Camera Motion Estimation for Multi-Camera Systems

Jae-Hak Kim

A thesis submitted for the degree of Doctor of Philosophy of
The Australian National University

August 2008

This thesis is submitted to the Department of Information Engineering, Research School of Information Sciences and Engineering, The Australian National University, in fullfilment of the requirements for the degree of Doctor of Philosophy.

This thesis is entirely my own work, except where otherwise stated, describes my own research. It contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning.

Jae-Hak Kim
31 July 2008

Supervisory Panel:

Prof. Richard Hartley	The Australian National University
Dr. Hongdong Li	The Australian National University
Prof. Marc Pollefeys	ETH, Zürich
Dr. Shyjan Mahamud	National ICT Australia

In summary, this thesis is based on materials from the following papers, and my percived contributions to the relevant chpaters of my thesis are stated:

Jae-Hak Kim and Richard Hartley, "Translation Estimation from Omnidirectional Images," Digital Image Computing: Technqiues and Applications, 2005. DICTA 2005. Proceedings, vol., no., pp. 148-153, Dec 2005, **(80 per cent of my contribution and related to chapter 6)**

Brian Clipp, Jae-Hak Kim, Jan-Michael Frahm, Marc Pollefeys and Richard Hartley, "Robust 6DOF Motion Estimation for Non-Overlapping, Multi-Camera Systems," Applications of Computer Vision, 2008. WACV 2008. IEEE Workshop on , vol., no., pp.1-8, 7-9 Jan 2008, **(40 per cent of my contribution and related to chapter 7)**

Jae-Hak Kim, Richard Hartley, Jan-Michael and Marc Pollefeys, "Visual Odometry for Non-overlapping Views Using Second-Order Cone Programming," Asian Conference on Computer Vision, Tokyo, Japan, ACCV (2) 2007, pp. 353-362 (also published in Lecture Notes in Computer Sciences, Springer, Volume 4844, 2007), **(80 per cent of my contribution and related to chapter 8)**

Hongdong Li, Richard Hartley, and Jae-Hak Kim, "Linear Approach to Motion Estimation using a Generalized Camera," IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), Anchorage, Alaska, USA, 2008, **(30 per cent of my contribution and related to chapter 9)**

Jae-Hak Kim, Hongdong Li and Richard Hartley, "Motion Estimation for Multi-Camera Systems using Global Optimization," IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), Anchorage, Alaska, USA, 2008, **(80 per cent of my contribution and related to chapter 10)**

Jae-Hak Kim, Hongdong Li and Richard Hartley, "Motion Estimation for Non-overlapping Multi-Camera Rigs: Linear Algebraic and L_∞ Geometric Solutions," submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, **(70 per cent of my contribution and related to chapters 9 and 10)**

'Peace be with you.' *John 20:21*

Acknowledgements

I would like to sincerely thank Prof. Richard Hartley for giving me the opportunity to study at the Australian National University and at NICTA and for supervising me during my Ph.D. course. His guidance gave me a deep understanding of multiple view geometry and truly let me know how beautiful geometry is. I would like to thank Dr. Hongdong Li, who gave me a lot of advice on my research work and encouraged me to conceive new ideas. I would also like to thank Prof. Marc Pollefeys, who gave me a chance to visit his vision group at the University of North Carolina, Chapel Hill, as a visiting student and inspired me to find a research topic for my Ph.D. thesis. I also thank Dr. Shyjan Mahamud, who directed my approach to solving problems. I would like to thank National ICT Australia (NICTA) for providing Ph.D. scholarships during the last four years.

I would like to thank Dr. Jan-Michael Frahm for many discussions about this research as well as other members of the vision group at the UNC-Chapel Hill – Dr. Jean-Sebastien Franco, Dr. Philippos Mordohai, Brian Clipp, David Gallup, Sudipta Sinha, Paul Merrel, Changchang Wu, Li Guan and Seon-Ju Kim. They gave me a warm welcome and provided good research atmosphere during my visit to UNC-Chapel Hill.

I would like to thank Prof. Berthold K. P. Horn and Prof. Steve Maybank who answered my questions and discussed the history and terminologies of epipolar geometry via emails. I would also like to thank Dr. Jun-Sik Kim who pointed out an error in a figure that has been used in this thesis.

I also thank Prof. Joon H. Han, Prof. Daijin Kim, Prof. Seungyong Lee, Prof. Yongduek Seo and Prof. Jong-Seung Park. They have been my strongest support from Korea, and they have shown me how exciting and interesting computer vision is.

I would also like to thank people and organizations for providing me pictures and illustrations that are used in my thesis – AAAS, AIST, Breezesystems Inc., Dr. Carsten Rother, Google, The M.C. Escher Company, NASA/JPL-Caltech, Point Grey Research Inc., Prof. Richard Hartley, Sanghee Seo, Dr. Simon Baker, Timothy Crabtree, UNC-Chapel Hill and Wikipedia.

I am grateful to the RSISE graduate students and academic staff – Andreas M. Maniotis, Dr. Antonio Robles-Kelly, Dr. Arvin Dehghani, Dr. Brad Yu, Dr. Chanop Silpa-Anan, Dr. Chunhua Shen, Desmond Chick, Fangfang Lu, Dr. Guohua Zhang, Dr. Hendra Nurdin, Jae Yong Chung, Dr. Jochen Trumpf, Junae Kim, Dr. Kaiyang Yang, Dr. Kristy Sim, Dr. Lei Wang, Luping Zhou, Manfed Doudar, Dr. Nick Barnes, Dr. Paulette Lieby, Dr. Pei Yean Lee,

Pengdong Xiao, Peter Carr, Ramtin Shams, Dr. Robby Tan, Dr. Roland Goecke, Sung Han Cha, Surya Prakash, Tamir Yedidya, Teddy Rusmin, Vish Swaminathan, Dr. Wynita Griggs, Yifan Lu, Yuhang Zhang and Zhouyu Fu. They welcomed me to RSISE and helped me survive as a Ph.D. student in Australia.

I would also like to thank Hossein Fallahi and Dr. Andy Choi, who were residents with me at Toad Hall.

I would like to thank my close friends from Korea – Jongseung Kim, Jaenam Kim, Hui-jeong Kim, Yechan Ju, Hochan Lim, Kyungho Kim and Jaewon Jang – who supported me.

I also thank my friends in POSTECH – Hyukmin Kwon, Semoon Kil, Byunghwa Lee, Jiwoon Hwang, Dr. Kyoo Kim, Yumi Kim, Dr. Hyeran Kim, Hyosin Kim, Dr. Sunghyun Go, Dr. Changhoon Back, Minseok Song, Dr. Hongjoon Yeo, Dr. Jinmong Won, Dr. Gilje Lee, Sookjin Lee, Hyekyung Lim, Chanjin Jeong and Chunkyu Hwang.

I would like to thank the Korean students in Canberra – Anna Jo, Christina Yoon, Eunhye Park, Eunkyung Park, Haksoo Kim, Inho Shin, Jane Hyo Jin Lee, Kyungmin Lee, Mikyung Moon, Miseon Moon, Sanghoon Lee, Sangwoo Ha, Se-Heon Oh, Sung-Hun Lee, Taehyun Kim, Thomas Han and Wonkeun Chang. They have been so kind to me as the same international student studying in Australia. I will never forget the good times that we had together.

In particular, I would like to thank Fr. Albert Se-jin Kwon, Fr. Michael Young-Hoon Kim, Br. Damaso Young-Keun Chun and Fr. Laurie Foote. for provding spiritual guidance.

Last, but not the least, I would like to thank my relatives and familiy – Clare Kang, Gloria Kim, Natalie Kim, Yunmi Kim, Hyuncheol Kim, Hocheol Kim, Yeonmi Kim, my father and my mother. I would especially like to thank my wife, Eun Young Kim, who supported me with her love and sincere belief. Also, thank my little child to be borned soon.

Thanks to all people I do not remember now, and as always, thanks be to God.

Abstract

The estimation of motion of multi-camera systems is one of the most important tasks in computer vision research. Recently, some issues have been raised about general camera models and multi-camera systems. Using many cameras as a single camera is studied [60], and the epipolar geometry constraints of general camera models is theoretically derived. Methods for calibration, including a self-calibration method for general camera models, are studied [78, 62]. Multi-camera systems are an example of practically implementable general camera models and they are widely used in many applications nowadays because of both the low cost of digital charge-coupled device (CCD) cameras and the high resolution of multiple images from the wide field of views. To our knowledge, no research has been conducted on the relative motion of multi-camera systems with non-overlapping views to obtain a geometrically optimal solution.

In this thesis, we solve the camera motion problem for multi-camera systems by using linear methods and convex optimization techniques, and we make five substantial and original contributions to the field of computer vision. First, we focus on the problem of translational motion of omnidirectional cameras, which are multi-camera systems, and present a constrained minimization method to obtain robust estimation results. Given known rotation, we show that bilinear and trilinear relations can be used to build a system of linear equations, and singular value decomposition (SVD) is used to solve the equations. Second, we present a linear method that estimates the relative motion of generalized cameras, in particular, in the case of non-overlapping views. We also present four types of generalized cameras, which can be solvable using our proposed, modified SVD method. This is the first study finding linear relations for certain types of generalized cameras and performing experiments using our proposed linear method. Third, we present a linear 6-point method (5 points from the same camera and 1 point from another camera) that estimates the relative motion of multi-camera systems, where cam-

eras have no overlapping views. In addition, we discuss the theoretical and geometric analyses of multi-camera systems as well as certain critical configurations where the scale of translation cannot be determined. Fourth, we develop a global solution under an L_∞ norm error for the relative motion problem of multi-camera systems using second-order cone programming. Finally, we present a fast searching method to obtain a global solution under an L_∞ norm error for the relative motion problem of multi-camera systems, with non-overlapping views, using a branch-and-bound algorithm and linear programming (LP). By testing the feasibility of LP at the earlier stage, we reduced the time of computation of solving LP.

We tested our proposed methods by performing experiments with synthetic and real data. The Ladybug2 camera, for example, was used in the experiment on estimation of the translation of omnidirectional cameras and in the estimation of the relative motion of non-overlapping multi-camera systems. These experiments showed that a global solution using L_∞ to estimate the relative motion of multi-camera systems could be achieved.

Contents

Chapter 1

Introduction

In this thesis, we investigate the relative motion estimation problem of multi-camera systems to develop linear methods and a global solution. Multi-camera systems have many benefits such as rigid motion for all six degrees of freedom without 3D reconstruction of the scene points. Implementations of multi-camera systems can be found in many applications but few studies have been done on the motion of multi-camera systems so far.

In this chapter, we give a general introduction to multi-camera systems and their applications, followed by our contributions and an overview of this thesis.

Recently, the popularity of digital cameras such as digital SLR (single-lens reflex) cameras, compact cameras and mobile phones with built in camera has increased due to their decreased cost. Barry Hendy from Kodak Australia [29] plotted the "pixels per dollar" as a basic measure of the value of a digital camera and used the information to recommend a retail price for Kodak digital cameras. This law is referred to as "Hendy's Law". On the basis of this law, it can be concluded that the resolution of a digital camera is becoming higher and the price per pixel of the camera sensor is becoming lower every year. It is no longer difficult or expensive to set up an application that uses several cameras.

It is considered that multicamera systems (a cluster of cameras or a network of cameras) have many benefits in real applications such as visual effects and scientific research. The first study on virtualized reality projects that use virtual views captured by a network of cameras was conducted by Kanade et al. in 1995 [54]. Their system was used to capture touchdowns in the Super Bowl, which is the championship game of professional American football, and it was used to look around the event from other point of virtual views. In 1999, a similar visual

Figure 1.1: *A software controlling 120 cameras using 5 laptops.* `www.breezesys.com` *(Courtesy of Breezesystems, Inc)*

effect known as “bullet time” was implemented in the film “The Matrix”, where the camera appears to orbit around the subject of the scene. This was done by placing a large number of cameras around the subject of the scene. Digital Air is a well-known company that produces Matrix-like visual effects for commercial advertisements [9]. Another company, Breezesys, Inc. [6], sells consumer-level software that allows the simultaneous capture of multiple images by multiple cameras controlled by a single laptop, as shown in Figure 1.1. Thus, the use of multi-camera systems in various applications is becoming popular and their use is expected to increase in the near future.

In the last two decades, many studies have been conducted on the theory and geometry of single-camera systems which are used to capture images from two views, three views and multiple views [11, 10, 27]. However, the theory and geometry of multi-camera systems have not been fully studied or clarified yet. This is because in addition to recording multiple views of a scene using a network of cameras or an array of cameras, there are more challenging tasks such as obtaining spatial and temporal information as the multi-camera system moves around the environment.

This process of obtaining the orientation and position information is known as the “visual odometry” problem or “the problem of estimation of relative motion of multi-camera systems”. A good example of this is as follows: The Mars Exploration Rovers, Spirit and Opportunity,

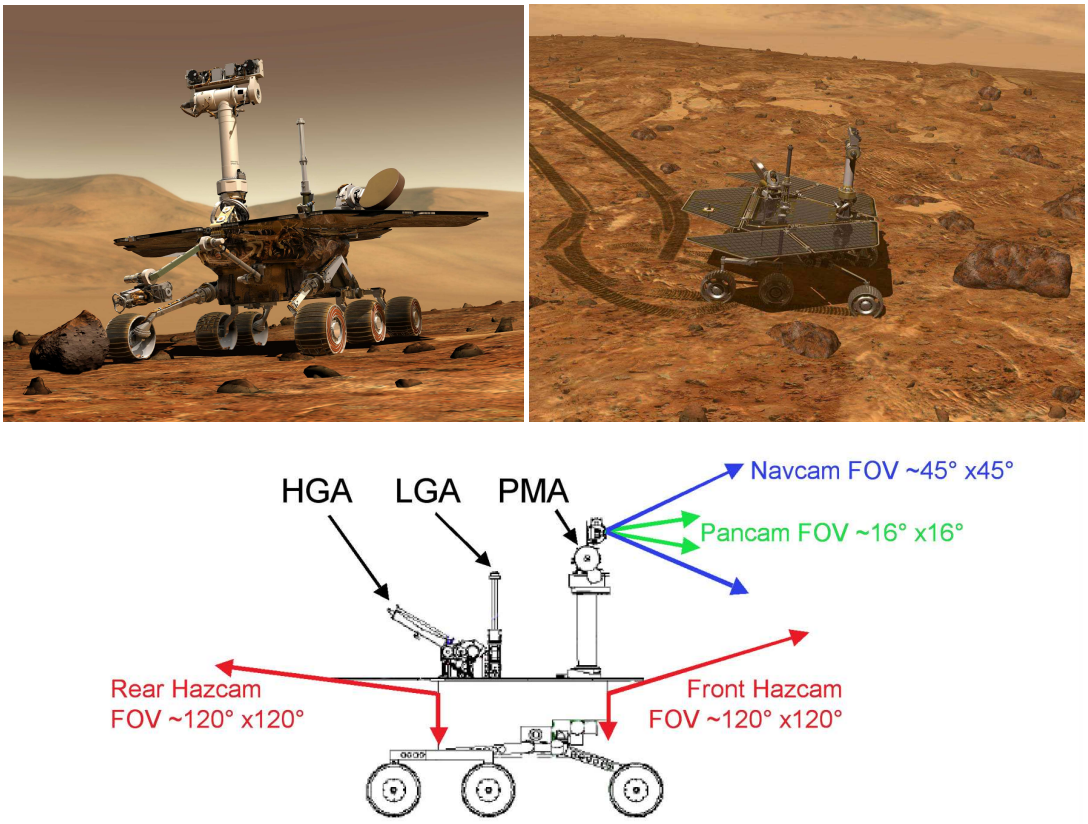


Figure 1.2: ***The Mars Exploration Rovers in motion.*** *The rovers are equipped with 9 cameras: four Hazcams are mounted on the front and rear ends for hazard avoidance, two Navcams are mounted on the head of the rovers for navigation, two Pancams are mounted on the head to capture panoramas, and one micoscopic camera is mounted on the robotic arm. (Courtesy NASA/JPL-Caltech)*

landed on Mars in January 2004. As shown in Figure 1.2, these rovers were equipped with nine cameras distributed between their heads, legs and arms. Although the rovers were equipped with navigation sensors such as IMU (inertial measurement unit) and odometry sensors on their wheels, the estimated distance travelled by the rovers on Mars was not very accurate. This could have been due to several reasons, for example, the rover wheels could not obtain a proper grip on the ground on Mars, which caused the wheels to spin without moving. This resulted in the recording of false measurements by the odometry unit. Another reason could have been the accidental failure of the IMU and odometry equipment. In such a case, visual sensors such as the nine cameras might be used to determine the location of the rovers on Mars. To our knowledge, no research has been conducted on getting an optimal solution to predict the

motion of multi-camera systems. Hence, if we develop an optimal solution, it can be applied to control the motion of planetary rovers, UAVs (unmanned aerial vehicles), AUVs (autonomous underwater vehicles) and domestic robots such as Spirit and Opportunity on Mars, Aerosonde, REMUS and iRobot's Roomba.

In general, the motions of camera systems can be considered to be Euclidean motions that have six degrees of freedom in three-dimensional (3D) space. So, the main aim of this study is to estimate the motion for all six degrees of freedom. However, in single-camera systems that capture two images, the relative motion can be estimated for only five degrees of freedom: three degrees for rotation and two degrees for translation direction. The scale of translation cannot be estimated from the single-camera system unless 3D structure is recovered. However, in the case of non-overlapping multiple rigs, 3D structure recovery problem is not as easy as in the case of systems with overlapping views such as stereo systems and monocular SLAM (Simultaneous Localization and Mapping) systems.

1.1 Problem definition

In this thesis, we investigate the motion of multi-camera systems. We investigate motion estimation problems such as the translational motion of an omni-directional camera, the motion of a non-overlapping 8-camera system on a vehicle using a linear method and the motion of a 6-camera system (Ladybug2 camera) using second-order cone programming (SOCP) or linear programming (LP) under L_∞ norm.

In general, the motion of multi-camera systems is a rigid motion. Therefore, there are 6 degrees of freedom for rotation and translation. Taking advantage of the spatial information (exterior calibration parameters) of cameras in multi-camera systems, we can estimate the relative motion of multi-camera systems for six degrees of freedom.

Given known camera parameters, we capture image sequences using a multi-camera system. Then, pairs of matching points are detected and found using feature trackers. Using these pairs of matching points, we estimate the relative motion of multi-camera systems for all the six degrees of freedom.

1.2 Contributions

In this thesis,

1. We show that if the rotation of the camera across multiple views is known, it is possible to estimate the translation more accurately using a constrained minimization method based on singular value decomposition (SVD).

2. We also show that the motion of non-overlapping images can be estimated from a minimal set of 6 points of which 5 points are from one camera and 1 point is from another camera. Theoretical analysis of the critical configuration that makes it impossible to solve the relative motion of multi-camera systems is also studied.

3. A linear method to estimate the orientation and position of a multi-camera system (or a general camera model) is studied by considering the rank deficiency of equations and experiments. To our knowledge, no experiments using linear methods have been performed by other researchers in the field of computer vision.

4. Using global optimization and the convex optimization techniques, we solved the problem of estimation of motion using SOCP.

5. We solved the problem of estimation of motion using LP with a branch-and-bound algorithm. Approaches 4 and 5 provide a framework to obtain a global solution for the problem of estimation of relative motion in multi-camera systems (even with non-overlapping views) under the L_∞ norm.

We performed experiments with synthetic and real data to verify our algorithms, and they mostly showed robust and good results.

1.3 Overview

In chapter 1, we provide a general overview of the problems in the estimation of multi-camera systems and demonstrate how multi-camera systems can be used in real applications.

In chapters 2 to 4, we provide brief overviews of the single-camera system, two-camera system, three-camera system, multi-camera system and their motion estimation problems. In chapter 5, we discuss previous related works.

The main work of this thesis is presented in chapters 6, 7, 8, 9 and 10. In chapter 6, we show how constrained minimization allows the robust estimation from omnidirectional images. In chapter 7, we show how using six points, we can estimate the relative motion of non-overlapping views, and we also show that there is a degeneracy configuration that makes it impossible to estimate the motion of non-overlapping multi-camera rigs. In chapter 8, we reveal a linear method for estimation of the motion of a general camera model or non-overlapping multi-camera systems along with an intensive analysis of the rank deficiency in generalized epipolar constraint equations. In chapter 9, we study the geometry of multi-camera systems and demonstrate how using their geometry, we can convert the motion problem to a convex optimization problem using SOCP. In chapter 10, we attempt to improve the method proposed in chapter 9 by developing a unified framework to derive a global solution for the problem of estimation of camera motion in multi-camera systems using LP and a branch-and-bound algorithm. Finally, in chapter 11, conclusions and discussions are presented.

Chapter 2

Single-Camera Systems

2.1 Geometry of cameras

In this section, we revisit the geometry of single-camera systems and present a detailed analysis of the projection of points in space onto an image plane and the rigid transformations of points and cameras.

Let us assume that the world can be represented using a projective space $\mathbb{P}^3$. The structures and shapes of objects are represented using points in the form of 4-vectors such as $\mathbf{X}$ in $\mathbb{P}^3$. The motion of these points is represented by a 3×3 rotation matrix $\mathtt{R}$ and a 3-vector translation $\mathbf{t}$. Let us now consider transformations of points and cameras in the projective space $\mathbb{P}^3$.

Three coordinate systems are used to describe the positions of points, the locations of cameras in the projective space $\mathbb{P}^3$ and the image coordinates in $\mathbb{P}^2$. In this study, we have used right-hand coordinate systems, as shown in Figure 2.1. The first coordinate system is the *world coordinate system*, which is used to represent the positions of points and cameras in

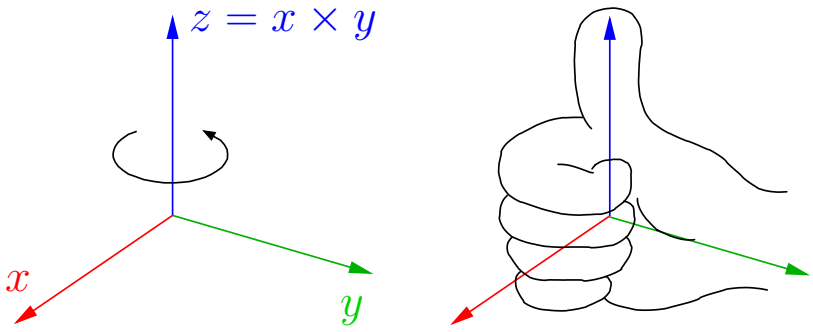
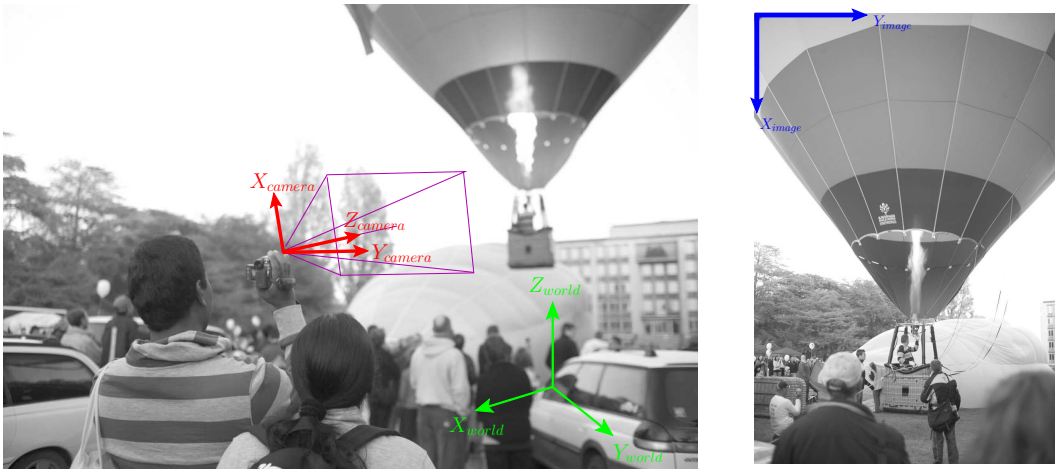


Figure 2.1: *Right-hand coordinate system.*



(a) Camera and scene structure in the world coordinate system (b) Projected image

Figure 2.2: *(a) The camera coordinate system (indicated in red) is represented by the basis vectors X_{camera}, Y_{camera} and Z_{camera}, and the world coordinate system (indicated in green) is represented by the basis vectors X_{world}, Y_{world} and Z_{world} in 3D space. (b) The image coordinate system is represented by two vectors X_{image} and Y_{image} in 2D space.*

the world. Hence, the positions of all points and cameras can be represented by an identical measurement unit such as "metre". The second system is the *camera coordinate system*, in which the positions of the points are based on the viewpoints of the cameras in $\mathbb{P}^3$. It should be noted that a point in space can be expressed both in the world coordinate system and in the camera coordinate system. The final coordinate system is the *image coordinate system*, which is specifically used to define the coordinates of pixels in images. Unlike the first two coordinate systems, the image coordinate system is in $\mathbb{P}^2$. The image coordinate system uses "pixels" as the unit of measurement.

Figure 2.2 shows the three coordinate systems. In Figure 2.2(a), we observe that the person holding the camera is taking a picture of a balloon. A camera has its own two-dimensional (2D) coordinate system for images. This 2D coordinate system is shown in Figure 2.2(b). The camera is positioned with respect to a reference point in the world coordinate system. The position of the balloon in the air can also be defined with respect to the reference point in the world coordinate system. Therefore, the positions of the camera and balloon (structure) are expressed in the world coordinate system (indicated in green). The orgin of the camera

coordinate system (indicated in red) is positioned at the centre of the camera and points toward the object of interest.

2.1.1 Projection of points by a camera

If we assume that the z-axis of the camera is aligned with the z-axis of the world coordinate system, and the two coordinate systems are placed at the origin, then the camera projection matrix can be represented by a 3×4 matrix as follows:

$$\mathtt{P} = [\mathtt{I} \mid \mathbf{0}] \tag{2.1}$$

where $\mathtt{I}$ is a 3×3 identity matrix.

Let a 4-vector $\mathbf{X}_{cam}$ be a point in space and $\mathbf{X}_{cam}$ be represented in the camera coordinate system. Then, $\mathbf{X}_{cam}$ may be projected onto the image plane of the camera through a lens. The image plane uses a 2D image coordinate system, as shown in Figure 2.2(b). Therefore, the projected point $\mathbf{x}$ is represented as a 3-vector in $\mathbb{P}^2$ and can be denoted as follows:

$$\mathbf{x} = [\mathtt{I} \mid \mathbf{0}]\mathbf{X}_{cam} \tag{2.2}$$

It should be noted that $\mathbf{x}$ still uses the same unit (say "metre") as that of the world coordinate system in (2.2). However, as we are dealing with images, this unit needs to be converted to a pixel unit. Most digital cameras have a charge-coupled device (CCD) image sensor that is only a few millimetres in size. For instance, the Sony ICX204AK[1] is a $6\text{-}mm\ (= 0.24\ in)$ diagonal, interline CCD solid-state image sensor with a square pixel array, and it has total of 1024×768 active pixels. The unit cell size of each pixel is $4.65\mu m \times 4.65\mu m$[2]. Therefore, the units needed to be converted in order to obtain the coordinates of a pixel in the image. For instance, in Sony ICX204AK CCD sensors, the size of a pixel is 4.65×10^{-6} metres. Hence, this value is multiplied by $1/(4.65 \times 10^{-6})$ in order to convert the unit from metres to pixels.

It is also necessary to consider other parameteres such as the focal length, the principal

[1] Sony ICX204AK technical document [33]

[2] $1\mu m$ (micrometre) = $10^{-6}m$ (metre) = $3.93700787 \times 10^{-5}$ in.

points where the optical axis meets the image plane, and the skewness of the image sensor. All these parameters are included in a 3×3 matrix, which is termed a "calibration matrix". The calibration matrix may be added in (2.2) and it is given as follows:

$$\mathbf{x} = \mathtt{K}[\mathtt{I} \mid \mathbf{0}]\mathbf{X}_{cam} \tag{2.3}$$

where $\mathtt{K}$ has focal lengths f_x and f_y, and the skew parameter s, and it is defined as

$$\mathtt{K} = \begin{bmatrix} f_x & s & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{2.4}$$

The units of the focal lengths f_x and f_y should be converted from metres, the unit of the world coordinate system, to pixels, the measurement unit of images.

