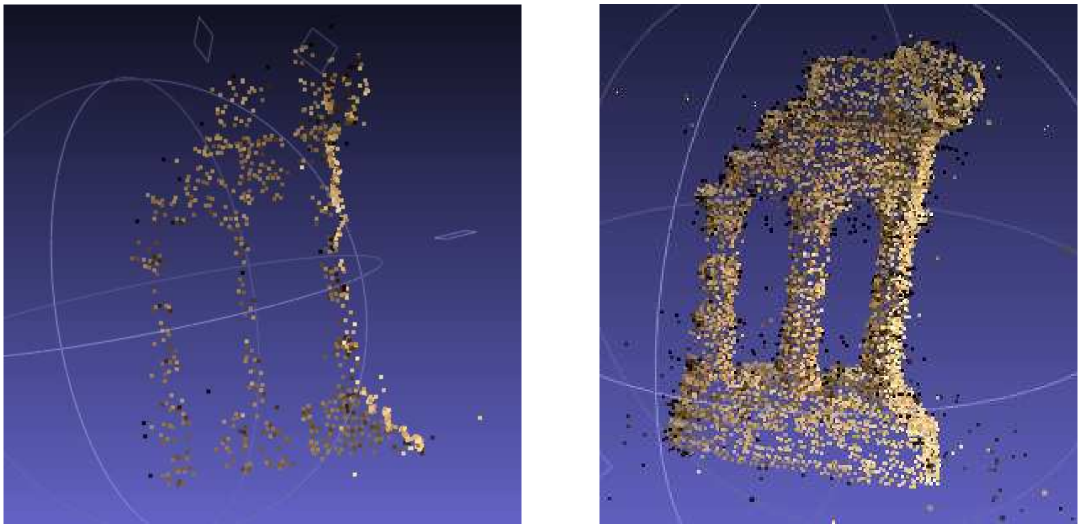


Figure 5.6: Bundler: left (16 images), right (47 images)

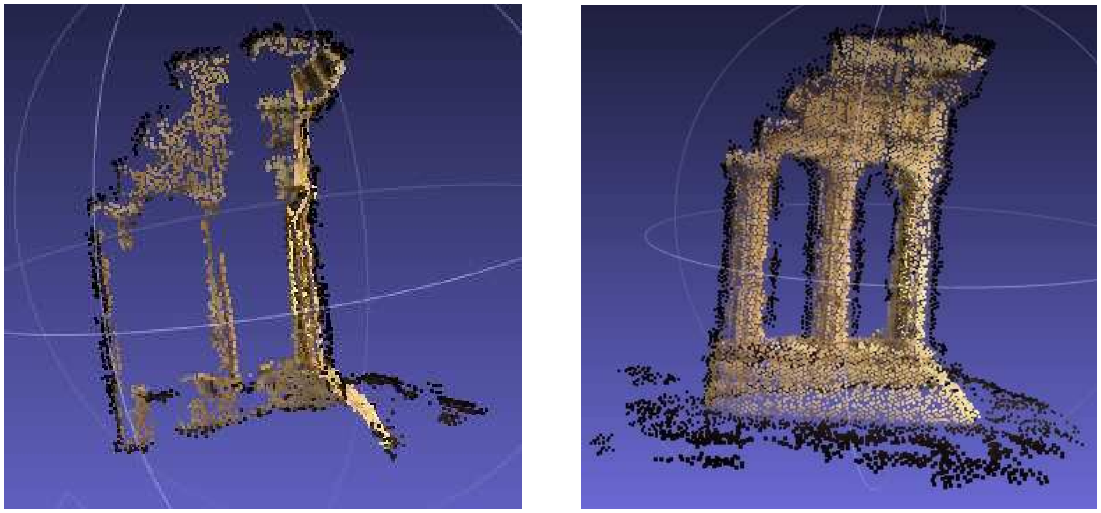


Figure 5.7: PMVS: left (16 images), right (47 images)

In the 47-image temple-ring sequence test results given in Tab. 5.2, Bundler and PMVS generate similar numbers of 3D points while the proposed method generates 4.04 times more points. The right-hand side images in Figs. 5.6∼5.8 show the results for the 47-image temple-ring sequence. It can be seen from Figs. 5.6 and 5.7 that Bundler and PMVS reconstructions are more detailed for the 16-image sequences, however, surface insufficiencies still exist. In contrast, reconstruction results from the proposed approach in Fig. 5.8 present higher density point clouds with a more complete model; however, the results are deteriorated by overlapping points. This

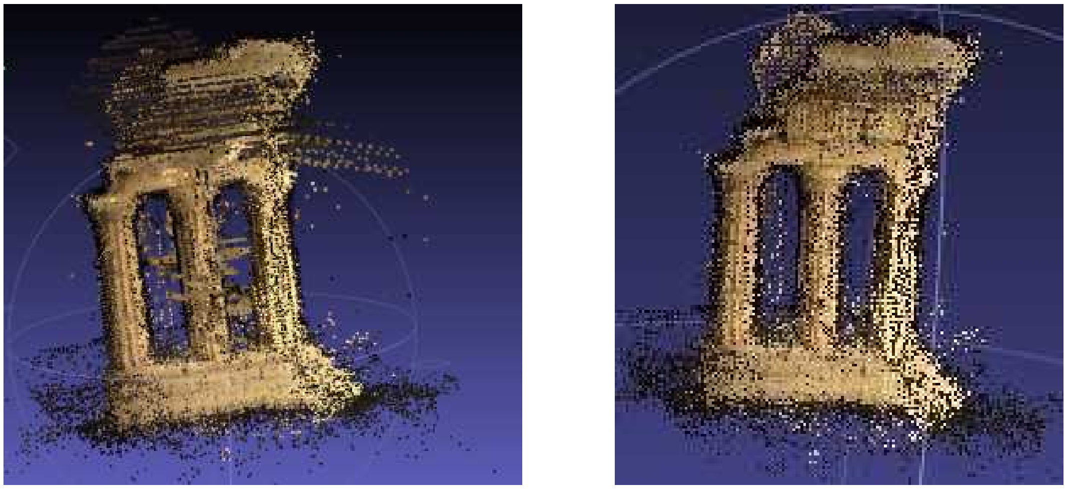


Figure 5.8: Proposed algorithm: left (16 images), right (47 images)

Table 5.3: Performance of proposed algorithm on different image sequences

Object	Image Number	Image Size	Number of 3D points	Mean Reprojection Error
Fountain	11	3072× 2048	34265	0.721
Building	25	720 × 576	38211	0.535
Flower	20	2048 × 1536	35430	0.591
Bag	17	1536 × 2048	19965	0.854

chapter assumes that the camera intrinsic parameters are constant during the image sequences. In practice, however, parameters such as focal length differ slightly between images. This variance in focal length thus varies the camera intrinsic parameters to result in 3D points overlapping as shown in the results of Fig. 5.8.

Implementation on different image sequences

The proposed method was also evaluated on four real world image sequences, shown in Fig. 5.9. Image sequence 1 (fountain) is sourced from [178] and image sequence 2 (building) was captured with a Sony camcorder NP70 by the authors; both sequences have fine details and a complex Lambertian surface. Image sequence 3 (flower) and sequence 4 (bag) in Fig. 5.9 were captured with a Sony P5 digital camera; the objects in sequences (3) and (4) possess non-Lambertian surfaces. The image resolutions range from 3 to 6 megapixels, as shown in Table 5.3. The height of the tar-

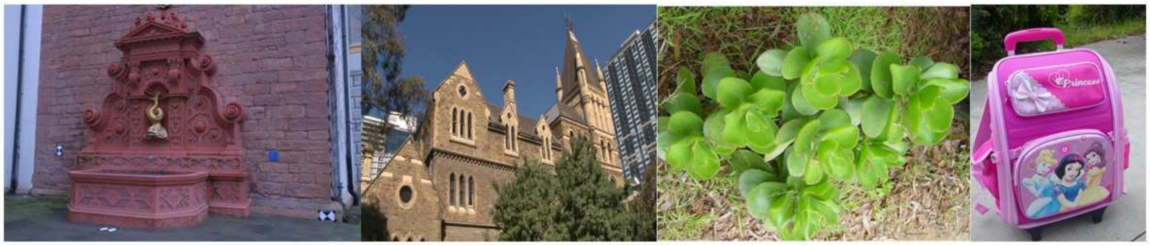


Figure 5.9: Real-world test images

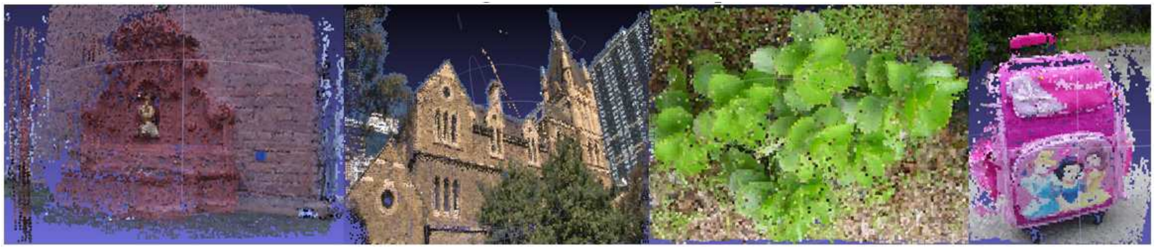


Figure 5.10: Reconstructed images

get objects ranges from 60cm to 6.5m and the number of images ranges from 11 to 25.

The reconstruction results are given in Tab. 5.3 and Fig. 5.10. As can be seen from Fig. 5.10, the reconstruction recovers major details of the target. However, in the fountain image set, the image sequences have a much larger baseline than building sequences. For example, the boundaries of the fountain differ greatly while the camera viewpoints change. The proposed region growing algorithm assumes that the two images are quite similar. However, if two matching areas vary greatly, the (7×7) correlation windows cannot be matched properly and the dense propagation will not start in these areas. Thus, the two sides of fountain are blurred in the result of Fig. 5.10. In reconstructing non-Lambertian surfaces, although the flower and the bag sequences in Fig. 5.10 both have some areas highlighted, the proposed algorithm shows the ability to handle highlighted areas. However, since the proposed method is based on key point detection, it is not suitable for smooth surfaces that lack features distinct in intensity, colour, texture or shape; this is particularly apparent for the top pink surface of the bag.

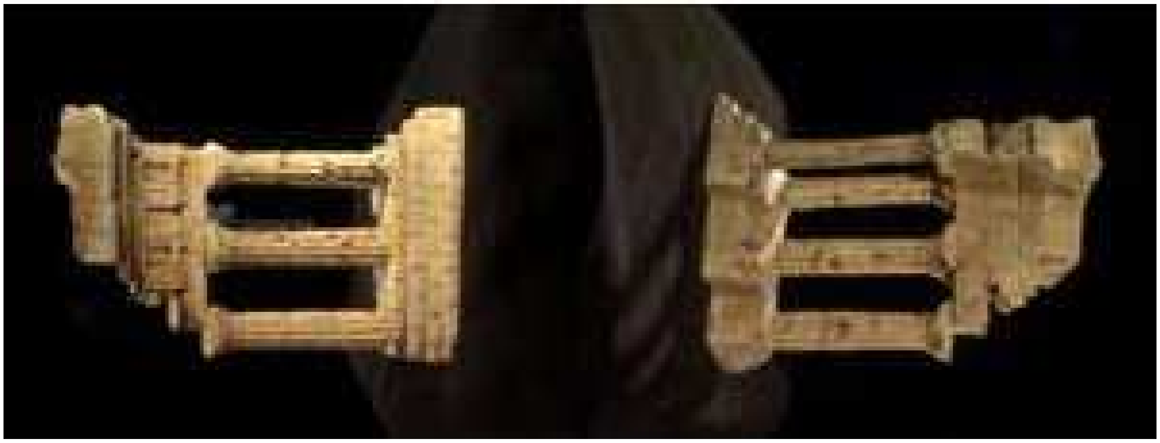


Figure 5.11: the TempleRing Dataset

5.6 3D Reconstruction Application for Handedness Constraint Validation

To demonstrate and evaluate the reliability of the handedness constraint to resolve the cheirality problem as presented in Chapter 3, a 3D model reconstruction from a 360° image sequence was conducted in two stages. The evaluation image sequences used were sourced from the TempleRing dataset, composed of 47 views sampled on a 360° ring about the object Fig. 5.11, where all images were of size 640 × 480 pixels. The original camera positions from the data set are shown in Fig. 5.12, where each white point represents one camera centre in one image.

In the preliminary reconstruction stage, the initial reconstruction is conducted using the first two images. The world origin is assumed at the first image, and the camera projective matrix is defined as $P_1 = A \begin{bmatrix} I|0 \end{bmatrix}$. To evaluate the camera projective matrix of the second image, the camera motion parameters R and t are determined from the SVD of the essential matrix. To select the correct camera motion from the eight possible solutions in Eq. (3.18) and Eq. (3.21), the proposed three-constraint algorithm in Eqs. (3.69) to (3.71) is applied and P_2 is then refined by the

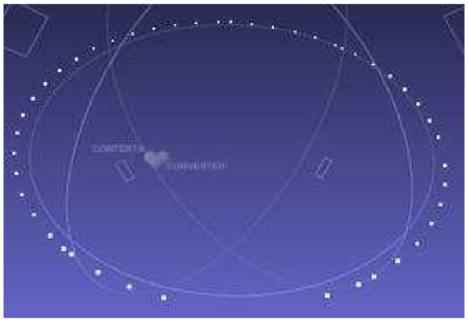


Figure 5.12: Original camera position

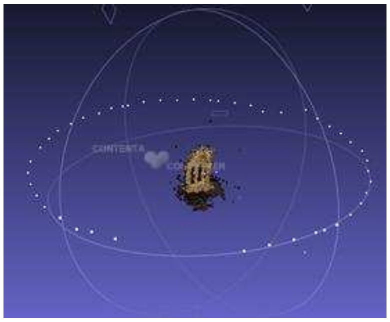


Figure 5.13: Test camera position

Levernberg-Marquardt algorithm [49].

In the multi-view reconstruction stage, the subsequent images are merged into the preliminary reconstruction image-by-image in two steps. First, the matches that correspond to the points already reconstructed are used to compute the new projective matrix. Then, the reconstruction is updated by initializing new points for new matches, refining these points and deleting incorrect points. For example, following preliminary reconstruction using images 1 and 2, image 3 is merged into the surface reconstruction. By using matching points corresponding to the same tracks in images 1 and 2, P_3 can be initialized using the DLT algorithm [30]. Then, the rotation matrix R_3 is computed from the QR factorization of P_3, and the sign of $det(R_3)$ is evaluated to ensure that P_3 follows the right-hand rule. P_3 is further refined by the Levenberg-Marquardt algorithm, and the new tracks for image 3 are added with the linear triangulation method. Using the initial values of P_1, P_2, P_3, the image points and the 3D points as input, the bundle adjustment method is used to refine the camera motion, P_1,P_2,P_3 and 3D points. Finally, this entire multiview reconstruction procedure is repeated until there are no remaining images in the sequence. In Fig. 5.13, the 47 camera centres computed from the camera projective matrix are added into the 3D model, as indicated by the white points. Thus, the cheirality problem of camera projections is resolved with the proposed cheirality constraint, where the same right handed coordinate system is ensured.

5.7 Summary

Building upon the camera motion calibration techniques proposed in Chapters 3 and 4, this chapter proposed a high density approach to surface reconstruction from a sequence of uncalibrated images based on SfM. The proposed approach addresses deficiencies in the surface integrity resulting from existing approaches, and presents a flexible automatic methodology with the simple interface of ‘videos to 3D model’; these improvements are vital for 3D modeling and visualization. Experimental results indicate that the proposed algorithm performs comparably to existing benchmark sparse and dense reconstruction approaches, and works reliably on real-world objects.

Part III

Real-Time 3D Reconstruction from Video Images

Chapter 6

Robust Real-Time Multi-View Reconstruction: Geometric Modelling Iterated EKF-SLAM

Despite the accurate and flexible advantages of the SfM 3D reconstruction technique proposed in Chapter 5, the main disadvantages of existing dense SfM multiple-view reconstruction approaches are the expense in time and memory. SfM reconstruction needs about 15 minutes for 47 images. To address these SfM shortcomings, this chapter proposes a real-time dense 3D reconstruction method based on Simultaneous Localization and Mapping (SLAM).

Over the past decade, the SLAM algorithm has been extensively applied to research in camera tracking and 3D mapping in robotic automation and computer vision. Potential applications range from camera motion tracking, real-time reconstruction of objects from video sequences, object/human motion tracking and recognition, 3D shape recognition, 3D navigation, and 3D obstacle detection for mobile robotics and augmented reality. In the highly successful MonoSLAM [55] system,

Figure 6.1: Microsoft Kinect [2]

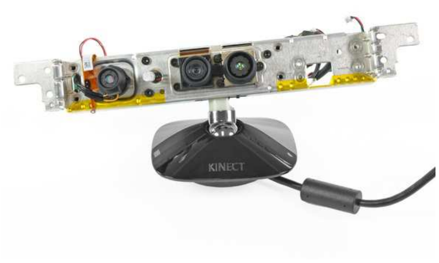


Figure 6.2: Kinect Inside [2]

camera poses and an incremental map of 3D landmarks are computed using a standard Extended Kalman Filter. Ever since Smith and Cheeseman [179] first employed an EKF as the central estimator, EKF-SLAM strategies have been widely used and have been improved significantly in feature selection criteria [180], robot navigation [181] and automatic re-localisation [182].

The EKF filter linearizes a non-linear system around the current state estimate, and thus produces errors when propagating the error covariance estimate through the model. Due to the unknown depth information in a monocular camera system, the landmark initialization is problematic for the EKF because of the combination of nonlinearity with large uncertainty in the non-measured DOF. Recent low-cost RGB-Depth (RGB-D) cameras, such as the Microsoft Kinect in Fig. 6.1, provide synchronized colour and per-pixel depth information in real-time. By using a depth sensor, the Kinect avoids the complexity of robust visual correspondence computation for depth estimation from stereo matching. As shown in Fig. 6.2, the depth sensor consists of one IR projector, one IR camera and one RGB camera, and the relative geometry between the IR image and the projector pattern can be easily measured [183]. The depth data output by the Kinect for each frame is the 'true' 3D information that addresses the real-time feature initialization in monocular camera EKF processing for camera motion and 3D structure estimation [184],[185].