2.1.2 Rigid transformation of points

A rigid transformation $\mathtt{M}$ of a point $\mathbf{X}$ in $\mathbb{P}^3$ is given as follows:

$$\mathbf{X}' = \mathtt{M}\mathbf{X} , \tag{2.5}$$

where $\mathtt{M}$ is a 4×4 matrix used for transformation and $\mathbf{X}'$ is the position of $\mathbf{X}$ after transformation of $\mathbf{X}$. This transformation may be considered to represent the point $\mathbf{X}$ after rotation and translation. Thus, (2.5) may be rewritten as follows:

$$\mathbf{X}' = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \mathbf{X} , \tag{2.6}$$

where $\mathtt{R}$ is a 3×3 rotation matrix and $\mathbf{t}$ is a 3-vector translation. Please note that the point $\mathbf{X}$ is translated by $\mathbf{t}$ first and then rotated by $\mathtt{R}$ with respect to the world coordinate system. This is shown in Figure 2.3.

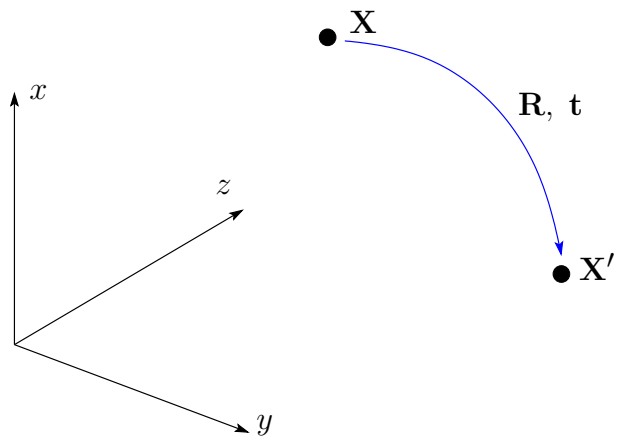


Figure 2.3: ***Rigid transformation of a point.*** *A point* $\mathbf{X}$ *is moved to a different position* $\mathbf{X}'$ *by a rigid motion comprising rotation* $\mathtt{R}$ *and translation* $\mathbf{t}$.

2.1.3 Rigid transformation of cameras.

Let us now consider the rigid transformation of the coordinates of a camera, as shown in Figure 2.4. The camera is placed in the world coordinate system, so its coordinate transformation has rotation and translation parameters similar to the transformation of points.

A camera aligned with the axis of the world coordinate system at the origin is represented by a 3×4 matrix as follows:

$$\mathtt{P} = [\mathtt{I} \mid \mathbf{0}] \,, \tag{2.7}$$

where $\mathtt{I}$ is a 3×3 identity matrix.

If the camera is positioned at a point $\mathbf{c}$, the camera matrix is represented as follows:

$$\mathtt{P} = \begin{bmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 1 & -c_z \end{bmatrix} \,, \tag{2.8}$$

where the vector $\mathbf{c} = [c_x, c_y, c_z]^\top$ is the centre of the camera. The left 3×3 submatrix in $\mathtt{P}$ is not changed because the camera is still aligned with the world coordinate system.

If the camera is rotated by $\mathtt{R}$ with respect to the world coordinate system, then the newly

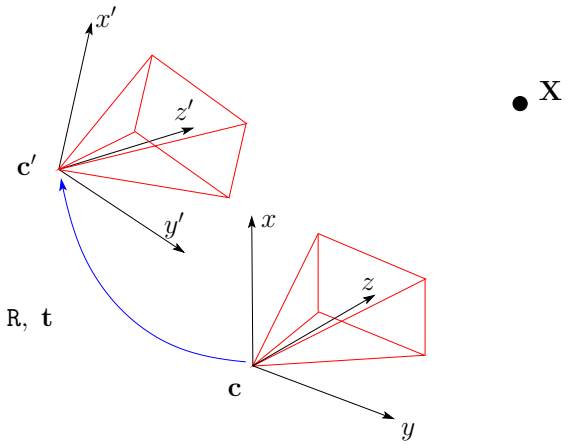


Figure 2.4: ***Rigid transformation of a camera.*** *A camera at* $\mathbf{c}$ *is moved to a position* $\mathbf{c}'$ *by a rigid motion comprising rotation* $\mathtt{R}$ *and translation* $\mathbf{t}$.

positioned camera matrix can be represented as follows:

$$\mathtt{P} = \mathtt{R}[\mathtt{I} \mid -\mathbf{c}] = [\mathtt{R} \mid -\mathtt{R}\mathbf{c}] = [\mathtt{R} \mid \mathbf{t}] \,, \tag{2.9}$$

where $\mathbf{t} = -\mathtt{R}\mathbf{c}$ is a vector represented by the translation[3].

In particular, note that the camera is first translated by $\mathbf{t}$ and is then rotated by $\mathtt{R}$ with respect to the world coordinate system. Finally, the camera is positioned at $\mathbf{c}$. A point $\mathbf{X}$ in $\mathbb{P}^3$ is projected onto an image point $\mathbf{v}$ in $\mathbb{P}^2$ by the camera matrix $\mathtt{P}$ as follows:

$$\mathbf{v} = \mathtt{P}\mathbf{X} = \mathtt{R}[\mathtt{I} \mid -\mathbf{c}]\mathbf{X} \,, \tag{2.10}$$

where $\mathbf{v}$ is a 3-vector in $\mathbb{P}^2$ and is represented in the image coordinates. Hence, $\mathbf{v}$ can be considered as an image vector originating from the centre of the camera to the point $\mathbf{X}$. If $\mathbf{X}$ is displaced by the motion matrix $\mathtt{M}$, then the projection of $\mathbf{X}$ is also displaced as follows:

$$\mathbf{v}' = \mathtt{P}\mathtt{M}\mathbf{X} = \mathtt{R}[\mathtt{I} \mid -\mathbf{c}]\mathtt{M}\mathbf{X} \,. \tag{2.11}$$

[3]The vector $\mathbf{t}$ is also called a translation in other articles. However, probably it is more reasonable to define $\mathbf{c}$ as a translation instead of $\mathbf{t}$ because it is more relevant to our geometrical concepts. For better understanding, in this thesis, the vector $\mathbf{c}$ is called as the centre of the camera and the vector $\mathbf{t}$ is denoted as the direction of translation.

Instead of moving $\mathbf{X}$, let us imagine that the camera is moved to make the position of the projected point the same as that of $\mathbf{v}'$. Therefore, from (2.11), the matrix $\mathtt{P}'$ of the transformed camera matrix is written as:

$$\mathtt{P}' = \mathtt{PM} = \mathtt{R}[\mathtt{I} \mid -\mathbf{c}]\mathtt{M} \,. \tag{2.12}$$

Let us consider two rigid transformations $\mathtt{M}_1$ and $\mathtt{M}_2$. Let the transformations be applied in the order $\mathtt{M}_1$ and $\mathtt{M}_2$ to a point $\mathbf{X}$. The transformed point is denoted as $\mathbf{X}' = \mathtt{M}_2\mathtt{M}_1\mathbf{X}$. In the same way, the transformed camera matrix can be given by $\mathtt{P}' = \mathtt{P}\mathtt{M}_2\mathtt{M}_1$ instead of moving points.

2.2 Epipolar geometry of two views

In this section, we revisit the geometry of single-camera systems used to capture two images from two different locations and also re-introduce methods to estimate the relative motion of a camera between two views. In the following section, we distinguish between two terms "views" and "cameras" in order to better understand multi-camera systems.

2.2.1 Definitions of views and cameras

Views. Views are defined as images taken by a single camera at different locations. As the same camera is used, each view has the same image size and the same calibration parameters. The phrase "two views", implies that physically a single camera device is used to capture two images from two different positions in space. On the other hand, the phrase "multiple views" (say n views) implies that physically a single camera device is used to capture multiple images, which form a single image sequence, from n different positions.

Cameras. Cameras are physical devices used to capture images. The image sizes and calibration parameters vary from camera to camera. Even if the cameras are identical and are manufactured by the same company, they may have different focal lengths and/or different principal points. The cameras may be located in the same positions while capturing images but are generally placed in different positions. Whenever we use the phrase "two cameras", it

refers to two physically separated camera devices that are used together to capture two image sequences. The phrase "multiple cameras" implies that n camera devices are used together to capture n image sequences. Therefore, the phrase "3 views of 4 cameras", means that four cameras are used to capture four image sequences from three different positions (a total of 12 images).

2.2.2 History of epipolar geometry

The history of epipolar geometry is closely connected to the history of photogrammetry. The first person to analyze geometric relationships was Guido Hauck in 1883 [28]. In his article published in "Journal of Pure and Applied Mathematics", he used the German term Kernpunkt (epipole) as follows [28]:

> Es seien (Fig. 1. a) S' und S'' zwei Projectionsebenen, O_1 und O_2 die zugehörigen Projectionscentren. Die Schnittlinie g_{12} der zwei Projectionsebenen nennen wir den *Grundschnitt*. Die Verbindungslinie O_1O_2 möge die zwei Projectionsebenen in den Punkten o'_2 und o''_1 schneiden, welche wir die *Kernpunkte* der zwei Ebenen nennen.

The English translation may be as given below:

> Let S' and S'' be two projection planes, and O_1 and O_2 the corresponding projection centres (Fig. 1. a). We will call the intersection line of the two projection planes the *Grundschnitt (basic cut)*. Let the line joining O_1O_2 cuts the two projection planes in the points o'_2 and o''_1, which we will call the *Kernpunkte (epipoles)* of the two planes.

Figure 2.5 shows the epipolar geometry and the two epipoles (Kernpunkte) o''_1 and o'_2, as illustrated by Guido Hauck in his paper [28].

Epipolar geometry was studied first by German mathematicians and was introduced to the English in the first half of the 20th century. As pointed out by J. A. Salt [65] in 1934, most of the literature on photogrammetry until that time had appeared in German. In 1908, Von Sanden

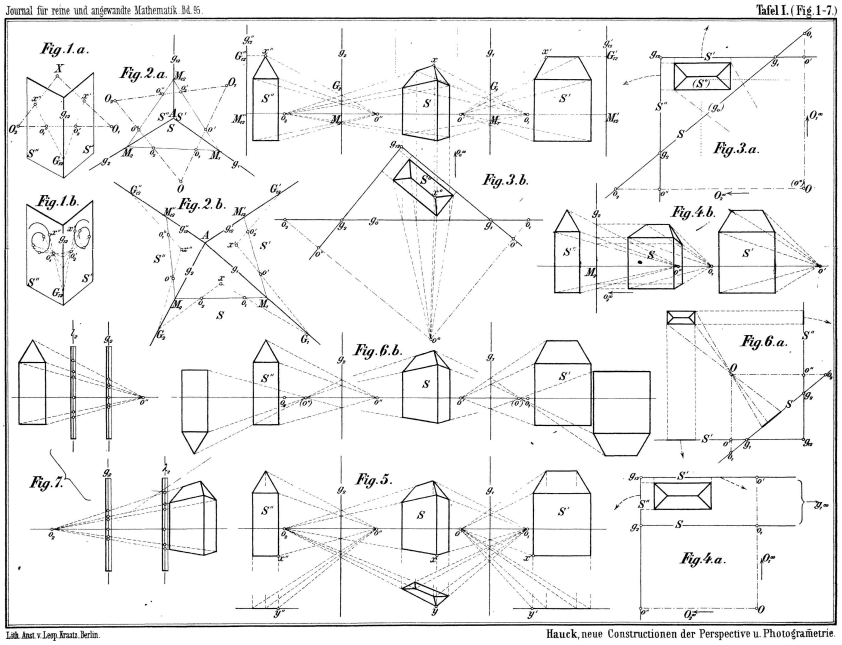


Figure 2.5: *Illustrations from Guido Hauck's paper (Courtesy of wikipedia.org. The copyright of the image has expired).*

presented the first comprehensive description of how to determine the epipole in his Ph.D. thesis [84]. In 1934, a German book entitled "Lehrbuch der Stereophotogrammetrie (Text book of Stereophotogrammetry)" by Baeschlin and Zeller was published [3], and it was translated into English in 1952 by Miskin and Powell with the title "Text book of Photogrammetry" [88]. It was the book that introduced English equivalent terms such as epipoles and epipolar planes.

The usage of the words related to epipolar geometry in photogrammetry is somewhat different from their usage in computer vision because it is assumed that aerial photographs are used in phogrammetry. However, the essential meaning of the words is the same. According to the glossary in the "Manual of Photogrammetry". The terms epipoles, epipolar plane and epipolar ray are defined as follows [70]:

> **epipoles** – In the perspective setup of two photographs (two perspective projections), the points on the planes of the photographs where they are cut by the air base[4] (extended line joining the two perspective centers). In the case of a pair

[4]air base (photogrammetry) – The line joining two air stations, or the length of this line; also, the distance (at the scale of the stereoscopic model) between adjacent perspective centers as reconstructed in the plotting in-

of truly vertical photographs, the epipoles are infinitely distant from the principal points.

epipolar plane – Any plane which contains the epipoles; therefore, any plane containing the air base. Also called basal plane.

epipolar ray – The line on the plane of a photograph joining the epipole and the image of an object. Also expressed as the trace of an epipolar plane on a photograph.

The concept of an essential matrix in computer vision is also related to that in photogrammetry. In 1959, Thompson first presented an equation composed of a skew-symmetric matrix and an orthogonal matrix to determine the relative orientation in photogrammetry [81]. In 1981, in computer vision, Longuet-Higgins was the first to introduce a 3×3 matrix similar to that in Thompson's equation. This matrix was later termed an essential matrix and was used to explain the relationships between points and the lines corresponding to these points in the two views [46].

Following this, several studies were made to derive methods to determine the relative orientation and translation of the two images. In 1991, Horn presented an iterative algorithm to estimate the relative orientation [31]. In 1997, Hartley presented a linear algorithm known as the "normalized 8-point algorithm" to estimate the fundamental matrix, which is the same as the essential matrix except in this case, the cameras are not calibrated [25]. In 1996, Phillip [59] introduced a linear method for estimating essential matrices using five point correspondences, and it obtains the solutions by finding the roots of a 13th-degree polynomial. In 2004, Nister improved on Philip's method by finding the roots of a 10th-degree polynomial [57]. In 2006, Stewénius presented a minimal 5-point method that uses five matching pairs of points and finds the solutions using a Gröbner basis [73, 74].

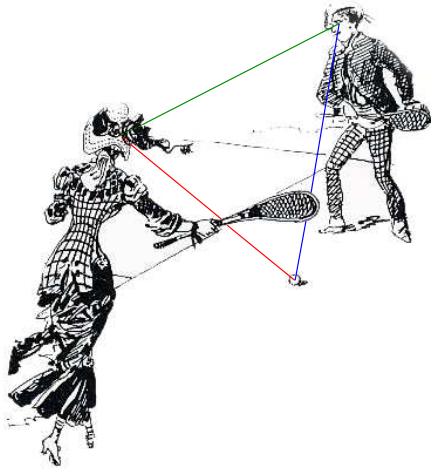


Figure 2.6: *Intuitive illustration of epipolar geometry.*

2.2.3 Interpretation of epipolar geometry

In this section, we first present a simple illustration of epipolar geometry, as shown in Figure 2.6, before defining its mathematical equations. Let us imagine that there are two persons, a lady and a gentleman, playing with a ball. From the viewpoint of the gentleman, he can see both the ball and the lady. Although his eye is directly focused on the ball, both the image of the ball and the lady are projected onto the retina of his eyes. Now, suppose we draw a line from the eye of the lady to the ball. He can now perceive the ball, the eye of the lady and the line. In epipolar geometry, the eye of the lady observed by the gentleman is called an epipole. In addition, the line seen by the gentleman is known as an epipolar line. The epipolar line corresponds to the image of the ball seen by the lady. In the same way, considering the viewpoint of the lady, the gentleman's eye perceived by the lady is called an epipole. If we draw a line from the eye of the gentleman to the ball, the line observed by the lady is another epipolar line. Therefore, given an object in two views, we have two epipoles and two epipolar lines. It is apparent that the ball, the eye of the gentleman and the eye of the lady form a triangle that lies in a single plane. In other words, they are coplanar. In epipolar geometry, this property is known as the epipolar constraint, and it yields an epipolar equation that is used to construct an

strument [70]. air station (photogrammetry) – the point in space occupied by the camera lens at the moment of exposure; also called camera station or exposure station [70].

essential matrix.

2.2.4 Mathematical notation of epipolar geometry

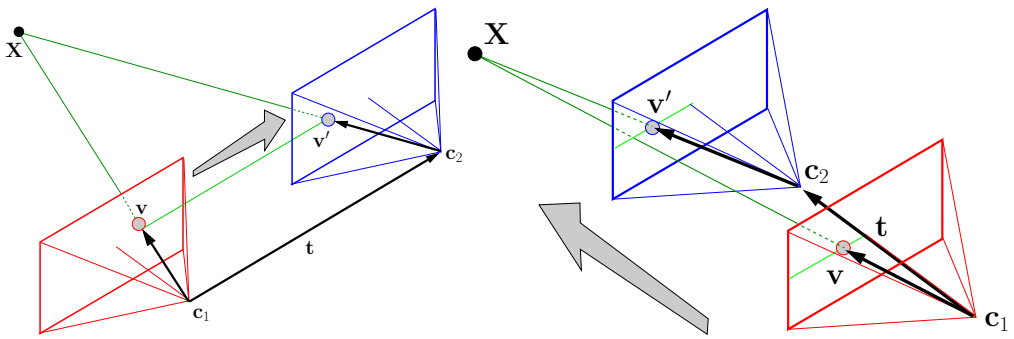
Epipolar geometry is used to explain the geometric relationships between two images. The two images are captured by a single camera that is shifted from one place to another, or they can be captured by two cameras at different locations. Assuming that the cameras are calibrated, the epipolar geometry can be represented by a 3×3 matrix, which is called an essential matrix. The essential matrix describes the relationships between the pairs of matching points in the two images.

Let $\mathbf{v}$ and $\mathbf{v}'$ be points in the first image and in the second image, respectively, that form a matching pair. Without loss of generality, let us assume that a single camera is used to capture the two images, hence, although the camera moves from one position to another, its intrinsic parameters such as the focal length and principal points remain the same.

2.2.4.1 Pure translation (no rotation) case

If we assume that the motion of the camera is translational as it shifts between two positions to capture two images, the essential matrix $\mathtt{E}$, which is used to explain the relationships between point correspondence $\mathbf{v}$ and $\mathbf{v}'$, becomes the simple form of a skew-symmetric matrix as follows:

$$\begin{aligned}
\mathbf{v'}^{\top}\mathtt{E}\mathbf{v} &= \mathbf{v'}^{\top}[\mathbf{t}]_{\times}\mathbf{v} \\
&= \mathbf{v'}^{\top}(\mathbf{t} \times \mathbf{v}) \\
&= \mathbf{v}^{\top}(\mathbf{v}' \times \mathbf{t}) \\
&= \mathbf{t}^{\top}(\mathbf{v} \times \mathbf{v}') \\
&= \begin{vmatrix} t_1 & v_1 & v'_1 \\ t_2 & v_2 & v'_2 \\ t_3 & v_3 & v'_3 \end{vmatrix} = 0 \,,
\end{aligned} \tag{2.13}$$



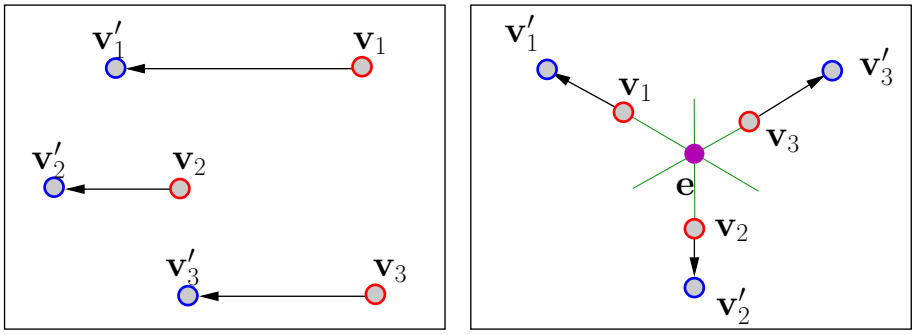
(a) Pure translation to the side (b) Pure translation forward

Figure 2.7: ***Epipolar geometry for a pure translational motion.*** *The camera (indicated in red) at position* $\mathbf{c}_1$ *moves to position* $\mathbf{c}_2$ *(indicated in blue) by pure translation indicated by* $\mathbf{t}$*. A 3D point* $\mathbf{X}$ *is projected to image points* $\mathbf{v}$ *and* $\mathbf{v}'$ *in the first and second view, respectively. The three vectors* $\mathbf{v}$*,* $\mathbf{v}'$ *and* $\mathbf{t}$ *are on an epipolar plane.*

where $\mathbf{t}$ is the translation of the camera and $[\mathbf{a}]_\times$ is a skew-symmetric matrix of any 3-vector $\mathbf{a}$. The translation vector $\mathbf{t}$ and the matching pairs of points $\mathbf{v}$ and $\mathbf{v}'$ can be written as $\mathbf{t} = (t_1, t_2, t_3)^\top$, $\mathbf{v} = (v_1, v_2, v_3)^\top$ and $\mathbf{v}' = (v_1', v_2', v_3')^\top$.

Equation (2.13) is in the form of a scalar triple product of three vectors, $\mathbf{v}$, $\mathbf{v}'$ and $\mathbf{t}$, but it is nothing more than a coplanar constraint on the three vectors. As shown in Figure 2.6, the triangle is formed by three line segments joining three points such as the lady's eye, the gentleman's eye and the ball. This triangle should lie in a single plane. There are three coordinate systems in this situation. The first two coordinate systems are 2D coordinate systems used by the two images taken by the camera. The third coordinate system is the world coordinate system, which shows the position of the two cameras (viewpoints of the two persons) and the ball. Because there is no rotation in this particular pure translation case, the directions of these three vectors are not affected by other coordinate systems. Therefore, it is simply a coplanar condition for three vectors to lie on a plane. The plane is called an epipolar plane in the epipolar geometry.

As shown in Figure 2.7, the vector $\mathbf{v}$ is a projected image vector of a 3D point $\mathbf{X}$ in the first view $\mathbf{c}_1$. The vector $\mathbf{v}'$, corresponding to $\mathbf{v}$, is a projected image vector of the 3D point $\mathbf{X}$ in the second view $\mathbf{c}_2$. The translation vector $\mathbf{t}$ is the same as the displacement of camera positions. Because of the purely translational motion of the cameras, the translation vector $\mathbf{t}$



(a) Pure translation to the side (b) Pure translation to the forward

Figure 2.8: ***Overlapped image vectors*** $\mathbf{v}_i$ ***and*** $\mathbf{v}'_i$ ***on an image.*** *(a) The vectors are parallel for sideways translational motion. (b) The vectors coincide at an epipole* $\mathbf{e}$ *for forward motion.*

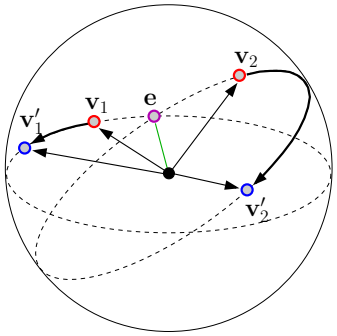


Figure 2.9: *Image vectors on a sphere with an epipole* $\mathbf{e}$ *for a pure translation.*

is in the epipolar plane containing the two image vectors $\mathbf{v}$ and $\mathbf{v}'$. Therefore, a great cirle (plane) joining $\mathbf{v}$ and $\mathbf{v}'$ also contains the translation direction vector $\mathbf{t}$.

We now define a property of pure translational motion. Suppose the image vectors $\mathbf{v}_i$ and $\mathbf{v}'_i$ overlap, as shown in Figure 2.8. Then, for sideways translational motion, the overlapped image vectors $\mathbf{v}_i$ and $\mathbf{v}'_i$ will be parallel. On the other hand, in the case of forward motion, $\mathbf{v}_i$ and $\mathbf{v}'_i$ will meet at a single point. This point is the same as the epipole in the first view.

In other words, this property can also be explained as follows. Suppose the image vectors $\mathbf{v}$ and $\mathbf{v}'$ are on a sphere, as shown in Figure 2.9. The image vectors $\mathbf{v}$ and $\mathbf{v}'$ join a plane (great circle). If there are more than two pairs of matching points such as $\mathbf{v}_i$ and $\mathbf{v}'_i$, where $i = 1, \ldots, n$, and n is the number of point correspondences, then the intersection of these planes forms an epipolar axis containing two epipoles. This property will be used in chapter 10

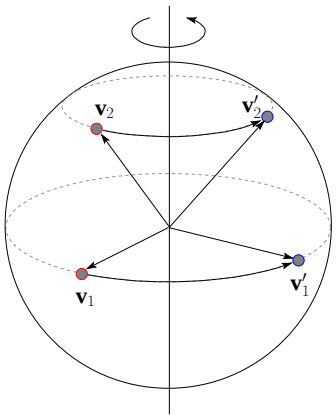


Figure 2.10: *Image vectors on a sphere for the pure rotation case.*

to estimate the relative orientation of two views.

2.2.4.2 Pure rotation (no translation) case

If the motion of the camera is purely rotational when the two images are captured by the camera, the geometric relationships of $\mathbf{v}$ and $\mathbf{v}'$ can be represented as a simple rotation about an axis, as shown in Figure 2.10.

2.2.4.3 Euclidean motion (rotation and translation) case

If the motion of the camera is both rotational and translational, a general form of the essential matrix $\mathtt{E}$ for a pair of matchings points $\mathbf{v}$ and $\mathbf{v}'$ may be written as follows[5]:

$$\begin{aligned} \mathbf{v}'^{\top}\mathtt{E}\mathbf{v} &= \mathbf{v}'^{\top}[\mathbf{t}]_{\times}\mathtt{R}\mathbf{v} && (2.14)\\ &= \mathbf{v}'^{\top}\mathtt{R}[\mathtt{R}^{\top}\mathbf{t}]_{\times}\mathbf{v} && (2.15)\\ &= \mathbf{v}'^{\top}\mathtt{R}[\mathbf{c}]_{\times}\mathbf{v}\ , && (2.16) \end{aligned}$$

where $\mathtt{R}$ is a relative rotation matrix and $\mathbf{t}$ is a translation direction vector. This can be explained as rotating the image vector $\mathbf{v}$ in the first view by $\mathtt{R}$ in order to align the image plane in the first view with that in the second view. After all, $\mathtt{R}\mathbf{v}$ is the image vector in the first image rotated into a coordinate system of the second camera.

[5]See Appendix A.2.