Motivated by the low cost, reliable depth sensing and real-time speed of the Kinect camera, this chapter proposes a robust two-step Geometric Modelling Iterated Extended Kalman Filter (GMIEKF) SLAM algorithm to recover the 3D trajectory of a free moving RGB-D camera in real-time for multi-view reconstruction applications. Operating recursively on the measurement stream over time, the first step of GMIEKF-SLAM geometrically models the camera motion using 3D depth data from the RGB-D camera, where the results from this first step are applied as predictions to the second step, which employs the IEKF to update the states. This work focuses on accurate and robust modelling of the free-moving camera in an unknown environment, and provides accurate prior camera motion into the EKF process to provide more robust and consistent estimations compared to the standard EKF algorithm. With known 3D depth from the Kinect, this problem of uncertainty in camera motion thus becomes a geometric pose estimation problem. By using the depth from the RGB-D camera as the true 3D data for the geometric camera motion estimation, the proposed geometric modelling method fundamentally avoids the linear assumption errors of the camera motion model, and is an ubiquitous solution for any unknown camera motion and unknown environment. Further, the recursive nature of the GMIEKF-SLAM algorithm enables a more efficient solution to 3D reconstruction compared to purely geometric approaches, such as SfM [5], bundle adjustment [132] and PMVS [118].

This chapter presents the proposed GMIEKF-SLAM algorithm as part of a general system for autonomous multi-view reconstruction. The geometric modelling method includes feature extraction and matching, feature initialization and geometric state modelling. In the feature extraction and matching stage, the robust estimation algorithm RANSAC [11] is employed to remove the effect of mismatched points. This chapter proposes the management of feature points to control the number of points in the map by dynamically adding visible features with reliable 3D information or removing the occluded features during the evolution. Efficient rendering of complex geometric objects is then performed using surfel (surface element) representations of the depth data. Section 6.3 evaluates the proposed GMIEKF-SLAM approach, where the performance of GMIEKF-SLAM is compared to the standard EKF-SLAM algorithm [55] using a circle camera trajectory recovery experiment in a real indoor environment, with 3D reconstruction of an indoor scene and small object performed to demonstrate the camera motion estimation performance.

6.1 Background

6.1.1 Kalman Filter

The Kalman filter addresses the general problem of estimating the state $s \in R^n$ of a discrete-time controlled process that is governed by the linear stochastic difference equation:

$$s_k = As_{k-1} + Bu_{k-1} + w_{k-1} \tag{6.1}$$

A is the state transition model which is applied to the previous state s_{k1}; B is the control-input model which is applied to the control vector u_{k-1}; the random variables w_k and v_k represent the process and measurement noise, respectively, assumed to be independent, white and with normal probability distributions with covariance Q and R:

$$p(w) \sim N(0, Q) \tag{6.2}$$

$$p(v) \sim N(0, R) \tag{6.3}$$

At time k an observation (or measurement) z_k of the true state x_k is made according to:

$$z_k = Hs_k + v_k \tag{6.4}$$

where H is the observation model which maps the true state space into the observed space and v_k is the observation noise which is assumed to be zero mean Gaussian white noise with covariance R.

Defining $\hat{s}_k^- \in R^n$ to be the *a priori* state estimate at step k given knowledge of the process prior to step k, and $\hat{s}_k \in R^n$ to be the *a posteriori* state estimate at step k given measurement z_k, the *a priori* and *a posteriori* estimate errors can be defined as:

$$e_k^- = s_k - \hat{x}_k^- \tag{6.5}$$

$$e_k = s_k - \hat{s}_k \tag{6.6}$$

Then, the *a priori* estimate error covariance is given by:

$$L_k^- = E\begin{bmatrix} e_k^- & e_k^{-T} \end{bmatrix} \tag{6.7}$$

and the *a posteriori* estimate error covariance is:

$$L_k = E\begin{bmatrix} e_k & e_k^T \end{bmatrix} \tag{6.8}$$

The *a posteriori* state estimate $\hat{s}_k$ is a linear combination of an *a priori* estimated $\hat{s}_k^-$ and a weighted difference between an actual measurement z_k and a measurement prediction $H\hat{z}_k^-$:

$$\hat{s}_k = \hat{s}_k^- + K(z_k - H\hat{z}_k^-) \tag{6.9}$$

The difference $(z_k - H\hat{z}_k^-)$ in Eq. (6.9) is known as the measurement residual. The $n \times m$ matrix K is the gain factor with the form:

$$K_k = L_k^- H^T (H L_k^- H^T + R)^{-1} \tag{6.10}$$

The Kalman filter estimates a process by using a form of feedback control: the filter estimates the process state at some time and then obtains feedback in the form of (noisy) measurements. As such, the equations for the Kalman filter fall into two groups: prediction and measurement update equations. The prediction equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the *a priori* estimates for the next time step. The measurement update equations are responsible for the feedback incorporating a new measurement into the *a priori* estimate to obtain an improved *a posteriori* estimate. The prediction equations have the form:

$$\hat{s}_k^- = A\hat{s}_{k-1} + Bu_{k-1} \tag{6.11}$$

$$L_k^- = AL_{k-1}A^T + Q \tag{6.12}$$

And the measurement update equations are:

$$K_k = L_k^- H^T (H L_k^- H^T + R)^{-1} \tag{6.13}$$

$$\hat{s}_k = \hat{s}_k^- + K_k(z_k - H\hat{s}_k^-) \tag{6.14}$$

$$L_k = (I - K_k H)L_k^- \tag{6.15}$$

After each prediction and measurement update pair, the process is repeated with the previous *a posteriori* estimates used to project or predict the new *a priori* estimates. In the actual implementation of the filter, the measurement noise covariance R is usually measured prior to operation of the filter. It is possible to take some off-line sample measurements in order to determine the variance of the measurement noise. The determination of the process noise covariance is generally more difficult since it is hard to directly observe the process. Sometimes a relatively simple process model can produce acceptable results if one 'injects' enough uncertainty into the process via the selection of Q. Generally, the tuning of the filter parameters is performed off-line with the assumption that the process measurements and noise are reliable.

6.1.2 The Extended Kalman Filter

In estimation theory, the Extended Kalman Filter (EKF) is the nonlinear version of the Kalman filter which linearizes about an estimate of the current mean and covariance. The EKF recursively estimates a state vector s over the unknown parameters from measurements gathered by a sensor and the dynamic state:

$$s_k = f_k s_{k-1} + u_{k-1} \tag{6.16}$$

$$z_k = h(s_k) + v_k \tag{6.17}$$

where the dynamic model in Eq. (6.16) describes how the state vector s evolves in each time step k by the state transition function f and the control vector u. In Eq. (6.17), the $k-th$ measurements z are expressed as function h of the unknown state s, plus measurement noise v. The EKF-SLAM algorithm consists of the two stages of prediction and update as follows:

Prediction: In the prediction step, the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state:

$$\hat{s}_{k|k-1} = f(\hat{s}_{k-1|k-1}, u_{k-1}) \tag{6.18}$$

$$L_{k|k-1} = F_{k-1}L_{k-1|k-1}F_{k-1}^T + Q_{k-1} \tag{6.19}$$

Update: Once the outcome of the next measurement (necessarily corrupted with some amount of error, including random noise) is observed, these estimates are updated using a weighted average, with more weight being given to estimates with higher certainty.

$$K_k = L_{k|k-1}H_k^T(B_k + H_kL_{k|k-1}H_k^T)^{-1} \tag{6.20}$$

$$\hat{s}_{k|k} = \hat{s}_{k|k-1} + K_k(z_k - h(\hat{s}_{k|k-1})) \tag{6.21}$$

$$L_{k|k} = L_{k|k-1} - K_kH_kL_{k|k-1} \tag{6.22}$$

where K_k is the Kalman gain at step k, F and H are Jacobians of f and h. The process and image measurement noise are both assumed to be zero mean multivariate Gaussian noise with covariance matrices Q and B. The state vector is accompanied by a single covariance matrix L. Let $\hat{s}_{k|k-1}$ denote the *a priori* state estimate at step k, and $\hat{s}_{k|k}$ is the *a posteriori* state estimate at step k given measurement z_k. The *a priori* estimate error is $e_{k|k-1} = s_k - \hat{s}_{k|k-1}$ and the *a priori* estimate error covariance is then $L_{k|k-1} = E[e_{k|k-1}, e_{k|k-1}^T]$. The state prediction equations Eqs. (6.18) and (6.19) describe how the system models the state. The process update equations Eqs. (6.21) and (6.22) propagate the state estimate $\hat{s}_{k-1}$ through the dynamics of the system in Eqs. (6.16) and (6.17) and the covariance matrix accordingly.

6.1.3 EKF-SLAM

The SLAM algorithm is a process by which a mobile sensor can build a map of an environment and concurrently use the map to deduce its location. Consider a mobile sensor moving through an environment taking relative observations of a number of unknown landmarks as shown in Fig. 6.3. At a time instant k, the following quantities are defined:

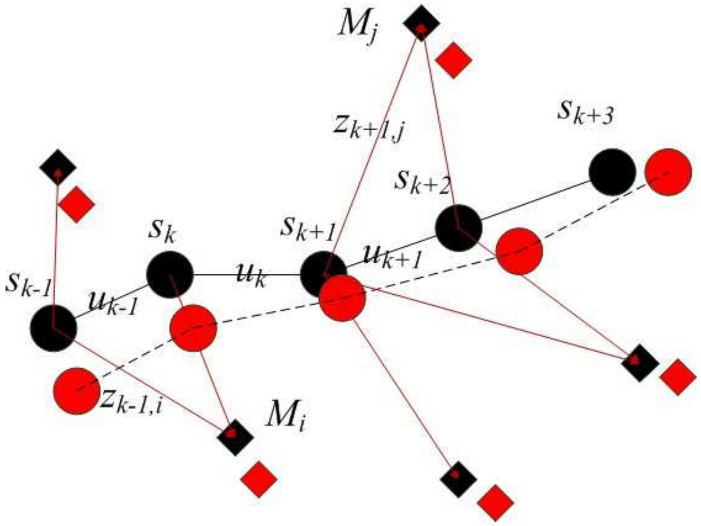


Figure 6.3: The MonoSLAM Problem

- s_k: The state vector including the location and orientation of the sensor, and the landmarks visible in this frame.

- u_k: The control vector, applied at time $k-1$ to drive the sensor to a state x_k at time k.

- M_i: A vector describing the location of the $i-th$ landmark whose true location is assumed to be time invariant.

- z_{ki}: An observation taken from the sensor at the location of the $i-th$ landmark at time k. When there are multiple landmark observations at any one time or when the specific landmark is not relevant to the discussion, the observation will be simply denoted as z_k.

6.2 The Geometric Modelling Iterated Extend Kalman Filter

This chapter proposes a two-step Geometric Modelling Iterated Extended Kalman Filter to estimate free-moving camera motion. The algorithm has the following architecture:

- Step One: A non-linear least squares optimization method is proposed to accurately estimate the camera motion.
  - Feature extraction and matching: Extract the feature points from two consecutive images, and match the corresponding points.
  - Feature initialization: Initialize the 3D feature points from 2D image points and the measurement functions.
  - Geometric State Modelling: Using the geometric method to evaluate the camera motion, this result is used as the prediction of the state and state covariance matrix.
- Step Two: Iterated-EKF is employed to update the state, where the differences between the expected measurements and matched points are used to update the predicted state with IEKF.

In the first step, the 3D positions of features are firstly initialized with the proposed geometric transformation. These 3D points are then back-projected to the matched image. The residual, defined as the difference between a matched value and the measured value (back-projected value), is formulated and the problem of optimization against the predicted state parameters becomes a least square problem of minimizing the measurement residual error. Simultaneously, the priori covariance matrix of states can be calculated from the deviations of the state value between the *a priori* and optimized value. States are then optimally obtained by treating the first-step state estimates as predictions. Addressing issue 3 of EKF-SLAM as discussed in section 2.6, the proposed approach taken in the second step of the GMIEKF-SLAM does not neglect the higher order terms of the Taylor series expansions and an iterative EKF is applied to perform a nonlinear least squares fit. This step of applying the IEKF re-linearizes the measurement equation by an iterative Newton-Raphson algorithm around the updated state. Since the iterated approaches inevitably increase the computational complexity, this chapter makes a trade-off between estimation accuracy and computational cost by setting *a priori* iteration times.

6.2.1 GMIEKF Step One

Feature Extraction and Matching

The feature points from two consecutive images are extracted using the Scale Invariant Feature Transform (SIFT) [81], where this work utilizes the SIFTGPU package [176]. The points with valid Kinect 3D depth are selected as key points; in particular, the visibilities of the feature points are constrained by the back-projections of the corresponding 3D points within the range of the image. The feature points on the two consecutive images are matched with the Zero-mean Normalized Cross-Correlation (ZNCC) algorithm. Typically, these matched feature points contain a significant number of outliers. In this chapter, the robust estimation algorithm RANSAC [11] is employed to remove the effect of mismatched points (outliers). Fig. 6.4 and Fig. 6.5 show two consecutive images with different view angles and Fig. 6.6 shows the matched feature points. In Fig. 6.6, the two images are superimposed on each other: the red circles and green crosses are the feature points of the images in Figs. 6.4 and 6.5. Applying RANSAC results in a refined set of essential feature points, as shown in Fig. 6.7. RANSAC computes a relation that best fits the data and classifies the data as inliers (correct matches) and outliers. The classification employs a cost function

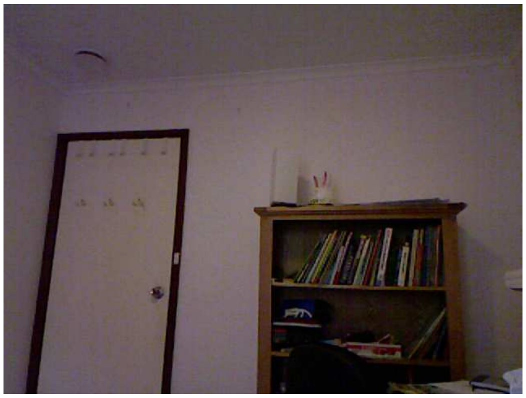


Figure 6.4: Left Image

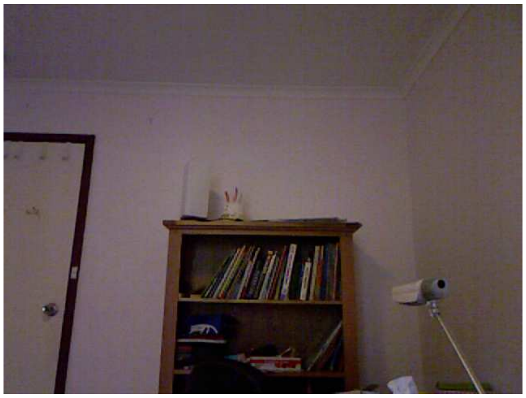


Figure 6.5: Right Image

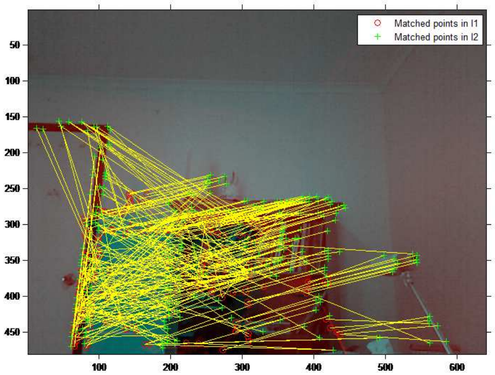


Figure 6.6: The matches before RANSAC

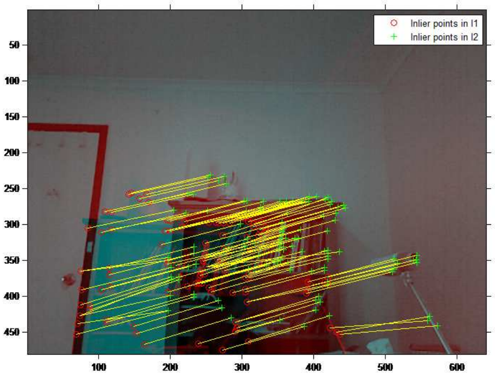


Figure 6.7: The matches after RANSAC

together with a threshold that depends on the expected measurement noise. This threshold is directly correlated with the number of feature points. In this chapter, to trade-off between accuracy and real-time processing, the number of inliers is bounded to around $N = 100$ for each time step in practice.

New features are only added into the system if the number of features in the current time step is less than the threshold $\frac{N}{2}$. A feature point is deleted from the system if a match point cannot be found in the new image.

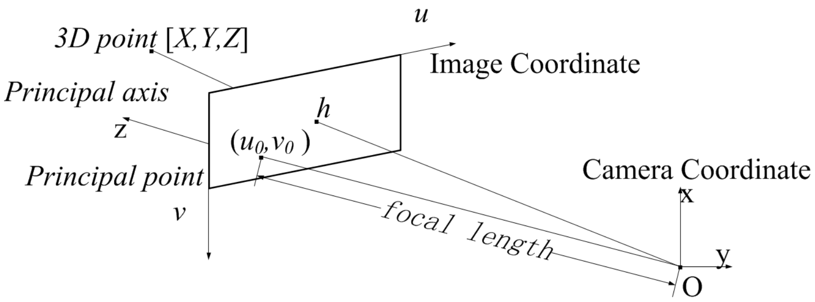


Figure 6.8: The relationship between camera and image coordinates

Feature Initialization

In this chapter, the state vector is composed of the camera motion information and the 3D features:

$$\hat{s} = \begin{bmatrix} \hat{s}_v & \hat{M}_1 & \hat{M}_2 & \dots \end{bmatrix}^T \tag{6.23}$$

The camera motion information is given by:

$$\hat{s}_v = \begin{bmatrix} t & q \end{bmatrix}^T \tag{6.24}$$

where the camera′s state vector comprises a metric 3D translation vector $t = \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^T$, and the camera rotation is represented by orientation quaternion $q = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T$. This chapter uses the 'hat' to indicate an estimate of s_v, and the 3D feature locations are stored as: $\hat{M}_i = \begin{bmatrix} X_i & Y_i & Z_i \end{bmatrix}^T$.