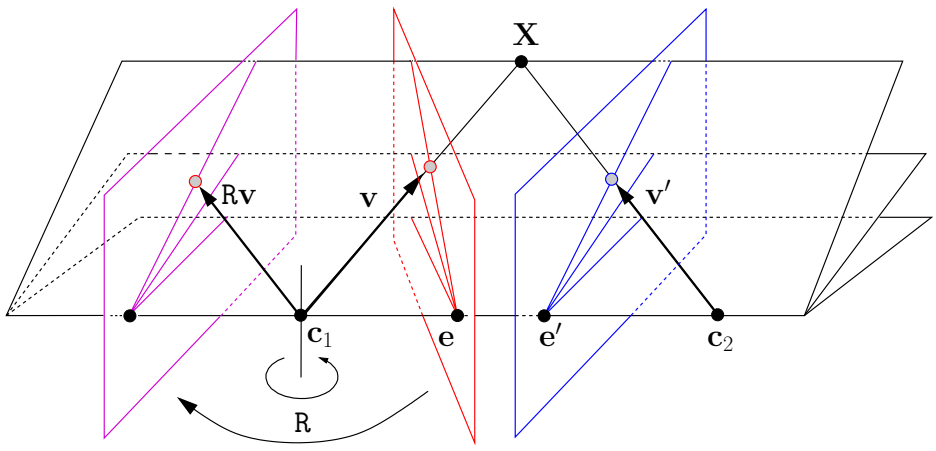


Figure 2.11: *Alignment of the first view (indicated in red) with the second view (indicated in blue) in order to make the two views the same as those in the pure translation case. The virtually aligned view is marked as purple.*

As shown in Figure 2.11, on aligning the image planes, the two image planes become parallel, resulting in a situtation that is similar to the pure translation case. Instead of using the vector $\mathbf{v}$, a rotated image vector $\mathtt{R}\mathbf{v}$ can be used as the vector corresponding to the image vector $\mathbf{v}$. Because the aligned view (indicated in purple) is parallel to the second view (indicated in blue) as shown in Figure 2.11, the image point vectors $\mathbf{v}'$ and $\mathtt{R}\mathbf{v}$ also satisfy the epipolar co-planar constraint as follows:

$$\mathbf{v}'^{\top}[\mathbf{t}]_{\times}(\mathtt{R}\mathbf{v}) = 0\,. \tag{2.17}$$

2.2.4.4 Essential matrix from two camera matrices

Let the two camera matrices be $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ and $\mathtt{P}' = [\mathtt{R} \mid -\mathtt{R}\mathbf{c}] = [\mathtt{R} \mid \mathbf{t}]$, where $\mathtt{R}$ is the relative orientation, $\mathbf{c}$ is the centre of the second view and $\mathbf{t} = -\mathtt{R}\mathbf{c}$ is a translation direction vector. As explained in the previous section, for a given pair of matching points, $\mathbf{v}$ and $\mathbf{v}'$, the essential matrix may be written from the two camera matrices as follows:

$$\mathbf{v}'^{\top}\mathtt{E}\mathbf{v} = \mathbf{v}'^{\top}[\mathbf{t}]_{\times}\mathtt{R}\mathbf{v} = 0\,. \tag{2.18}$$

where $\mathtt{E}$ is the essential matrix from cameras $\mathtt{P}$ and $\mathtt{P}'$.

For a general form of two camera matrices such as $\mathtt{P}_1 = [\mathtt{R}_1 \mid -\mathtt{R}_1\mathbf{c}_1]$ and $\mathtt{P}_2 = [\mathtt{R}_2 \mid -\mathtt{R}_2\mathbf{c}_2]$, the essential matrix from the general form of two camera matrices may be written as follows:

$$\mathbf{v'}^{\top}\mathtt{E}\mathbf{v} = \mathbf{v'}^{\top}\mathtt{R}_2[\mathbf{c}_1 - \mathbf{c}_2]_{\times}\mathtt{R}_1^{\top}\mathbf{v} = 0\,. \tag{2.19}$$

It can be derived from (2.18) by multiplying a 4×4 matrix with the camera matrices $\mathtt{P}_1$ and $\mathtt{P}_2$ as follows:

$$\begin{aligned} \mathtt{P}_1\mathtt{H} &= [\mathtt{R}_1 \mid -\mathtt{R}_1\mathbf{c}_1] \begin{bmatrix} \mathtt{R}_1^{\top} & \mathbf{c}_1 \\ \mathbf{0}^{\top} & 1 \end{bmatrix} && (2.20)\\ &= [\mathtt{I} \mid \mathbf{0}] && (2.21) \end{aligned}$$

and

$$\begin{aligned} \mathtt{P}_2\mathtt{H} &= [\mathtt{R}_2 \mid -\mathtt{R}_2\mathbf{c}_2] \begin{bmatrix} \mathtt{R}_1^{\top} & \mathbf{c}_1 \\ \mathbf{0}^{\top} & 1 \end{bmatrix} && (2.22)\\ &= [\mathtt{R}_2\mathtt{R}_1^{\top} \mid \mathtt{R}_2\mathbf{c}_1 - \mathtt{R}_2\mathbf{c}_2] && (2.23)\\ &= \mathtt{R}_2\mathtt{R}_1^{\top}[\mathtt{I} \mid \mathtt{R}_1(\mathbf{c}_1 - \mathbf{c}_2)]\,. && (2.24) \end{aligned}$$

From (2.21) and (2.24), the essential matrix can be constructed as follows:

$$\begin{aligned} \mathtt{E} &= [\mathtt{R}_2(\mathbf{c}_1 - \mathbf{c}_2)]_{\times}\mathtt{R}_2\mathtt{R}_1^{\top} && (2.25)\\ &= \mathtt{R}_2[\mathbf{c}_1 - \mathbf{c}_2]_{\times}\mathtt{R}_1^{\top}\,. && (2.26) \end{aligned}$$

2.2.4.5 Fundamental matrix

The fundamental matrix is basically the same as the essential matrix except that a calibration matrix is not considered. When calibrated cameras are given, point coordinates in images are represented in pixel units. However, if we assumme that the cameras are calibrated, we can eliminate the pixel units by multiplying the inverse of the calibration matrix with the coordinates of the points. In the fundamental matrix, such image points can be considered as

directional vectors to the corresponding 3D points. Given calibrated cameras and directional vectors of the image points, the essential matrix can be easily obtained. On the other hand, if uncalibrated cameras and pixel coordinates of the image points are provided, we can obtain the fundamental matrix. Simply, given a point correspondence $\mathbf{x}$ and $\mathbf{x}'$ in pixel units, because of the presence of directional image vectors $\mathbf{v} = \mathtt{K}^{-1}\mathbf{x}$ and $\mathbf{v}' = \mathtt{K}^{-1}\mathbf{x}'$, where $\mathtt{K}$ is a calibration of the camera, the fundamental matrix $\mathtt{F}$ may be written as follows:

$$\mathbf{v}'^{\top}\mathtt{E}\mathbf{v} = (\mathtt{K}^{-1}\mathbf{x}')^{\top}\mathtt{E}(\mathtt{K}^{-1}\mathbf{x}) = \mathbf{x}'^{\top}\mathtt{K}^{-\top}\mathtt{E}\mathtt{K}^{-1}\mathbf{x} = \mathbf{x}'^{\top}\mathtt{F}\mathbf{x}\,. \tag{2.27}$$

Therefore, $\mathtt{F} = \mathtt{K}^{-\top}\mathtt{E}\mathtt{K}^{-1}$.

Elements of the fundamental matrix Given a fundamental matrix $\mathtt{F}$, its elements $\mathtt{F}_{ij}$ may be written as

$$\mathtt{F} = \begin{bmatrix} \mathtt{F}_{11} & \mathtt{F}_{12} & \mathtt{F}_{13} \\ \mathtt{F}_{21} & \mathtt{F}_{22} & \mathtt{F}_{23} \\ \mathtt{F}_{31} & \mathtt{F}_{32} & \mathtt{F}_{33} \end{bmatrix}. \tag{2.28}$$

For this $\mathtt{F}$, a pair of matching points $\mathbf{x} = (x_1, x_2, x_3)\top$ and $\mathbf{x}' = (x'_1, x'_2, x'_3)^{\top}$; hence, the equation of epipolar constraints can be given as

$$(x'_1, x'_2, x'_3)\mathtt{F}(x_1, x_2, x_3)^{\top} = 0\,. \tag{2.29}$$

The coefficients of the term $x'_i x_j$ in (2.29) correspond to the elements of $\mathtt{F}$. These elements of $\mathtt{F}$ can be determined from two camera matrices and the position of a 3D point using a bilinear constraint, which will be explained in the following paragraphs.

Bilinear constraints Let $\mathtt{A}$ and $\mathtt{B}$ be two camera matrices. Then, a 3D point $\mathbf{X}$ can be projected by the two camera matrices as $k\mathbf{x} = \mathtt{A}\mathbf{X}$ and $k'\mathbf{x}' = \mathtt{B}\mathbf{X}$, where k and k' are any

non-zero scalar values. These two projections of $\mathbf{X}$ may be written as

$$\begin{bmatrix} \mathtt{A} & \mathbf{x} & \mathbf{0} \\ \mathtt{B} & \mathbf{0} & \mathbf{x}' \end{bmatrix} \begin{pmatrix} \mathbf{X} \\ -k \\ -k' \end{pmatrix} = \mathbf{0} \,. \tag{2.30}$$

If we rewrite (2.30) using the row vectors of the matrices $\mathtt{A}$ and $\mathtt{B}$, and the elements of $\mathbf{x}$ and $\mathbf{x}'$, we can determine the elements of the fundamental matrix $\mathtt{F}$. Suppose the two camera matrices are

$$\mathtt{A} = \begin{bmatrix} \mathbf{a}_1^\top \\ \mathbf{a}_2^\top \\ \mathbf{a}_3^\top \end{bmatrix} \tag{2.31}$$

and

$$\mathtt{B} = \begin{bmatrix} \mathbf{b}_1^\top \\ \mathbf{b}_2^\top \\ \mathbf{b}_3^\top \end{bmatrix} \,, \tag{2.32}$$

then, (2.30) is written as

$$\begin{bmatrix} \mathbf{a}_1^\top & x_1 & & \\ \mathbf{a}_2^\top & x_2 & & \\ \mathbf{a}_3^\top & x_3 & & \\ \mathbf{b}_1^\top & & x_1' & \\ \mathbf{b}_2^\top & & x_2' & \\ \mathbf{b}_3^\top & & x_3' & \end{bmatrix} \begin{pmatrix} \mathbf{X} \\ -k \\ -k' \end{pmatrix} = \mathtt{D} \begin{pmatrix} \mathbf{X} \\ -k \\ -k' \end{pmatrix} = \mathbf{0} \,. \tag{2.33}$$

From the above equation (2.33), the coefficient of the term $x_i' x_j$ is determined by eliminating two rows and the last two columns of the matrix $\mathtt{D}$, and by calculating the determinant of the remaining 4×4 matrix. Therefore, the entries of the fundamental matrix may be written

as follows:

$$\mathtt{F}_{ji} = (-1)^{i+j} det \begin{pmatrix} \sim \mathbf{a}_i^\top \\ \sim \mathbf{b}_j^\top \end{pmatrix} \tag{2.34}$$

where $\sim \mathbf{a}_i^\top$ is a 2×3 matrix created after omitting the i-th row $\mathbf{a}_i^\top$ from the matrix $\mathtt{A}$ and $\sim \mathbf{b}_i^\top$ is a 2×3 matrix is created after omitting the i-th row $\mathbf{b}_i^\top$ from the matrix $\mathtt{B}$. Equation (2.34) is called a "bilinear relation" for two views. The relations for three and four views are known as trilinear relations and quadlinear relations, respectively.

2.3 Estimation of essential matrix

2.3.1 8-point algorithm

Longuet-Higgins was the first to develop the 8-point algorithm in computer vision, which estimates the essential matrix using 8 pairs of matching points across two views [46]. Unlike Thompson's iterative method using 5 point correspondences [81], which solves five third-order equations iteratively, the 8-point method directly obtains the solution from linear equations.

Given the point correspondences $\mathbf{v} = (v_1, v_2, v_3)^\top$ and $\mathbf{v}' = (v_1', v_2', v_3')^\top$, the 3×3 essential matrix $\mathtt{E}$ can be dervied as follows:

$$\mathbf{v'}^\top \mathtt{E} \mathbf{v} = (v_1', v_2', v_3') \begin{bmatrix} \mathtt{E}_{11} & \mathtt{E}_{12} & \mathtt{E}_{13} \\ \mathtt{E}_{21} & \mathtt{E}_{22} & \mathtt{E}_{23} \\ \mathtt{E}_{31} & \mathtt{E}_{32} & \mathtt{E}_{33} \end{bmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = 0 \, . \tag{2.35}$$

A linear equation may be obtained from (2.35) as follows:

$$(v'_1v_1,\ v'_1v_2,\ v'_1v_3,\ v'_2v_1,\ v'_2v_2,\ v'_2v_3,\ v'_3v_1,\ v'_3v_2,\ v'_3v_3) \begin{bmatrix} \mathtt{E}_{11} \\ \mathtt{E}_{12} \\ \mathtt{E}_{13} \\ \mathtt{E}_{21} \\ \mathtt{E}_{22} \\ \mathtt{E}_{23} \\ \mathtt{E}_{31} \\ \mathtt{E}_{32} \\ \mathtt{E}_{33} \end{bmatrix} = 0\ . \tag{2.36}$$

It can be observed that equation (2.36) has nine unknowns parameters. However, if we assume that the value of last coordinate of the matching points is one, for example, $v_3 = 1$ and $v'_3 = 1$, the equation has eight unknowns to be solved. Therefore, as there are eight independent equations for eight pairs of matching points, equation (2.36) can be solved directly.

In order to determine the relative orientation and translation of the camera system from the estimated essential matrix, Longuet-Higgins proposed a method wherein the translation vector can be obtained by multiplying the transpose of the essential matrix with (2.18) as follows:

$$\begin{aligned} \mathtt{E}\mathtt{E}^\top &= ([\mathbf{t}]_\times \mathtt{R})([\mathbf{t}]_\times \mathtt{R})^\top && (2.37)\\ &= ([\mathbf{t}]_\times \mathtt{R}\mathtt{R}^\top [\mathbf{t}]_\times^\top && (2.38)\\ &= [\mathbf{t}]_\times [\mathbf{t}]_\times^\top\ . && (2.39) \end{aligned}$$

If we perform the trace of $\mathtt{E}\mathtt{E}^\top$, it becomes $\mathrm{Tr}(\mathtt{E}\mathtt{E}^\top) = \mathrm{Tr}([\mathbf{t}]_\times [\mathbf{t}]_\times^\top) = 2||\mathbf{t}||^2$. By assuming $\mathbf{t}$ to be a unit vector, i.e., $||\mathbf{t}|| = 1$, the trace of $\mathtt{E}\mathtt{E}^\top$ can be given as

$$\mathrm{Tr}(\mathtt{E}\mathtt{E}^\top) = 2\ . \tag{2.40}$$

Therefore, the essential matrix $\mathtt{E}$ can be normalized by dividing it by $\sqrt{\frac{1}{2}\mathrm{Tr}(\mathtt{E}\mathtt{E}^\top)}$. After

obtaining the normalized essential matrix, the direction of the translation vector $\mathbf{t}$ is determined using the main diagonal of $\mathtt{E}\mathtt{E}^\top$ as follows:

$$\mathtt{E}\mathtt{E}^\top = \begin{bmatrix} t_3^2 + t_2^2 & -t_2 t_1 & -t_3 t_1 \\ -t_2 t_1 & t_3^2 + t_1^2 & -t_3 t_2 \\ -t_3 t_1 & -t_3 t_2 & t_2^2 + t_1^2 \end{bmatrix} = \begin{bmatrix} 1 - t_1^2 & -t_2 t_1 & -t_3 t_1 \\ -t_2 t_1 & 1 - t_2^2 & -t_3 t_2 \\ -t_3 t_1 & -t_3 t_2 & 1 - t_3^2 \end{bmatrix} , \tag{2.41}$$

where $t_1^2 + t_2^2 + t_3^2 = 1$ because $\mathbf{t}$ is a unit vector. From the main diagonal of $\mathtt{E}\mathtt{E}^\top$, we can obtain three independent elements of the translation vector $\mathbf{t}$. However, the scale of $\mathbf{t}$ cannot be determined.

In order to find a relative orientation, Longuet-Higgins used the fact that each row of the rotation matrix is orthogonal to each row of the essential matrix. Let us suppose $\mathbf{q}_i$ and $\mathbf{r}_i$ are the i-th column vectors of the essential matrix $\mathtt{E}$ and the rotation matrix $\mathtt{R}$ contained in $\mathtt{E}$, respectively. They may be written as

$$\mathtt{E} = \begin{bmatrix} \mathbf{q}_1 & \mathbf{q}_2 & \mathbf{q}_3 \end{bmatrix} \tag{2.42}$$

and

$$\mathtt{R} = \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \end{bmatrix} . \tag{2.43}$$

Then, because $[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}$ satisfies for any 3-vector $\mathbf{a}$ and $\mathbf{b}$, we can derive the following relations from (2.18) as follows:

$$\mathbf{q}_i = \mathbf{t} \times \mathbf{r}_i , \tag{2.44}$$

where $\mathbf{q}_i$ is the i-th column vector of the essential matrix $\mathtt{E}$, and $\mathbf{r}_i$ is the i-th column vector of the rotation matrix $\mathtt{R}$, where $i = 1, \ldots, 3$.

Because $\mathbf{r}_i$ is orthogonal to $\mathbf{q}_i$ and is coplanar with $\mathbf{t}$, the vector $\mathbf{r}_i$ can be written as a linear combination of $\mathbf{q}_i$ and $\mathbf{q}_i \times \mathbf{t}$. If we define a new vector $\mathbf{w}_i = \mathbf{q}_i \times \mathbf{t}$, then

$$\mathbf{r}_i = \lambda_i \mathbf{t} + \mu_i \mathbf{w}_i \tag{2.45}$$

where λ_i and μ_i are any scalar values. Here, the unknown scalar μ_i is determined to be $\mu_i = 1$ by substituting (2.45) into (2.44) as follows:

$$\begin{aligned}
\mathbf{q}_i &= \mathbf{t} \times \mathbf{r}_i && (2.46)\\
&= \mathbf{t} \times (\lambda \mathbf{t} + \mu_i \mathbf{w}_i) && (2.47)\\
&= \mu_i (\mathbf{t} \times \mathbf{w}_i) && (2.48)\\
&= \mu_i \mathbf{t} \times (\mathbf{q}_i \times \mathbf{t}) && (2.49)\\
&= \mu_i \mathbf{q}_i \ . && (2.50)
\end{aligned}$$

Because the rotation matrix R is an orthogonal matrix, the cross products of any two column vectors of R are the same as the elements of the remaining column vector of R. For example, $\mathbf{r}_1 = \mathbf{r}_2 \times \mathbf{r}_3$. Therefore, from (2.46), (2.45) and $\mu = 1$, we obtain

$$\begin{aligned}
\mathbf{r}_1 &= \mathbf{r}_2 \times \mathbf{r}_3 \\
\lambda_1 \mathbf{t} + \mathbf{w}_1 &= (\lambda_2 \mathbf{t} + \mathbf{w}_2) \times (\lambda_3 \mathbf{t} + \mathbf{w}_3) \\
&= (\lambda_2 \mathbf{t} + \mathbf{w}_2) \times (\lambda_3 \mathbf{t} + \mathbf{w}_3) \\
&= \lambda_2 \lambda_3 \mathbf{t} \times \mathbf{t} + \lambda_2 \mathbf{t} \times \mathbf{w}_3 + \lambda_3 \mathbf{w}_2 \times \mathbf{t} + \mathbf{w}_2 \times \mathbf{w}_3 \\
&= \lambda_2 (\mathbf{t} \times \mathbf{w}_3) - \lambda_3 (\mathbf{t} \times \mathbf{w}_2) + \mathbf{w}_2 \times \mathbf{w}_3 \\
&= \lambda_2 \mathbf{q}_3 - \lambda_3 \mathbf{q}_2 + \mathbf{w}_2 \times \mathbf{w}_3 \\
&= \lambda_2 \mathbf{q}_3 - \lambda_3 \mathbf{q}_2 + (\mathbf{q}_2 \times \mathbf{t}) \times (\mathbf{q}_3 \times \mathbf{t}) \\
&= \lambda_2 \mathbf{q}_3 - \lambda_3 \mathbf{q}_2 + \det(\mathbf{q}_2\ \mathbf{t}\ \mathbf{t}) \mathbf{q}_3 - \det(\mathbf{q}_2\ \mathbf{t}\ \mathbf{q}_3) \mathbf{t} \\
&= \lambda_2 \mathbf{q}_3 - \lambda_3 \mathbf{q}_2 + \mathbf{q}_2^\top (\mathbf{t} \times \mathbf{t}) \mathbf{q}_3 - \mathbf{q}_2^\top (\mathbf{t} \times \mathbf{q}_3) \mathbf{t} \\
&= \lambda_2 \mathbf{q}_3 - \lambda_3 \mathbf{q}_2 - \mathbf{q}_2^\top (\mathbf{t} \times \mathbf{q}_3) \mathbf{t} \ .
\end{aligned}$$

Because $\mathbf{w}_1$, $\mathbf{q}_3$ and $\mathbf{q}_4$ are all orthogonal to $\mathbf{t}$ and the last term on the right in (2.51) is a multiple of $\mathbf{t}$, the above equation becomes

$$\lambda_1 \mathbf{t} = -\mathbf{q}_2^\top (\mathbf{t} \times \mathbf{q}_3) \mathbf{t} = \mathbf{w}_2 \times \mathbf{w}_3 \ . \qquad (2.51)$$

On substituting the above equation into (2.45), we obtain the final equation of each column vector of the roation matrix R as follows:

$$\mathbf{r}_1 = \mathbf{w}_1 + \mathbf{w}_2 \times \mathbf{w}_3 \tag{2.52}$$

$$\mathbf{r}_2 = \mathbf{w}_2 + \mathbf{w}_3 \times \mathbf{w}_1 \tag{2.53}$$

$$\mathbf{r}_3 = \mathbf{w}_3 + \mathbf{w}_1 \times \mathbf{w}_2 \, . \tag{2.54}$$

Although we have estimated the relative orientation and translation using 8 pairs of matching points, there are four possible solutions if we consider signs of the orientations and translations. In order to identify the signs, Longuet-Higgins proposed a 3D-point reconstruction method and determined the signs of the translation and rotation on the basis of the values of the last coordinates of the reconstructed 3D points. If the values of the last coordinates of a pair of 3D points are negative, then the sign of the translation changes. If the values of the last coordinates of the 3D points are opposite in sign to each other, then the sign of the rotation is reversed.

2.3.2 Horn's nonlinear 5-point method

Horn proposed a method to determine the relative orientation (rotation) and baseline (translation) of the motion of a camera system using 5 pairs of matching points across the two views [30]. The rotation of the first camera coordinate system with respect to the second camera is known as the relative orientation. There are five unknowns parameteres – 3 for rotation and 2 for translation – in the essential matrix.

Given a pair of matching points $\mathbf{v}$ and $\mathbf{v}'$, $\mathtt{R}\mathbf{v}$ is the image vector in the first view rotated into the coordinate system of the right view (or camera), where R is the relative rotation with respect to the other view. For these two views, there is a coplanar condition, known as the epipolar constraint, for the image vectors $\mathtt{R}\mathbf{v}$, $\mathbf{v}'$ and the translation vector $\mathbf{t}$ as follows:

$$\mathbf{v}'^{\top} [\mathbf{t}]_{\times} \mathtt{R}\mathbf{v} = 0 \, . \tag{2.55}$$

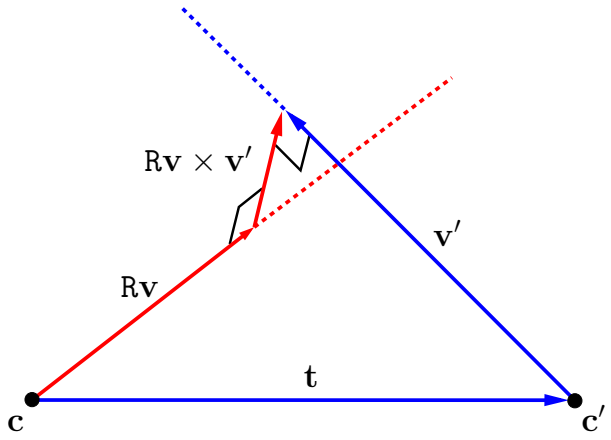


Figure 2.12: *The shortest distance is a line segment between the two image vectors* $\mathbf{v}'$ *and* $\mathtt{R}\mathbf{v}$.

Considering the cost function to be minimized to solve these 5 unknowns for the essential matrix, the shortest distance between two rays is that between two image vectors $\mathtt{R}\mathbf{v}$ and $\mathbf{v}'$. Figure 2.12 shows the shortest distance. This shortest distance is determined by measuring the length of the line segment that intersects $\mathbf{v}'$ and $\mathtt{R}\mathbf{v}$, which is parallel to $\mathtt{R}\mathbf{v} \times \mathbf{v}'$. Because the sum of $\mathbf{t}$ and $\mathbf{v}'$ is the same as the sum of $\mathtt{R}\mathbf{v}$ and $\mathtt{R}\mathbf{v} \times \mathbf{v}'$, we obtain the following equations:

$$\alpha \mathtt{R}\mathbf{v} + \gamma(\mathtt{R}\mathbf{v} \times \mathbf{v}') = \mathbf{t} + \beta \mathbf{v}' \,, \tag{2.56}$$

where the values of α and β are proportional to their distances along the first and second image vector to the points where they approach closely, while the value of γ is proportional to the value of the shortest distance between the image vectors. By calculating the dot product of (2.56) with $\mathtt{R}\mathbf{v} \times \mathbf{v}'$, $\mathbf{v}' \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')$ and $\mathtt{R}\mathbf{v} \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')$, we obtain the following equations as follows:

For γ:

$$\alpha \mathtt{R}\mathbf{v} + \gamma(\mathtt{R}\mathbf{v} \times \mathbf{v}') = \mathbf{t} + \beta\mathbf{v}' \tag{2.57}$$

$$\alpha(\mathtt{R}\mathbf{v})^\top(\mathtt{R}\mathbf{v} \times \mathbf{v}') + \gamma||\mathtt{R}\mathbf{v} \times \mathbf{v}'||^2 = \mathbf{t}^\top(\mathtt{R}\mathbf{v} \times \mathbf{v}') + \beta\mathbf{v}'^\top(\mathtt{R}\mathbf{v} \times \mathbf{v}') \tag{2.58}$$

$$\gamma||\mathtt{R}\mathbf{v} \times \mathbf{v}'||^2 = \mathbf{t}^\top(\mathtt{R}\mathbf{v} \times \mathbf{v}') \tag{2.59}$$

$$\gamma||\mathtt{R}\mathbf{v} \times \mathbf{v}'||^2 = \mathbf{v}'^\top[\mathbf{t}]_\times\mathtt{R}\mathbf{v} \tag{2.60}$$

for α:

$$\alpha \mathtt{R}\mathbf{v} + \gamma(\mathtt{R}\mathbf{v} \times \mathbf{v}') = \mathbf{t} + \beta\mathbf{v}' \tag{2.61}$$

$$\begin{aligned}\alpha(\mathtt{R}\mathbf{v})^\top(\mathbf{v}' \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')) + \gamma(\mathtt{R}\mathbf{v} \times \mathbf{v}')^\top(\mathbf{v}' \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')) \\ = \mathbf{t}^\top(\mathbf{v}' \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')) + \beta\mathbf{v}'^\top(\mathbf{v}' \times (\mathtt{R}\mathbf{v} \times \mathbf{v}'))\end{aligned} \tag{2.62}$$

$$\alpha||\mathtt{R}\mathbf{v} \times \mathbf{v}'||^2 = \mathbf{t}^\top(\mathbf{v}' \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')) \tag{2.63}$$

$$\alpha||\mathtt{R}\mathbf{v} \times \mathbf{v}'||^2 = (\mathtt{R}\mathbf{v} \times \mathbf{v}')^\top(\mathbf{t} \times \mathbf{v}') \tag{2.64}$$

for β:

$$\alpha \mathtt{R}\mathbf{v} + \gamma(\mathtt{R}\mathbf{v} \times \mathbf{v}') = \mathbf{t} + \beta\mathbf{v}' \tag{2.65}$$

$$\begin{aligned}\alpha(\mathtt{R}\mathbf{v})^\top(\mathtt{R}\mathbf{v} \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')) + \gamma(\mathtt{R}\mathbf{v} \times \mathbf{v}')^\top(\mathtt{R}\mathbf{v} \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')) \\ = \mathbf{t}^\top(\mathtt{R}\mathbf{v} \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')) + \beta\mathbf{v}'^\top(\mathtt{R}\mathbf{v} \times (\mathtt{R}\mathbf{v} \times \mathbf{v}'))\end{aligned} \tag{2.66}$$

$$0 = \mathbf{t}^\top(\mathtt{R}\mathbf{v} \times (\mathtt{R}\mathbf{v} \times \mathbf{v}')) - \beta||\mathtt{R}\mathbf{v} \times \mathbf{v}'||^2 \tag{2.67}$$

$$\beta||\mathtt{R}\mathbf{v} \times \mathbf{v}'||^2 = (\mathtt{R}\mathbf{v} \times \mathbf{v}')^\top(\mathbf{t} \times \mathtt{R}\mathbf{v}) \tag{2.68}$$

For a given rotation, Horn showed a closed form of the least squares solution for the baseline direction by minimizing

$$\sum_{i=1}^{n} w_i \mathbf{t}^\top(\mathtt{R}\mathbf{v}_i \times \mathbf{v})^2 \tag{2.69}$$

where w_i is a weighting factor. This 5-point algorithm adjusts the rotation and the baseline

iteratively until a desired value of error is obtained. Further details can be found in [30].

2.3.3 Normalized 8-point method

Longuet-Higgins introduced an 8-point method for a given set of 8 point correspondences in two images [46]. In uncalibrated cameras, the fundamental matrix has the same properties as an essential matrix, except for the use of pixel coordinates for images. However, Longuet-Higgins's 8-point method can not be used for uncalibrated cameras in practical applications because of its sensitivity to noise. An improved and robust method of estimating a fundamental matrix was presented by Hartley in which coordinates of the points in the images are normalized [25].

Hartley pointed out that the main reason for errors in the 8-point method was the acceptable range of pixel coordinates of homogeneous 3-vectors, and it eventually relates to the condition number of the SVD. The pixel coordinates usually range from zero to a few thousands and they are in the first two elements of the homogeneous 3-vector of the points. However, the value of the last element of the homogeneous 3-vector is always one. Therefore, the SVD of the equation of epipolar constraints returns one huge singular value but relatively small singluar values for other elements of the solution vector. In order to resolve this problem, normalization of the homogeneous point coordinates is performed by moving them to an origin at the centroid of all the points and by scaling them to have a mean distance $\sqrt{2}$ in [25].

Let $\mathtt{T}$ be the transformation matrix of all 2D image coordinates. Then, a fundamental matrix $\mathtt{F}_n$ can be expressed using the transformation matrix and image coordinates $\mathbf{x}$ and $\mathbf{x}'$ as follows:

$$\mathbf{x}'^{\top}\mathtt{T}^{\top}\mathtt{F}_n\mathtt{T}\mathbf{x} = 0\,. \tag{2.70}$$

Therefore, the fundamental matrix $\mathtt{F}$ for unnormalized image coordinates $\mathbf{x}$ and $\mathbf{x}'$ is obtained by multiplying the inverse of a matrix $\mathtt{T}^{\top}$ and $\mathtt{T}$ with each side of (2.70) as follows:

$$\mathtt{F} = \mathtt{T}^{-\top}\mathtt{F}_n\mathtt{T}^{-1}\,. \tag{2.71}$$

2.3.4 5-point method using a Gröbner basis

Stewénius et al. derived a solution to the minimal five-point relative pose problem by using a Gröbner basis [71]. The minimal five-point solver requires five point correspondences and it returns up to 10 real solutions. These 10 solutions can be found by solving polynomial equations using a Gröbner basis.

There exist three epipolar constraints for the minimal five-point problem: the coplanar constraint, the rank constraint and the trace constraint. They are given as follows:

$$\mathbf{v}'^{\top}\mathtt{E}\mathbf{v} = 0 \tag{2.72}$$

$$\det(\mathtt{E}) = 0 \tag{2.73}$$

$$2\mathtt{E}\mathtt{E}^{\top}\mathtt{E} - \mathrm{trace}(\mathtt{E}\mathtt{E}^{\top})\mathtt{E} = 0\ , \tag{2.74}$$

where $\mathtt{E}$ is a 3×3 essential matrix. The rank constraint is derived from the fact that the rank of the essential matrix is two. The trace constraint is derived by Philip in [59].

Using these constraints, Stewénius et al. derived 10 polynomial equations of three unknown parameters, and then, they obtained up to 10 solutions of the polynomial equations using a Gröbner basis.

Li and Hartley also proposed a 5-point method which solves the relative motion of two views [44]. Their method provides a simpler algorithm than Stewénius's method.

2.3.5 The L_∞ method using a branch-and-bound algorithm

Hartley and Kahl performed a study to obtain a global solution for the essential matrix in terms of the geometric relations between two views [21]. There were no algorithms before this that proved a geometrical optimality for the essential matrix in L_∞ norm minimization.

Unlike previous methods of estimation of essential matrix (the 8-point method, the normalized 8-point method and Stewénius's 5-point method), Hartley and Kahls's method provides a method to search a global solution under L_∞ norm using a branch-and-bound algorithm, which makes the search faster than a exhaustive search over all the rotation space. They also

showed that the speed of searching over the rotation space can be remarkably reduced if we test the feasibility of linear programming in an earlier step.

Chapter 3

Two- and Three-camera Systems

Because we focus on multi-camera systems which have more than three cameras, it is not an essential topic in this thesis to investigate details on two-camera systems and three-camera systems. However, in this chapter, we give a brief introduction to the usage of two-camera systems, a trifocal tensor and three-camera systems in multiple view geometry.

A two-camera system comprises a set of two cameras that are physically connected together and that simultaneously capture images. Similarly, a three-camera system is a set of three cameras that are connected together and that capture images at the same time. Stereo (binocular) cameras are well-known examples of two-camera systems. In this chapter, we discuss the characteristics and rigid motion of the two/three-camera systems.

3.1 Two-camera systems (stereo or binocular)

A stereo camera is a type of camera that has two lenses that enable it to capture two pictures at the same time from different positions. Similar to the manner in which a human being uses two eyes to obtain the depth of an object in front their eyes, a stereo camera performs stereoscopy of the two images to determine the depth of the object in front of the camera.

There are many terminologies related to stereoscopy such as stereopsis, binocular vision, stereoscopic imaging and 3D imaging. Streopsis was first invented by Charles Wheatstone in 1838 and his research on binocular vision is available in [85]. Stereoscopy is used in photogrammetry to obtain 3D geographic data from aerial photographs. Stereo cameras were widely used as scientific equipment and also as devices for artistic purposes in the early 20th century. One of the stereo camera manufactured by the Eastman Kodak company is shown

Figure 3.1: *Stereo camera and photographs taken by the stereo camera: Kodak Brownie Hawkeye Stereo Model 4, manufactured between 1907 and 1911. (Courtesy of Timothy Crabtree from* `www.deviantart.com`*, All copyrights reserved* ©*2008. Reprinted with permission.)*

in Figure 3.1. In computer vision, stereo cameras are used to determine the depth of an object using close-range photogrammetry, as shown in Figure 3.2.

3.2 Motion estimation using stereo cameras

Not only a single image from a stereo camera system, but also two images from the stereo camera system can be considered as one application of the two-camera systems. There are many studies about motion estimation from stereo images as follows. In 1983, Moravec et al. developed a robot known as Stanford Cart, which is equipped with nine-eyed stereo cameras [51]. The robot used the stereo camera system to avoid obstacles on the path of the robot. Matthies and Shafer introduced an ellipsoid (3D Gaussian) error model for stereo navigation system in 1987 [49]. Young and Chellappa proposed a motion model for a stereo camera system using a Kalman filter in 1990 [87]. Zhang and Faugueras showed a method to estimate the motion of a stereo camera system using pairs of matching 3D line segments and an extended Kalman

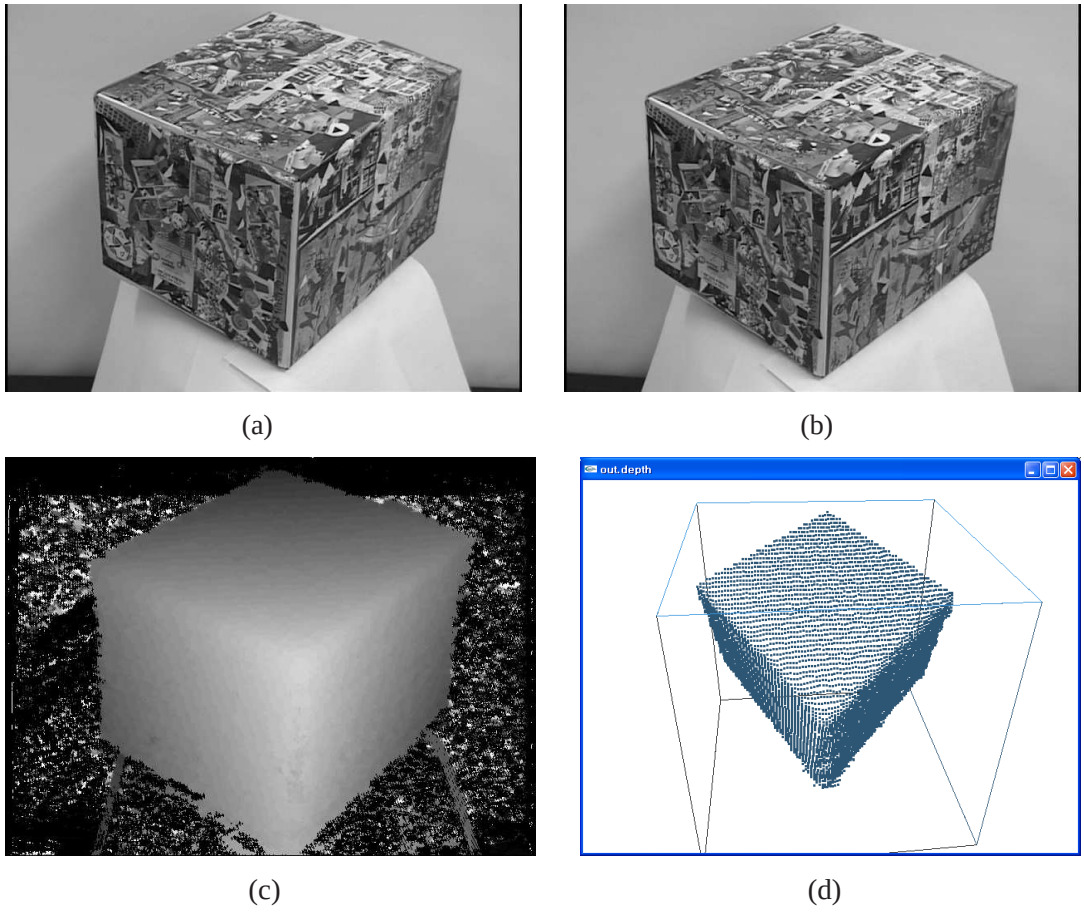


Figure 3.2: ***Stereoscopy and dense disparity map reconstruction.*** *(a) Left image, (b) right image, (c) dense disparity map and (d) 3D view of the stereo reconstruction*

filter in 1992 [89]. In 1995, Matthies et al. developed a real-time stereo vision system to detect obstacles in terrain data [48]. In 1998, Se and Brady presented a stereo vision system to detect obstacles for partially sighted people [68]. Ferrari et al. developed a simple real-time stereo system to avoid obstacles in unknown environment in 1990 [12]. In 1997, Konolige showed SRI's small vision module (SVM) which is used to compute dense stereo range images in real time [41]. Molton and Brady introduced multiple stereo match hypotheses and a Kalman filter for tracking 3D reconstructed points [50].

3.3 Three-camera systems (trinocular)

A three-camera system is a set of three cameras that physically connected together and that simultaneously capture images from different positions. Unlike stereo camera systems, three-camera systems have not been widely used in commercial products. However, it is still worth studying three-camera systems for scientific purporses.

3.4 Trifocal tensor

The geometric relationships between three views or between three cameras may be described mathematically using a trifocal tensor. The trifocal tensor $\mathtt{T}_i$, where $i = 1, \ldots, 3$, consists of three 3×3 matrices with 18 independent degrees of freedom and yields geometric relationships between 3 pairs of lines and/or points such as the line-line-line correspondence, the point-line-line correspondence, the point-line-point correspondence, the point-point-line correspondence and the point-point-point correspondence.

Let a line in 3-space be represented as a 4-vector in projective space $\mathbb{P}^3$. Similarly, let a line projected in images be represented as a 3-vector in projective space $\mathbb{P}^2$. Suppose $\mathbf{l}$, $\mathbf{l}'$ and $\mathbf{l}''$ are 3 pairs of matching lines in the first, second and third view, respectively.

As shown in Figure 3.3, each of the lines $\mathbf{l}$, $\mathbf{l}'$ and $\mathbf{l}''$ back-project to the planes π, π' and π'', respectively. Suppose that the three camera projection matrices are $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$, $\mathtt{P}' = [\mathtt{A} \mid \mathbf{a}_4]$ and $\mathtt{P}'' = [\mathtt{B} \mid \mathbf{b}_4]$ for the first, second and third camera, respectivley. Then, the back-projected

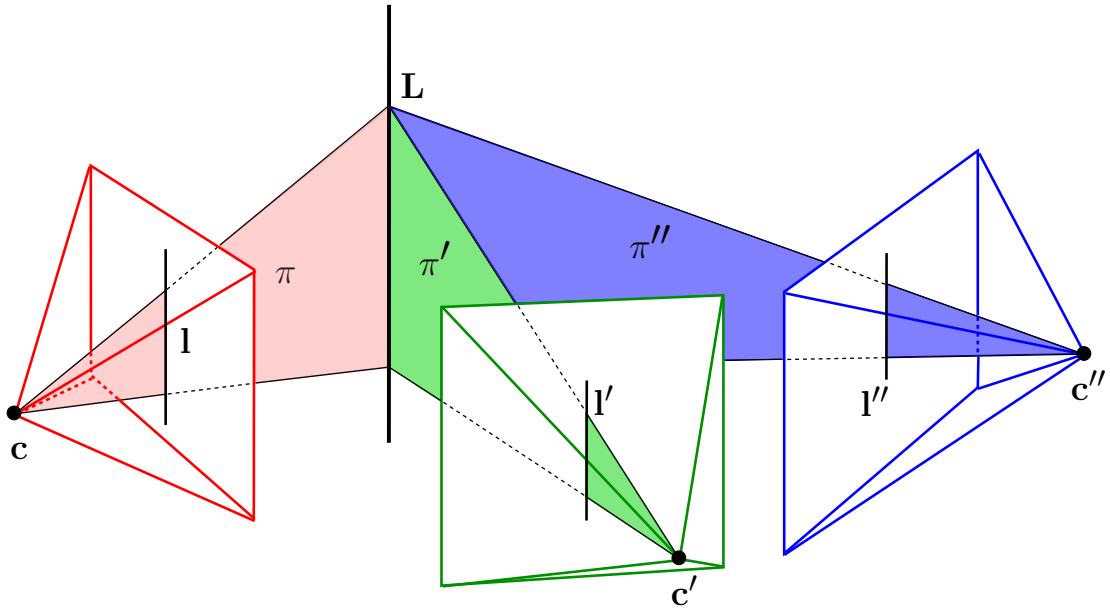


Figure 3.3: ***The line-line-line correspondence and trifocal tensor.*** *A line* $\mathbf{L}$ *is projected onto the images of the three cameras as* $\mathbf{l}$, $\mathbf{l}'$ *and* $\mathbf{l}''$ *for the first, second and third camera, respectively. The projected lines back-project to planes such as* π, π' *and* π'', *respectively. The trifocal tensor describes the geometric relationships between three cameras given the corresponding lines.*

planes can be represented as follows:

$$\pi = \mathtt{P}^\top \mathbf{l} = \begin{pmatrix} \mathbf{l} \\ \mathbf{0} \end{pmatrix} \tag{3.1}$$

$$\pi' = \mathtt{P}'^\top \mathbf{l}' = \begin{pmatrix} \mathtt{A}^\top \mathbf{l}' \\ \mathbf{a}_4{}^\top \mathbf{l}' \end{pmatrix} \tag{3.2}$$

$$\pi'' = \mathtt{P}''^\top \mathbf{l}'' = \begin{pmatrix} \mathtt{B}^\top \mathbf{l}'' l \\ \mathbf{b}_4{}^\top \mathbf{l}'' \end{pmatrix} . \tag{3.3}$$

A 4×3 matrix consisting of columns π, π' and π'' should have a rank of 2 that the three planes meet in a single line $\mathbf{L}$ in 3-space $\mathbb{P}^3$. From this constraint, we can derive three 3×3 matrices $\mathtt{T}_i$, where $i = 1, \ldots, 3$, as follows:

$$\mathtt{T}_i = \mathbf{a}_i \mathbf{b}_4{}^\top - \mathbf{a}_4 \mathbf{b}_i{}^\top , \tag{3.4}$$

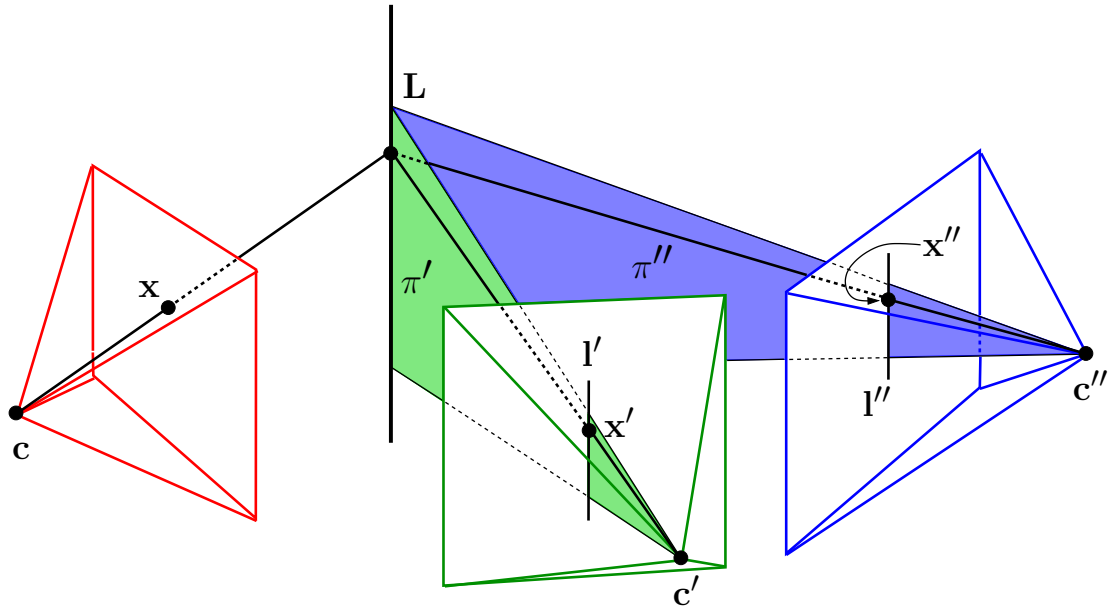


Figure 3.4: ***The point-line-line correspondence.*** *A line* $\mathbf{L}$ *in 3D space is projected as the lines* l' *and* l'' *in the images of the second and third camera, respectively. The projected lines* l' *and* l'' *back-project to planes such as* π' *and* π''*, respectively. A point* $\mathbf{X}$ *on the line* $\mathbf{L}$ *is projected on a point* $\mathbf{x}$ *by the first camera.*

where $i = 1, \ldots, 3$.

For a given set of corresponding lines $\mathrm{l} \leftrightarrow \mathrm{l}' \leftrightarrow \mathrm{l}''$, the relation between these three lines may be written as

$$\mathrm{l}^\top = (\mathrm{l}'^\top \mathtt{T}_1 \mathrm{l}'', \;\; \mathrm{l}'^\top \mathtt{T}_2 \mathrm{l}'', \;\; \mathrm{l}'^\top \mathtt{T}_3 \mathrm{l}'') = \mathrm{l}'^\top [\mathtt{T}_1, \mathtt{T}_2, \mathtt{T}_3] \mathrm{l}'' \,, \tag{3.5}$$

where $[\mathtt{T}_1, \mathtt{T}_2, \mathtt{T}_3]$ is a notation of three 3×3 matrices for the trifocal tensor.

Let us consider a point $\mathbf{x}$ in the first image and two lines l' and l'' in the other two images of a three-camera system. Suppose the point $\mathbf{x}$ and two lines l' and l'' are in a correspondence, as shown in Figure 3.4. Then, the point-line-line correspondence may be written using the trifocal tensor $\mathtt{T}_i$ as follows:

$$\mathrm{l}'^\top (\sum_{i=1}^{3} x^i \mathtt{T}_i) \mathrm{l}'' = 0 \quad \text{for a correspondence} \;\; \mathbf{x} \leftrightarrow \mathrm{l}' \leftrightarrow \mathrm{l}'' \tag{3.6}$$

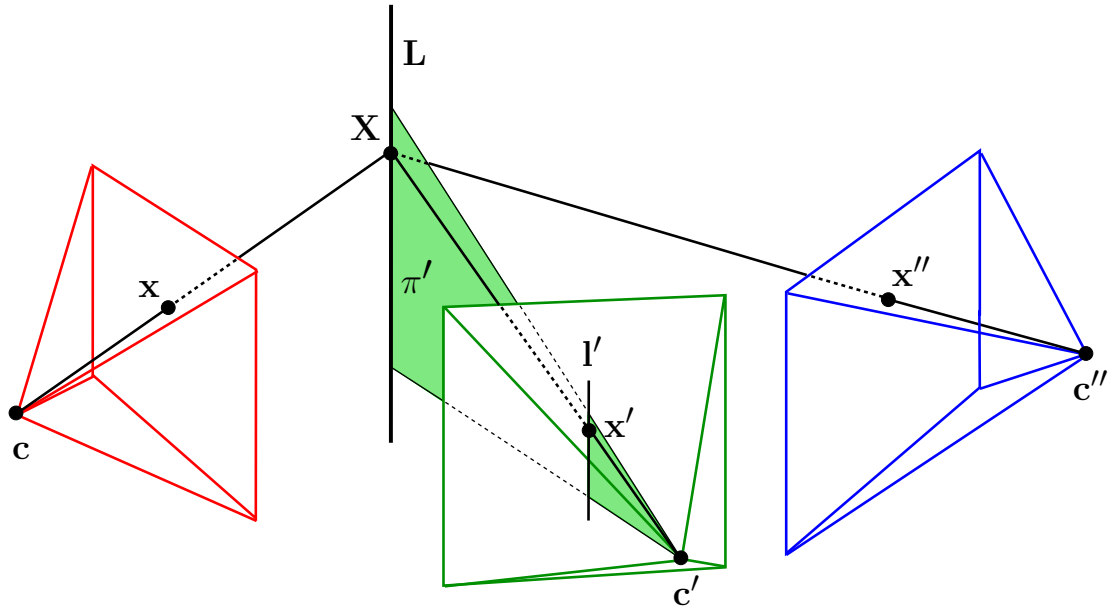


Figure 3.5: ***The point-line-point correspondence.*** *A line* $\mathbf{L}$ *in 3D space is projected as the line* $\mathbf{l}'$ *in the image of the second camera. The projected line* $\mathbf{l}'$ *back-projects to a plane* π' *for the second camera. A point* $\mathbf{X}$ *on the line* $\mathbf{L}$ *in 3D space is projected onto the points* $\mathbf{x}$ *and* $\mathbf{x}''$ *by the first and third camera, respectively.*

where x^i is the i-th coordinate of $\mathbf{x}$.

Let us consider a point-line-point correspondence such as $\mathbf{x}$, $\mathbf{l}'$ and $\mathbf{x}''$. The trifocal tensor $\mathtt{T}_i$ describes the geometrical relationships of point-line-point correspondence of the three cameras as follows:

$$\mathbf{l}'^{\top}(\sum_{i=1}^{3} x^i \mathtt{T}_i)[\mathbf{x}'']_{\times} = \mathbf{0}^{\top} \quad \text{for a correspondence } \mathbf{x} \leftrightarrow \mathbf{l}' \leftrightarrow \mathbf{x}'' \,, \tag{3.7}$$

where x^i is the i-th element of the vector $\mathbf{x}$. An example of this configuration is shown in Figure 3.5.

For a point-point-point correspondence of the three cameras, as shown in Figure 3.6, the geometric relationships between the points $\mathbf{x}$, $\mathbf{x}'$ and $\mathbf{x}''$ can be represented using the trifocal

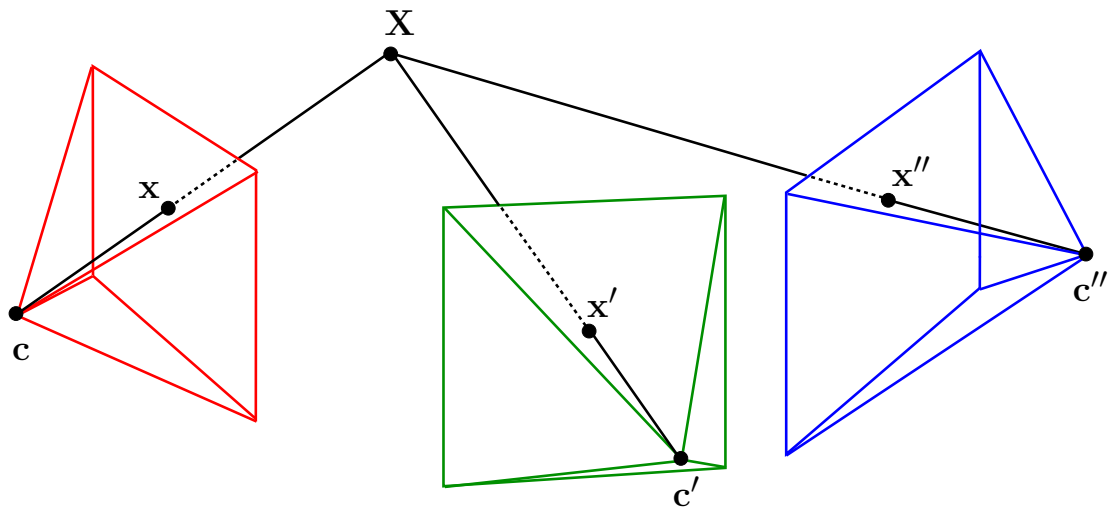


Figure 3.6: ***The point-point-point correspondence.*** *A point* $\mathbf{X}$ *in 3D space is projected onto the points* $\mathbf{x}$, $\mathbf{x}'$ *and* $\mathbf{x}''$ *by the first, second and third camera, respectively.*

tensor as follows:

$$[\mathbf{x}']_\times(\sum_{i=1}^{3} x^i \mathrm{T}_i)[\mathbf{x}'']_\times = 0_{3\times 3} \quad \text{for a correspondence } \mathbf{x} \leftrightarrow \mathbf{x}' \leftrightarrow \mathbf{x}'' \,, \tag{3.8}$$

where x^i is the i-th element of the vector $\mathbf{x}$.

It should be noted that the point-line-line and the point-line-point correspondences do not indicate a unique correspondence between the three cameras. In the point-line-point case, consider that a point $\mathbf{X}$ in 3D space may project to points $\mathbf{x}$ and $\mathbf{x}''$ in the image planes of the first and third camera, respectively. However, the corresponding line $\mathbf{l}'$ of the second camera may back-project to a plane containing the point $\mathbf{X}$, however, any line $\mathbf{L}$ on the plane can be projected onto the line $\mathbf{l}'$. Therefore, the point-line-point correspondence is not unique. In the case of the point-line-line configuration, if a plane π' (back-projected by the line $\mathbf{l}'$) is an epipolar plane between the first and second camera, the point $\mathbf{X}$ in 3D space should also lie in the epipolar plane. It implies that any line on the plane π' that passes through $\mathbf{X}$ can be the corresponding line of the point $\mathbf{x}$ and the line $\mathbf{l}'$. Hence, the point-line-line configuration is also not unique.

However, the line-line-line and point-point-point correspondences are unique and have individual trifocal tensor representations.

3.5 Motion estimation using three cameras

There are a few three-camera system to estimate the motion of cameras and to obtain the strucure of the environment. In 1998, Murray and Little developed a trinocular system for building simple grid maps of the environment in real time [53].

Chapter 4

Multi-camera Systems

4.1 What are multi-camera systems?

Multi-camera systems are a system having many cameras (usually more than three cameras) securely mounted on an object which have a rigid motion.

There exist many kinds of camera systems comprising multiple cameras such as the stereo camera system, omnidirectional camera system and multi-camera system, as shown in Figure 4.1. A stereo camera has two lenses and two CCD (charge-coupled device) sensors, and it takes two images simultaneously. An omnidirectional camera comprises either multiple cameras or a camera and mirrors. It can capture 360° images. A multi-camera system is a set of cameras firmly connected together, but they need not share a field of view. This is a more general type of camera system than the omnidirectional camera system.