The pinhole camera perspective geometry is shown in Fig. 6.8, and the camera and image coordinates are related by the perspective projection equations:

$$\frac{x - x_0}{f_x} = \frac{x_c}{d} \tag{6.25}$$

$$\frac{y - y_0}{f_y} = \frac{y_c}{d} \tag{6.26}$$

where f_x and f_y are the distances from the centre of projection to the image plane, $\begin{bmatrix} x_0 & y_0 \end{bmatrix}^T$ is the coordinate of the camera centre, and d is the depth of the image point $m = \begin{bmatrix} x & y \end{bmatrix}^T$. $m_c = \begin{bmatrix} x_c & y_c & 1 \end{bmatrix}^T$ is the homogeneous representation of m in camera coordinates. Thus, 3D point M can be estimated from:

$$M = R^{-1} m_c + t \tag{6.27}$$

where R is the rotation matrix representation of the quaternion orientation, and t is the translation vector; both t and R are estimated by the proposed GMIEKF-SLAM approach for each time step. Upon system initialization, the first image centre is assumed as the origin point, with $q = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$ and $t = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$. In this chapter, the Kinect intrinsic parameters including the camera centre $\begin{bmatrix} x_0 & y_0 \end{bmatrix}$ and the focal length f_x and f_y are assumed as known and constant, where the values of intrinsic parameters are based on Burrow's Kinect calibration results [186]. The initial 3D feature points can then be obtained from Eqs. (6.25) and (6.26).

With a perspective camera, the position at which the feature point is expected to be found in the image has the form:

$$m = \begin{bmatrix} x & y \end{bmatrix}^T \tag{6.28}$$

The measurement function h at time-step k is initialized by:

$$m_c = R(M - t) \tag{6.29}$$

Then the estimate of the feature on image has the form:

$$m = \begin{bmatrix} \hat{x} & \hat{y} \end{bmatrix}^T = \begin{bmatrix} (x_0 - f_x \frac{m_{cx}}{m_{cz}}) & (y_0 - f_y \frac{m_{cy}}{m_{cz}}) \end{bmatrix}^T \tag{6.30}$$

Geometric State Modelling

In this chapter, let an unknown vector χ_k present the unknown predicted states. The measurement residual at step k can be written as:

$$\epsilon_k = z_k - h(\chi_k) \tag{6.31}$$

where z_k contains the measurement value of the matched points and h is the measurement equations of Eqs. (6.28) and (6.29). The cost function for the measurement residual of all features is then:

$$\sum_{i=1}^{N} \epsilon_k = \sum_{i=1}^{N} (z_k^i - h(\chi_k)^i) \tag{6.32}$$

where N is the number of the features. The optimal estimator quantifies the prediction error of the model parametrized by minimizing the cost function Eq. (6.31), which is designed to measure how well the model fits the observations z_k^i and the measurements $h(\chi_k)^i$. The 3D feature points can be initialized using the matched image points via Eqs. (6.25) and (6.26), and only the camera motion parameters R_k (with the quaternion $q_k = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}$) and t_k in the unknown vector χ_k need to be optimized. The new cost function can be written as:

$$R_k, t_k = \arg\min_{R_k, t_k} \sum_{i=1}^{N} (z_k^i - h(\chi_k)^i) \tag{6.33}$$

Using the *a priori* state parameters as an initial guess, the non-linear least squares optimization of χ is conducted with the Levernberg-Marquardt algorithm [125]. In this chapter, the estimates of the first-step are treated as the state prediction:

$$\hat{s}_{k|k-1} = \chi_k \tag{6.34}$$

Using this optimal state as the new measurement input, the state deviations e have the form:

$$e_{k|k-1} = s_k - \chi_k \tag{6.35}$$

And the *a priori* covariance of the state deviations is:

$$L_{k|k-1} = E\left[e_{k|k-1}e^T_{k|k-1}\right] \tag{6.36}$$

6.2.2 GMIEKF Step Two - Iterated EKF Update

In the second step of the proposed GMIEKF algorithm, the higher order terms of Taylor series are not neglected. The EKF estimate optimality of the unknown parameters can be determined with respect to a cost function J minimized at each time step based on the estimate from the previous time step:

$$J_k = \frac{1}{2}[z_k - h(s_k)]^T B_k^{-1}[z_k - h(s_k)] + \frac{1}{2}(s_k - \hat{s}_{k|k-1})^T L^{-1}_{k|k-1}(s_k - \hat{s}_{k|k-1}) \tag{6.37}$$

where B_k is the measurement noise covariance matrix in time step k. Expanding J in a second order Taylor series about the i-th iterated value of the estimate of s_k, denoted as s^i_k:

$$J_k = J^i_k + (J^i_{s_k})^T(\hat{s}_k - \hat{s}^i_k) + \frac{1}{2}(\hat{s}_k - \hat{s}^i_k)^T J^i_{s_k s_k}(\hat{s}_k - \hat{s}^i_k) \tag{6.38}$$

$$J^i_k = J|_{s_k = s^i_k} \tag{6.39}$$

The gradient of J is:

$$J^i_{s_k} = -(h^i_k)^T B_k^{-1}(z_k - h_k) + L^{-1}_{k|k-1}(\hat{s}^i_{k|k} - \hat{s}^i_{k|k-1}) \tag{6.40}$$

The Hessian of J, retaining only up to the first derivative of h, is:

$$J^i_{s_k s_k} = (H^i_k)^T B_k^{-1} H^i_k + L^{-1}_{k|k-1} \tag{6.41}$$

where $H^i_k = \dfrac{\partial h^i_k}{\partial \hat{s}^i_{k|k}}$

The Newton-Raphson algorithm is used to find the optimal estimation that minimizes the cost function Eq. (6.37) by setting its derivative to zero yielding the next value of s_k in the iteration as:

$$\hat{s}^{i+1}_k = \hat{s}^i_k - (J^i_{s_k s_k})^{-1} J^i_{s_k} \tag{6.42}$$

Substituting Eq. (6.39) and Eq. (6.40) into Eq. (6.41), the update equations of GMIEKF-SLAM are thus:

$$\hat{s}^{i+1}_k = \hat{s}^i_k + L^i_{k|k}(H^i_k)^T B_k^{-1}(z_k - h^i_k) - L^i_{k|k} L^{-1}_{k|k-1}(\hat{s}^i_{k|k} - \hat{s}_{k|k-1}) \tag{6.43}$$

$$L^i_{k|k} = L_{k|k-1} - L_{k|k-1}(H^i_k)^T [H^i_k L_{k|k-1}(H^i_k)^T + B_k]^{-1} H^i_k L_{k|k-1} \tag{6.44}$$

As a result, GMIEKF-SLAM repeatedly calculates an intermediate posterior state $\hat{s}^i_k$, where i is the iteration number. GMIEKF-SLAM starts from the *a priori* state, where $\hat{s}^0_k = \hat{s}_{k|k-1}, H^0_k = H_k\ , L^0_{k|k} = L_{k|k-1}$. At each iteration, the previous iteration's estimate and covariance matrix are used as the new *a priori* information. When the consecutive values differ by less than a preselected threshold:

$$\hat{s}^{i+1}_k - \hat{s}^i_{k|k} < \xi \tag{6.45}$$

or after a certain number of iterations, the iteration is stopped, which indicates the approach of the cost function in Eq. (6.44). GMIEKF-SLAM decreases the linearization error by re-linearizing the measurement model and tries to find the best estimate of the state.

6.3 EKF and IEKF Simulation Comparison

A range and bearing camera model [12] with synthetic map and landmark data was used to compare the consistency between EKF and IEKF under variant image noise. The camera′s true trajectory is known as a circle of centre $(0, 20)m$ and $20m$ radius. The landmarks that are visible in the semi-circular field of view of the camera are selected in each time step. The experimental parameters are set with $\delta\dot{t} = 1.0m/s^2$, $\delta\dot{\theta} = 3.0rad/s^2$, $\delta v = 1.0pixels$ and $\Delta T = 0.1s$, where the measurement noise is assumed to be 3 pixels ($\delta v = 3.0pixels$). Assuming the z axis to be zero, where the camera only moves on the xy plane, the camera state at time step k is simplified into the form:

$$s_k = \begin{bmatrix} t_x & t_y & \theta_k & \dot{t}_k & \dot{\theta}_k & x_1 & y_1 & \ldots & x_N & y_N \end{bmatrix}^T = \begin{bmatrix} s_{vk} \\ M_{1 \ldots N} \end{bmatrix} \tag{6.46}$$

The Normalised Estimation Error Squared (NEES) method [137] is used to characterize the filter performance:

$$\varepsilon_k = (s_k - \hat{s}_{k|k})^T P_{k|k}^{-1} (s_k - \hat{s}_{k|k}) \tag{6.47}$$

where s_k is the ‘true’ state vector from the synthetic data and $\hat{s}_{k|k}$ is the estimated value. Each filter was run for two loops of the trajectory with each simulation repeated 50 times. Figs. 6.9 and 6.10 show the NEES results of EKF and IEKF averaged over the 50 simulation repetitions, where the x and y axes indicate time step and NEES error, respectively (the total number of time steps is $k = 277$). In Fig. 6.9,

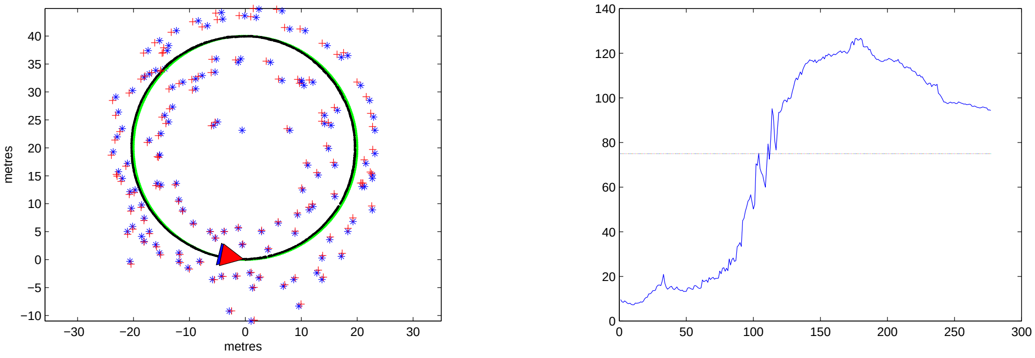


Figure 6.9: The simulation path and NEES of EKF ($\delta v = 1.0 pixels$)

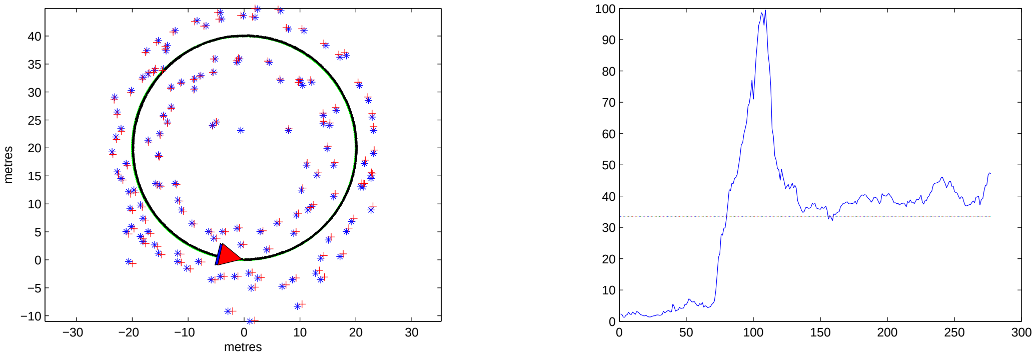


Figure 6.10: The simulation path NEES of IEKF ($\delta v = 1.0 pixels$)

the NEES of EKF has a quick increase from $k = \begin{bmatrix}50 & 150\end{bmatrix}$, maintaining this high error level NEES until the simulation finishes. When $k = \begin{bmatrix}100 & 150\end{bmatrix}$, the camera closes the first loop. In the second loop, the measurement error is accumulated, and EKF maintains this error into the second loop. In Fig. 6.10, the NEES of IEKF peaks when the camera closes the first loop then quickly converges; thus, it can be seen that the IEKF observations are more accurate than EKF since the NEES mean value is 33.1 lower than the EKF.

In the second experiment, measurement noise of $\delta v = 3.0$ pixels is added, whilst all other parameters are kept unchanged. Figs. 6.11 and 6.12 are the NEES of EKF and IEKF, respectively. In comparison to Figs. 6.9 and 6.10, the mean NEES values

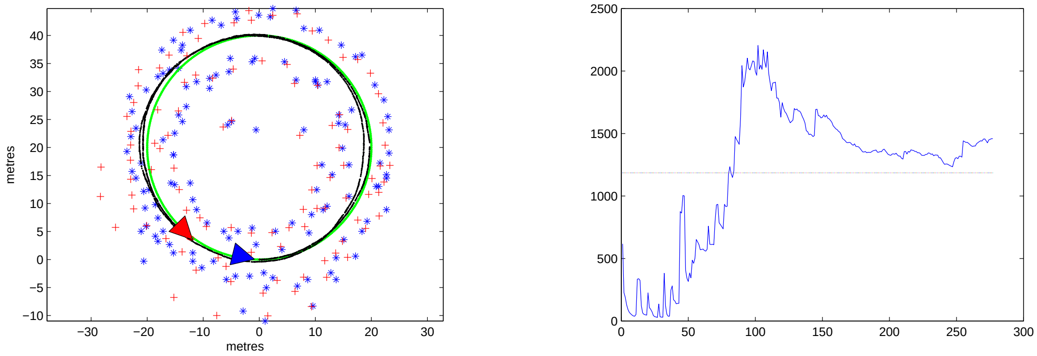


Figure 6.11: The simulation path and NEES of EKF ($\delta v = 3.0 pixels$)

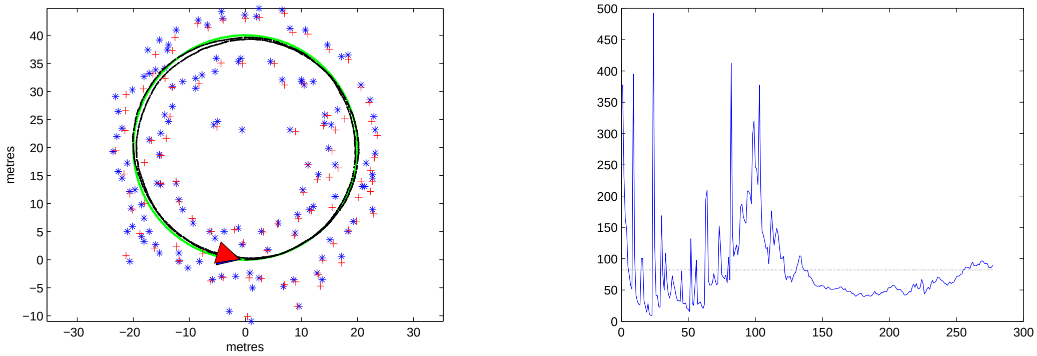


Figure 6.12: The simulation path and NEES of IEKF ($\delta v = 3.0 pixels$)

of both algorithms show an increase: the EKF increases from 78.87 to 1289.63, whilst the IEKF increases from 33.1 to 79.23. Thus, in a simulation environment, the IEKF shows a better consistency against increased measurement noise compared to EKF.

6.4 Experimental Results

6.4.1 Estimation Accuracy Comparison Under Nonlinear Acceleration

This section presents an experiment under nonlinear camera motion model (irregular accelerations). This experiment evaluates the accuracy of the proposed GMIEKF-SLAM algorithm for camera trajectory recovery of a real room environ-

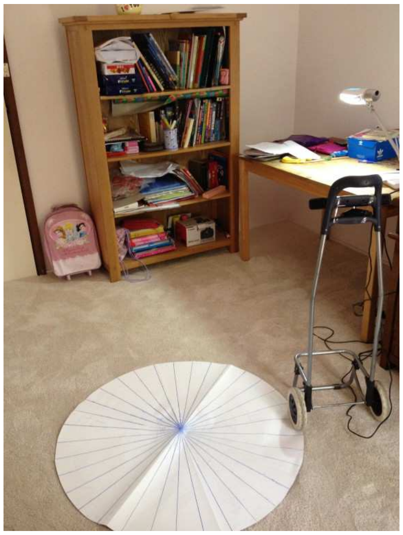


Figure 6.13: The real room environment

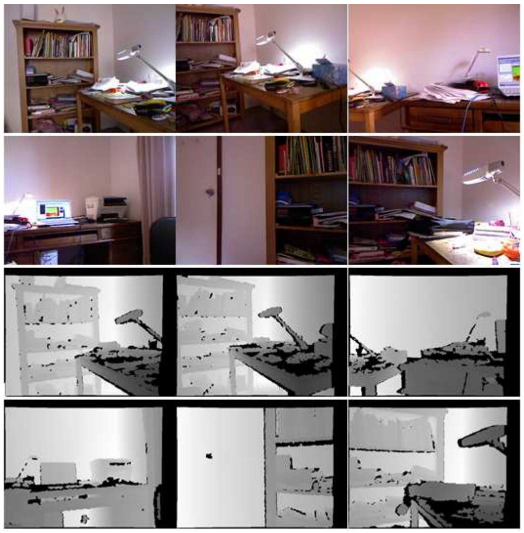


Figure 6.14: The sample images and depth

ment, comparing GMIEKF to the standard EKF-SLAM approach [55]. The Kinect is fixed on a trolley of 0.8m in height, where the wheels can only move back and forth to constrain movement to the XY plane. The trolley moves along a 360° circle with radius 0.6m. As shown in Fig. 6.13, the left wheel of the trolley walks along the edge of paper circle, where the circle is divided into 10° segments. In this chapter, the Kinect motion simulates nonlinear acceleration motion – the Kinect walks in an arbitrary velocity along the circle and pauses at each segment for an arbitrary time period i.e., 36 stops in total. Then the images and corresponding depth data of the Kinect are sampled frame-by-frame. The whole sequence has 446 images, where each frame image size is 640 × 480; Fig. 6.14 shows the sample images and corresponding depth images obtained from the sequence. In this experiment, both EKF-SLAM and the proposed GMIEKF-SLAM are run on this same image sequence to perform camera trajectory recovery.