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

Figure 4.1: ***Examples of multi-camera systems.*** *(a) Stereo camera system (Bumblebee^{TM}2), (b) Omnidirectional camera system (Ladybug^{TM}2), (c) Multi-camera system (ProFUSION 25, Courtesy of Point Grey Research Inc.) and (d) Multi-camera system (Camera for the UrbanScape project, Courtesy of UNC-Chapel Hill).*

4.1.1 Advantages of multi-camera systems

The amount of information obtained increases with the number of cameras used. Multi-camera systems usually have more than three cameras and they need not share a field of view. These systems have a large field of view and a complex structure of view. They can be distributed in a network. Like most omnidirectional cameras, multi-camera systems can take panoramic photos. Moreover, they can be used in a factory or building for surveillance or be mounted on a moving vehicle. They can also be worn on the body. However the larger the number of cameras used, the greater is the complexity of the multi-camera system. Estimating the motions of all the cameras is not easy, unlike the case of a single-camera system.

4.2 Geometry of multi-camera systems

In this section, the geometry of multi-camera systems is considered. These systems comprise a set of cameras that are connected firmly and the movement of the cameras are described by a rigid transformation. Without loss of generality, the projection matrices for each camera in the multi-camera system are written as follows:

$$\mathtt{P}_i = [\mathtt{I} \mid -\mathbf{c}_i] \,, \tag{4.1}$$

, where $i = 1, \ldots, n$ is the index number of the cameras, n is the total number of cameras and $\mathbf{c}_i$ is the centre of the i-th camera. In this form of projection matrices, the rotational part of the extrinsic parameters is removed to simplify the formulas. This removal can easily be performed by multiplying the inverse of the rotation matrix with image vectors. For instance, if the original shape of the projection matrices is $\mathtt{P}_i = \mathtt{R}[\mathtt{I} \mid -\mathbf{c}_i]$, then the original projected image point is $\mathbf{x} = \mathtt{R}[\mathtt{I} \mid -\mathbf{c}_i]\mathbf{X}$. In this case, by multiplying the image point with $\mathtt{R}^{-1}$, we obtain $\mathbf{v} = \mathtt{R}^{-1}\mathbf{x} = \mathtt{R}^{-1}\mathtt{R}[\mathtt{I} \mid -\mathbf{c}_i]\mathbf{X} = [\mathtt{I} \mid -\mathbf{c}_i]\mathbf{X}$. To remove the rotational component from the original projection matrices, we need to know the rotation matrices. We assume that all the extrinsic parameters, rotation and translation of cameras with respect to the world coordinate system, are already known. This concept is illustrated in Figure 4.2.

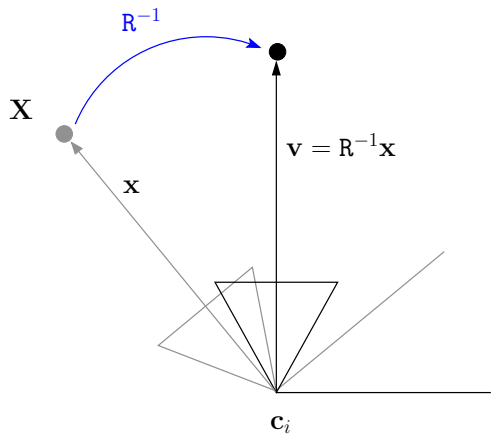


Figure 4.2: ***Removing the rotational component of camera projection matrices in multi-camera systems.*** *A camera (a grey coloured triangle) is placed at the centre $\mathbf{c}_i$ and is rotated by $\mathtt{R}$. A point $\mathbf{X}$ is projected by the camera and the projection of $\mathbf{X}$ is denoted by $\mathbf{x}$. If the point $\mathbf{X}$ is rotated by the inverse of the rotation $\mathtt{R}^{-1}$, then, the vector $\mathbf{v} = \mathtt{R}^{-1}\mathbf{X}$ is the projection of the point when the camera has no rotation component in the camera projection matrix.*

4.2.1 Rigid transformation of multi-camera systems

Rigid transformation of cameras was discussed in section 2.1.3, and the rigid transformation of multi-camera systems will be explained in this section. The Euclidean motion of multi-camera systems can be written as follows:

$$\mathtt{M} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{4.2}$$

where $\mathtt{R}$ and $\mathbf{t}$ are rotation and translation, respectively.

Using (4.1) and (4.2), all the cameras in the multi-camera system are moved to new positions by motion $\mathtt{M}$ as follows:

$$\mathtt{P}'_i = \mathtt{P}_i\mathtt{M} = \left[\mathtt{R} \,\middle|\, -\mathtt{R}\mathbf{t} - \mathbf{c}_i\right]. \tag{4.3}$$

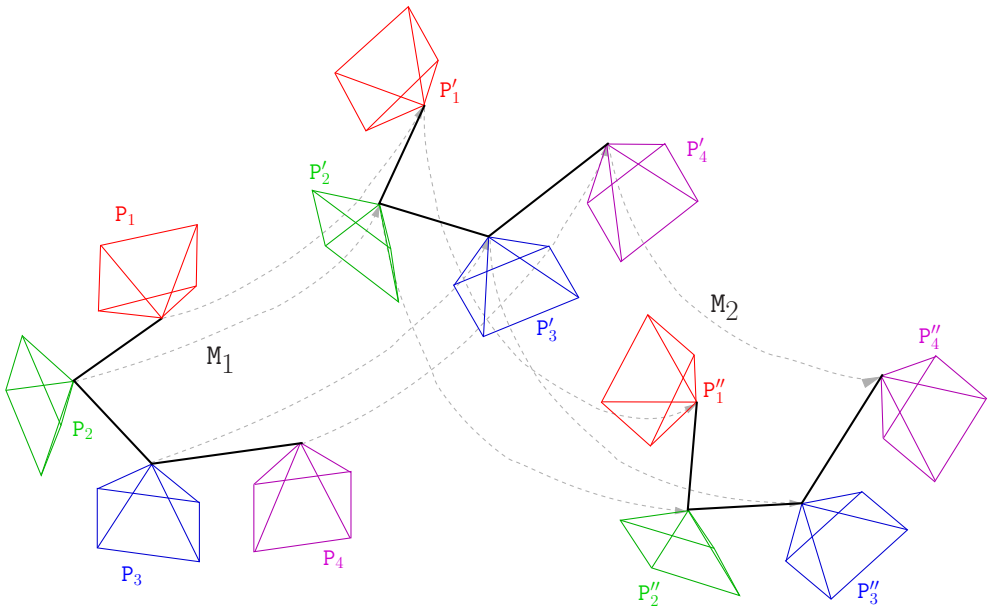


Figure 4.3: *Example of two motions* $\mathtt{M}_1$ *and* $\mathtt{M}_2$ *of multi-camera systems.*

For consecutive motions $\mathtt{M}_1, \mathtt{M}_2, \ldots, \mathtt{M}_m$,

$$\mathtt{M}_j = \begin{bmatrix} \mathtt{R}_j & -\mathtt{R}_j\mathbf{t}_j \\ \mathbf{0} & 1 \end{bmatrix} , \tag{4.4}$$

where $j = 1, \ldots, m$ and m is the number of motions, the positions of the cameras after these motions will be as follows:

$$\mathtt{P}'_i \;=\; \mathtt{P}\mathtt{M}_m\mathtt{M}_{m-1}\cdots\mathtt{M}_1 \; . \tag{4.5}$$

An example of a multi-camera system subjected to such motions is shown in Figure 4.3.

4.3 Essential matrices in multi-camera systems

In single-camera systems, there is a geometric relationship between two images taken by a single camera in motion. This geometric relationship is represented by a 3×3 matrix, the essential matrix. Similar to the essential matrix for two images, multiple essential matrices represent geometric relationships in multi-camera systems.

For instance, suppose there are four cameras that are securely connected to each other and move along a path as shown in Figure 4.3. We term the cameras the "four-camera system". Let $\mathtt{P}_1$, $\mathtt{P}_2$, $\mathtt{P}_3$ and $\mathtt{P}_4$ be the camera projection matrices of these four cameras and let $\mathtt{P}'_1$, $\mathtt{P}'_2$, $\mathtt{P}'_3$ and $\mathtt{P}'_4$ be their camera projection matrices after being subjected to the motion.

If we define these camera projection matrices as $\mathtt{P}_1 = [\mathtt{I} \mid -\mathbf{c}_1]$, $\mathtt{P}_2 = [\mathtt{I} \mid -\mathbf{c}_2]$, $\mathtt{P}_3 = [\mathtt{I} \mid -\mathbf{c}_3]$ and $\mathtt{P}_4 = [\mathtt{I} \mid -\mathbf{c}_4]$, then the camera projection matrices after the motion are written as $\mathtt{P}'_1 = [\mathtt{R} \mid -\mathtt{R}\mathbf{t} - \mathbf{c}_1]$, $\mathtt{P}'_2 = [\mathtt{R} \mid -\mathtt{R}\mathbf{t} - \mathbf{c}_2]$, $\mathtt{P}'_3 = [\mathtt{R} \mid -\mathtt{R}\mathbf{t} - \mathbf{c}_3]$ and $\mathtt{P}'_4 = [\mathtt{R} \mid -\mathtt{R}\mathbf{t} - \mathbf{c}_4]$, where $\mathtt{R}$ and $\mathbf{t}$ are the rotation and translation for the Euclidean motion of the cameras.

Therefore, a relationship between two cameras $\mathtt{P}_i$ and $\mathtt{P}'_i$, where $i = 1, \ldots, m$, may be written as an essential matrix from (2.19) as follows:

$$\mathtt{E}_i = \mathtt{R}[\mathbf{c}_i - \mathbf{t} - \mathtt{R}^\top \mathbf{c}_i]_\times \mathtt{I} \,, \tag{4.6}$$

where $i = 1, \ldots, m$ and m is the total number of cameras in the multi-camera system.

4.4 Non-perspective camera systems

The Dutch graphic artist Maurits C. Escher created a lithographic print displaying reflections in a mirror, as shown in Figure 4.4. Just like Escher's interest in artworks showing imaginary scenes and scenes difficult to depict in the non-perspective world, there are studies in the field of computer vision pertaining to photographs captured by a non-perspective camera system.

A general type of conceptual non-perspective camera was first studied by Grossberg and Nayar in [18]. They considered the projection as a mapping from the incoming scene rays to the photo-sensitive elements on the image sensor. These elements are called "raxels" and contain information on geometric, radiometric and optical properties of the incoming scene rays. Four types of non-perspective imaging systems are described in their paper. These are a catadioptric system, a dioptric wide-angle system, an imaging system comprising a camera cluster and a compound camera made of individual sensing elements.

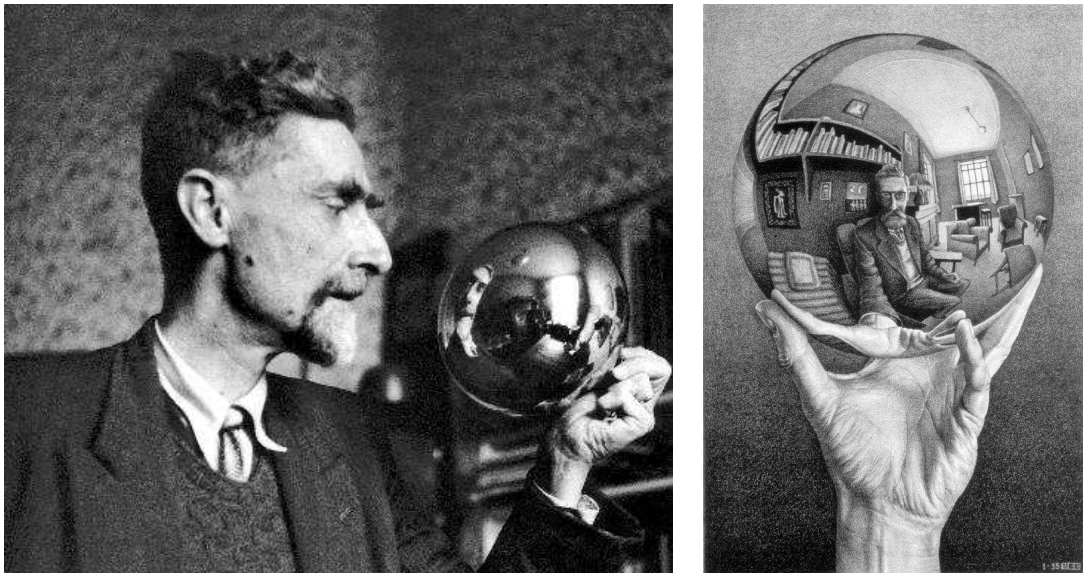


Figure 4.4: *M.C. Escher holding a curved mirror and his artwork "Hand with Reflecting Sphere" in 1935. In the curved mirror, he obtains a wider view of his surroundings than that obtained by looking directly without the mirror. Most of his surroundings such as his room and his whole body are seen in the mirror. Just like artists see the world with mirrors, computer vision researchers also consider a camera system with mirrors, the non-perspective camera system. (All M.C. Escher's works ©2008 The M.C. Escher Company – the Netherlands. All rights reserved. Used by permission.* `www.mcescher.com`*)*

Figure 4.5: *Examples of implementation of catadioptric sensors which are composed of cameras and curved mirrors. (Images from [4]. Reprinted with permission from the author, S. Baker.)*

Figure 4.6: ***Dioptric imaging system.*** *Sigma $8-mm$ fisheye lens for a DSLR camera and the photograph taken by a Sigma SD9 camera with the fisheye lens. Dioptric camera systems have a lens with a wide field of view to obtain a large field of view. The lens is a concave lens so as to capture a large number of incoming light rays. (Photograph by Jae-Hak Kim)*

The catadioptric sensor, as shown in Figure 4.5, contains both mirrors (catoptrics) and lenses (dioptrics). The word "catadioptric" is originally related to the terminology used in telescope design; however, in computer vision, the catadioptric sensor is used as a panoramic or omni-directional sensor, and it is built by using a perspective camera and curved mirrors. The image reflected by the curved mirror is captured on the perspective camera. Because of the reflection on the curved mirror, incoming light rays are no longer mapped by the perspective projection.

The dioptric wide-angle system shown in Figure 4.6 has a large concave lens to obtain a wide field of view. For instance, a fisheye lens may provide a view angle of around 180° . Because of this wide angle, most fisheye lenses do not have a single centre of projections. Therefore, the dioptric wide-angle system is also a type of non-perspective projection camera.

The camera cluster is a set of cameras that are physically connected to each other as shown in Figure 4.7. There is no limit on the number of cameras in the camera cluster. In this thesis, we call this type of camera cluster "multi-camera systems" to distinguish them from "multiple views", which refers to a large number of images taken by a single camera at multiple locations. For providing panoramic or omnidirectional images, each individual camera in multi-camera

Figure 4.7: ***Camera cluster.*** *The Stereo Omnidirectional System (SOS) by Japan's National Institute of Advanced Industrial Science and Technology and developed in collaboration with the National Rehabilitation Center for Persons with Disabilities. It provides not only omnidirectional images but also depth from stereo for people using wheelchairs. (Copyright ©National Institute of Advanced Industrial Science and Technology (AIST), Japan. All rights reserved. Reprinted with permission from AIST).*

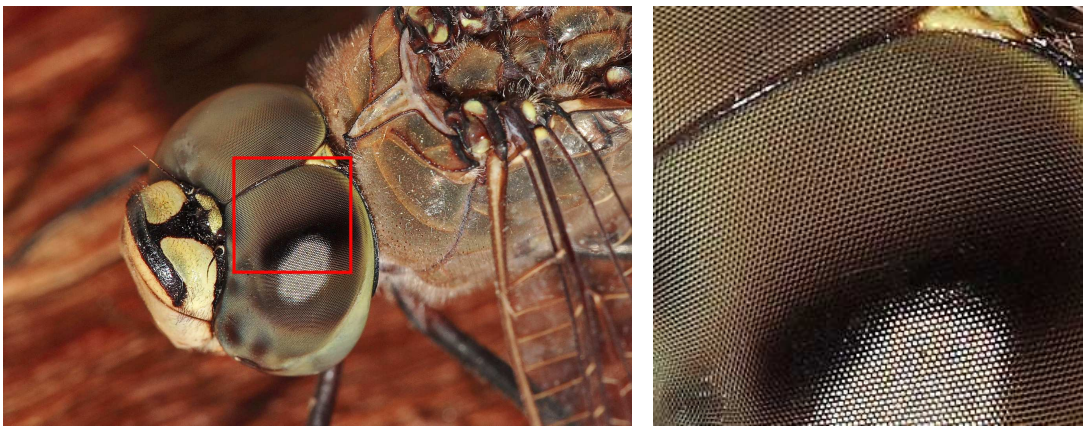


Figure 4.8: *Compound eye of a dragonfly comprising units called ommatidia. Each ommatidium is hexagonal in shape. The compound eye of the dragonfly is a type of apposition eye that is divided into two groups. (Images from* `www.wikipedia.org` *and reprinted with permission under the terms of the GNU Free Documentation Licence).*

systems needs to provide a view angle of 360° and should have a small amount of overlapping views.

There is a compound camera of which structure of the lens is similar to the structure of the eyes of insects, as shown in Figure 4.8. A series of artificial compound eyes has been created by a team of bioengineers at the University of California, Berkeley, as shown in Figure 4.9. It can be used as a camera to obtain a wider field of view than that of a fisheye lens.

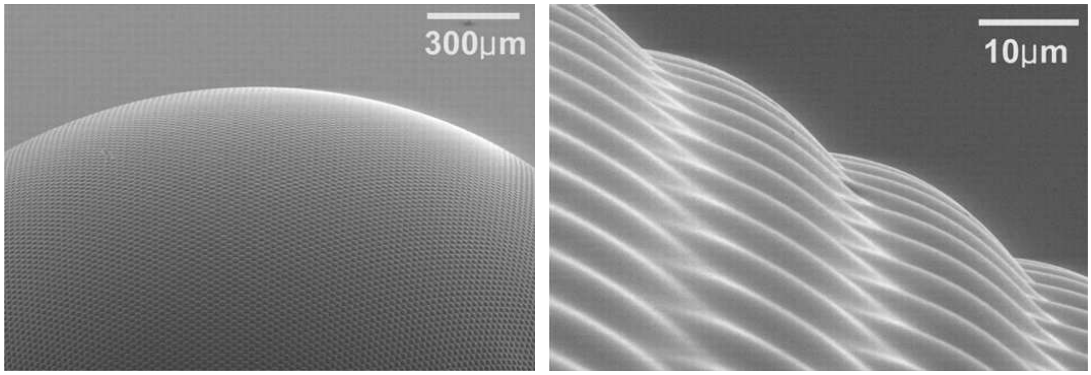


Figure 4.9: *(Left) Artificial compound eye in which 8,370 hexagonal microlenses are spherically arranged. (Right) The spherical arrangement. Courtesy: from [34]. Reprinted with permission from AAAS (American Association for the Advancement of Science).*

Chapter 5

Previous Related Work

The concept of "structure from motion", i.e., the reconstruction of the shape of an object and estimation of camera motion from videos and images, has been introduced in the field of computer vision research. However, most research was performed for conventional camera systems such as single-camera systems, stereo cameras and omnidirectional cameras. Very little research has been dedicated to multi-camera systems. In this chapter, we discuss some previous work that is related to the next following chapters, which contain the main contributions of this thesis.

In summary, plane-based projective reconstruction (see section 5.1.1) and linear multi-view reconstruction and camera recovery (section 5.1.2) relate to chapter 6. Recovering camera motion using L_∞ minimization (section 5.2) and Lie-algebraic averaging of motions (section 5.3.2) relate to chapters 6, 7, 9 and 10. The general imaging model (see section 5.4) relates to chapters 7 and 8. Convex optimization in multiple view geometry (section 5.5) relates to chapters 9 and 10.

5.1 Motion estimation using a large number of images

5.1.1 Plane-based projective reconstruction

Kaucic, Dano and Hartley have studied a linear method of projective reconstruction using planar homography [38]. Their experimental results are shown in Figure 5.1. Kaucic's method uses four points located on the same plane. The four points are used as reference points to determine the planar homography matrix. Thus, this method requires only four points isible in

Figure 5.1: *An image sequence and its reconstruction by Kaucic's method. (Courtesy of Richard Hartley)*

all the images to perform the projective reconstruction.

For a given large number of images captured by a single camera, it is possible to extract and match feature points. Since this method is based on planar homography, it is necessary to select a reference plane. The four points that are located on the reference plane are used for matching over all the images. Once the four points are identified, the planar homography is estimated from the four points. The planar homography shows the relationship between points on a single plane visible in two views.

Using the estimated planar homographies, the first 3×3 part of the camera matrices can be determined, however, the last column or translation part of the camera matrices is unknown. This translation part can be estimated using the constraints derived from the fundamental matrix of two views, the trifocal tensor of three views and the quadrifocal tensor of four views. These constraints are used to develop linear equations that are easily solved by singular value decomposition (SVD).

For instance, the linear equations to be solved in the case of 8 views are as follows:

$$\begin{bmatrix} \mathtt{S}^1_{12}\mathtt{T}^1_{12} & \mathtt{S}^1_{34}\mathtt{T}^1_{34} & & \\ & \mathtt{S}^2_{34}\mathtt{T}^2_{34} & \mathtt{S}^2_{56}\mathtt{T}^2_{56} & \\ & & \mathtt{S}^3_{56}\mathtt{T}^3_{56} & \mathtt{S}^3_{78}\mathtt{T}^3_{78} \\ \mathtt{S}^4_{12}\mathtt{T}^4_{12} & & & \mathtt{S}^4_{78}\mathtt{T}^4_{78} \end{bmatrix} \mathbf{t} = 0\,, \tag{5.1}$$

where $\mathbf{t}$ is a $3m$-vector of all the translation parts of m cameras (in this example, $m = 8$). A $n \times 9$ matrix $\mathtt{S}^p_{jk}$ is a $n \times 9$ matrix is constructed from n point correspondences of the j-th and k-th views at p frame. A 9×6 matrix $\mathtt{T}^p_{jk}$ is obtained from n pairs of matching points of the j-th and k-th views at p frame.

The matrix $\mathtt{T}^p_{jk}$ is constructed using a bilinear relationship derived from the fundamental matrix. It can be written as follows:

$$\mathtt{T}^p_{jk} = \begin{bmatrix}
0 & -\begin{vmatrix}\mathbf{A}_3\\ \mathbf{B}_2\\ \mathbf{B}_3\end{vmatrix} & \begin{vmatrix}\mathbf{A}_2\\ \mathbf{B}_2\\ \mathbf{B}_3\end{vmatrix} & 0 & -\begin{vmatrix}\mathbf{A}_2\\ \mathbf{A}_3\\ \mathbf{B}_3\end{vmatrix} & \begin{vmatrix}\mathbf{A}_2\\ \mathbf{A}_3\\ \mathbf{B}_2\end{vmatrix} \\
0 & \begin{vmatrix}\mathbf{A}_3\\ \mathbf{B}_1\\ \mathbf{B}_3\end{vmatrix} & -\begin{vmatrix}\mathbf{A}_2\\ \mathbf{B}_1\\ \mathbf{B}_3\end{vmatrix} & \begin{vmatrix}\mathbf{A}_2\\ \mathbf{A}_3\\ \mathbf{B}_3\end{vmatrix} & 0 & -\begin{vmatrix}\mathbf{A}_2\\ \mathbf{A}_3\\ \mathbf{B}_1\end{vmatrix} \\
0 & -\begin{vmatrix}\mathbf{A}_3\\ \mathbf{B}_1\\ \mathbf{B}_2\end{vmatrix} & \begin{vmatrix}\mathbf{A}_2\\ \mathbf{B}_1\\ \mathbf{B}_2\end{vmatrix} & -\begin{vmatrix}\mathbf{A}_2\\ \mathbf{A}_3\\ \mathbf{B}_2\end{vmatrix} & \begin{vmatrix}\mathbf{A}_2\\ \mathbf{A}_3\\ \mathbf{B}_1\end{vmatrix} & 0 \\
\begin{vmatrix}\mathbf{A}_3\\ \mathbf{B}_2\\ \mathbf{B}_3\end{vmatrix} & 0 & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{B}_2\\ \mathbf{B}_3\end{vmatrix} & 0 & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_3\\ \mathbf{B}_3\end{vmatrix} & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_3\\ \mathbf{B}_2\end{vmatrix} \\
-\begin{vmatrix}\mathbf{A}_3\\ \mathbf{B}_1\\ \mathbf{B}_3\end{vmatrix} & 0 & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{B}_1\\ \mathbf{B}_3\end{vmatrix} & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_3\\ \mathbf{B}_3\end{vmatrix} & 0 & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_3\\ \mathbf{B}_1\end{vmatrix} \\
\begin{vmatrix}\mathbf{A}_1\\ \mathbf{B}_1\\ \mathbf{B}_2\end{vmatrix} & 0 & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{B}_1\\ \mathbf{B}_2\end{vmatrix} & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_3\\ \mathbf{B}_2\end{vmatrix} & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_3\\ \mathbf{B}_1\end{vmatrix} & 0 \\
-\begin{vmatrix}\mathbf{A}_2\\ \mathbf{B}_2\\ \mathbf{B}_3\end{vmatrix} & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{B}_2\\ \mathbf{B}_3\end{vmatrix} & 0 & 0 & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_2\\ \mathbf{B}_3\end{vmatrix} & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_2\\ \mathbf{B}_2\end{vmatrix} \\
\begin{vmatrix}\mathbf{A}_2\\ \mathbf{B}_1\\ \mathbf{B}_3\end{vmatrix} & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{B}_1\\ \mathbf{B}_3\end{vmatrix} & 0 & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_2\\ \mathbf{B}_3\end{vmatrix} & 0 & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_2\\ \mathbf{B}_1\end{vmatrix} \\
-\begin{vmatrix}\mathbf{A}_2\\ \mathbf{B}_1\\ \mathbf{B}_2\end{vmatrix} & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{B}_1\\ \mathbf{B}_2\end{vmatrix} & 0 & -\begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_2\\ \mathbf{B}_2\end{vmatrix} & \begin{vmatrix}\mathbf{A}_1\\ \mathbf{A}_2\\ \mathbf{B}_1\end{vmatrix} & 0
\end{bmatrix}, \tag{5.2}$$

where $\mathbf{A}_i$ and $\mathbf{B}_i$ are the i-th row vectors of matrices $\mathtt{A}$ and $\mathtt{B}$, respectively, which are 3×3 submatrices of the camera matrices. The camera matrices at the j-th view and k-th view are written as $\mathtt{P}_j = [\mathtt{A} \mid \mathbf{a}]$ and $\mathtt{P}'_k = [\mathtt{B} \mid \mathbf{b}]$, respectively.

Let $\mathbf{x}_j^i$ and $\mathbf{x}_k'^i$ be the i-th pair of matching points in views j and k. The point coordinates of $\mathbf{x}_j^i$ and $\mathbf{x}_k'^i$ are $(x^i, y^i, z^i)^\top$ and $(x'^i, y'^i, z'^i)^\top$, respectively. The matrix $\mathtt{S}_{jk}^p$ is derived from point correspondences as follows:

$$\mathtt{S}_{jk}^p = \begin{bmatrix} x_j'^1 x_j^1 & x_j'^1 y_j^1 & x_j'^1 & y_j'^1 x_j^1 & y_j'^1 y_j^1 & x_j^1 & y_j^1 & 1 \\ x_j'^2 x_j^2 & x_j'^2 y_j^2 & x_j'^2 & y_j'^2 x_j^2 & y_j'^2 y_j^2 & x_j^2 & y_j^2 & 1 \\ & & & \vdots & & & & \\ x_j'^n x_j^n & x_j'^n y_j^n & x_j'^n & y_j'^n x_j^n & y_j'^n y_j^n & x_j^n & y_j^n & 1 \end{bmatrix}. \tag{5.3}$$

Because the planar homographies are already estimated using the four points, the 3×3 matrices $\mathtt{A}$ and $\mathtt{B}$ can be easily determined. Further, $\mathtt{T}_{jk}^p$ is calculated from the two cameras, and $\mathtt{S}_{jk}^p$ is derived using the n pairs of matching points. Finally, after substituting the obtained result into (5.1), the translation vector $\mathbf{t}$ can be estimated by SVD.

The method proposed by Kaucic, Dano and Hartley can be compared with the factorization method proposed by Sturm and Triggs [79]. However, Kaucic's method does not require all the points to be visible in all images as Sturm's method does. Only four points located on a plane visible in all images are required to solve the projective reconstruction problem. However, both methods only concern images captured by a single-camera system and multi-camera systems were not utilized in many applications. In chapter 6, we show a method to determine the translation of for omnidirectional cameras. In chapters 7, 9 and 10, we present methods to estimate the motion of multi-camera systems.

5.1.2 Linear multi-view reconstruction and camera recovery

As shown in Figure 5.2, Rother and Carlsson proposed a method for a simultaneous computation of 3D shapes and estimation of the camera motion from multiple views using four points located on a reference plane visible in all images [63, 64]. They used SVD to solve the linear equations. Rother's method constructs linear equations by mapping the bases of 3D points to the bases of 2D points.

Using four coplanar points visible in all images, homographies can be estimated. This also

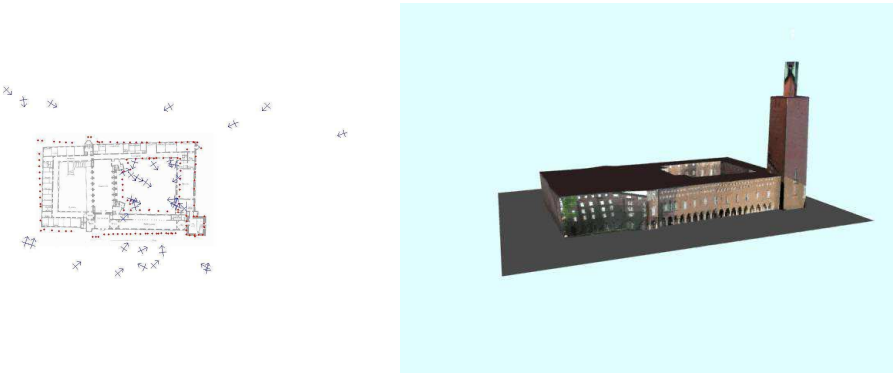


Figure 5.2: *Camera position and 3D shape recovered by Rother's method. (Courtesy of Carsten Rother ©all rights reserved)*

gives us the first 3×3 half of the camera matrix. We now know the first 3×3 half of the camera matrix and the coordinates of the pairs of matching points. The unknown parameters are the translation part of the camera matrix and the coordinates of the 3D points. Using Rother's method, the two unknowns – the 3D point and translation part – can be calculated simultaneously.

Without a detailed description of the equations, it may be written as follows:

$$\begin{bmatrix} x\mathbf{A}_3{}^\top - \mathbf{A}_1^\top & -1 & 0 & x \\ y\mathbf{A}_3{}^\top - \mathbf{A}_2^\top & 0 & -1 & y \end{bmatrix} \begin{pmatrix} \tilde{\mathbf{X}} \\ t_1 \\ t_2 \\ t_3 \end{pmatrix} = \mathbf{0}\,, \tag{5.4}$$

where $\mathbf{A}_i{}^\top$ is the i-th row of the first 3×3 matrix $\mathtt{A}$ from the camera matrix $\mathtt{P} = [\mathtt{A} \mid \mathbf{t}]$, $\mathbf{t} = (t_1, t_2, t_3)^\top$, (x, y) are non-homogeneous coordinates of a 3D point and $\tilde{\mathbf{X}}$ is the projected point of the 3D point.

If there are m views and each view has n matching points, a set of $2nm$ equations can be generated and $3n + 3m$ unknown parameters exist in the equations. These equations can be solved using SVD.

Similar to Kaucic's method, Rother's method also suffers in the presence of noise on the measurement of coordiantes of mathcing pairs, and it also requires overlapped views between

images and four visible points on a reference plane over all the images.

5.2 Recovering camera motion using L_∞ minimization

Sim and Hartley proposed a method for estimation of camera motion for a single-camera system using L_∞ minimization [69]. In this method, the rotations of the camera are determined first, and then the translation is estimated using second-order cone programming (SOCP) to determine the camera motion on the basis of the matched points.

In Sim's method, it is assumed that the cameras are calibrated. In the first step, given pairs of matching points, the relative orientations between the pairs are computed using the essential matrix. Because the translation is up to scale, they minimized a maximum of the angle error between a unit vector of the estimated translation and a unit vector of the true translation. This minimization problem is solved using SOCP.

Although their approach deals with global optimization techniques, only the translation part uses convex optimization. Hence, it is still not an optimal solution in terms of the estimation of both rotation and translation. Similar to the methods discussed previously, this method also requires overlapped views to estimate the motion.

5.3 Estimation of rotation

5.3.1 Averaging rotations

Curtis et al. proposed a method for averaging rotations using SVD [8]. Given two rotation matrices, an algebraic average of the two rotations – summing them and then dividing the result by two – does not yield a correct approximation of two rotation matrices. Curtis et al. proposed a method to theoretically obtain the correct average of the two given rotation matrices. This method is useful to obtain a reasonable rotation matrix when the estimated rotation matrices are inaccurate because of measurement errors.

Let $\mathtt{R}_1$ and $\mathtt{R}_2$ be two rotation matrices. In the method given by Curtis et al. the average of two rotations is calculated by computing the SVD of the sum of two rotations and by using the

result to determine the first and last orthogonal matrices as follows:

$$\mathtt{R}_a = \mathtt{U}\mathtt{V}^\top \ , \tag{5.5}$$

where $\mathtt{U}\mathtt{D}\mathtt{V}^\top = \mathtt{R}_1 + \mathtt{R}_2$. Here, $\mathtt{U}$ and $\mathtt{V}$ are orthogonal matrices, and $\mathtt{D}$ is a diagonal matrix with non-negative entries. This SVD approach to averaging rotations is based on the orthonormal procrustes problem, which is a matrix approximation problem for two matrices proposed by Schönemann [66].

A 3D rotation matrix can be represented by a quaternion as well. Given two quaternions $\mathbf{q}_1$ and $\mathbf{q}_2$, Curtis obtained the average of the two quaternions as follows:

$$\mathbf{q}_a = \begin{cases} \frac{(\mathbf{q}_1+\mathbf{q}_2)}{\lambda} & \text{if } \mathbf{q}_1^\top \mathbf{q}_2 \geq 0 \\ \frac{(\mathbf{q}_1-\mathbf{q}_2)}{\mu} & \text{otherwise} \end{cases} \ , \tag{5.6}$$

where $\lambda = ||\mathbf{q}_1 + \mathbf{q}_2||$ and $\mu = ||\mathbf{q}_1 - \mathbf{q}_2||$.

The weighted averages of more than two rotations can also be obtained using the two methods listed above.

5.3.2 Lie-algebraic averaging of motions

Govindu showed a method to average the motions of an image sequence using Lie-algebra [17]. Given m images, $\frac{m(m-1)}{2}$ pairwise relative motions can be used to calculate globally consistent averages of motions over an image sequence.

5.4 General imaging model

A general imaging model was first introduced by Grossberg and Nayar in [18]. They described a general imaging model for light rays incident on an image and proposed a new concept of light rays as “raxels” which contain geometric, radiometric and optical information. They also proposed a calibration method for general image models using structured light patterns.

5.5 Convex optimization in multiple view geometry

In this section, we briefly re-introduce convex optimization and outline its use in multiple view geometry problems in computer vision. Furture details on the convex optimization in multiple view geometry can be found in [22].

Convex optimization is a method to find an optimal solution to a problem that has the shape of a convex function and the domain of a convex set. Because of the shape of the convex function, there exists only a single minimum solution to the problem and this makes it easier to obtain a globally optimal solution as compared to other nonlinear optimization methods which usullay risk not converging, or converge to a local minimum.

Convex set. A convex set is a subset S of $\mathbb{R}^n$, provided that the line segment joining any two points in S is contained in S. The convex set can be defined as

$$(1-\alpha)\mathbf{x}_0 + \alpha\mathbf{x}_1 \in S \text{ for all } \mathbf{x}_0, \mathbf{x}_1 \in S \text{ and } \alpha \in [0,\ 1]\ . \tag{5.7}$$

Convex function. A convex function is a function f that has the domain of a convex set such that all for $\mathbf{x}_0, \mathbf{x}_1 \in \text{domain}(f)$, and $0 \leq \alpha \leq 1$.

$$f((1-\alpha)\mathbf{x}_0 + \alpha\mathbf{x}_1) \leq (1-\alpha)f(\mathbf{x}_0) + \alpha f(\mathbf{x}_1)\ . \tag{5.8}$$

Convex optimization problem. Given a convex function f, we find the minimum of f in the domain of f. The convex optimization problem can be solved by algorithms that depend on the function f and the domain D.

Ideally, it would be most suitable if multiple view geometry problems were in the form of convex optimization problem; however, most cost functions in multiple view geometry are not in the form of convex functions. However, another approach to solve multiple view geometry problems is to use quasi-convex optimization.

Quasi-convex functions. A function f is a quasi-convex function if its α-sublevel set is convex for all α as follows:

$$S_\alpha = \{\mathbf{x} \in D \mid f(\mathbf{x}) \leq \alpha\} \, . \tag{5.9}$$

This property of quasi-convex functions is important for computer vision researchers because some cost functions in multiple view geometry may be considered as forms of quasi-convex functions. A quasi-convex function has no local minimum but the shape of the quasi-convex function is not convex. Instead, it becomes a convex function only if a certain sublevel of the quasi-convex function is considered. If the sublevel is specified by the value of α, we refer to it as an α-sublevel set of the quasi-convex function. Hence, the strategy is to determine an α-sublevel set of the quasi-convex function in order to obtain a global minimum. Further information of convex optimization is provided by Boyd and Vanderberghe in [5].

Convex optimization has attracted many researchers in computer vision since 2004. In 2004, Hartley and Schaffalitzky [23] first introduced the L_∞ cost function for the multiview triangulation problem and the motion reconstruction problem of omnidirectional images. Following that, in 2005, a convex optimization technique was introduced to solve the problems of multiple view geometry by two separate research groups at different locations but almost at the same time. Kahl [35] introduced a quasi-convex optimization method to solve the multiview triangulation problem, the camera resectioning problem and the homography estimation problem using SOCP. Ke and Kanade [39] also presented a quasi-convex optimization method to solve the multiview triangulation problem, the camera resectioning problem and the multiview reconstruction problem with known rotations using SOCP or linear programming (LP).

Chapter 6

Translation Estimation from Omnidirectional Images

There are two known approaches for reconstructing camera motion and structure from an image sequence when there are missing feature tracks in the image sequence. One is to compute both camera motion and structure at the same time as Rother's method [63, 64]. The other is to compute camera motion first and then obtain the structure as Kaucic's method [37]. However, in the presence of noise, both methods are extremely sensitive to noise, and they could fail to achieve accuracy of estimation. When features are detected and tracked from the image sequence, the length of the tracked features affects estimation results in both methods. Rother's method needs feature tracks visible across large number of views to obtain robust results when there are measurement errors in the data. Kaucic's method also requires the long feature tracks to get robust results.

In this chapter, we present a method which does not require the use of long feature track lengths as the above-mentioned methods. Instead of using the long track lengths, we use a constrained minimization to get a more reliable result by changing an equation which is used in the plane-based translation estimation method [37]. We assume that relative rotation between two views is known. The relative motion can be estimated by using Kaucic's method or Longuet-Higgins's algorithm [24, 46]. In particular, note that we would like to solve the motion problem throughout all views in an image sequence.

From our experiments, the proposed method showed a more robust result than the other two methods and the time of computation of the proposed method is similar as that of the

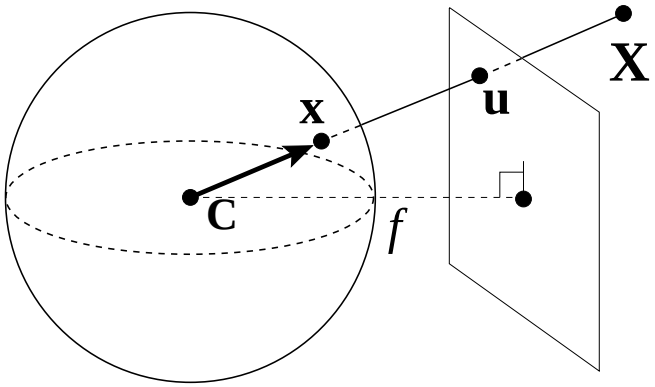


Figure 6.1: *Model of an omnidirectional camera: The centre of the unit sphere is the centre of the omnidirectional camera. The unit direction vector* $\mathbf{x}$ *is the image of* $\mathbf{X}$.

previous methods.

6.1 Omnidirectional camera geometry

An image vector for an omnidirectional image is represented by a three-dimensional direction vector $\mathbf{x}$ which starts from the origin of a coordinate system to a point on the surface of a unit sphere $\mathcal{S}^2$, as shown in Figure 6.1.

Let us suppose that a unit sphere is placed of the origin of a coordinate system. Then, an image vector can be easily represented by a unit vector. Let $\mathbf{X}$ be a 3D point in oriented projective space $\mathcal{P}^3$. Then, the point $\mathbf{X}$ is projected onto a spherical image which is modelled by a unit sphere [75]. The projected point $\mathbf{x}$ is represented by a direction vector from the centre of the unit sphere $\mathbf{C}$ in real space $\mathcal{R}^3$ to the direction of pointing $\mathbf{X}$. Suppose that the centre of the unit sphere is the centre of the omnidirectional camera. Accordingly, the directional image vector $\mathbf{x}$ is simply obtained by $\mathbf{x} = (\tilde{\mathbf{X}} - \tilde{\mathbf{C}})/||\tilde{\mathbf{X}} - \tilde{\mathbf{C}}||$ where $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{C}}$ in $\mathcal{R}^3$ are inhomogeneous coordinates of the point $\mathbf{X}$ and the centre $\mathbf{C}$ in $\mathcal{P}^3$, respectively. For instance, if $\mathbf{X} = (2,\ 4,\ 8,\ 2)^\top$, then $\tilde{\mathbf{X}} = (1,\ 2,\ 4)^\top$. The directional image vector $\mathbf{x}$ is also represented

by a projection of $\mathbf{X}$ and normalization on the unit sphere as follows:

$$\begin{aligned}\mathbf{u} &= \mathtt{K}[\mathtt{I} \mid \mathbf{0}]\mathbf{X} = [f\mathbf{X}_1,\ f\mathbf{X}_2,\ \mathbf{X}_3]^\top \\ \mathbf{x} &= \frac{[\mathbf{u}_1,\ \mathbf{u}_2,\ f\mathbf{u}_3]^\top}{||[\mathbf{u}_1,\ \mathbf{u}_2,\ f\mathbf{u}_3]^\top||},\end{aligned}$$

where $\mathtt{K} = diag(f,\ f,\ 1)$ and $\mathbf{u}$ in $\mathcal{P}^2$ is a projected image point on the plane from the camera centre $\mathbf{C}$. Since $\mathbf{u}$ is on the line $\overline{\mathbf{XC}}$, $\mathbf{x}$ is represented by normalizing $\mathbf{u}$.

The position of a camera can be represented by a rotation and translation with respect to the origin of a world coordinate frame. In an omnidirectional camera, we can set the focal length f to one, so the matrix $\mathtt{K}$ becomes an identity matrix. These can be written as follows:

$$\begin{aligned}\mathbf{u} &= \mathtt{KR}[\mathtt{I} \mid -\tilde{\mathbf{C}}]\mathbf{X} \\ &= \mathtt{R}[\mathtt{I} \mid -\tilde{\mathbf{C}}]\mathbf{X} \\ &= [\mathtt{R} \mid -\mathtt{R}\tilde{\mathbf{C}}]\mathbf{X} \\ &= [\mathtt{R} \mid \mathbf{t}]\mathbf{X} \qquad (6.1) \\ \mathbf{x} &= \frac{\mathbf{u}}{||\mathbf{u}||}. \qquad (6.2)\end{aligned}$$

Equation (6.1) shows that the point $\mathbf{X}$ is projected on a plane and the point is transformed by a rotation $\mathtt{R}$ and translation $\mathbf{t} = -\mathtt{R}\tilde{\mathbf{C}}$. The image point $\mathbf{u}$ projected on the plane is projected onto the unit sphere as shown in Figure 6.1. This is the same as normalizing $\mathbf{u}$ with respect to the camera centre $\mathbf{C}$ as shown in (6.2).

Therefore, any direction vector $\mathbf{x}$ for an omnidirectional image can be represented by mapping the point through rigid transformation, followed by projecting onto an image plane and normalizing on a unit sphere.

Remark. If a point $\mathbf{X}$ in $\mathcal{P}^3$ is projected by an omnidirectional camera, then a three-dimensional direction vector $\mathbf{x}$ in an omnidirectional image is represented by $\mathbf{x} = \mathbf{u}/||\mathbf{u}||$, where $\mathbf{u} = [\mathtt{R} \mid \mathbf{t}]\,\mathbf{X}$ and $\mathtt{R}$ and $\mathbf{t}$ are rotation and translation of the omnidirectional camera, respectively.

Definition 1. An omnidirectional camera projection matrix $\mathtt{P}$ is expressed as $\mathtt{P} = [\mathtt{R} \mid \mathbf{t}]$ where

$\mathbf{t} = -\mathtt{R}\tilde{\mathbf{C}}$, and $\tilde{\mathbf{C}}$ is the centre of the camera, $\mathtt{R}$ is a rotation matrix and $\mathbf{t}$ is a translation vector.

If we know the rotation of an omnidirectional camera, the camera projection matrix of the omnidirectional camera may be further simplified by multiplying it by the inverse of the rotation matrix.

Remark. Given an omnidirectional camera projection matrix $\mathtt{P} = \mathtt{R}[\mathtt{I} \mid -\tilde{\mathbf{C}}]$, we can obtain a simplified projection matrix $\hat{\mathtt{P}} = \mathtt{R}^{-1}\mathtt{P} = [\mathtt{I} \mid -\tilde{\mathbf{C}}]$ by multiplying $\mathtt{P}$ by the inverse of the rotation matrix $\mathtt{R}^{-1}$. Note that a point $\mathbf{x}$ projected by the camera matrix $\mathtt{P}$ is written as a point $\hat{\mathbf{x}} = \mathtt{R}^{-1}\mathbf{x}$.

6.2 A translation estimation method

Kaucic et al. proposed a plane-based projective reconstruction method with missing data [37]. In their paper, the projective reconstruction method was applied in the case of a conventional camera. However, if we apply this method to omnidirectional images, it becomes a translation estimation method for an omnidirectional camera. In this thesis, we assume that rotations in all views are already known. This assumption is similar as that of Kaucic's method because Kaucic et al. assumed that homographies in all views are already computed in their plane-based projective reconstruction. More details about Kaucic's method are explained in section 5.1.1. Practically, these rotations may be estimated from essential matrices. A singular value decomposition (SVD) method may be used to extract a rotation part and a translation part from an essential matrix [24, 46].

6.2.1 Bilinear relations in omnidirectional images

Let $\mathbf{x} \leftrightarrow \mathbf{x}'$ be a point correspondence in two omnidirectional views. This point correspondence is obtained by the projection of a point $\mathbf{X}$ in space onto two omnidirectional views. Let $\mathtt{P} = [\mathtt{I} \mid -\mathbf{a}]$ and $\mathtt{P}' = [\mathtt{I} \mid -\mathbf{b}]$ be the two omnidirectional camera projection matrices corresponding to the two views. Let us suppose that the rotation in each view is already known. Therefore, the left 3×3 sub-matrix of the camera projection matrix may become a form of

an identity matrix by multiplying the camera projection matrices by the inverse of the rotation matrix.

With a known calibration matrix and rotation matrix, according to [27, Equation (9.2) on page 244], the fundamental matrix corresponding to the two omnidirectional cameras becomes as follows:

$$\mathtt{F} = [\mathbf{b} - \mathbf{a}]_\times = [\mathbf{t}]_\times \,, \tag{6.3}$$

where $[\mathbf{t}]_\times$ is a 3×3 skew-symmetric matrix and the 3-vector $\mathbf{t}$ is the translation from the centre of the first camera to the centre of the second camera. Therefore, the fundamental matrix for the two omnidirectional cameras is expressed by the 3-vector $\mathbf{t}$ because we already know the rotations. To check this, note that the rotation matrix part of the fundamental matrix is an identity matrix, so the fundamental matrix has only a skew-symmetric matrix part.

Lemma 1. *Let* $\mathtt{P}$ *and* $\mathtt{P}'$ *be two camera projection matrices for omnidirectional images and write as* $\mathtt{P} = [\mathtt{I} \mid -\mathbf{a}]$ *and* $\mathtt{P}' = [\mathtt{I} \mid -\mathbf{b}]$ *where* $\mathbf{a}$ *and* $\mathbf{b}$ *are the centres of each camera. Then, a fundamental matrix* $\mathtt{F}$ *for the two omnidirectional cameras is written as a* 3×3 *skew-symmetric matrix* $\mathtt{F} = [\mathbf{t}]_\times$*, where* $\mathbf{t} = \mathbf{b} - \mathbf{a}$.

Given a point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$ represented as unit vectors in omnidirectional images, from lemma 1, we can obtain the following epipolar constraint equation as follows:

$$\begin{aligned} \mathbf{x}'^\top \mathtt{F} \mathbf{x} &= \mathbf{x}'^\top [\mathbf{t}]_\times \mathbf{x} \\ &= \mathbf{x}'^\top (\mathbf{t} \times \mathbf{x}) \\ &= (\mathbf{x} \times \mathbf{x}')^\top \mathbf{t} = 0 \,. \end{aligned} \tag{6.4}$$

Equation (6.4) is zero because of epipolar constraints and the vector $\mathbf{t}$ which consists of two camera centres $\mathbf{a}$ and $\mathbf{b}$ may be decomposed to build a system of linear equations with the centres $\mathbf{a}$ and $\mathbf{b}$ as follows:

$$(\mathbf{x} \times \mathbf{x}')^\top \mathbf{t} = (\mathbf{x} \times \mathbf{x}')^\top \begin{bmatrix} -\mathtt{I} & \mathtt{I} \end{bmatrix} \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} = 0 \,. \tag{6.5}$$

Note that this is an easy way to derive the bilinear constraint when image vectors are represented in omnidirectional images. The same derivation could be obtained by a more difficult way for central projection cameras as explained in section 2.2.4.5 and [27, Section 17.1 on page 412].

Accordingly, given the point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$, equation (6.5) gives us a system of linear equations in terms of the camera centres $\mathbf{a}$ and $\mathbf{b}$. These equations are the same as the bilinear relationships shown in [37]. The 3-vector $\mathbf{t}$ in (6.5) is obtained from a skew-symmetric fundamental matrix. Therefore, the vector $\mathbf{t}$ in (6.5) is written as follows:

$$\mathbf{t} = \mathbf{f}_{ji} = \mathtt{T}_{ij}\mathbf{c}_{ij} \ , \tag{6.6}$$

where $\mathbf{f}_{ji}$ is a 3-vector of a skew-symmetric fundamental matrix; $\mathtt{T}_{ij}$ is a 3×6 matrix of a bilinear relation from two views i and j; and $\mathbf{c}_{ij}$ is a 6-vector consisting of two 3-vectors $\mathbf{a}$ and $\mathbf{b}$ which come from the last column in each camera projection matrix. The vector $\mathbf{c}_{ij}$ gives us the centres of two cameras. Then, in summary, the matrix $\mathtt{T}_{ij}$ and the vector $\mathbf{c}_{ij}$ are written as follows:

$$\mathtt{T}_{ij} = \begin{bmatrix} -\mathtt{I} & \mathtt{I} \end{bmatrix} \tag{6.7}$$

$$\mathbf{c}_{ij} = \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \end{pmatrix} , \tag{6.8}$$

where $\mathbf{a}$ and $\mathbf{b}$ are the centres of two cameras at view i and j.

Lemma 2. *Let* $\mathtt{P} = [\mathtt{I} \mid -\mathbf{a}]$ *and* $\mathtt{P}' = [\mathtt{I} \mid -\mathbf{b}]$ *be two omnidirectional cameras for view* i *and* j*. Given point correspondences* $\mathbf{x} \leftrightarrow \mathbf{x}'$ *in the two omnidirectional views* i *and* j*, we can compute a fundamental matrix* $\mathtt{F}_{ji}$ *from* $\mathbf{x}'^{\top}\mathtt{F}_{ji}\mathbf{x} = 0$ *where* $\mathtt{F}_{ji}$ *is a skew-symmetric matrix according to lemma 1 . A 3-vector* $\mathbf{f}_{ji}$ *defining a matrix* $\mathtt{F}_{ji}$ *can be expressed by a bilinear relation* $\mathtt{T}_{ij}$ *and a vector* $\mathbf{c}_{ij} = [\mathbf{a}^{\top},\ \mathbf{b}^{\top}]^{\top}$ *from two cameras such as* $\mathbf{f}_{ji} = \mathtt{T}_{ij}\mathbf{c}_{ij}$*.*

6.2.2 Trilinear relations

The trilinear relations are expressed in the same way as bilinear relations by using a trifocal tensor instead of a fundamental matrix. Let $\mathbf{x} \leftrightarrow \mathbf{x}' \leftrightarrow \mathbf{x}''$ be a point correspondence in three omnidirectional views. Let $\mathtt{P} = \mathtt{R}[\mathtt{I} \mid -\tilde{\mathbf{C}}_1]$, $\mathtt{P}' = \mathtt{R}'[\mathtt{I} \mid -\tilde{\mathbf{C}}_2]$ and $\mathtt{P}'' = \mathtt{R}''[\mathtt{I} \mid -\tilde{\mathbf{C}}_3]$ be three omnidirectional cameras. Then, the simplified cameras become $\hat{\mathtt{P}} = \mathtt{R}^{-1}\mathtt{P} = [\mathtt{I} \mid -\tilde{\mathbf{C}}_1]$, $\hat{\mathtt{P}}' = \mathtt{R}'^{-1}\mathtt{P}' = [\mathtt{I} \mid -\tilde{\mathbf{C}}_2]$ and $\hat{\mathtt{P}}'' = \mathtt{R}''^{-1}\mathtt{P}'' = [\mathtt{I} \mid -\tilde{\mathbf{C}}_3]$. Now, the vectors $\tilde{\mathbf{C}}_1$, $\tilde{\mathbf{C}}_2$ and $\tilde{\mathbf{C}}_3$ are centres of cameras in the world coordinate system. The camera matrices become $\bar{\mathtt{P}} = [\mathtt{I} \mid \mathbf{0}]$, $\bar{\mathtt{P}}' = [\mathtt{I} \mid -\mathbf{a}]$ and $\bar{\mathtt{P}}'' = [\mathtt{I} \mid -\mathbf{b}]$, where $\mathbf{a} = \tilde{\mathbf{C}}_2 - \tilde{\mathbf{C}}_1$ and $\mathbf{b} = \tilde{\mathbf{C}}_3 - \tilde{\mathbf{C}}_1$. The trifocal tensor corresponding to these three omnidirectional cameras becomes as follows:

$$\mathcal{T}_i^{jk} = \delta_i^j \mathbf{b}^k - \delta_i^k \mathbf{a}^k \,. \tag{6.9}$$

The trifocal tensor relation for point-point-point correspondence is

$$x^i x'^p x''^q \epsilon_{pjs} \epsilon_{qkt} \mathcal{T}_i^{jk} = 0_{st} \tag{6.10}$$

and by substituting (6.9) we can obtain the following equations:

$$\begin{aligned}
x^i x'^p x''^q \epsilon_{pjs} \epsilon_{qkt} (\delta_i^j \mathbf{b}^k - \delta_i^k \mathbf{a}^k) &= 0_{st} \\
x^i x'^p x''^q \epsilon_{pjs} \epsilon_{qkt} \delta_i^j \mathbf{b}^k - x^i x'^p x''^q \epsilon_{pjs} \epsilon_{qkt} \delta_i^k \mathbf{a}^k &= 0_{st} \\
x^j x'^p x''^q \epsilon_{pjs} \epsilon_{qkt} \mathbf{b}^k - x^k x'^p x''^q \epsilon_{pjs} \epsilon_{qkt} \mathbf{a}^k &= 0_{st} \\
x^j (x'' \times \mathbf{b})_t ([x']_\times)_{js} - x^k (x' \times \mathbf{a})_s ([x'']_\times)_{kt} &= 0_{st} \\
(x \times x')_s (x'' \times \mathbf{b})_t - (x \times x'')_t (x' \times \mathbf{a})_s &= 0_{st} \,.
\end{aligned} \tag{6.11}$$

Equation (6.11) is a system of linear equations of the two relative camera centres $\mathbf{a}$ and $\mathbf{b}$. In the same way, the trilinear relationship can be used. Therefore, The trilinear relationship is

written as follows:

$$\mathcal{T}_i^{jk} = (-1)^{(i+1)} det \begin{bmatrix} \sim \mathtt{P}^i \\ \mathtt{P}^j \\ \mathtt{P}^k \end{bmatrix} , \tag{6.12}$$

where the expression $\sim \mathtt{P}^i$ means the matrix $\mathtt{P}$ with row i omitted.