The camera motion model of standard EKF-SLAM [55] is:

$$n = \begin{bmatrix} \delta \dot{t} \\ \delta \dot{\theta} \end{bmatrix} \tag{6.48}$$

The covariance matrix of the processing noise vector n is assumed as diagonal, representing the translation and rotation components of the uncorrelated noise. And the camera motion model of [55] is:

$$f = \begin{bmatrix} t_{k+1} \\ q(\theta_{k+1}) \\ \dot{t}_{k+1} \\ \dot{\theta}_{k+1} \end{bmatrix} = \begin{bmatrix} t_k + (\dot{t}_k + \delta\dot{t}\Delta T)\Delta T \\ q(\theta_k) \times q(\dot{\theta}_k + \delta\dot{\theta}\Delta T)\Delta T \\ \dot{t}_k + \delta\dot{t}\Delta T \\ \dot{\theta}_k + \delta\dot{\theta}\Delta T \end{bmatrix} \tag{6.49}$$

where the notation $q(\dot{\theta}_k + \delta\dot{\theta}\Delta T)\Delta T$ denotes the quaternion rotation vector $(\dot{\theta}_k + \delta\dot{\theta}\Delta T)\Delta T$.

In the experimental practice, the parameters of the EKF-SLAM are set as follows: constant angular velocity is $\dot{\theta} = 0rad/s$; velocity is $\dot{t} = 0m/s$; processing noise vector is initialized with $\delta\dot{t} = 0.007m/s^2$ and $\delta\dot{\theta} = 0.001rad/s^2$; measurement noise is $v = 1pixel$; measurement noise covariance is $\delta v = 0.0025pixel/s^2$; and, $\Delta T = 1s$. In the proposed GMIEKF-SLAM approach, the measurement noise and covariance are set to the same values as for EKF-SLAM for accurate comparisons. The thresholds of features measurement (e.g., the number of measured features in each images is 50; the SIFT thresholds for feature detection is 0.2 and cross-correlation threshold is 0.9; the desired probability thresholds of RANSAC algorithm is 0.99) were kept the same as EKF-SLAM. Then, the raw depth values are converted into real 3D values. In the depth images, the raw depth values range from 0 to 2047. This chapter uses the

Table 6.1: Comparison of estimation performance

parameters	Algorithm	MSE of x(m)	MSE of y(m)	MSE z(m)	Time Consumption(ms)
tuned	EKF-SLAM	0.1487	0.173	0.0106	33.32
tuned	GMIEKF-SLAM	0.1009	0.0845	0.0082	167.94
$\dot{t} = 0.03m/s$ $\dot{\theta} = 0.18rad/s$	EKF-SLAM	0.2104	0.1907	0.008	32.54
$\dot{t} = 0.03m/s$ $\dot{\theta} = 0.18rad/s$	GMIEKF-SLAM	0.093	0.0845	0.0082	178.32
$imagenoise = 3pixel$	EKF-SLAM	0.1866	0.1862	0.0144	34.77
$imagenoise = 3pixel$	GMIEKF-SLAM	0.1105	0.0862	0.0101	177.74

conversion parameters given by ROS Kinect [186] to convert the raw depth values into real 3D values. Lastly, in each time step, the camera motion parameters t and R can be obtained from the estimated state $\hat{s}_k$, and the Kinect camera centres are calculated from:

$$P_k = A \left\{ R_k | t_k \right\} \tag{6.50}$$

where P_k has the form:

$$P_k = \left\{ P_k^3 | p_k^4 \right\} \tag{6.51}$$

$$C_k = -(P_k^3)^{-1} p_k^4 \tag{6.52}$$

and A represents the Kinect intrinsic parameters [186], P_k^3 is the left 3×3 matrix of P_k and p_k^4 is the $4th$ column of P_k.

In this experiment, the target camera trajectory should be a circle with 0.6m radius on the XY plane, where the z axis remains constant. Fig. 6.15 and Fig. 6.16 show the camera centre estimates from EKF-SLAM and the proposed GMIEKF-SLAM algorithm for four snapshots ($1st$, $50th$, $350th$ and $446th$ frame) from the 446 images captured. On the left image, the green crosses and red circles are the match point positions, whilst the estimated results of camera centre path are shown on the right image.

In this chapter, the start point coordinates are set as $\begin{bmatrix} 0 & 0 & 0.8 \end{bmatrix}$ and the circle radius is 0.6 m; thus, the ground truth 3D coordinates of each stop are known. Fig. 6.17 shows the trajectory errors compared to the ground true values for the 36 stops

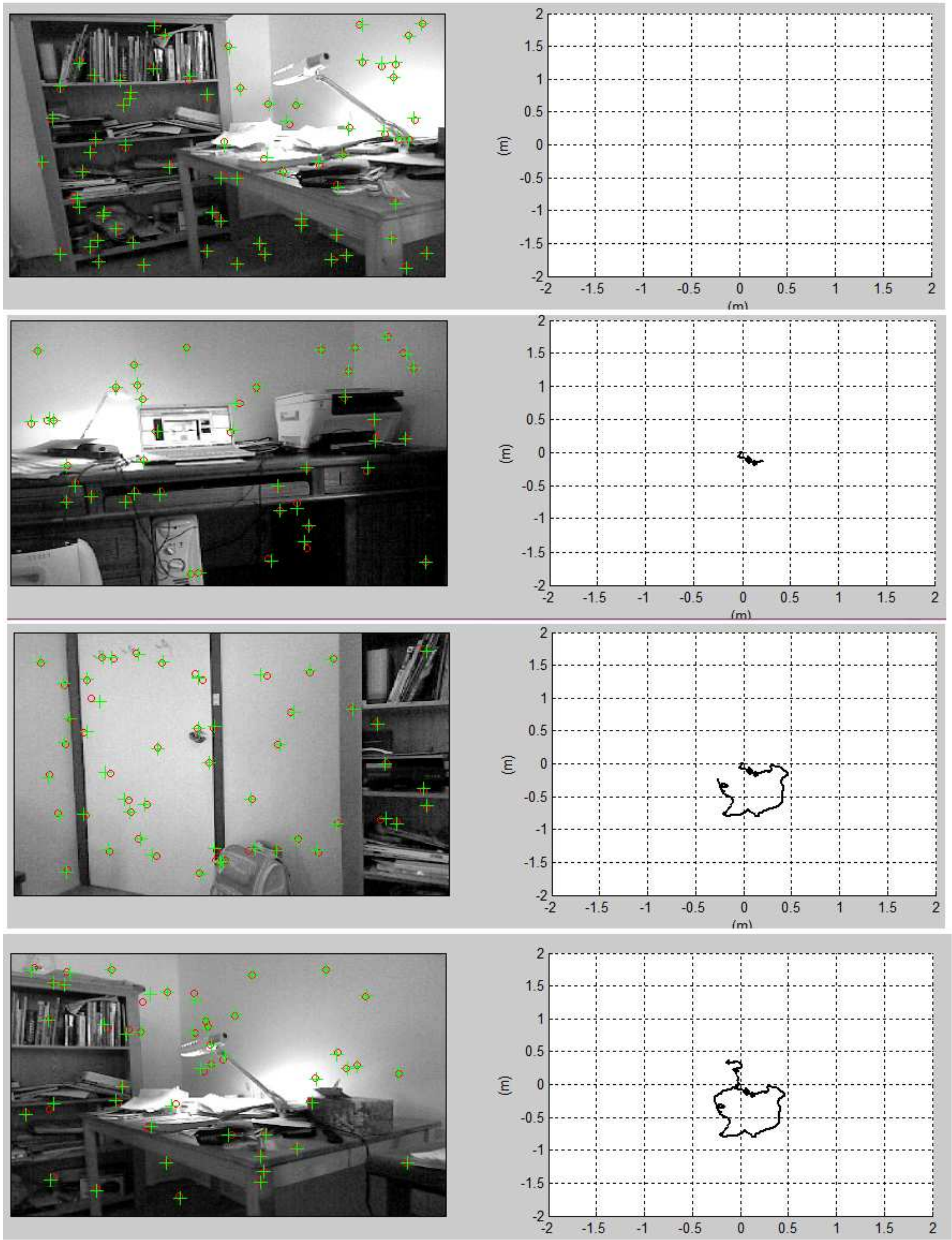


Figure 6.15: EKF-SLAM trajectory test

tested by EKF-SLAM and GMIEKF-SLAM; the blue line indicates the X, Y, Z errors of EKF-SLAM, whilst the red line shows the X, Y, Z errors of GMIEKF-SLAM. The

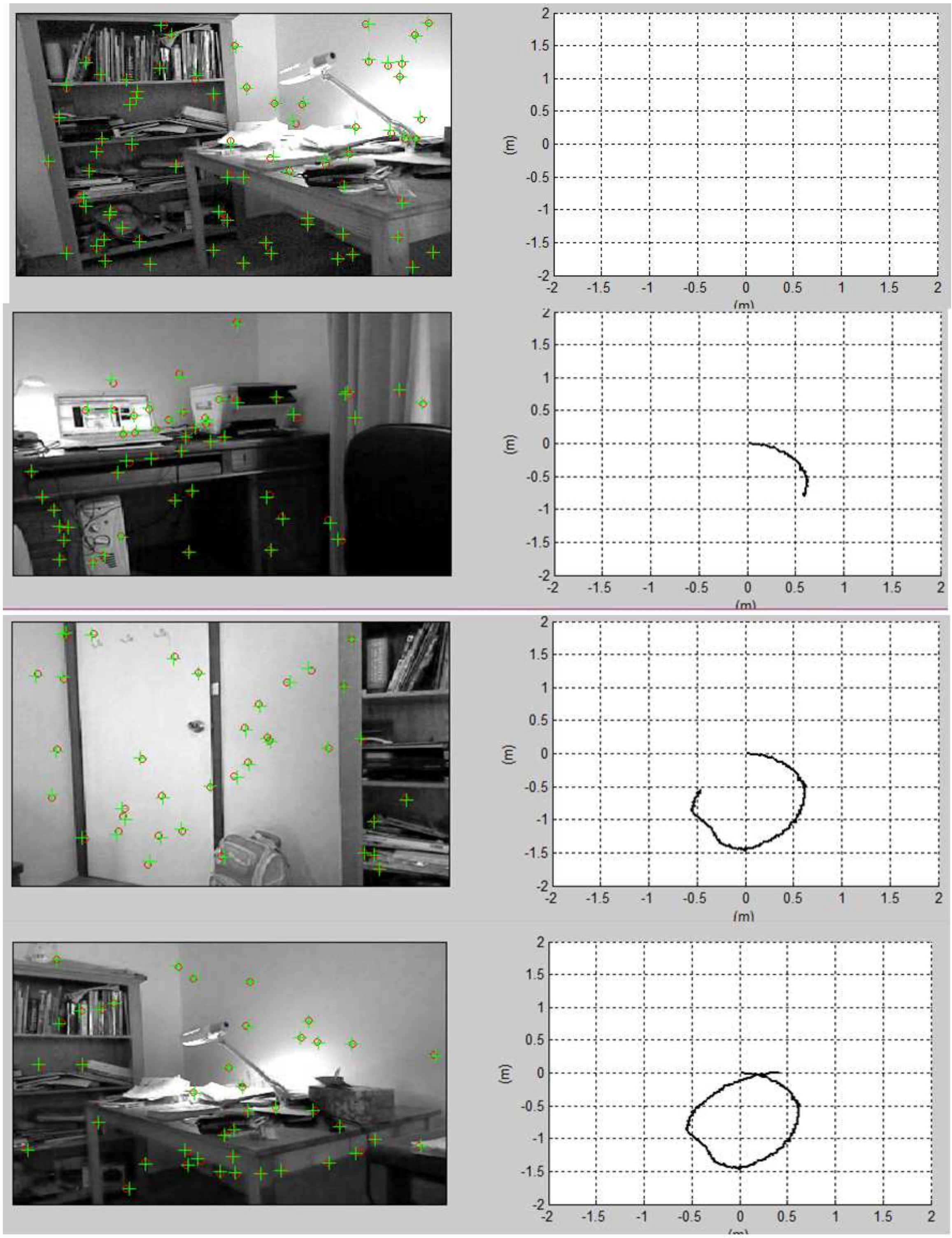


Figure 6.16: GMIEKF-SLAM trajectory test

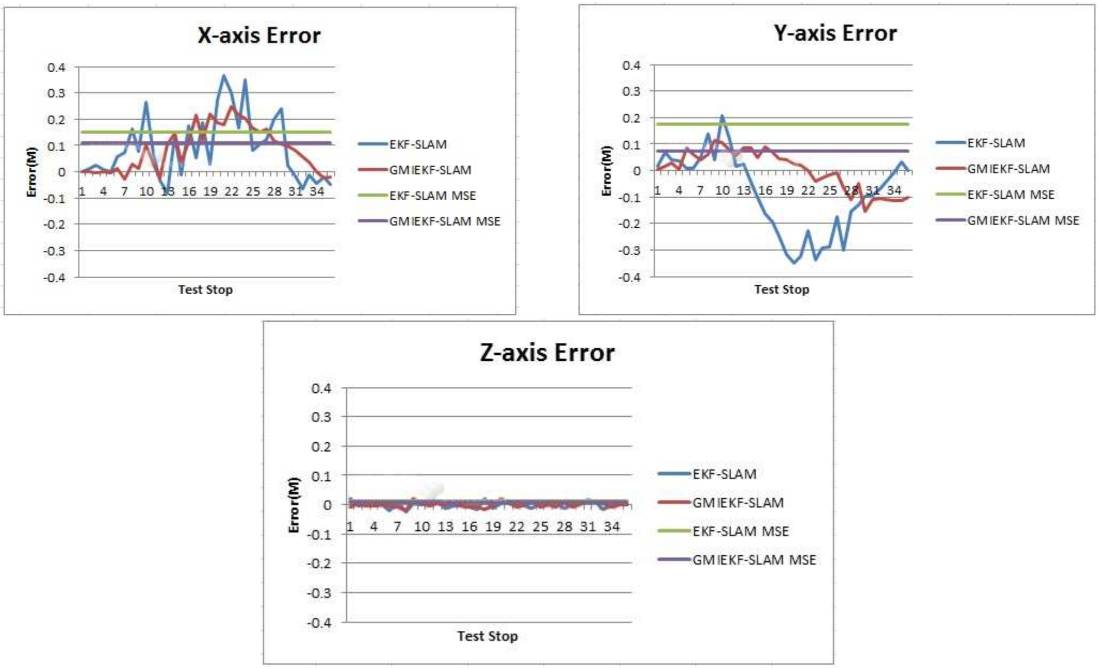


Figure 6.17: The X,Y,Z-error comparison (tuned)

Mean Squared Errors (MSE) of EKF-SLAM and GMIEKF-SLAM are shown by the green and purple lines, respectively. In Fig. 6.17, the X-axis error of EKF-SLAM propagates with increasing test stops, the Y-axis value has the largest error at the $20th$ sample point and the Z-axis error exhibits a small MSE. In Fig. 6.17, the X-axis error and Y-axis error of GMIEKF-SLAM both exhibit a smaller MSE compared to EKF-SLAM.

Tab. 6.1 displays the MSE and the processing time required for each image using EKF-SLAM and GMIEKF-SLAM. It can be seen from Tab. 6.1 that the GMIEKF-SLAM exhibits a lower MSE compared to the EKF-SLAM, which indicates more accurate state estimates by the proposed GMIEKF-SLAM approach. In EKF-SLAM, the camera motion is assumed as constant velocity and constant angular velocity with Gaussian acceleration. The camera motion is thus expected to be smooth and slow moving, where large acceleration or speed (sudden or fast motion) is problematic. Moreover, EKF-SLAM assumes that the processing noise is Gaussian and represents the state uncertainty by approximating the mean and variance. However, this approximation is an inadequate model for free camera movement, especially for nonlinear motion. Further, EKF-SLAM linearizes the system at each time step using only the *a priori* information and current measurements; however, the linearization can be poor and thus the EKF-SLAM estimations become erroneous. In contrast, the camera motion model is tested from the geometric camera pose estimation in GMIEKF-SLAM, which is ubiquitous for any unknown movement and any unknown environment. The non-linear least squares optimization of the measurement residual error allows the estimate to move away from the biased value, thus reducing the system error and uncertainty. However, whilst the GMIEKF-SLAM is more computationally complex with elapsed CPU time for GMIEKF-SLAM around 180ms for each time step (approximately fivefold the EKF-SLAM processing time), the GMIEKF-SLAM nonlinear optimization and iterated re-linearization processing is 5 frames/s.

6.4.2 Estimation Robustness Comparison

This second experiment compares the robustness of the EKF-SLAM and GMIEKF-SLAM by investigating the effect of tuning parameters on the process and measurement updates. Two experiments are conducted to investigate the tuning of process and measurement parameters, respectively.

Experiment 1: Changing the Process Parameters

Changing the process parameters investigates the effect of varying velocity and angular velocity on the trajectory estimates computed by EKF-SLAM and GMIEKF-SLAM. In this experiment, the velocity components of the translation and the angular

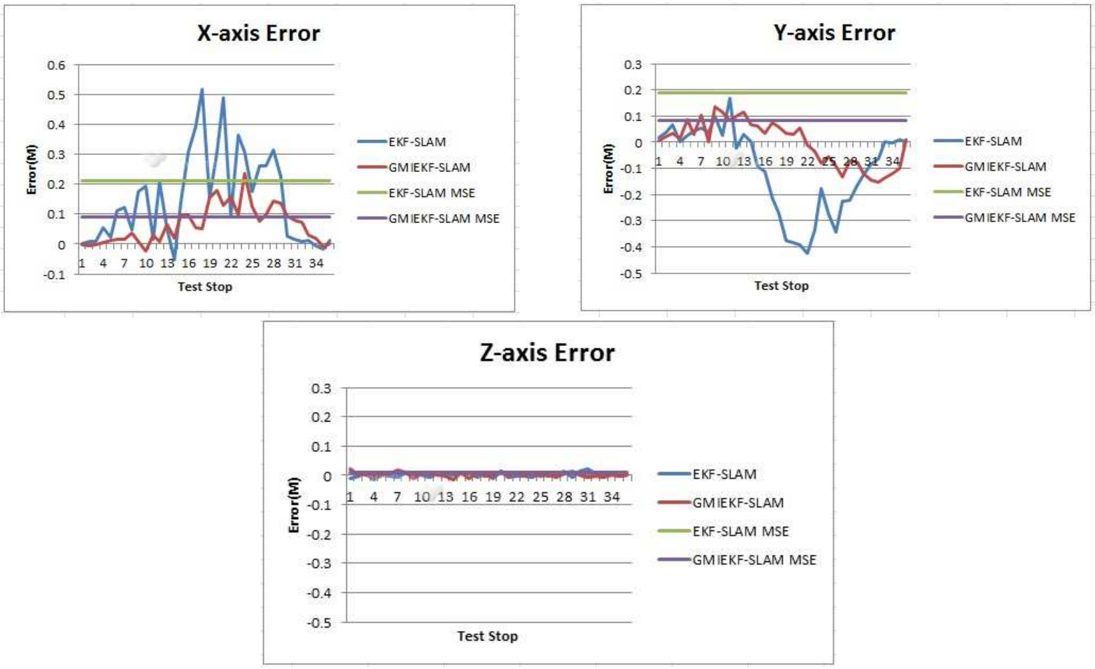
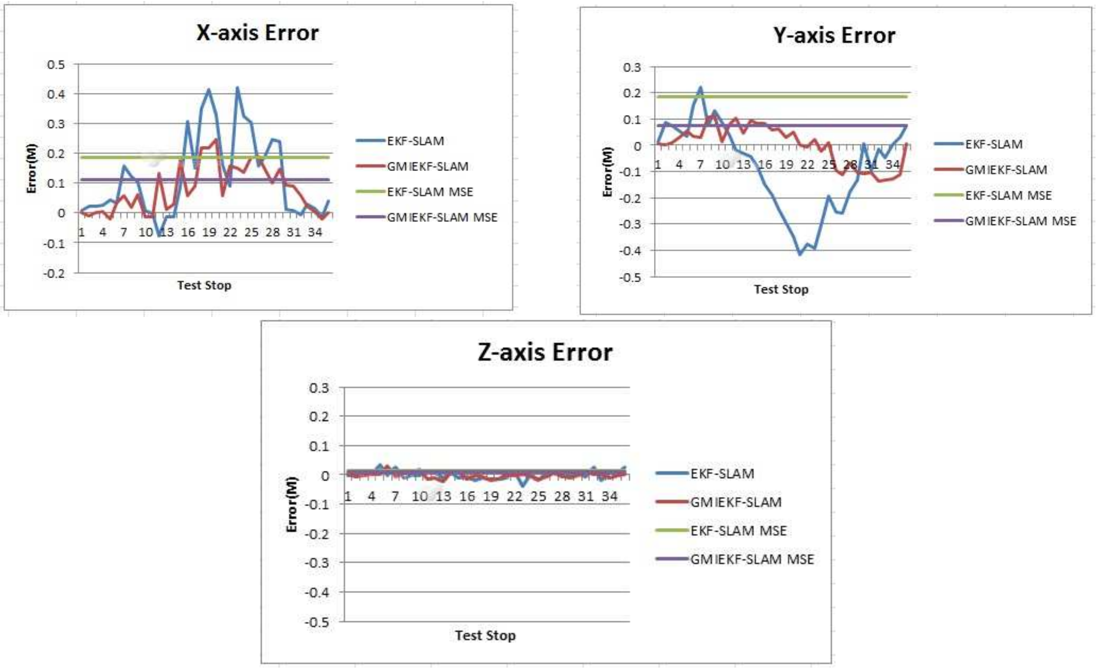


Figure 6.18: The X,Y,Z-error comparison ($\dot{t} = 0.03m/s$ and $\dot{\theta} = 0.18rad/s$)

velocity are changed to 0.03 m/s and 0.18 rad/s, respectively, to generate new motion. The MSE results shown in Fig. 6.18 indicate that the estimation accuracy of GMIEKF-SLAM is better than EKF-SLAM with lower MSE exhibited across the X, Y and Z axes. Comparing the two EKF-SLAM results in Fig. 6.17 and Fig. 6.18, it can be seen that the changes in the velocity lead to a divergence of EKF estimation, where the MSE of X-axis increases from 0.1487 m to 0.2104 m, and the Y-axis error increases from 0.173m to 0.1907m. In EKF-SLAM, the linearization depends on the nonlinear behaviour of the function f in Eq. (6.31) about the state estimate $\hat{s}_{k|k-1}$. In practice, only prior tuning of the state vector and covariance matrix can give consistent and reliable state estimates using EKF-SLAM. In contrast, GMIEKF-SLAM is able to adapt to velocity changes, where the MSE results in Fig. 6.18 are not significantly different from the tuned results reported in Fig. 6.17. The GMIEKF-SLAM uses the optimized state to model the behavior of the camera motion in the process function. When enough feature points (N >50 in practice) are


X-axis Error

Y-axis Error

Z-axis Error

Figure 6.19: The X,Y,Z-error comparison (image noise = 3 pixel)

taken, the state predictions $\hat{s}_{k|k-1}$ are consistent and accurate due to the covariance matrix reliably estimating the state errors in Eq. (6.21), with the predicted state a good approximation of the *a priori* state $\hat{s}_{k|k-1}$.

Experiment 2: Changing the Measurement Noise

This experiment compares the EKF-SLAM and GMIEKF-SLAM measurement update performance, where the measurement noise is increased from 1 pixel to 3 pixels whilst other parameters are kept constant. Fig. 6.19 shows the MSE obtained from EKF-SLAM and GMIEKF-SLAM and it can be seen that the GMIEKF-SLAM is more accurate with lower MSE in the trajectory estimation compared to the EKF-SLAM. In Fig. 6.19, the EKF-SLAM shows divergent behavior of the X, Y and Z MSE values, compared to the tuned results in Fig. 6.17. In contrast, in Fig. 6.19 the GMIEKF-SLAM only exhibits slight derivations in the XYZ MSE values compared to the tuned GMIEKF-SLAM results in Fig. 6.17. In the EKF-SLAM, the

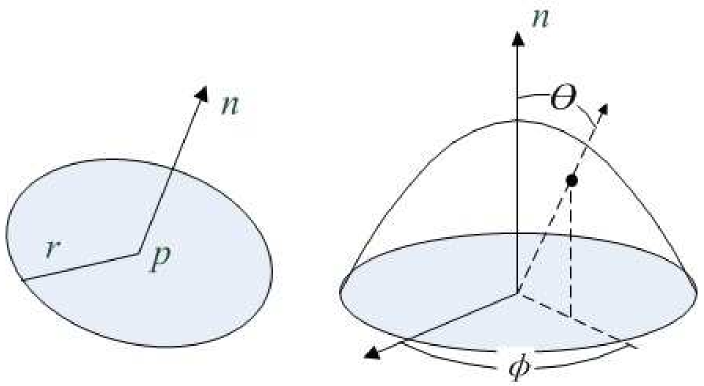


Figure 6.20: A surfel is described by its position p, normal n, radius r and visibility confidence

measurement model in Eq. (6.2) also involves nonlinear transformations; this results in an uncertain updated state estimate $\hat{s}_{k|k}$ around which the filter linearizes the measurement function. In EKF-SLAM, the linearization of $h(s_k)$ is expanded into a first order Taylor series, where the high order terms are ignored. In contrast, the GMIEKF-SLAM repeatedly calculates the intermediate posterior state estimate $\hat{s_k^i}$, which can reduce the error between estimated and measurement values.

6.4.3 Multiple View Stereo 3D Reconstruction

After obtaining the camera motion trajectory using the tuned parameters, this experiment utilises a surfel-based surface representation [187] to render complex geometric objects for 3D object reconstruction. The depth map is generated for the current model using the rigid transformation $\begin{bmatrix} R & t \end{bmatrix}$ and the camera intrinsic parameters. Each pixel is evaluated whether it is an inlier or outlier based on a maximum distance threshold on the absolute depth difference between virtual rendering and input scan. If the total ratio $\frac{outliers}{inliers+outliers} < 0.05$, the registration is successful.

Surfel Representation

Motivated by [187], this chapter utilises an explicit surface representation by representing the model surface as a set of surfels (surface elements). As shown in Fig. 6.20, the surfel has a position p, normal vector n, radius r and visibility confidence v. The unstructured set of surfels can be easily kept consistent throughout any modification compared to the triangle mesh, which needs considerable efforts when adding, updating or removing any vertices. Surfel visibility confidence consists of a polar angle θ and an azimuth angle Φ to indicate a view direction for each surfel. In order to estimate its reliability, each surfel is assigned a visibility confidence $v_i \in \left[0, \;\; 64\right]$. A surfel is assumed to be correct when its position is confirmed by several observations from different directions. In this chapter, a surfel has high visibility confidence if it has been observed in at least 6 bins.

Surfels are updated by integrating the depth measurements into the old scan. New surfels are created for parts of the scan that are not explained by the current model. Surfels that are not consistent and in conflict with the current scan are removed. In this experiment, surface merging based on surfel representation is performed using two operations: Surfel update and surfel addition. A surfel is updated when the new surfel satisfies three conditions: (1) if the depth of the surfel is valid, the re-projection of this surfel is in the range of the current image; (2) if the normal angle between the new and old camera principal axes is less than the pre-defined maximum angle of 60°, the surfel is considered as visible in the new image; (3) if two different surfels correspond to the same object then comparing the depth value of existing and new surfels, the one closer to the camera is considered as visible. Otherwise, if condition (1) cannot be satisfied, this surfel is omitted. If conditions (2) or (3) cannot be satisfied, this surfel is removed from the surface queue. After all existing surfels have been updated, surfels are added in the re-

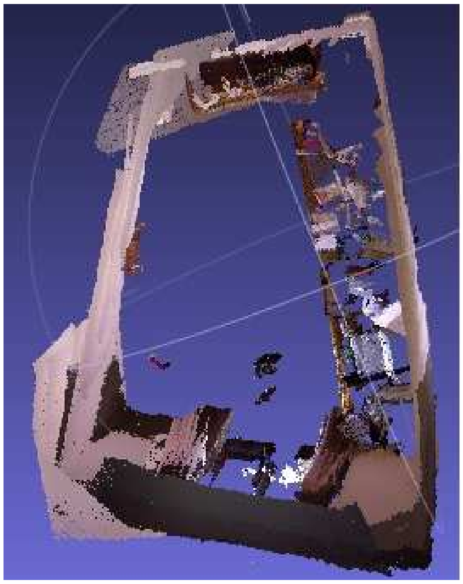


Figure 6.21: 3D reconstruction result of EKF-SLAM

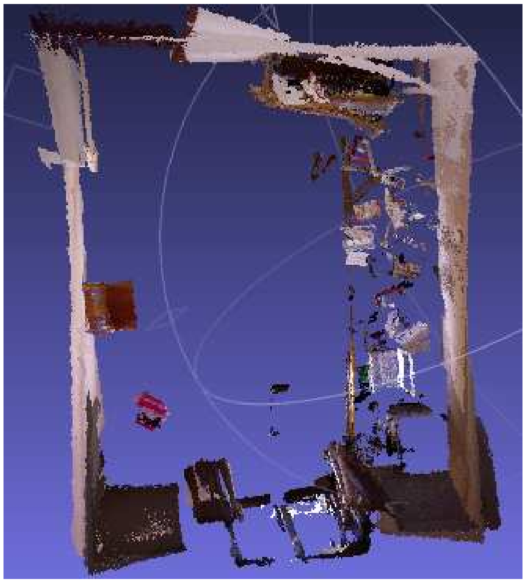


Figure 6.22: 3D reconstruction result of GMIEKF-SLAM

gions where the new depth map is not covered by existing surfels. After each surfel addition and update, the surfel visibility confidence histogram is updated accordingly.

The proposed method GMIEKF-SLAM has been evaluated on the multi-view reconstruction of an indoor scene and a small object. The camera motion estimation error directly impacts the 3D reconstruction result, as shown in the EKF-SLAM result of Fig. 6.21 and Fig. 6.23. The EKF linearizes the non-linear camera motion model and assumes Gaussian system noise, which cannot accurately estimate the true camera motion trajectory, thus in Fig. 6.21 and Fig. 6.23, the 3D scene shows reconstruction object overlaps when errors are accumulated. In contrast, the GMIEKF-SLAM benefits from geometric camera motion prediction and the re-linearization of the measurement model, thus the results show an improvement in estimation and reconstruction accuracy. The GMIEKF-SLAM results in Fig. 6.22 shows a complete reconstruction of the rectangle shape of the room in Fig. 6.22, and the toy in Fig. 6.24 is more accurately reconstructed compared to the EKF-SLAM results in Fig. 6.23.

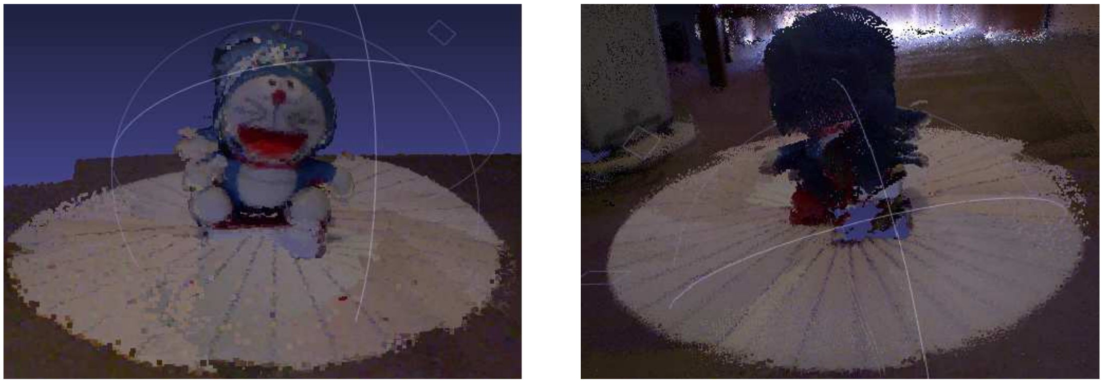


Figure 6.23: 3D Reconstruction Result of EKF-SLAM

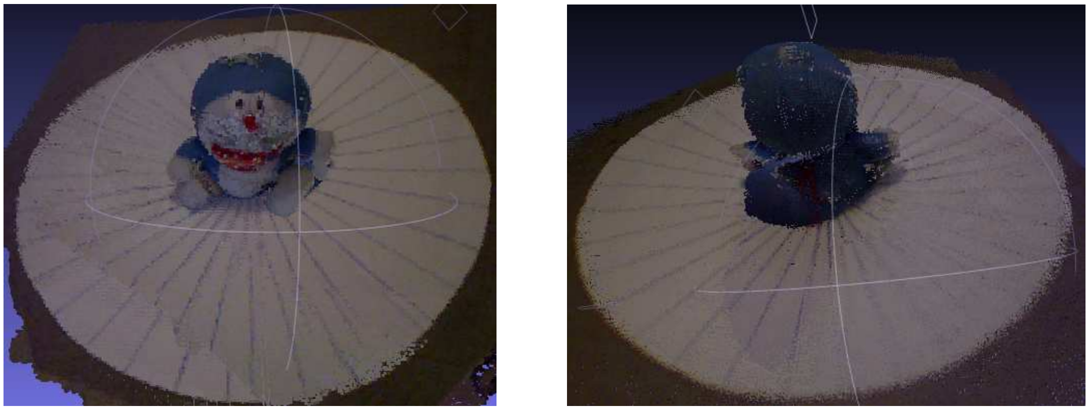


Figure 6.24: The 3D Reconstruction Result of GMIEKF-SLAM

6.5 Conclusion

To improve upon the speed performance of the SfM algorithm proposed in Chapter 5, this chapter proposed the Geometric Modelling Iterated EKF-SLAM (GMIEKF-SLAM) algorithm for real-time (5 frame/s) accurate and robust localization and mapping with RGB-D data from a Kinect camera. It has been shown that mechanism of geometrical camera pose estimation and iterative measurement linearization can be integrated into a novel GMIEKF-SLAM algorithm. The geometric modelling method for dynamic camera motion modelling fundamentally avoids the linear assumption errors of the camera motion model, and is an ubiquitous solution for any unknown camera motion and unknown environment. Compared to the traditional EKF-SLAM approach, experimental results show that GMIEKF-SLAM can improve the camera motion estimation performance in the presence of *a priori* prediction statistics and alleviate the linearization error by iterating the measurement estimation around the update state. In particular, the robustness of GMIEKF-SLAM has been exhibited through experiments that vary the process parameters and measurement noise and apply GMIEKF-SLAM to the 3D reconstruction of a real indoor room environment and small object; experimental results demonstrated that GMIEKF-SLAM provided an improved estimation and reconstruction accuracy compared to EKF-SLAM. However, the improvements obtained by using GMIEKF-SLAM are at a computational cost, with the current GMIEKF-SLAM processing rate around 5 frame/s. An efficient implementation taking advantage of modern GPU hardware could be developed to reduce the processing time for GMIEKF-SLAM, and immediate future work (detailed in Chapter 7) will test more complicated environments including 3D reconstruction of outdoor environments, scenes with significant occlusions and more varying motions to further evaluate the accuracy and robustness of GMIEKF in realistic conditions.

Chapter 7

Conclusion and Future Work

This thesis has demonstrated novel 3D reconstruction methods for efficient and high quality reconstructions. In this thesis, the algorithms presented include eight possible solution for camera motion calibration, handedness constraint in orientation of camera projection, dense 3D reconstruction using SfM and real-time dense 3D reconstruction using SLAM via a RGB-D camera e.g., Microsoft Kinect. Detailed analysis and comparisons with existing algorithms have shown that the proposed techniques produce highly competitive results; where the algorithms presented in this thesis may be applied to efficiently solve 3D reconstruction problems and computer vision problems.

7.1 Thesis Contribution Highlights

Given a set of uncalibrated images or video sequence of a scene, this thesis proposes techniques to reconstruct the 3D locations of points in a scene from multiple views through geometrical constraints in the images. In camera self-calibration, the feature correspondences address the matching of points or features between two or more images such that the matched points correspond to the same 3D point in the observed scene. The geometrical relationship is then expressed with the fundamental matrix, which can be estimated from the feature correspondences between these views. With more than three images, the fundamental matrix computed from the point correspondences is sufficient for recovering the camera intrinsic parameters and camera motion. In SfM reconstruction, the camera projective matrix is expressed by the intrinsic matrix and camera motion, and triangulation then applies projective geometry to determine the 3D location by using the position of two fixed points to a known distance apart. To move from two to multi-view geometry, tracks are calculated separately for overlapping feature correspondences of all the views in the sequence. In real-time dense reconstruction, a SLAM algorithm has been proposed to estimate the motion of a moving camera and 3D localization of its surroundings.