6.2.3 Constructing an equation

Bilinear relations. Given multiple images in omnidirectional cameras, we can choose any two views from an image sequence to construct bilinear relation equations. Point correspondences between the two selected views are obtained by using a well known feature matching method [47]. Note that because the point correspondence is not required to be seen in all views, we are dealing with a missing data problem. Then, we obtain fundamental matrices for every pair of views. Each fundamental matrix $\mathtt{F}_{ji}$ for view i and j can be defined by a 3-vector $\mathbf{f}_{ji}$ which comes from a skew-symmetric fundamental matrix lemma 1. Then, the following equation is satisfied for the multiple images with missing data:

$$\mathtt{S}_{ij}\mathbf{f}_{ji} = \mathbf{0} , \tag{6.13}$$

where $\mathtt{S}_{ij}$ is a $n \times 3$ matrix of the point correspondences which are extracted from views i and j, and n is the number of point correspondences. This matrix $\mathtt{S}_{ij}$ is the same matrix used to compute a normalized 8-point fundamental matrix.

By substituting (6.6) into (6.13), we obtain the following equation:

$$\mathtt{S}_{ij}\mathtt{T}_{ij}\mathbf{c}_{ij} = \mathbf{0} . \tag{6.14}$$

If we select any two consecutive views, then we make (6.14) from the two views. If we select again and repeat for all other two consecutive frames, then we can construct a large matrix $\mathtt{E}$ consisting of a set of equations (6.14) for the consecutive frames [37, 27]. We used consecutive frames and added more frames to make a system of linear equations solvable.

However, there is a simple way to make the linear system of equations. The number of possible ways of selecting two views from m views is the combination of choice number, ${}_mC_2 = \frac{m!}{2!(m-2)!} = \frac{m(m-1)}{2}$. For instance, given four views, the number of ways of selecting two views becomes {1,2},{1,3},{1,4},{2,3},{2,4} and {3,4}, so there are ${}_4C_2 = \frac{4 \cdot 3}{2} = 6$ ways. This can be easily written by using two for-loop sentences in any programming language.

For example, let us see how to make the large matrix $\mathtt{E}$ when we have four views. The left part $\mathtt{S}_{ij}\mathtt{T}_{ij}$ in (6.14) is added to the large matrix $\mathtt{E}$ to compose all relations of translation vectors through all views. Let Δ_i and Δ_j be the first $n \times 3$ and last $n \times 3$ part of $\mathtt{S}_{ij}\mathtt{T}_{ij}$. There are 6 ways of selecting two views. Therefore, the large matrix $\mathtt{E}$ has $6n$ rows and $4m$ columns, where n is the number of point correspondences and m is the number of views. Then, the large matrix $\mathtt{E}$ can be expressed as follows:

$$\mathtt{E} = \begin{bmatrix} \Delta_1 & \Delta_2 & & \\ \Delta_1 & & \Delta_3 & \\ \Delta_1 & & & \Delta_4 \\ & \Delta_2 & \Delta_3 & \\ & \Delta_2 & & \Delta_4 \\ & & \Delta_3 & \Delta_4 \end{bmatrix}, \tag{6.15}$$

where $[\Delta_i \mid \Delta_j] = \mathtt{S}_{ij}\mathtt{T}_{ij}$ and Δ_k is a $n \times 3$ matrix. Therefore, the equation we have to solve becomes

$$\mathtt{E}\mathbf{c} = \mathbf{0}\,, \tag{6.16}$$

where $\mathbf{c}$ is a $3m$-vector consisting of all camera centres, and m is the number of views.

Trilinear relations. Similar to the bilinear relations, three views can be used to construct a system of linear equations using trilinear relations. Suppose that there are n number of point correspondences across three views. Then, for three views, view i, view j and view k, we have

an equation as follows:

$$\mathtt{S}_{ijk}\mathbf{f}_{kji} = \mathbf{0}\ , \tag{6.17}$$

where $\mathtt{S}_{ijk}$ is a $n \times 6$ matrix of the point correspondences which are extracted from views i, j and k. This matrix $\mathtt{S}_{ijk}$ is the same matrix as a system of linear equations for the trifocal tensor from point correspondences.

6.2.4 A simple SVD-based least-square minimization

The simple way to solve (6.16) is to use singular value decomposition (SVD) [16, 61]. If the SVD of $\mathtt{E}$ is $\mathtt{E} = \mathtt{U}\mathtt{D}\mathtt{V}^\top$, then the last column of matrix $\mathtt{V}$ is the vector $\mathbf{c}$ that minimizes $||\mathtt{E}\mathbf{c}||$. This minimization method is used by Kaucic et al. [37] but it gives unexpected results when there is a significant level of noise in the data. Figures 6.3 and 6.5 show results from the simple SVD-based method. You can notice the spiral and sharply changing camera motion in the figures. Because their method is not appropriate for real applications with noise data, we need a more reliable method to solve this problem.

6.3 A constrained minimization

In this section, we present a robust minimization method for translation estimation from omnidirectional images, which gives better results than the previous method [37]. The previous method which is based on SVD does not show a robust result when there is noise in data. Surprisingly, a slight modification can easily improve the result by changing given equations and introducing a constrained minimization method.

Equation (6.15) can be rewritten by dividing it into two components, $\mathtt{S}_{ij}$ and $\mathtt{T}_{ij}$, as follows:

$$\mathtt{E} = diag \begin{pmatrix} \mathtt{S}_{12} \\ \mathtt{S}_{13} \\ \mathtt{S}_{14} \\ \mathtt{S}_{23} \\ \mathtt{S}_{24} \\ \mathtt{S}_{34} \end{pmatrix} \begin{bmatrix} \mathtt{T}_1 & \mathtt{T}_2 & & \\ \mathtt{T}_1 & & \mathtt{T}_3 & \\ \mathtt{T}_1 & & & \mathtt{T}_4 \\ & \mathtt{T}_2 & \mathtt{T}_3 & \\ & \mathtt{T}_2 & & \mathtt{T}_4 \\ & & \mathtt{T}_3 & \mathtt{T}_4 \end{bmatrix}, \tag{6.18}$$

where $diag(\mathtt{S}_{ij})$ is a block diagonal matrix from $\mathtt{S}_{ij}$ and $\mathtt{T}_k$ is a 3×3 matrix. Therefore, $\mathtt{T}_{ij} = [\mathtt{T}_i \mid \mathtt{T}_j]$.

Let (6.18) be $\mathtt{E} = \mathtt{AG}$ as follows:

$$\mathtt{A} = diag \begin{pmatrix} \mathtt{S}_{12} \\ \mathtt{S}_{13} \\ \mathtt{S}_{14} \\ \mathtt{S}_{23} \\ \mathtt{S}_{24} \\ \mathtt{S}_{34} \end{pmatrix} \quad \text{and} \quad \mathtt{G} = \begin{bmatrix} \mathtt{T}_1 & \mathtt{T}_2 & & \\ \mathtt{T}_1 & & \mathtt{T}_3 & \\ \mathtt{T}_1 & & & \mathtt{T}_4 \\ & \mathtt{T}_2 & \mathtt{T}_3 & \\ & \mathtt{T}_2 & & \mathtt{T}_4 \\ & & \mathtt{T}_3 & \mathtt{T}_4 \end{bmatrix}. \tag{6.19}$$

Then, the original equation to be solved is as follows:

$$\begin{aligned} \mathtt{E}\mathbf{c} &= 0 \\ \mathtt{AG}\mathbf{c} &= 0\,, \end{aligned}$$

where $\mathbf{c}$ is a $3m \times 1$ vector consisting of the last columns of each projection matrix in m views. For example, in four views, $\mathbf{c} = [\mathbf{c}_{12}^\top,\ \mathbf{c}_{13}^\top,\ \mathbf{c}_{14}^\top,\ \mathbf{c}_{23}^\top,\ \mathbf{c}_{24}^\top,\ \mathbf{c}_{34}^\top]^\top$ becomes a 18×1 vector. Therefore, the problem becomes a constrained least-squares problem. All we have to do is to find a vector $\mathbf{x}$ that minimizes $||\mathtt{A}\mathbf{x}||$ subject to the condition $||\mathbf{x}|| = 1$ and $\mathbf{x} = \mathtt{G}\mathbf{c}$. We can find a vector $\mathbf{x}$ and also the translation vector $\mathbf{c}$ by using the algorithm A5.6 in [27, p 596].

The condition $||\mathbf{x}|| = 1$ gives a normalization of fundamental matrices instead of translations. The condition $\mathbf{x} = \mathtt{G}\mathbf{c}$ constrains $\mathbf{x}$ to lie in the column space of $\mathtt{G}$.

By slightly modifying the equations and adding constraints, we improved the translation estimation result dramatically. The results are shown in the following section.

6.4 Algorithm

The proposed algorithm estimating the translation from omnidirectional images is shown in algorithm 1.

Algorithm 1: *Translation estimation from omnidirectional images using trilinear relations.*

Input: (1) A set of point correspondences $\mathbf{x} \leftrightarrow \mathbf{x}' \leftrightarrow \mathbf{x}''$ across three views from a total of m views in an image sequence; (2) Rotations between views.
Output: Estimated translation.

for $i = 1, \ldots, m-2$ **do**
  for $j = i, \ldots, m-1$ **do**
    for $k = j, \ldots, m$ **do**
      Get a point correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}'_j \leftrightarrow \mathbf{x}''_k$ from three views.
      Multiply $\mathbf{x}_i$, $\mathbf{x}'_j$ and $\mathbf{x}''_k$ by the inverse of each rotation matrix at view i, j and k, respectively.
      Construct a matrix $\mathtt{S}_{ijk}$ from the point correspondence.
      Put $\mathtt{S}_{ijk}$ into a block diagonal matrix $\mathtt{A}$.
      Compute matrices $\mathtt{T}_i$, $\mathtt{T}_j$ and $\mathtt{T}_k$ from the point correspondences and put them into a matrix $\mathtt{G}$.
    end
  end
end
Find $\mathbf{c}$ which minimizes $||\mathtt{AG}\mathbf{c}||$ subject to $||\mathtt{G}\mathbf{c}|| = 1$.
Extract all 3-vectors of translations from $\mathbf{c}$.

6.5 Experiments

6.5.1 Synthetic experiments

Randomly distributed data in space is synthesized for omnidirectional cameras which have a circular motion on a plane. This is shown in figure Figure 6.2. A circle in the figure indicates a

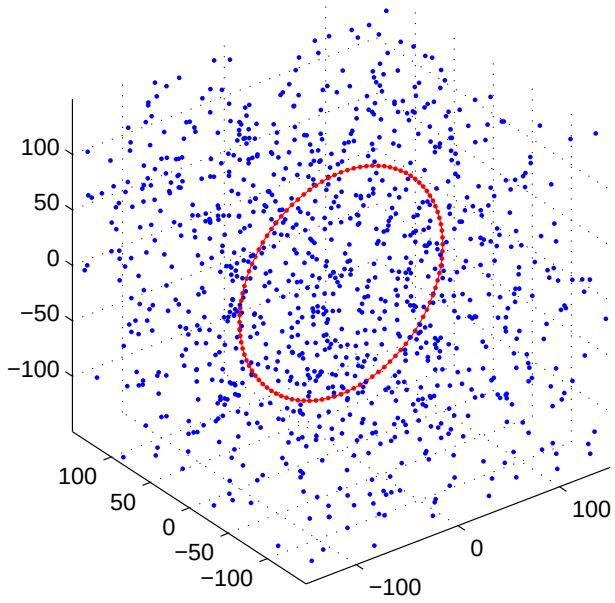
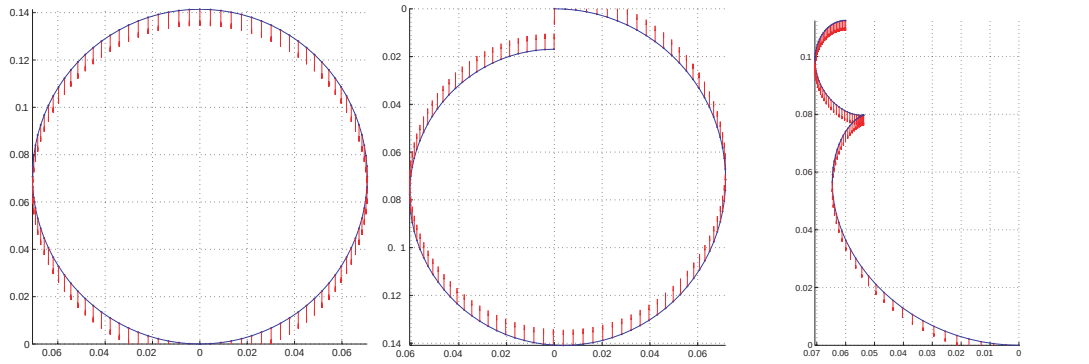


Figure 6.2: *Synthesized data for 101 omnidirectional cameras which have a circle planar motion and uniformly random distributed 1000 points. Each camera can see only a part of the data with a given visible range.*

circular motion of cameras. In particular, note that only a limited number of points are visible to each camera. The radius of the circle which is used for the camera motion is 100 units.

In Figure 6.3, we show the omnidirectional camera motion recovered using a fundamental matrix based method for changing levels of noise in the data. The result shows unexpected spirals and sharply changing trajectories of camera motion. It becomes a problem in the case of previous methods. Kaucic's method have the same problem when we use the same noisy data. The problem disappears when we use a long track length, i.e. more than 10 track lengths. However, we are concerned about the worst case such as when tracked features are short and have a lot of noise.

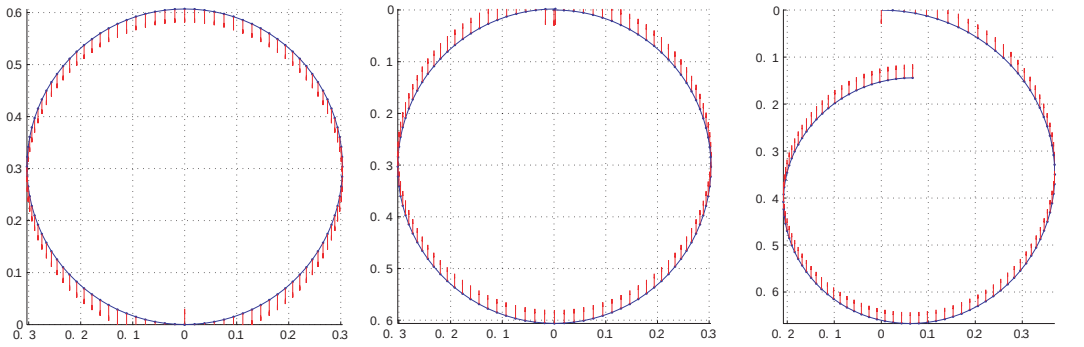
In Figure 6.4, we show the omnidirectional camera motion that is recovered using constrained minimization and a fundamental matrix based method for varying noise levels. As can be shown in Figure 6.4, it gives a better recovery than the result in Figure 6.3. Specifically, note



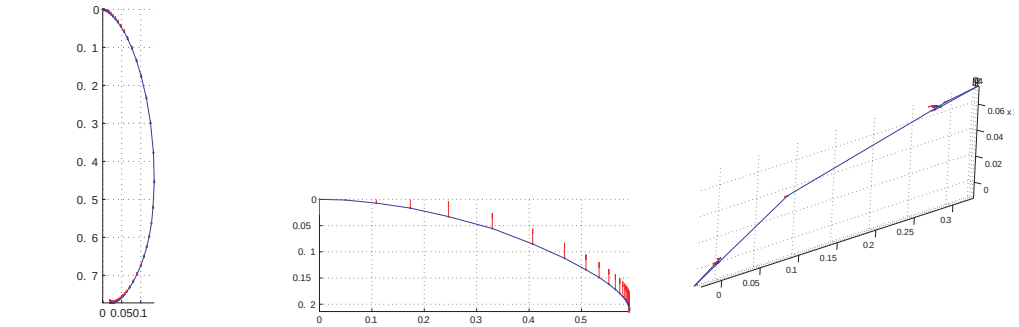
(a) $\sigma = 1.0 \times 10^{-7}$ (b) $\sigma = 1.2 \times 10^{-5}$ (c) $\sigma = 1.0 \times 10^{-4}$

(d) $\sigma = 5.0 \times 10^{-4}$ (e) $\sigma = 1.0 \times 10^{-3}$ (f) $\sigma = 1.0 \times 10^{-2}$

Figure 6.3: ***Bilinear + SVD.*** *The results obtained for a camera moving in a circular motion in a plane. The camera motion was recovered using the fundamental matrix and the SVD-based method for varying a standard deviation σ of Gaussian noise with a zero-mean. No track had a length greater than three.*

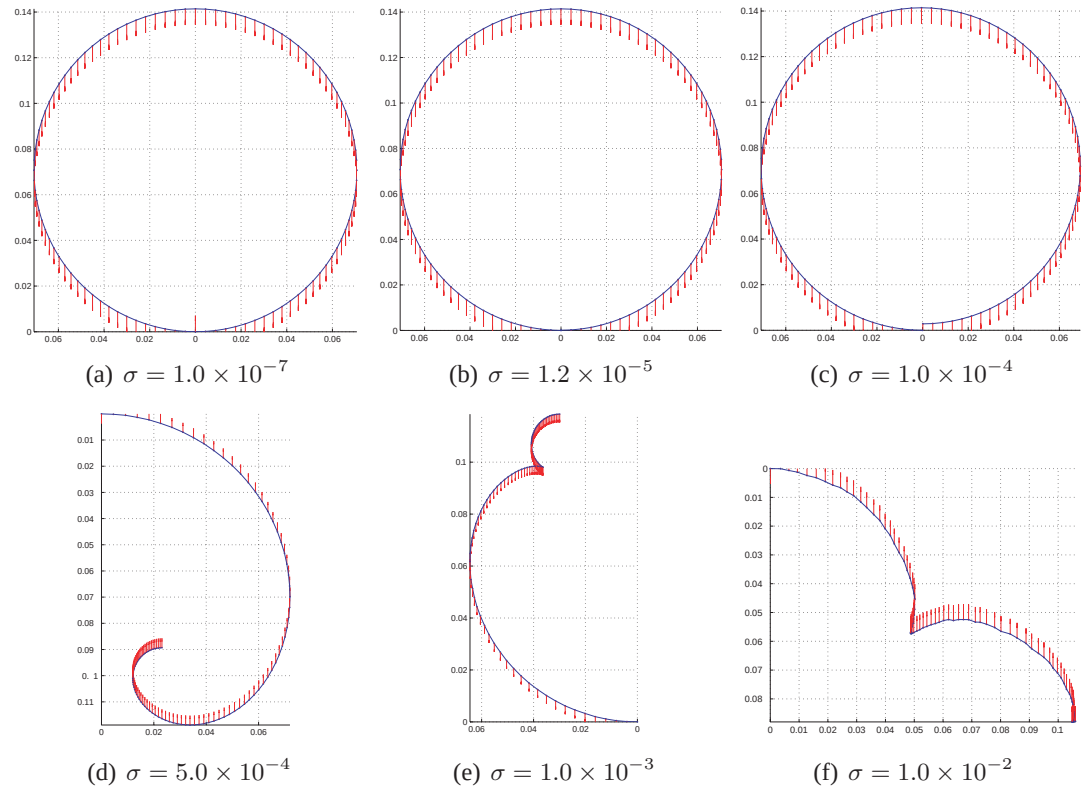


(a) $\sigma = 1.0 \times 10^{-7}$ (b) $\sigma = 1.2 \times 10^{-5}$ (c) $\sigma = 1.0 \times 10^{-4}$



(d) $\sigma = 5.0 \times 10^{-4}$ (e) $\sigma = 1.0 \times 10^{-3}$ (f) $\sigma = 1.0 \times 10^{-2}$

Figure 6.4: ***Bilinear + Constrained SVD.*** *The results obtained for a camera moving in a circular motion in a plane. The camera motion was recovered using the fundamental matrix and the constrained minimization for varying a standard deviation σ of Gaussian noise with a zero-mean. No track had a length greater than three.*



(a) $\sigma = 1.0 \times 10^{-7}$ (b) $\sigma = 1.2 \times 10^{-5}$ (c) $\sigma = 1.0 \times 10^{-4}$

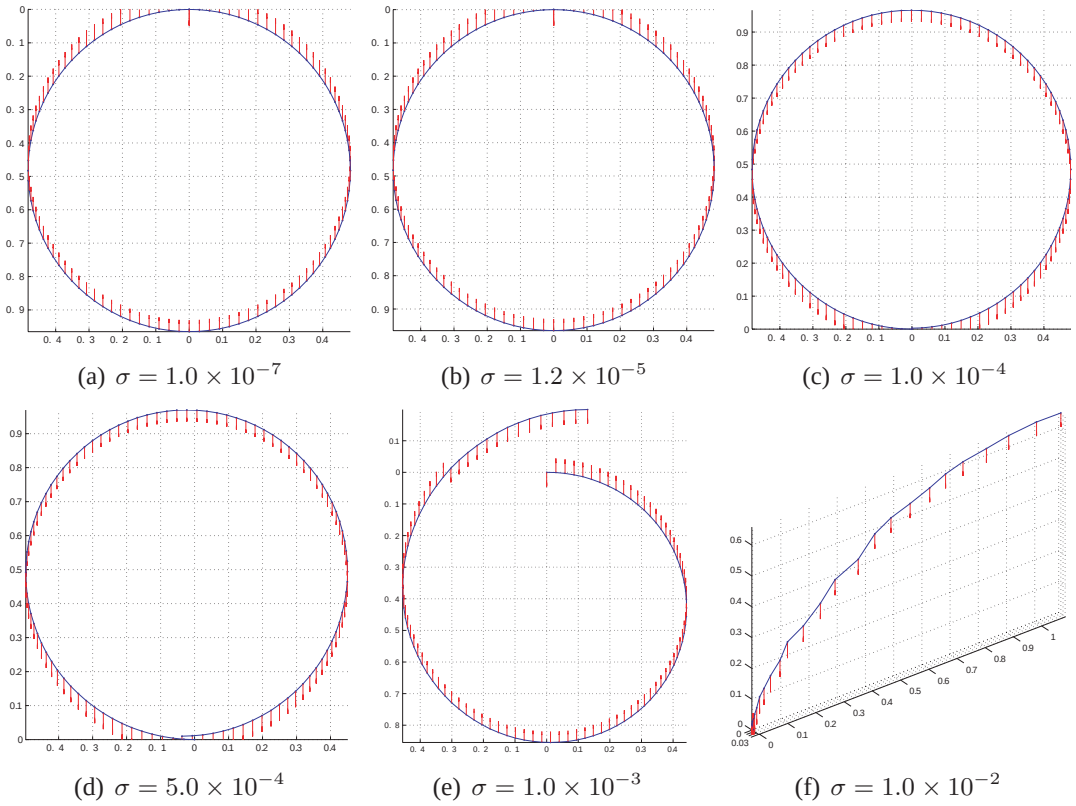
(d) $\sigma = 5.0 \times 10^{-4}$ (e) $\sigma = 1.0 \times 10^{-3}$ (f) $\sigma = 1.0 \times 10^{-2}$

Figure 6.5: ***Trilinear + SVD.*** *The results obtained for a camera moving in a circular motion in a plane. The camera motion was recovered using the trifocal tensor and the SVD-based method for varying a standard deviation σ of Gaussian noise with a zero-mean. No point correspondences were tracked across more than three views.*

that there are no spirals or unexpected changing trajectories in Figure 6.4 as in Figure 6.3. Particularly, at the same level of noise, Figure 6.4-(c) shows a significant improvement compared with Figure 6.3-(c).

In Figure 6.5, we show the omnidirectional camera motion recovered using a trifocal tensor by changing the noise in the data. By adding more noise, it gives unstable recovery. Particularly, Figure 6.5-(d), (e) and (f) show spirals or unexpected changing trajectories. This result using a trifocal tensor, as shown in Figure 6.5, is much better than the result obtained using a fundamental matrix, as shown in Figure 6.3. The result is improved by increasing the number of views.

In Figure 6.6, we show the omnidirectional camera motion that is recovered using a constrained minimization and a trifocal tensor based method for varying noise levels. The result



(a) $\sigma = 1.0 \times 10^{-7}$ (b) $\sigma = 1.2 \times 10^{-5}$ (c) $\sigma = 1.0 \times 10^{-4}$

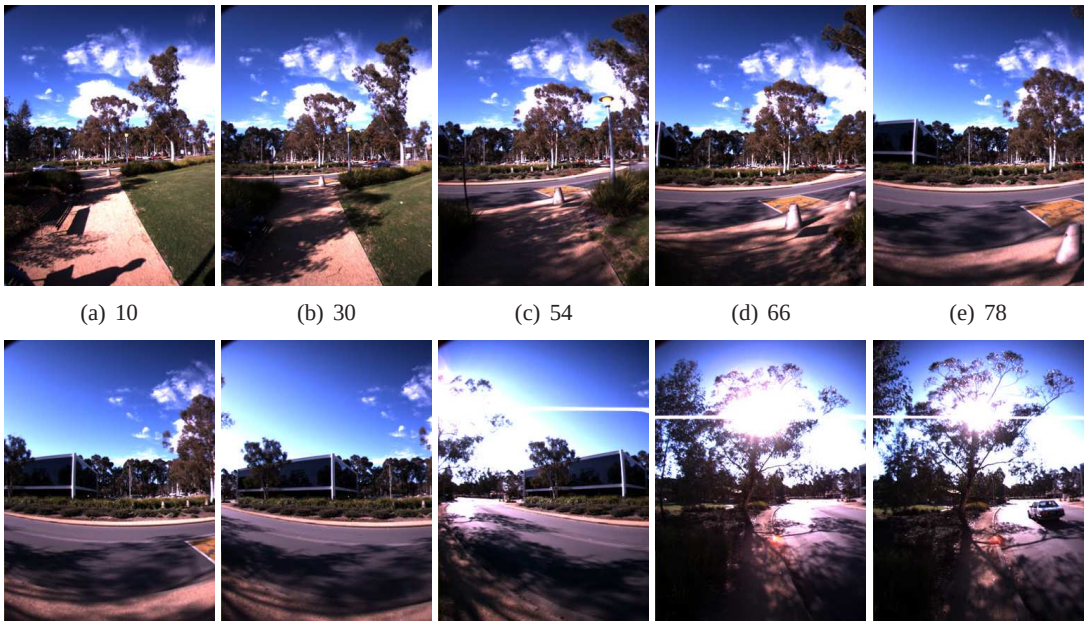
(d) $\sigma = 5.0 \times 10^{-4}$ (e) $\sigma = 1.0 \times 10^{-3}$ (f) $\sigma = 1.0 \times 10^{-2}$

Figure 6.6: ***Trilinear + Constrained SVD.*** *The results obtained for a camera moving in a circular motion in a plane. The camera motion was recovered using the trifocal tensor and the constrained minimization method for varying a standard deviation σ of Gaussian noise with a zero-mean. No track had a length greater than three.*

in Figure 6.6 shows a better recovery than the result in Figure 6.5. Specifically, note that there are no spirals or unexpected changing trajectories in Figure 6.6 as it is in Figure 6.5. Particularly, at the same level of noise, Figure 6.6-(d) shows a significant improvement compared with Figure 6.5-(d).