This thesis has focused on the following three main areas of work:

- Chapter 3 revisited the cheirality problem in oriented projective geometry, proposing and showing the root cause of cheirality to be a handedness problem in camera motion estimation. Starting from 3D projective geometry in Euclidean coordinates and homogeneous coordinates, Chapter 3 developed a 4D rotation visualization by treating time as movement and examining snapshots of the 4D model at various points in time. The handedness problem within the cheirality of points X and $\neg X$ can be easily and directly analyzed under the 4D simulator. This thesis extended existing SVD solutions of the essential matrix that focused on geometrically static scenes to dynamic continuous camera motion and proposed eight possible solutions of camera motion from two views with geometrical analyses and derivation. The eight possible solutions convey the continuous camera orientation information between successive frames. Subsequently, the cheirality problem can be resolved by confining all rotations in multiple views to the right-hand rule, equivalent to applying the $det(R) = 1$

constraint. For a projective reconstruction, the same cheirality problem exists in the Singular Value Decomposition(SVD) of essential matrix for camera motion estimation, and the Direct Linear Transformation (DLT) of the projection matrix, and projective transformation. This ability to resolve the long-standing ambiguity inherent to the cheirality of camera projection can thus be applied to the development of camera tracking, augmented reality, structure from motion and 3D modeling applications in computer vision, as explored in Chapters 5 and 6. An OpenGL Camera motion simulator to demonstrate and verify proposed camera motion calibration and eight solutions from the essential matrix was presented in Chapter 4.

- A high density approach to surface reconstruction from a sequence of uncalibrated images based on SfM was proposed in Chapter 5. The main contributions of this work were threefold: First, Chapter 5 introduced a new region growing algorithm to increase the feature points density to overcome the sparseness of the points of interest. Second, the proposed SfM method presented a flexible automatic methodology with the simple one-stop interface of 'videos to 3D model'. The proposed approach works for largely separated images and requires fewer images than the standard approach, producing a high density of points that can be used for direct surface reconstruction. Third, new surface reconstruction algorithms in the proposed SfM approach integrate both 3D data points and 2D images: The new cost functions based on the re-projection error have far fewer minima than those derived from 2D data alone to result in more stable and more efficient reconstruction results.

- Chapter 6 presented the Geometric Modelling Iterated EKF-SLAM (GMIEKF-SLAM) algorithm for real-time, accurate and robust localization and mapping with RGB-D data from a Kinect camera. Geometrical camera pose estimation and iterative measurement linearization is integrated into the novel GMIEKF-SLAM algorithm, where the geometric modelling method for dynamic camera motion modelling fundamentally avoids the linear assumption errors of the camera motion model, and is an ubiquitous solution for any unknown camera motion and unknown environment. The proposed GMIEKF-SLAM algorithm can improve the camera motion estimation performance in the presence of *a priori* prediction statistics and alleviate linearization error by iterating the measurement estimation around the update state.

7.2 Further Work

Extending on the work presented in this thesis, future work on dense real-time 3D reconstruction could extend beyond small scale and indoor environments, to test on large-scale environments e.g., 3D reconstruction of outdoor environments. For large scale scene reconstruction, one key challenge is computational complexity. As mentioned in Chapter 5, a key problem of bundle adjustment is the significant computational time required for large-scale problems. An efficient implementation taking advantage of modern GPU hardware could be developed to reduce the processing time. In Wang's multiple core solution [53], the CPU based system is up to 10 times faster whilst the GPU system is up to 30 times faster than typical single-core implementations. Extended implementations could also utilise low level libraries such as Basic Linear Algebra Subprograms (BLAS) [188] and Linear Algebra PACKage (LAPACK) [189], which have already been optimized for parallel hardware implementation. Immediate future work could aim to generate solutions for the GPU SfM reconstruction of 3D objects on mobile devices.

Robustness is another challenge with large-scale 3D reconstruction applications. The effectiveness of the SLAM approach is due to fully correlated posterior estimation over feature points and camera positions. However, the SLAM algorithm suffers from computational complexity and incorrect data association problems. Thus, real-time SLAM implementation in an increasingly unstructured large-scale outdoor environment poses a number of challenges; for example, the loop-closure problem, where the camera revisits previous positions during a large traversal, is especially difficult to resolve. Some approaches focused on resolving the loop closure problem: Newman et al. [190] use salient visual image features to detect possible loop closure events. FastSLAM [191] applies the Rao-Blackwellized Particle Filter (RBPF) in combination with scan matching; the RBPF represents the probability distribution over all possible trajectories and is therefore capable of closing the loop. However, since the dimensionality of the trajectory grows over time, the number of particles increases exponentially to avoid eliminating the potential for loop closure. Clemente et al. [57] used a combination of geometric compatibility and random sampling in monocular SLAM to perform map-to-map matching to detect loop closure. Correspondences were then found between landmarks common to the two submaps using both their visual appearance and their geometry.

Inspired by [57], this thesis could investigate a submap SLAM algorithm that combines the advantage of SfM and SLAM to achieve large-scale environment reconstruction. First, a sequence of local maps of limited size are built independently into submaps, then these submaps are registered using the SfM technique proposed in Chapter 5. The advantage of SfM is in the recovery of the camera pose for unorganized images, while SLAM is accurate and robust for local maps reconstruction. Further, two issues inherent to EKF-based SLAM algorithms as mentioned in Chapter 6 are: First, the processing time associated with the EKF update is

O(n2) in the number of map features; second, the accumulative linearization errors in the EKF ultimately contribute to biased state estimates. The proposed sub-maps technique can overcome both issues: First, by segmenting the problem into smaller sub-maps, the computational time of the filter is bounded; second, since each local map effectively resets the base frame, linearization errors only accumulate within a local map and not between maps.

Although the GMIEKF algorithm presented in Chapter 6 proposes repetitive linearization of the nonlinear measurement model to provide a running estimate of camera motion, EKF-based SLAM is still difficult to implement and tune. A new linear filter, the Unscented Kalman Filter (UKF) proposed by Julier [143], generalizes the Kalman filter for nonlinear systems by transforming approximations of the probability distributions through nonlinear process and measurement functions. The UKF linearizes process and measurement functions by statistical linear regression of the functions through sampling points in the uncertainty region around the state estimate. Such sampling points are then propagated through the non-linear functions, from which the mean and covariance of the estimate are then recovered. The UKF thus more accurately captures the true mean and covariance compared to the EKF, and potential future work in the monocular SLAM algorithm lies the extension of GMIEKF to GMIUKF. Extending the GMIUKF SLAM framework into nonlinear systems can therefore more precisely estimate the mean and covariance of a continuous nonlinear transformation.

Finally, motivated by research into 4D space-time frameworks for oriented projective geometry, as the problem of 3D reconstruction from multi-view images matures, future research could address in the area of 4D reconstruction from multi-view image sequences; the current advancements in 3D reconstruction from multi-view images will become the foundation for future research into 4D reconstructions.

Appendix A

Geometric Representation of Eight Possible Solutions

Fig. A.1 illustrates the transformation of the world coordinate system to the camera coordinate system, where the point O is expressed in the world coordinate system whilst camera centre C exists in the camera coordinate system. The two coordinate systems are related via a geometric transform composed of rotation R and translation t.

Camera centre The camera centre is a column vector $C = \begin{bmatrix} a & b & c & \gamma \end{bmatrix}^T$, defined by $PC = 0$, such that

$$a = det(\begin{bmatrix} p_2 & p_3 & p_4 \end{bmatrix}), b = -det(\begin{bmatrix} p_1 & p_3 & p_4 \end{bmatrix}) \tag{A.1}$$

$$c = det(\begin{bmatrix} p_1 & p_2 & p_4 \end{bmatrix}), \gamma = -det(\begin{bmatrix} p_1 & p_2 & p_3 \end{bmatrix}), \tag{A.2}$$

where p_i is the i-th column of P.

Principal point The principal point (shown as p in Fig. A.1) is the intersection of the image plane with the principal axis, which passes through the camera centre

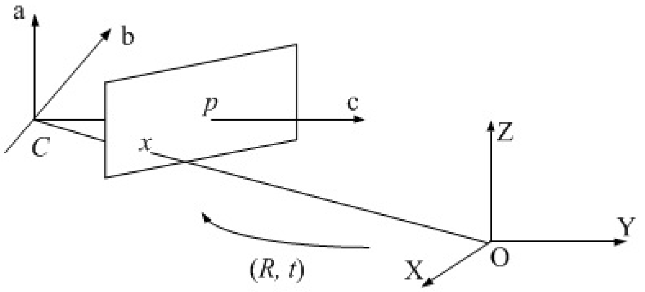


Figure A.1: Transformation between world and camera coordinate systems

Table A.1: Key model elements with eight combinations of (R, t)

R	t	Camera centre	Reconstructed point	Image point
R_1	t	$\begin{bmatrix}a & b & c & \gamma\end{bmatrix}^T$	$\begin{bmatrix}A_{x1} & A_{y1} & A_{z1} & A_{w1}\end{bmatrix}^T$	x_1
R_1	$-t$	$\begin{bmatrix}-a & -b & -c & \gamma\end{bmatrix}^T$	$\begin{bmatrix}A_{x1} & A_{y1} & A_{z1} & -A_{w1}\end{bmatrix}^T$	x_1
R_2	t	$\begin{bmatrix}a & b & c & -\gamma\end{bmatrix}^T$	$\begin{bmatrix}A_{x2} & A_{y2} & A_{z2} & A_{w2}\end{bmatrix}^T$	x_2
R_2	$-t$	$\begin{bmatrix}-a & -b & -c & -\gamma\end{bmatrix}^T$	$\begin{bmatrix}A_{x2} & A_{y2} & A_{z2} & -A_{w2}\end{bmatrix}^T$	x_2
$-R_1$	t	$\begin{bmatrix}a & b & c & -\gamma\end{bmatrix}^T$	$\begin{bmatrix}-A_{x1} & -A_{y1} & -A_{z1} & -A_{w1}\end{bmatrix}^T$	x_1
$-R_1$	$-t$	$\begin{bmatrix}-a & -b & -c & -\gamma\end{bmatrix}^T$	$\begin{bmatrix}-A_{x1} & -A_{y1} & -A_{z1} & -A_{w1}\end{bmatrix}^T$	x_1
$-R_2$	t	$\begin{bmatrix}a & b & c & \gamma\end{bmatrix}^T$	$\begin{bmatrix}-A_{x2} & -A_{y2} & -A_{z2} & A_{w2}\end{bmatrix}^T$	x_2
$-R_2$	$-t$	$\begin{bmatrix}-a & -b & -c & \gamma\end{bmatrix}^T$	$\begin{bmatrix}-A_{x2} & -A_{y2} & -A_{z2} & -A_{w2}\end{bmatrix}^T$	x_2

and is perpendicular to the image plane. The principal point can be computed by $x_0 = Mm_3$, where M is the leftmost 3×3 block of P, m_3^T is the third row of M and the principal point remains constant in the eight possible essential matrix solutions.

Image points The image points are obtained according to $x = PX$.

Projective reconstructed points The point X is reconstructed by the linear triangulation method:

$$BX = 0, B = \begin{bmatrix} xP^{3T} - P^{1T} \\ yP^{3T} - P^{2T} \\ x'P'^{3T} - P'^{1T} \\ y'P'^{3T} - P'^{2T} \end{bmatrix} \tag{A.3}$$

The eight possible combinations of transform (R, t) and the relative key elements are thus computed, where the possible element combinations are outlined in Table A.1. The camera centres thus have four possibilities:

$$C_1 = C_{\begin{bmatrix} R_1|t \end{bmatrix}} = C_{\begin{bmatrix} -R_2|t \end{bmatrix}} = \begin{bmatrix} a & b & c & \gamma \end{bmatrix} \tag{A.4}$$

$$\neg C_2 = C_{\begin{bmatrix} R_1|-t \end{bmatrix}} = C_{\begin{bmatrix} -R_2|-t \end{bmatrix}} = \begin{bmatrix} -a & -b & -c & \gamma \end{bmatrix} \tag{A.5}$$

$$C_2 = C_{\begin{bmatrix} R_2|t \end{bmatrix}} = C_{\begin{bmatrix} -R_1|t \end{bmatrix}} = \begin{bmatrix} a & b & c & -\gamma \end{bmatrix} \tag{A.6}$$

$$\neg C_1 = C_{\begin{bmatrix} R_2|-t \end{bmatrix}} = C_{\begin{bmatrix} -R_1|-t \end{bmatrix}} = \begin{bmatrix} -a & -b & -c & -\gamma \end{bmatrix} \tag{A.7}$$

There are therefore eight possible positions for the reconstructed point X:

$$X_1 = \begin{bmatrix} A_{x1} & A_{y1} & A_{z1} & A_{w1} \end{bmatrix}^T \tag{A.8}$$

$$X_2 = \begin{bmatrix} A_{x2} & A_{y2} & A_{z2} & A_{w2} \end{bmatrix}^T \tag{A.9}$$

$$X_3 = \begin{bmatrix} A_{x1} & A_{y1} & A_{z1} & -A_{w1} \end{bmatrix}^T \tag{A.10}$$

$$X_4 = \begin{bmatrix} A_{x2} & A_{y2} & A_{z2} & -A_{w2} \end{bmatrix}^T \tag{A.11}$$

$$\neg X_1 = \begin{bmatrix} -A_{x1}, -A_{y1}, -A_{z1}, -A_{w1} \end{bmatrix}^T \tag{A.12}$$

$$\neg X_2 = \begin{bmatrix} -A_{x2}, -A_{y2}, -A_{z2}, -A_{w2} \end{bmatrix}^T \tag{A.13}$$

$$\neg X_3 = \begin{bmatrix} -A_{x1}, -A_{y1}, -A_{z1}, A_{w1} \end{bmatrix}^T \tag{A.14}$$

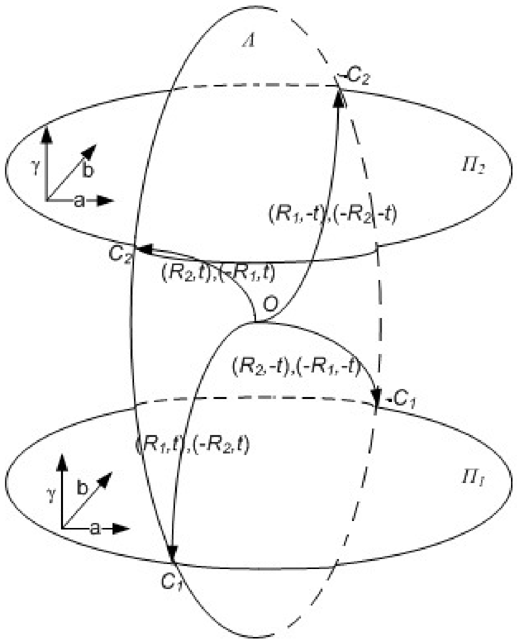


Figure A.2: Geometric model depicting the relationship between the camera centre, R and t.

$$\neg X_4 = \Big[-A_{x2}, -A_{y2}, -A_{z2}, A_{w2} \Big]^T \tag{A.15}$$

X_1 and X_3 only differ by the sign of T_1 in Table A.1. Similarly, X_2 and X_4 differ only by the sign of T_2. There are only two image points: x_1 corresponds to the value with R_1 regardless of the sign of R_1 and t. Similarly, x_2 corresponds to the value with R_2 irrespective of the sign of R_2 and t.

The geometric model of Fig. A.2 illustrates the eight possible transformations from the world coordinate system to the camera coordinate system. The 4D space of Table A.1 is reduced to 3D since the signs of (a, b, c) change in unison whilst the sign of γ changes independently; thus, the geometric model used in Fig. A.2 is shown using the axes (a, b, γ). In Fig. A.2, the camera centres $+C_1$ and $\neg C_1$ are shown in the Π_1 plane whilst the camera centres $+C_2$ and $\neg C_2$ are shown in the Π_2 plane, where C_1 and C_2 differ only in the sign of γ. The object point O in the world coordinate system is the centre of the object plane Λ, with radius equal to the norm of t (the sign of t indicates the different translation directions). There are

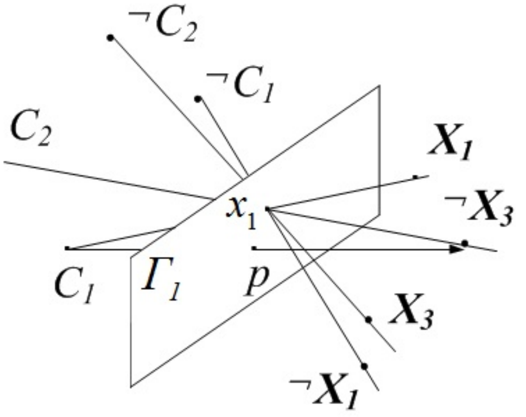


Figure A.3: Four solutions from R_1

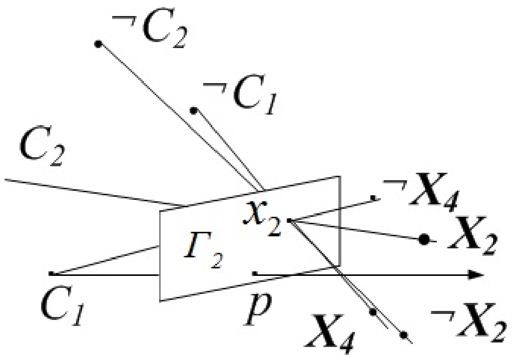


Figure A.4: Four solutions from R_2

four relative positions of the camera centres $+C_1$, $\neg C_1$, $+C_2$ and $\neg C_2$ resulting from the eight possible transformations of (R, t), however, only four transformations are unique. For example, in Table A.1 and Fig. A.2, for the solution of (R_1, t), the object point O translates t and rotates R_1 to C_1, while for the solution $(-R_2, t)$, O translates t and also rotates $-R_2$ to C_1. Thus, of the four possible camera centres $+C_1$, $\neg C_1$, $+C_2$ and $\neg C_2$, only one camera centre is correct and the object point O should be located in front of or facing that camera centre.

In the results shown in Table A.1, there is one principal point consistent across all transformations and two image points, x_1 and x_2. Figs. A.3 and A.4 show that the two image planes Γ_1 and Γ_2 differ in orientation but intersect at the common principal point p. On the image plane Γ_1, the four possible reconstructed points intersect at x_1 and relate to R_1. In contrast, the other four possible reconstructed points related to R_2 intersect at x_2 on the plane Γ_2. Fig. A.3 shows that when the rotation matrix equals R_1, the centre C_1 has two positive and negative solutions that correspond to the reconstructed points $+X_1$ and $\neg X_1$. Similarly, centre C_2 also has two positive and negative solutions that correspond to $+X_3$ and $\neg X_3$. Fig. A.4 illustrates that when the rotation matrix equals R_2, the centres $+C_1$ and $\neg C_1$ correspond to different

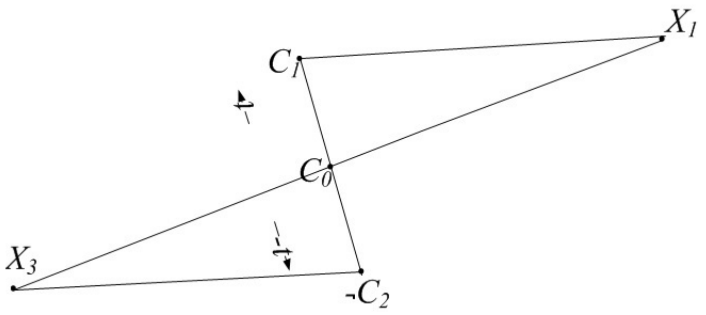


Figure A.5: In (R_1, t) and $(R_1, -t)$, the camera moves t and -t to camera centre C_1 and $\neg C_2$. The two reconstruction points X_1 and X_3 are symmetric to the baseline

reconstructed points $+X_4$ and $\neg X_4$, and centres $+C_2$ and $\neg C_2$ correspond to the points $+X_2$ and $\neg X_2$.

A.1 The Relationship Between the Solutions (R_1, t) and $(R_1, -t)$

Observing the two solutions of (R_1, t) and $(R_1, -t)$, the difference is only in the direction in which the translation vector is reversed. In Tab. A.1, the camera centre has two corresponding positions, C_1 and C_2, and the reconstruction points from the common image point x_1 are located at X_1 and X_3. The origin of the coordinate system is C_0. In Fig. A.5, (R_1, t) motion denotes camera movement from C_0 to C_1 with translation t, and the two projective images at C_0 and C_1 have the image point x_1 and reconstructed point X_1. $(R_1, -t)$ motion moves the camera from C_0 to $\neg C_2$ with translation $-t$, and the two projective images at C_0 and $\neg C_2$ have the image point x_1 and reconstructed point X_3. In the two solutions of (R_1, t) and $(R_1, -t)$, the two different camera centres with the same principal axis direction are symmetrically located on the two sides of C_0, and the two different reconstruction points are symmetrically located about the baseline. The relationship between the

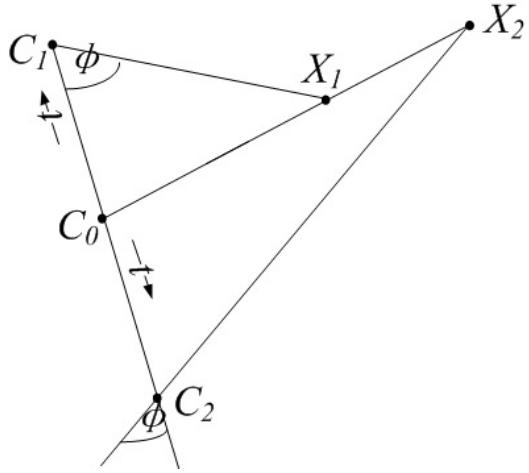


Figure A.6: In (R_1, t) and (R_2, t), The camera moves t to camera centre C_1 and C_2, the projective ray C_1X_1 and C_2X_2 are symmetric about the baseline.

solutions (R_2, t) and $(R_2, -t)$ exhibits similar behaviour to (R_1, t) and $(R_1, -t)$.

A.2 The Relationship Between the Solutions (R_1, t) and (R_2, t)

Observing the two solutions of (R_1, t) and (R_2, t), R_1 and R_2 are known as a "twisted pair" [29], and are symmetric about the baseline. As shown in Fig. A.6, with the origin of the coordinate system at $C_0 = (a_0, b_0, c_0, \gamma_0)$, the solution of (R_1, t) rotates R_1 from coordinates C_0 to coordinates $C_1 = (a, b, c, \gamma)$, while in the solution (R_2, t), coordinates $C_2 = (a, b, c, -\gamma)$ are formed by coordinates C_0 rotating with R_2. In Fig. A.6, (R_1, t) motion denotes the camera movement from C_0 to C_1 with translation t, and the two projective images at C_0 and C_1 have the image point x_1 and reconstructed point X_1. (R_2, t) motion moves the camera from C_0 to C_2 with translation t. Since these two solutions are symmetric about the baseline, the projective ray from C_2 is symmetrical with the projective ray of C_1 about the baseline. The two projective images at C_0 and C_2 thus have the image point x_2 and

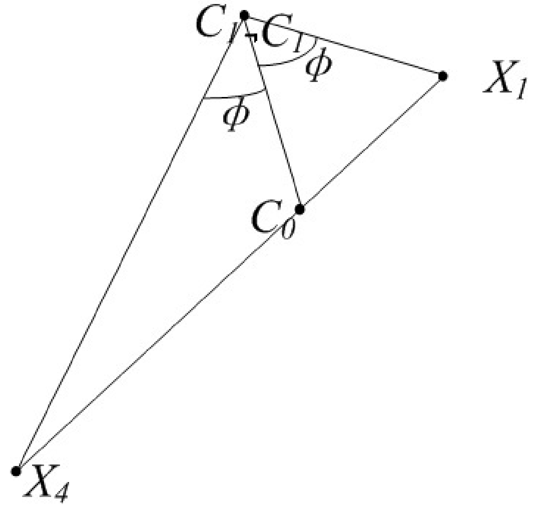


Figure A.7: In (R_1, t) and $(R_2, -t)$, the camera moves t and $-t$ to camera centres C_1 and $\neg C_1$. The projective ray C_1X_1 and $\neg C_1X_4$ are symmetrical around the baseline.

reconstructed point X_2. In the two solutions of (R_1, t) and (R_2, t), the principal axes of the two camera centres C_1 and C_2 are symmetric about the baseline and the two reconstructed points are independent. The relationship between the solutions $(-R_1, -t)$ and $(-R_2, -t)$ exhibits similar behaviour to (R_1, t) and (R_2, t).

A.3 The Relationship Between the Solutions (R_1, t) and $(R_2, -t)$

Observing the two solutions of (R_1, t) and $(R_2, -t)$, the rotation relationship is the same for (R_1, t) and (R_2, t). In Fig. A.7, (R_1, t) motion moves the camera from C_0 to C_1 with translation t, and two projective images from C_0 and C_1 have the image point x_1 and reconstructed point X_1. $(R_2, -t)$ motion moves camera from C_0 to $\neg C_1$ with $-t$. In Euclidean coordinates, C_1 and $\neg C_1$ are the same points but their principal axis is symmetric about the baseline; thus, two different 3D space points are reconstructed at X_1 and X_4. In the two solutions of (R_1, t) and $(R_2, -t)$, the two

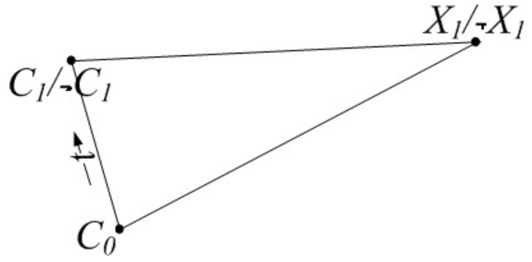


Figure A.8: In (R_1, t) and $(-R_1, -t)$, the two camera centres superpose each other, the principal axis and the 3D reconstruction points are also superposed.

camera centres C_1 and $\neg C_1$ therefore have the same Euclidean coordinates. Their principal axes are symmetric around the baseline and the two reconstructed points X_1 and X_4 are independent. The relationship between the solutions $(-R_1, -t)$ and $(-R_2, t)$ exhibits similar behaviour.

A.4 The Relationship Between the Solutions (R_1, t) and $(-R_1, -t)$

Observing the two solutions of (R_1, t) and $(-R_1, -t)$, the camera centre has two corresponding positions: C_1 and $\neg C_1$. As shown in Fig. A.8, (R_1, t) and $(-R_1, -t)$ are the same points in Euclidean coordinates, with the same principal axis and reconstruction points. Previous work has considered the solution $(-R_1, -t)$ to be the same as (R_1, t), discarding the solution of $(-R_1, -t)$ and the other three sign reversed solutions for $(R_1, -t)$, (R_2, t), $(R_2, -t)$ [152]. The four solutions of the essential matrix in Eq. (3.26) for relative orientation are usually achieved without considering the differences between Euclidean and homogeneous coordinates. The next section demonstrates that the relationship between the solutions (R_1, t) and $(-R_1, -t)$ is due to the handedness of the two sign-reversed rotations.

Bibliography

[1] M. K. Chandraker, "From Pictures to 3D: Global Optimization for Scene Reconstruction," *University of California, San Diego*, 2009.

[2] Z.Zhang, "Microsoft Kinect Sensor and Its Effect," *IEEE MultiMedia, Vol. 19, No. 2*, 2012.

[3] http://en.wikipedia.org/wiki/3Dreconstruction.

[4] M.Goesele, "Multi-view Stereo for Community Photo Collections," 2007.

[5] N. Snavely and S. Seitz, "Photo Tourism: Exploring Image Collections in 3D. ACM Transactions on Graphics," *ACM Transactions on Graphics*, 2006.

[6] X. Li, "Modeling and Recognition of Landmark Image Collections Using Iconic Scene Graphs," *ECCV*, 2008.

[7] S. M. S. N. Snavely, "Skeletal Graphs for Efficient Structure from Motion," *CVPR*, 2008.

[8] S. Agarwal, "Building Rome in a Day," *International Conference on Computer Vision, 2009*, 2009.

[9] http://www.google.com/earth/index.html, 2012.

[10] O. D. Cooper, "Robust Generation of 3D Models from Video Footage of Urban Scenes," *PHD thesis*, 2005.

[11] M. A. Fischler, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *comm. of the ACM*, 1981.

[12] H. C. Longuet-Higgins, "The Reconstruction of a Scene from Two Projections," *Nature*, vol. 293, no. 10, p. 3, 1981.

[13] B.K.P.Horn, "Relative Orientation," *International Journal of Computer Vision*, vol. 4, 1990.

[14] ——, "Relative Orientation Revisited," *Journal of the Optical Society of America A*, vol. 8, p. 9, 1991.

[15] S. Christy, “Euclidean Shape and Motion from Multiple Perspective Views by Affine Iterations,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 18, no. 11, 1996.

[16] R. Y. Tsai, “Uniqueness and Estimation of Three-Dimensional Motion Parameters of Rigid Objects with Curved Surfaces,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 6, no. 1, p. 14, 1984.

[17] O.D.Faugeras, “Motion from Point Matches: Multiplicity of Solutions,” *International Journal of Computer Vision*, vol. 4, no. 3, p. 21, 1990.

[18] S. J. Prince, “Augmented Reality Camera Tracking with Homographies,” *Computer Graphics and Applications*, vol. 22, no. 6, 2002.

[19] J.-H. Kim, “Motion Estimation for Nonoverlapping Multicamera Rigs: Linear Algebraic and L”infinity” Geometric Solutions,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, 2010.

[20] Laksameethanasan.D., “A Bayesian Reconstruction Method with Marginalized Uncertainty Model For Camera Motion In Microrotation Imaging,” *IEEE Transactions on biomedical engineering*, vol. 57, no. 7, 2010.

[21] J. Lim, “Estimating Relative Camera Motion from the Antipodal-Epipolar Constraint,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 10, 2010.

[22] A. Alamri, “AR-REHAB: An Augemented Reality Framework for Poststroke-Patient Rehabilitation,” *IEEE Transactions on Instrument Measurement*, 2010.

[23] E. M. Chutorian, “Head Pose Estimation and Augmented Reality Tracking: An Integrated System and Evaluation for Monitoring Driver Awareness,” *IEEE Transactions on Intelligent Transportation System*, vol. 11, no. 2, 2010.

[24] http://www.hitl.washington.edu/artoolkit/, “ARToolKit,” 2003.

[25] S.K.Ong, “Markerless Augmented Reality Using a Robust Point Transferring Method,” *Springer-Verlag Berlin Heidelberg*, 2007.

[26] M. Yuan, “Registration Based on Projective Reconstruction Technique for Augemented Reality Systems,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 3, 2005.

[27] ——, “Registration Using Natural Features for Augemented Reality Systems,” *IEEE Transactions on Visulization and Computer Graphics*, vol. 12, no. 4, 2006.

[28] H. Samet, *Foundations of Multidimensional and Metric Data Structures.* Morgan Kaufmann, 2006.

[29] R. Hartley, “Estimation of Relative Camera Positions for Uncalibrated Cameras,” 1992.

[30] ——, *Multiple View Geometry in Computer Vision.* Cambridge: Cambridge University Press, 2003.

[31] Z. Zhang, “An Automatic and Robust Algorithm for Determining Motion and Structure from Two Perspective Images,” p. 8, Sept,1995 1995.

[32] ——, “A New Multistage Approach to Motion and Structure Estimation: from Essential Parameters to Euclidean Motion via Fundamental Matrix,” *Journal of Optical Society of America*, vol. 14, no. 11, 1997.

[33] ——, “Motion of an Uncalibrated Stereo Rig: Self-Calibration and Metric Reconstruction,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 1, 1996.

[34] R. Hartley, “Cheirality Invariants,” in *Proc. DARPA Image Understanding Workshop*, 1993, p. 10.

[35] ——, “Cheirality,” *International Journal of Computer Vision*, vol. 26, no. 1, p. 34, 1998.

[36] J. Stolfi, *Oriented Projective Geometry: A Framework for Geometric Computations.* 1250 Sixth Avenue, San Diego: Academic Press, 1991.

[37] S. Laveau and O.Faugeras, “Oriented Projective Geometry for Computer Vision,” in *Proc. 4th European Conf. on Computer Vision*, 1996, p. 10.

[38] R. Hartley, E. Hayman, L. d. Agapito, and I. Reid, “Camera Calibration and the Search for Infinity,” p. 8, 1999.

[39] R. Hartley and F. Kahl, “Optimal Algorithms in Multiview Geometry,” 2007.

[40] F. Kahl, “Multiple-view Geometry under the L Norm,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 1, p. 10, 2008.

[41] D.Nister, “Calibration with Robust use of Cheirality by Quasi-affine Reconstruction of the Set of Camera Projection Centres,” 2001.

[42] B. Triggs, “Matching Constraints and the Joint Image,” 1995.

[43] T.Werner and T. Pajdla, “Oriented Matching Constriant,” 2001.

[44] M. Pollefeys, “Detailed Real-Time Urban 3D Reconstruction from Video,” *Interational Journal of Computer Vision*, 2007.

[45] V. Kolmogorov and R. Zabih, “Multi-camera Scene Reconstruction via Graph Cuts,” *In Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, 2002.

[46] K. N. Kutulakos, “A Theory of Shape by Space Carving,” *International Journal of Computer Vision*, 2000.

[47] M.Lhuillier, “Match Propogation for Image based Modeling and Rendering,” *IEEE Transcations on Patten Analysis and Machine Intelligence*, 2002.

[48] L. Tang, “Image Dense Matching based on Region Growth with Adaptive Window,” *Pattern Recognition Letters*, 2003.

[49] D. Marquardt, “An Algorithm for Least-Squares Estimation of Nonlinear Parameters,” *SIAM Journal on Applied Mathematics*, 1963.

[50] J. Nocedal and S. Wright, “Numerical Optimization,” *Springer*, 1999.

[51] K. Mitra, “A Scalable Projective Bundle Adjustment Algorithm using the L infinty Norm,” *Computer Vision, Graphics and Image Processing*, 2008.

[52] Z. Zheng you, “Incremental Motion Estimation Through Local Bundle Adjustment,” *Technical Report*, 2001.

[53] W. Chang chang, “Multicore Bundle Adjustment,” *CVPR*, 2011.

[54] L. Bing, “Accelerated Bundle Adjustment in Multiple-View Reconstruction,” *CVPR*, 2011.

[55] A. J. Davison, “MonoSLAM: Real-Time Single Camera SLAM,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, 2007.

[56] S. Perera, “Towards Realtime Handheld MonoSLAM in Dynamic Environments,” *Advances in Visual Computing Lecture Notes in Computer ScienceVolume*, 2006.

[57] L. A. Clemente, “Mapping Large Loops with a Single Hand-held Camera,” *In Proceedings of Robotics: Science and Systems*, 2007.

[58] H. Strasdat, “Scale Drift-Aware Large Scale Monocular SLAM,” *In Proceedings of Robotics: Science and Systems*, 2010.

[59] L. Ling, “Eight Solutions of the Essential Matrix for Continuous Camera Motion Tracking in Video Augmented Reality,” 11-15 July 2011 2011.

[60] L. Ling, E. Cheng, and I. Burnett, “Cheriality Revisit in Camera Projective Reconstruction,” *EuraSIP Signal Processing Image Communications*, submitted.

[61] L. Ling, “A New Flexible Registration Method for Video Augmented Reality,” 2011.

[62] ——, “A Dense 3D Reconstruction Approach from Uncalibrated Video Sequences,” p. 6, 2012.

[63] L. Ling, E. Cheng, and I. Burnett, “An Iterated Extended Kalman Filter for 3D mapping via Kinect Camera,” *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Vancouver, Canada*, 2013.

[64] ——, "Analyse Oriented Projective Geometry from the Camera Motion Viewpoint," *International Conference on Digital Image Computing*, submitted.

[65] Y. C. H. Wang, "Pinhole SPECT with Different Data Acquisition Geometrics: Usefulness of Unified Projection Operators in Homogeneous Coordinates," *IEEE Transactions on Medical Imaging*, vol. 26, no. 3, 2007.

[66] J. Blinn, "A Trip Down the Graphics Pipeline: Line Clipping," *IEEE Computer Graphics and Applications*, p. 8, 1991.

[67] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, 2000.

[68] R. Y. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology using Off-the-shelf TV Cameras and Lenses," *IEEE Transactions on Robotics and Automation*, vol. 3, no. 4, 1987.

[69] M.Pollefeys, "Metric 3D Surface Reconstruction from Uncalibrated Image Sequences," in *In Proc. SMILE Workshop*, p. 13.

[70] P.Torr, "Robust Parameterization and Computation of the Trifocal Tensor," *Image Vision Compute*, vol. 15, 1997.

[71] Q.Luong, "Camera Calibration, Scene Motion and Structure Recovery from Point Correspondences and Fundamental Matrices," *International Journal for Computer Vision*, vol. 22, no. 3, p. 28, 1997.

[72] ——, "Self-Calibration of a Moving Camera from Point Correspondences and Fundamental Matrices," *International Journal for Computer Vision*, vol. 22, no. 3, p. 28, 1997.

[73] S.M.Seitz, "A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms," in *Proc. IEEE international Conference on Computer Vision and Pattern Recognition*.

[74] R.Horaud, "Stereo Calibration from Rigid Motions," *IEEE Transcations on Patten Analysis and Machine Intelligence*, vol. 22, no. 12, p. 12, 2000.

[75] S.Maybank, "A Theory of Self Calibration of a Moving Camera," *International Journal for Computer Vision*, vol. 8, p. 30, 1992.

[76] P.Sturm, "Affine Stereo Calibration," *Technical Report, LIFIA,*, vol. 20, 1995.

[77] M.Pollefeys, "Self-Calibration and Matric 3D Reconstruction from Uncalibrated Image Sequnces," Ph.D. dissertation, 1999.

[78] B.Triggs, "Autocalibration and the Absolute Quadric," in *IEEE conference computer vision and pattern recognition*.

[79] E. Rosten and T. Drummond, “Machine Learning for Highspeed Corner Detection,” *In European Conference on Computer Vision*, 2006.

[80] R. P. E. Rosten and T. Drummond, “Faster and Better: A Machine Learning Approach to Corner Detection,” *IEEE Transcations on Pattern Analysis and Machine Intelligence*, 2010.

[81] D. G. Lowe, “Object Recognition from Local Scale-Invariant Features,” p. 8, 1999.

[82] T. T. H. Bay and L. V. Gool, “Surf: Speeded up Robust Features,” *In European Conference on Computer Vision*, 2006.

[83] K. K. Ethan Rublee, Vincent Rabaud and G. Bradski, “ORB: An Efficient Alternative to SIFT or SURF,” *ICCV*, 2011.

[84] C. S. M. Calonder, V. Lepetit and P. Fua, “Brief: Binary Robust Independent Elementary Features,” *ECCV*, 2010.

[85] M. Graybill, “An Introduction to the Theory of Statistics,” *McGraw Hill*, 2003.

[86] S. Chandran, “Introduction to kd-trees,” *University of Maryland Department of Computer Science.*

[87] R. I. Hartley, “In Defense of the Eight-Point Algorithm,” *IEEE Transaction on Pattern Recognition and Machine Intelligence*, vol. 19, no. 6, p. 14, 1997.

[88] Q. Luong, “The Fundamental Matrix: Theory, Algorithms, and Stability Analysis,” *International Joumal of Computer Vision*, 1996.

[89] H. Li, “Five Point Motion Estimation Make Easy,” in *Proceeding of the 18th International Conference on Pattern Recognition*, 2006.

[90] D. Nister, “An Efficient Solution to the Five Point Relative Pose Problem,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2004.

[91] Q. Luong, “A Method for the Solution of Certain Non-Linear Problems in Least Squares,” *International Joumal of Computer Vision*, 1996.

[92] R. Hartley, “Kruppas Equations Derived from the Fundamental Matrix,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1999.

[93] J. Li, “Bundle Depth-map Merging for Multiview Stereo,” p. 10, 2010.

[94] D. . R. J. Ducke, Benjamin ; Score, “Multiview 3D reconstruction of the Archaeological Site at Weymouth from Image Series,” *Computers and Graphics, 2011*, vol. 35, no. 2, p. 8, 2011.

[95] D. Nister, “Calibration with Robust Use of Cheirality by Quasi-Affine Reconstruction of the Set of Camera Projection Centres,” 2001.

[96] S. Sengupta, "Refinement in 3D Reconstruction using Cheirality Constraints," 2008.

[97] W. Xu, "Robust Relative Pose Estimation with Integrated Cheirality Constraint," 2008.

[98] T. Dang, "Continuous Stereo Self-Calibration by Camera Parameter Tracking," *IEEE Transcations on Image Processing*, vol. 18, no. 7, 2009.

[99] D.Bradley, "Accurate Multi-view Reconstruction Using Robust Binocular Stereo and Surface Meshing," in *Proc. IEEE International Conference Computer Vision and Pattern Recognition.*

[100] ——, "Markerless Garment Capture," *ACM Transactions on Graphics*, vol. 27, no. 3, 2008.

[101] M.Goesele, "Multiview Stereo Revisited," in *Proc. IEEE conf. Computer vision and Pattern Recog.*

[102] J. Li, "Bundled Depth-Map Merging for Multi-view Stereo," *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, vol. 3, no. 4, 2010.

[103] S. Shen, "Depth-Map Merging for Multi-View Stereo with High Resolution Images," *21st International Conference on Pattern Recognition, 2012 IEEE Conference on*, vol. 1, no. 2, 2012.

[104] P. Merrell, "Real-Time Visibility-Based Fusion of Depth Maps," *International Conference on Computer Vision (ICCV), Rio de Janeiro, Brazil*, 2007.

[105] Y.Liu, "Continuous Depth Estimation for Multi-view Stereo," in *Proc. IEEE International Conference on Computer vision and Pattern Recognition.*

[106] ——, "A Point Cloud Based Multi-view Stereo Algorithm for Free View Point Video," *IEEE Transcations on Visualizationa dn Computer Graphics*, vol. 16, no. 3, 2010.

[107] C. Zach, "A Globally Optimal Algorithm for Robust TV-l1 Range Image Integration," *ICCV, 2007 IEEE Conference on*, vol. 1, no. 2, 2007.

[108] Y. Deng, "Noisy Depth Maps Fusion for Multiview Stereo Via Matrix Completion," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 6, no. 5, 2012.

[109] P. Mordohai, "Real-Time Video-Based Reconstruction of Urban Environments," *3D Virtual Reconstruction and Visualization of Complex Architectures, Zurich, Switzerland*, 2007.

[110] Y.Boykov, "Fast Approcimate Energy Minimisation via Graph Cuts," *IEEE Transcations on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, 2001.

[111] ——, “Computing Geodesics and Minimal Surfaces via Graph Cuts,” in *ICCV*.

[112] G. Vogiatzis, “Multi-view Stereo via Volumetric Graph-cuts and Occlusion Robust Photo-Consistency,” *IEEE Transcations on Patten Analysis and Machine Intelligence*, vol. 32, no. 8, 2007.

[113] ——, “Multi-view Stereo via Volumetric Graph-cuts,” *CVPR, 2005 IEEE Conference on*, vol. 1, no. 2, 2005.

[114] Y.Boykov, “Markov Random Fields with Efficient Approximations,” in *International Conference on Computer Vision and Pattern Recognition*.

[115] A. Hornung, “Robust and Efficient Photo-Consistency Estimation for Volumetric 3D Reconstruction,” *International Conference on Computer Vision (ICCV)*, 2006.

[116] V. S. Lempitsky, “Global Optimization for Shape Fitting,” *Computer Vision and Pattern Recognition, IEEE Conference on*, 2007.

[117] A. Broadhurst, “A Probabilistic Framework for Space Carving,” *ICCV, IEEE Conference on*, 2001.

[118] Y.Furukawa, “Accurate,Dense and Robust Multiview Steropsis,” *IEEE Transcations on Patten Analysis and Machine Intelligence*, vol. 32, no. 8, 2010.

[119] M.Lhuillier, “A Quasi-dense Approach to Surface Reconstruction from Uncalibrated Images,” *IEEE Transcations on Patten Analysis and Machine Intelligence*, vol. 27, 2005.

[120] N. Snavely, “Modeling the World from Internet Photo Collections,” *International Journal for Computer Vision*, 2007.

[121] Y.Furukawa, “Towards Internet-scale Multi-view Stereo,” 2009.

[122] Z.Zhang, “Motion of an Uncalibrated Stereo Rig: Self-calibration and Metric Reconstruction,” *IEEE Transcations Robot and Automatics*, vol. 12, 1996.

[123] O. Hesse, “DDie Cubische Gleichung, Von Welcher Die Lsung des Problems der Homograpie von m. chasles abh ngt. J. Reine Angew,” *Math. Ann*, 1863.

[124] R. I. Hartley, “A Linear Method for Reconstruction from Lines and Points,” *In Proceedings of the International Conference on Computer Vision*, 1995.

[125] K. Levernberg, “A Method for the Solution of Certain Non-Linear Problems in Least Squares,” *Quarterly of Applied Mathematics 2*, 1944.

[126] P. Torr, “Robust Parameterization and Computation of the Trifocal Tensor,” *Image and Vision Computing*, 1997.

[127] L. Quan, “Invariants of 6 Points from 3 Uncalibrated Images,” *In Proceedings of the third European conference on Computer Vision*, 1994.

[128] A. Shashua and M. Werman, “On the Trilinear Tensor of Three Perspective Views and Its Underlying Geometry,” *In Proceedings of the International Conference on Computer Vision*, 1995.

[129] D. Nister, “Reconstruction from Uncalibrated Sequences with a Hierarchy of Trifocal Tensors,” *In Proc. European Conf. on Computer Vision*, 2000.

[130] Y. Xing, “An Improved Region-growth Algorithm for Dense Matching,” *International OCSCO World Press.*, 2006.

[131] M.Lourakis, “SBA: A Software Package for Generic Sparse Bundle Adjustment,” *ACM Trans. Math. Software*, vol. 36, no. 1, 2009.

[132] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon, “Bundle Adjustment A Modern Synthesis,” 1999.

[133] M. S. R. Smith and P. Cheeseman, “A Stochastic Map for Uncertain Spatial Relationships,” *In Fourth International Symposium of Robotics Research*, 1987.

[134] D. F. S. Thrun and W. Burgard, “A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots,” *Machine Learning*, 1998.

[135] Y. XiaoPing, “Design, Implementation, and Experimental Results of a Quaternion-Based Kalman Filter for Human Body Motion Tracking,” *Proceedings of the 2005 IEEE International Conference on Robotics and Automation Barcelona, Spain, April,2005*, 2005.

[136] W. J. Wilson, “Relative End-Effector Control Using Cartesian Position Based Visual Servoing,” *IEEE Transactions on Robotics and Automation*, 1996.

[137] T. Bailey, “Consistency of the EKF-SLAM Algorithm,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[138] J. Civera and A. J.Davison, “1Point RANSAC for Extended Kalman Filtering: Application to RealTime Structure from Motion and Visual Odometry,” *Journal of Field Robotics*, 2006.

[139] P. Alcantarilla, “Visual Odometry Priors for Robust EKF-SLAM,” *IEEE International Conference on Robotics and Automation*, 2010.

[140] P. Foo, “Combining the Interacting Multiple Model Method with Particle Filters for Manoeuvring Target Tracking,” *Radar, Sonar and Navigation*, 2011.

[141] J. A. Civera, Javier and J. Montiel, “Inverse Depth to Depth Conversion for Monocular SLAM,” *ICRA*, 2007.

[142] J. Montiel and J. Civera, “Unified Inverse Depth Parametrization for Monocular SLAM,” *In Proceedings of Robotics: Science and Systems*, 2006.

[143] S. Julier, “Unscented Filtering and Nonlinear Estimation,” *Proceedings of the IEEE*, 2004.

[144] M. Athans, “Suboptimal State Estimation for Continuous-Time Non-linear Systems from Discrete Noisy Measurements,” *IEEE Trans Automatic Control*, 1996.

[145] R. K. Mehra, “A Comparison of Several Nonlinear Filters for Reentry Vehicle Tracking,” *IEEE Trans Automatic Control*, 1971.

[146] D. Lerro, “Tracking with Debiased Consistent Converted Measurement vs. EKF,” *IEEE Trans Aerosp. Electron. Syst*, 1993.

[147] T. Lefebvre, “Kalman Filters for Nonlinear Systems: a Comparison of Performance,” *IEEE Trans Automatic Control*, 2001.

[148] F. Pagel, “Robust Monocular Egomotion Estimation Based on an IEKF,” *Canadian Conference on Computer and Robot Vision*, 2009.

[149] B. M. Bell, “The Iterated Kalman Filter Update as a Gauss-Newton Method,” *IEEE Trans Automatic Control*, 2001.

[150] A. Shademan, “Sensitivity Analysis of EKF and Iterated EKF Pose Estimation for Position-based Visual Servoing,” *Proceedings of IEEE Conference on Control Applications*, 2005.

[151] K. Shojaei, “Sensitivity Analysis of EKF and Iterated EKF Pose Estimation for Position-based Visual Servoing,” *Journal of Intelligent and Robotic Systems*, 2011.

[152] R. Y. Tsai, “Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch, II: Singular Value Decomposition,” *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-30, no. 4, p. 10, 1982.

[153] R. Tsai, “Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 6, p. 6, 1981.

[154] ——, “Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch III,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, no. 2, 1984.

[155] T. c. Toolbox, “http://www-cgi.cs.cmu.edu/afs/cs.cmu.edu/user/rgw/www/tsaicode.html,” 2001.

[156] M. Gupta, “Camera Calibration Technical Using Tsai’s Algorithm,” *International Journal of Enterprise Computing and Business Systems*, 2011.

[157] J. Heikkil, “Geometric camera calibration using circular control points,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2000.

[158] P. Merkle, “The effects of multiview,” *SIGNAL PROCESSING: IMAGE COMMUNICATION*, 2009.

[159] D. Dwarakanath, “Faster and more Accurate Feature-based Calibration for Widely Spaced Camera Pairs ,” *Digital Information and Communication Technology and it's Applications (DICTAP), 2012 Second International Conference on*, 2012.

[160] R. Hartley, “Triangulation,” *Computer Vision and Image Understanding*, vol. 68, no. 2, p. 12, 1997.

[161] R. F. Riesenfeld, “Homogeneous Coordinates and Projective Planes,” *IEEE Computer Graphics and Applications*, vol. 1, no. 1, 1981.

[162] L. Silverberg, *Unified Field Theory for the Engineer and the Applied Scientist*. Weinheim: Wiley Vch Verlag GmbH Co KGaA, 2009.

[163] A. A. Ramirez, “Presenting Methods for Unraveling the First Two Regular 4D Polytopes (4D Simplex and the Hypercube),” 6-8 November 2002 2002.

[164] W. Wang, “A SVD Decomposition of Essential Matrix with Eight Solutions for the Relative Positions of Two Perspective Cameras,” in *Proc. 15th Int. Conf. on Pattern Recognition*, 2001, p. 4.

[165] T.S.Huang, “Some Properties of the E Matrix in Two-View Motion Estimation,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 11, no. 12, 1989.

[166] B.Zielman, “Two Methods for Multidimensional Analysis of Three Way Skew-symmetric Matrics,” Ph.D. dissertation, 2002.

[167] G.H.Golub, “Singular Value Decomposition and Least Squares Solutions,” *Numerische Mathematik*, vol. 14, no. 5, p. 18, 1969.

[168] B. Palais, “A Disorienting Look at Euler's Theorem on the Axis of a Rotation,” *American Mathematical Monthly*, vol. 116, no. 10, 2009.

[169] cross product, “$http://en.wikipedia.org/wiki/cross_product$.”

[170] S. Gibson, “Accurate Camera Calibration for Off-line, Video-Based Augmented Reality,” *IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2002.

[171] J. Shi and C. Tomasi, “Good Features to Track,” *IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[172] Q. Luong, “Self-Calibration of a Moving Camera from Point Correspondences and Fundamental Matrices,” *Int. Journal of Computer Vision*, vol. 22, no. 3, p. 29, 1997.

[173] T. B. Benjamin Ummenhofer, “Dense 3D Reconstruction with a Hand-held Camera,” *Pattern Recognition (Proc. DAGM), Springer, LNCS, 2012*, 2012.

[174] C. Y. R. A.Dame, V. A. Prisacariu and I. D. Reid, “Dense Reconstruction Using 3D Object Shape Priors,” *Proc 17th Int Conf on Computer Vision and Pattern Recognition (CVPR 2013)*, 2013.

[175] V. Hiep, “Towards High Resolution Large Scale Multiple View Stereo,” *IEEE Proc, CVPR*, 2009.

[176] http://www.cs.unc.edu/ ccwu/siftgpu/, 2010.

[177] http://vision.middlebury.edu/mview/, p. http://vision.middlebury.edu/mview/data/, 2006.

[178] http://cvlab.epfl.ch/ strecha/multiview/denseMVS.html, 2010.

[179] P. C. R. Smith, “On the Representation of Spatial Uncertainty,” *Int. J. Robotics Research*, 1987.

[180] F. A. Cheein, “Feature Selection Criteria for Real Time EKF-SLAM Algorithm,” *International Journal of Systems Vol.6, No.3*, 2009.

[181] W. Brink, “Stereo Vision as a Sensor for EKF-SLAM,” *Intelligent Systems*, 2011.

[182] B. Williams, “Automatic Relocalisation for a Single Camera Simultaneous Localisation and Mapping System,” *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2007.

[183] K. Konolige and P. Mihelich, “Technical Description of Kinect Calibration,” *http://www.ros.org/wiki/kinectcalibration/technical*, 2011.

[184] S. L. Richard A. Newcombe, “KinectFusion:Real-Time Dense Surface Mapping and Tracking,” Octorber, 2011.

[185] A. H. Sebastian Lieberknecht, “RGB-D Camera-based Parallel Tracking and Meshing,” 2011.

[186] RGBDemo, “http://labs.manctl.com/rgbdemo/index.php/documentation/kinectcalibrationth
2010.

[187] P. Hanspeter, “Surfels: Surface Elements as Rendering Primitives,” *ACM SIGGRAPH*, 2000.

[188] BLAS, “$http://en.wikipedia.org/wiki/basic_linear_algebra_subprograms.$”

[189] LAPACK, “*http : //www.netlib.org/lapack/*.”

[190] P. Newman and K. Ho, “SLAM- Loop Closing with Visually Salient Features,” *ICRA*, 2005.

[191] M. Montemerlo, “FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping that Provably Converges,” *In Proc. International Joint Conference on Artificial Intelligence*, 2003.