6.5.2 Real experiments

Experiments with real data are carried out. An image sequence is acquired by Ladybug2 camera [32]. The Ladybug2 camera consists of 6 cameras and it capture a spherical image which covers around a hemisphere from the position of the camera. So, with the Ladybug2 camera, an omnidirectional image sequence can be captured. The Ladybug2 camera is mounted on a helmet and the helmet is worn on the head of a person who has equipped with a laptop computer



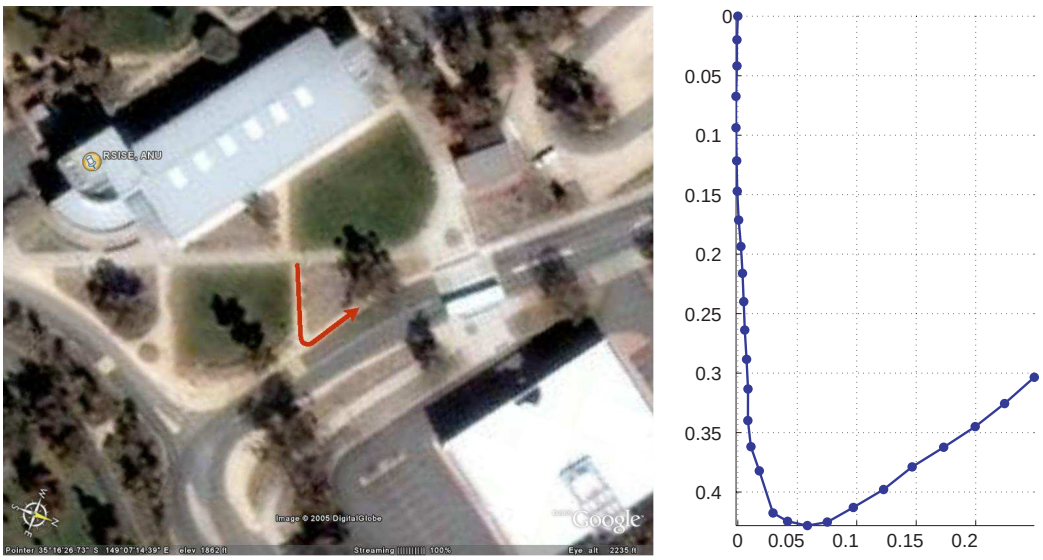
(a) 10 (b) 30 (c) 54 (d) 66 (e) 78

(f) 82 (g) 86 (h) 94 (i) 114 (j) 122

Figure 6.7: *Sample 10 images taken by a camera in the forward direction from Ladybug2 camera. The number at the bottom of each image is the frame number from total 136 images.*

which connected to the Ladybug2 camera to acquire a video. Then, while the person moves along a path, the Ladybug2 camera captures all 6 image sequences around the environment surrounding the person. Sample images from the Ladybug2 camera are shown in Figure 6.7. These images are taken from a camera in the forward direction of the moving of the person. The path which the person follows in this experiment is shown in Figure 6.8(a). The number of captured images is total 139 frames and the size of the images is 1024×768 pixels.

Features in the images are detected and matched to get a set of point correspondences across views by using the Kanade-Lucas-Tomasi (KLT) tracker [47]. Because of the wide field of view, the acquired images are distorted by radial distortion. The distortion in images are corrected by radial distortion information provided by Point Grey Inc. The pixel coordinates of the tracked features are also corrected in the same way as correcting radial distortion in the images. After that, the random sample consensus (RANSAC) approach is applied to remove outliers in the matched points [13]. Then, the set of point correspondences in all 6 cameras is transformed to have coordinates on a unit sphere. This whole process gives us image vectors



(a) The ground path on the map

(b) Estimated path

Figure 6.8: *(a) The ground path of cameras movement is indicated as a red arrow on a map from GoogleEarth,* `http://earth.google.com`*. (b) The estimated path from our proposed method. It starts from* $(0,\ 0)$*, and blue dots represent the positions of the centre of the Ladybug2 camera.*

which represent direction vectors from the centre of the omnidirectional camera to the point on a unit sphere as shown in Figure 6.1.

With all these transformed image vectors across two views or three views, a set of linear equations is constructed from bilinear relations or trilinear relations. The translations in camera motion are found by solving the constrained minimization problem as shown in algorithm 1.

The result is shown in Figure 6.8. The estimated path from translation estimation is superimposed over the Google map of the site where experiments are carried out. The estimated translation has some drift and jittering when the motion goes far from the initial starting point. However, the path is reasonably correct and it gives good estimation of rotation even when the person makes a turn left significantly. Note that the errors on the estimation are accumulated over all frames.

6.6 Conclusion

A translation estimation method from omnidirectional images is presented. The translation estimation method is based on the plane-based projective reconstruction problem with missing data. Therefore, it does not require that a point correspondence be seen in all views. We assume that the rotations of each camera are known. These rotations can be computed from other methods [24, 46]. This method uses a constrained minimization instead of a least-square minimization. This linear method gives a stable and reasonable recovery result, and therefore can be used as a good initial estimate for the next non-linear minimization step such as a bundle adjustment [82]. For the implementation of an omnidirectional camera, we may use two conventional cameras which are placed back-to-back to obtain the entire sphere of view [55].

Chapter 7

Robust 6 DOF Motion Estimation for Non-Overlapping Multi-Camera Rigs

Motion estimation of multi-camera systems has become of more interest with use of these systems for the capture of ground based or indoor video data to allow a reconstruction of the whole surrounding environment [83]. Capture systems need to provide a large field of view horizontally and vertically to cover the upper hemisphere of the environment. An efficient system, for example, is often build of two wide field of view cameras rigidly coupled together. Alternatively, each wide field of view camera is replaced by a camera cluster. To closely approximate the wide field of view camera, the optical centres of the cameras are as close together as possible and the cameras have no or very small overlap. This avoids parallax effects in between cameras. There are also systems that only consist of one camera cluster that captures the whole upper hemisphere. As we will show in our analysis, it is generally very challenging to recover a 6 degrees of freedom (DOF) motion for the latter type of cameras.

An example of a multi-camera system for the capture of ground based video is shown in Figure 7.1. This system consists of two camera clusters on each side of a vehicle. The cameras are attached tightly to the vehicle. Accordingly, they move in rigid motions. The shown system will later be used for experimental evaluation of our approach.

In this chapter, related work is discussed in the next section, and our novel 6 degrees of freedom motion estimation method for non-overlapping multi-camera rigs is introduced in section 7.3. In section 7.5, experiments with synthetic and real data are carried out.

Figure 7.1: *Example of a multi-camera system on a vehicle. (Courtesy of UNC-Chapel Hill)*

7.1 Related work

There has been a lot of study on the motion estimation of multi-camera systems [58, 14, 80]. Some approaches use stereo/multi-camera systems to estimate the ego-motion of the camera system. Nistér et al. proposed a technique that uses a calibrated stereo camera system for visual navigation in [58]. They used the stereo camera system to recover 3D points up to an unknown orientation. Frahm et al. introduced a 6 degrees of freedom estimation technique for a multi-camera system [14]. Their approach assumed overlapping views of the cameras to obtain the scale of the camera motion. Tariq and Dellaert proposed a 6 degrees of freedom tracker for a multi-camera system for head tracking using nonlinear optimization [80].

In this chapter, we propose an algorithm estimating 6 degrees of freedom motion of multi-camera systems. However, it does not require to have overlapping views and does not need to know the positions of the observed scene. In other words, 3D structure reconstruction is not required to estimate the 6 degrees of freedom motion.

Another type of approach is based on the generalized camera model [18, 60]. A stereo or multi-camera system is an example of generalized cameras. A generalized camera is a type of camera which may have different centres of projection. Without loss of generality, generalized cameras also can represent a type of single central projection camera. The single central projection cameras are an ordinary type of camera having all centres of projection identical. Nowadays, they are widely used by general customers. Accordingly, multi-camera systems

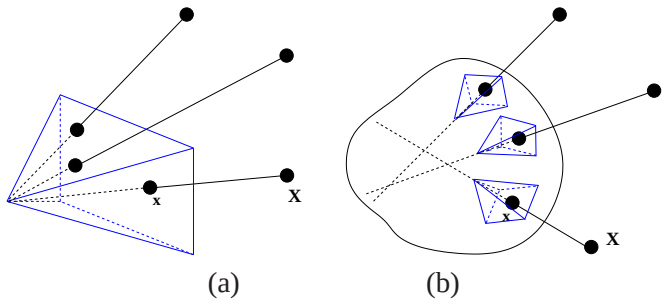


Figure 7.2: *(a) Ordinary camera and (b) Generalized camera.*

can be considered as a type of generalized cameras having multiple centres of projection for each physical camera [18, 60]. Figure 7.2 shows an illustration of an ordinary camera and a generalized camera.

The concept of generalized cameras was proposed by Grossberg and Nayar in [18]. Sturm showed a hierarchy of generalized camera models and multiview linear relations for generalized cameras [77]. A solution for the motion of a generalized camera is proposed by Stewénius et al [74]. They showed that there are up to 64 solutions for the relative position of two generalized cameras given 6 point correspondences. Their method delivers a rotation, translation and scale of a freely moving generalized camera. One of the limitations of the approach is that centres of projection can not be collinear. It means that their method can not solve a motion for the axial case of generalized cameras. The definition of axial cameras is shown in [77]. This limitation naturally excludes all two camera systems as well as a system of two camera clusters where the cameras of the cluster have approximately the same centre of projection. The approach of Stewénius et al. can not estimate the camera motion for pure translation at all, and the algorithm fails to give any result. Our method also can be affected by the pure translation, and may not return the 6 degrees of freedom of motion. However, at least, for the pure translation case, our proposed method can estimate 5 degrees of freedom motion without the scale of translation. Our method also uses 6 points to estimate the 6 degrees of freedom motion. The next section will introduce our novel approach for the 6 degrees of freedom estimation of a multi-camera system.

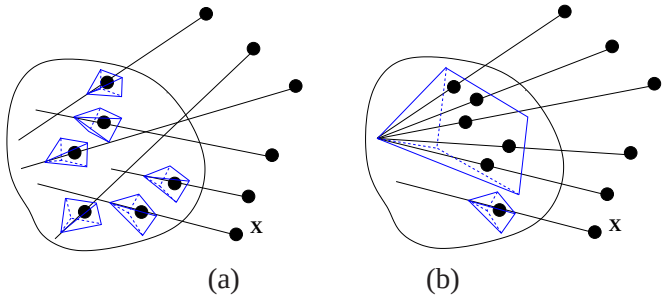


Figure 7.3: ***Examples of a generalized camera.*** *(a) Six rays never meet each other. It could be considered as that 6 rays are projected by each different camera. (b) Five rays meet on a single centre of projection and another ray does not meet the centre.*

7.2 6 DOF multi-camera motion

The proposed approach addresses the motion estimation of multi-camera systems. Multi-camera systems may have multiple centres of projection. In other words, they may consist of multiple conventional (central projection) cameras. For instance, a stereo system is one of the examples of multi-camera systems. However, multi-camera systems could have little overlapping views, for example, such as an omni-directional camera, Ladybug™2 [32]. These multi-camera systems are examples of generalized cameras. The most general type of generalized cameras may not have common centre of projection as shown in Figure 7.3. However, that case is rare in real applications. Practically, multi-camera systems are more frequently used. Our technique assumes that we observe at least five correspondences from one of the cameras and one correspondence from any additional camera. In practice this assumption is not a limitation as a reliable estimation of camera motion requires multiple correspondences due to noise.

Suppose that there is a set of calibrated cameras moving from one position to another. An essential matrix which describes the epipolar geometry of the calibrated camera can be estimated from five point correspondences in one camera. Nistér proposed an efficient algorithm for this estimation in [56]. It delivers up to ten valid solutions for the epipolar geometry. The ambiguity can be eliminated with one additional point correspondence. A rotation and a translation (up to scale) of the motion of the camera can be extracted from the essential matrix.

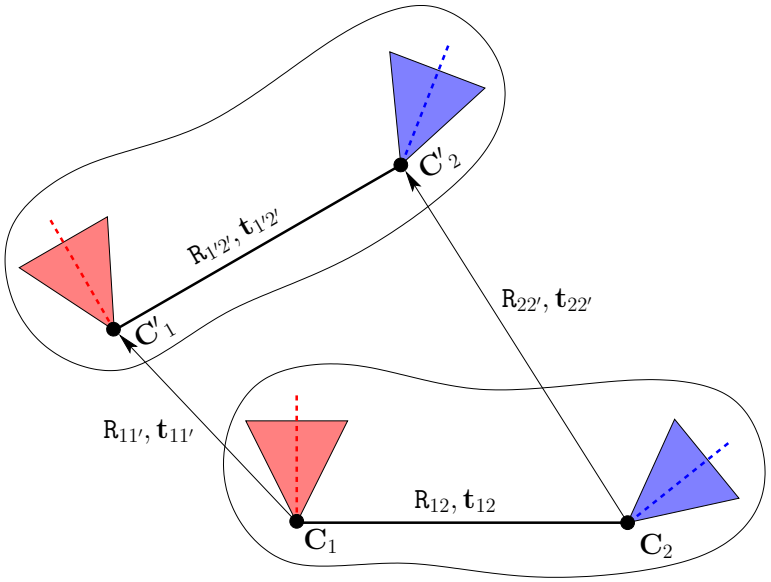


Figure 7.4: *Motion of a generalized camera which consists of two cameras attached together. A camera at $\mathbf{C}_1$ and a camera at $\mathbf{C}_2$ build a cluster of cameras and they become a multi-camera system. They move together from one place (position $\mathbf{C}_1$ and $\mathbf{C}_2$) to another (position $\mathbf{C}'_1$ and $\mathbf{C}'_2$) by a rigid motion.*

Consequently, a single camera provides 5 degrees of freedom of the camera motion. The remaining degree is the scale of the translation. Given these 5 degrees of freedom of the motion of the multi-camera system (rotation and translation direction) we can compensate for the rotation of the system. Our approach is based on the observation that the position of the epipole of each camera in the multi-camera system is restricted to a line in the image. Hence the scale as the remaining degree of freedom of the camera motion describes a linear subspace.

In the next following sections, we derive the mathematical basis of our approach to motion recovery.

7.3 Two camera system – Theory

We consider a set-up involving two cameras, rigidly configured with respect to each other. The two cameras are assumed to be calibrated. Figure 7.4 shows the configuration of the two-camera system.

A rigid motion of the two-camera system consists of a rotation and a translation between

two positions. The motion may be estimated from one of cameras in the two-camera system by using any relative motion estimation algorithm [27, 46, 56]. However, this gives only 5 degrees of freedom of motion, so the scale of translation is not solved.

In this section, mathematical derivations for two-camera systems are given and it is shown that in two-camera systems it is possible to find 6 degrees of freedom of motion of the two-camera systems, although the images may not overlap each other.

Theorem 2. *Let* $\mathtt{P}_1$ *and* $\mathtt{P}_2$ *be two cameras in a two-camera system, written as* $\mathtt{P}_1 = [\mathtt{I} \mid \mathbf{0}]$ *and* $\mathtt{P}_2 = [\mathtt{R}_2 \mid -\mathtt{R}_2\mathbf{C}_2]$. *Suppose that they move rigidly to a new position where the first camera is specified by* $\mathtt{P}'_1 = [\mathtt{R}'_1 \mid -\lambda\mathtt{R}'_1\mathbf{C}'_1]$. *Then, given a point correspondence* $\mathbf{x} \leftrightarrow \mathbf{x}'$ *from the second camera, the scale of translation* λ *is determined by an equation as follows:*

$$\mathbf{x}'^{\top}\mathtt{A}\mathbf{x} + \lambda\mathbf{x}'^{\top}\mathtt{B}\mathbf{x} = 0\,, \tag{7.1}$$

where $\mathtt{A} = \mathtt{R}_2\mathtt{R}'_1\,[(\mathtt{R}'^{\top}_1 - \mathtt{I})\mathbf{C}_2]_\times\mathtt{R}_2^{\top}$ *and* $\mathtt{B} = \mathtt{R}_2\mathtt{R}'_1\,[\mathbf{C}'_1]_\times\mathtt{R}_2^{\top}$.

In order to simplify the derivation, we assumed that a coordinate system is aligned with the initial position of the first camera, so that $\mathtt{P}_1 = [\mathtt{I} \mid \mathbf{0}]$. Any other coordinate system is easily transformed to this coordinate system by a Euclidean transformation. The first camera has moved to a new position at $\lambda\mathbf{C}'_1$.

Proof. Our immediate goal is to determine the camera matrix for the second camera after the motion. First note that the camera $\mathtt{P}'_1$ may be written as

$$\mathtt{P}'_1 = [\mathtt{I} \mid \mathbf{0}]\begin{bmatrix} \mathtt{R}'_1 & -\lambda\mathtt{R}'_1\mathbf{C}'_1 \\ \mathbf{0}^{\top} & 1 \end{bmatrix} = \mathtt{P}_1\mathtt{T}\,,$$

where the matrix $\mathtt{T}$, so defined, may be thought of as a Euclidean transformation induced by the motion of the camera pair. Since the second camera undergoes the same Euclidean motion,

we can compute the form of the camera $\mathtt{P}'_2$ to be

$$\begin{aligned}
\mathtt{P}'_2 &= \mathtt{P}_2\mathtt{T} \\
&= [\mathtt{R}_2 \mid -\mathtt{R}_2\mathbf{C}_2] \begin{bmatrix} \mathtt{R}'_1 & -\lambda\mathtt{R}'_1\mathbf{C}'_1 \\ \mathbf{0}^\top & 1 \end{bmatrix} \\
&= [\mathtt{R}_2\mathtt{R}'_1 \mid -\lambda\mathtt{R}_2\mathtt{R}'_1\mathbf{C}'_1 - \mathtt{R}_2\mathbf{C}_2] \\
&= \mathtt{R}_2\mathtt{R}'_1[\mathtt{I} \mid -(\lambda\mathbf{C}'_1 + {\mathtt{R}'_1}^\top\mathbf{C}_2)] \,. \qquad (7.2)
\end{aligned}$$

From the form of the two camera matrices $\mathtt{P}_2$ and $\mathtt{P}'_2$, we may compute the essential matrix for the second camera as follows:

$$\begin{aligned}
\mathtt{E}_2 &= \mathtt{R}_2\mathtt{R}'_1[\lambda\mathbf{C}'_1 + {\mathtt{R}'_1}^\top\mathbf{C}_2 - \mathbf{C}_2]_\times\mathtt{R}_2^\top \\
&= \mathtt{R}_2\mathtt{R}'_1[{\mathtt{R}'_1}^\top\mathbf{C}_2 - \mathbf{C}_2]_\times\mathtt{R}_2^\top + \lambda\mathtt{R}_2\mathtt{R}'_1[\mathbf{C}'_1]_\times\mathtt{R}_2^\top \qquad (7.3) \\
&= \mathtt{A} + \lambda\mathtt{B} \,.
\end{aligned}$$

Now, given a single point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$ as seen in the second camera, we may determine the value of λ, the scale of the camera translation. The essential matrix equation ${\mathbf{x}'}^\top\mathtt{E}\mathbf{x} = 0$ yields ${\mathbf{x}'}^\top\mathtt{A}\mathbf{x} + \lambda{\mathbf{x}'}^\top\mathtt{B}\mathbf{x} = 0$, and hence

$$\lambda = -\frac{{\mathbf{x}'}^\top\mathtt{A}\mathbf{x}}{{\mathbf{x}'}^\top\mathtt{B}\mathbf{x}} = -\frac{{\mathbf{x}'}^\top\left(\mathtt{R}_2\mathtt{R}'_1[{\mathtt{R}'_1}^\top\mathbf{C}_2 - \mathbf{C}_2]_\times{\mathtt{R}_2}^\top\right)\mathbf{x}}{{\mathbf{x}'}^\top\left(\mathtt{R}_2\mathtt{R}'_1[\mathbf{C}'_1]_\times\mathtt{R}_2^\top\right)\mathbf{x}} \,. \qquad (7.4)$$

□

7.3.1 Geometric interpretation

The situation may be understood via a different geometric interpretation as shown in Figure 7.5. We note from (7.2) that the second camera moves to a new position $\mathbf{C}'_2(\lambda) = {\mathtt{R}'_1}^\top\mathbf{C}_2 + \lambda\mathbf{C}'_1$. The locus of this point for varying values of λ is a straight line with direction vector $\mathbf{C}'_1$, passing through the point ${\mathtt{R}'_1}^\top\mathbf{C}_2$. From its new position, the camera observes a point at position $\mathbf{x}'$ in

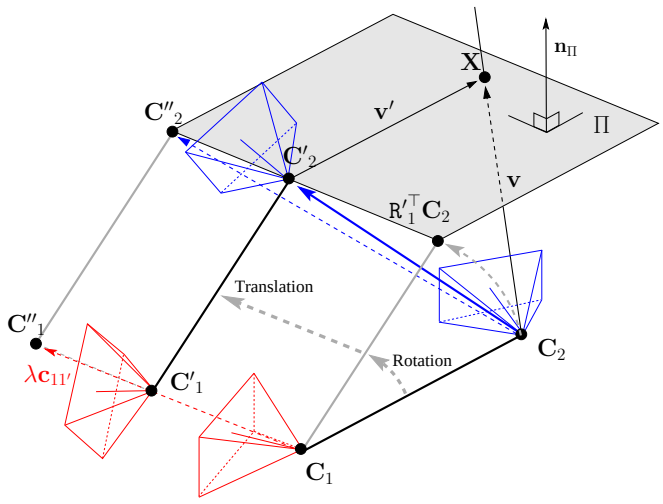


Figure 7.5: *The 3D point* $\mathbf{X}$ *must lie on the plane traced out by the ray corresponding to* $\mathbf{x}'$ *for different values of the scale* λ*. It also lies on the ray corresponding to* $\mathbf{x}$ *through the initial camera centre* $\mathbf{C}_2$*.*

its image plane. This image point corresponds to a ray $\mathbf{v}'$ along which the 3D point $\mathbf{X}$ must lie. If we think of the camera as moving along the line $\mathbf{C}'_2(\lambda)$, then this ray traces out a plane Π; The 3D point $\mathbf{X}$ must lie on this plane.

On the other hand, the point $\mathbf{X}$ is also seen (as image point $\mathbf{x}$) from the initial position of the second camera, and hence lies along a ray $\mathbf{v}$ through $\mathbf{C}_2$. The point where this ray meets the plane Π must be the position of the point $\mathbf{X}$. In turn this determines the scale factor λ.

7.3.2 Critical configurations

This geometric interpretation allows us to identify critical configurations in which the scale factor λ cannot be determined. As shown in Figure 7.5, the 3D point $\mathbf{X}$ is the intersection of the plane Π with a ray $\mathbf{v}$ through the camera centre $\mathbf{C}_2$. If the plane does not pass through $\mathbf{C}_2$, then the point $\mathbf{X}$ can be located as the intersection of plane and ray. Thus, the only possible critical configurations are where the plane Π passes through the second camera centre, $\mathbf{C}_2$.

According to the construction, the line $\mathbf{C}'_2(\lambda)$ (the locus of possible final positions of the second camera centre) lies on the plane Π. For different 3D points $\mathbf{X}$, and corresponding image measurement $\mathbf{x}'$, the plane will vary, but always contain the line $\mathbf{C}'_2(\lambda)$. Thus, the planes Π corresponding to different points $\mathbf{X}$ form a pencil of planes hinged around the axis line $\mathbf{C}'_2(\lambda)$.

Unless this line actually passes through $\mathbf{C}_2$, there will be at least one point $\mathbf{X}$ for which $\mathbf{C}_2$ does not lie on the plane Π, and this point can be used to determine the point $\mathbf{X}$, and hence the scale.

Finally, if the line $\mathbf{C}'_2(\lambda)$ passes through the point $\mathbf{C}_2$, then the method will fail. In this case, the ray corresponding to any point $\mathbf{X}$ will lie within the plane Π, and a unique point of intersection cannot be found.

In summary, if the line $\mathbf{C}'_2(\lambda)$ does not pass through the initial camera centre $\mathbf{C}_2$, almost any point correspondence $\mathbf{x}' \leftrightarrow \mathbf{x}$ may be used to determine the point $\mathbf{X}$ and the translation scale λ. The exceptions are point correspondences given by points $\mathbf{X}$ that lie in the plane defined by the camera centre $\mathbf{C}_2$ and the line $\mathbf{C}'_2(\lambda)$.

If on the other hand, the line $\mathbf{C}'_2(\lambda)$ passes through the centre $\mathbf{C}_2$, then the method will always fail. It may be seen that this occurs most importantly if there is no camera rotation, namely $\mathtt{R}'_1 = \mathtt{I}$. In this case, we see that $\mathbf{C}'_2(\lambda) = \mathbf{C}_2 + \lambda\mathbf{C}'_1$, which passes through $\mathbf{C}_2$. It is easy to give an algebraic condition for this critical condition. Since $\mathbf{C}'_1$ is the direction vector of the line, the point $\mathbf{C}_2$ will lie on the line precisely when the vector ${\mathtt{R}'_1}^\top\mathbf{C}_2 - \mathbf{C}_2$ is in the direction $\mathbf{C}'_1$. This gives a condition for singularity $({\mathtt{R}'_1}^\top\mathbf{C}_2 - \mathbf{C}_2) \times \mathbf{C}'_1 = \mathbf{0}$, or rearranging this expression, and observing that the vector $\mathbf{C}_2 \times \mathbf{C}'_1$ is perpendicular to the plane of the three camera centres $\mathbf{C}_2$, $\mathbf{C}'_1$ and $\mathbf{C}_1$ (the last of these being the coordinate origin), we may state:

Theorem 3. *The critical condition for singularity for scale determination is*

$$({\mathtt{R}'_1}^\top\mathbf{C}_2) \times \mathbf{C}'_1 = \mathbf{C}_2 \times \mathbf{C}'_1 \ .$$

In particular, the motion is not critical unless the axis of rotation is perpendicular to the plane determined by the three camera centres $\mathbf{C}_2$, $\mathbf{C}'_1$ *and* $\mathbf{C}_1$.

7.4 Algorithm

Figure 7.6 shows our proposed algorithm solving relative motion of two generalized cameras from 6 rays with two centres where 5 rays meet one centre and another ray meets another cen-

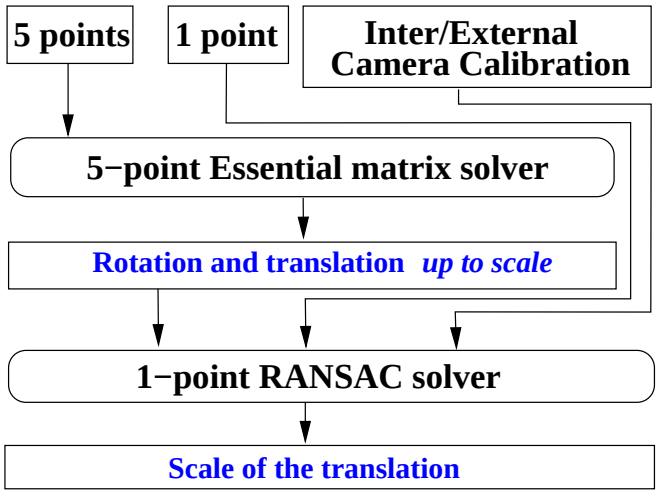


Figure 7.6: *Flow chart of algorithm for 5–1 camera configuration of generalized camera.*

tre. Firstly, we use 5 points in one ordinary camera to estimate an essential matrix between two views. Stewénius's method is used to estimate the essential matrix from 5 points [71]. There is also a simple derivation of the algorithm by Li et al. which uses 5 points and gives the same result [44]. The 5 points are selected by the random sample consensus (RANSAC) algorithm which gives us a guarantee that the selected 5 points are inliers [13]. A distance between the selected point and the corresponding epipolar line is used as criteria for the RANSAC algorithm. The essential matrix is decomposed to a skew-symmetric matrix of translation and a rotation matrix. When the essential matrix is decomposed, it should be considered that there existe an ambiguity on deciding a correct rotation matrix and a correct translation direction [27]. However, the translation is up to scale. So, we need to get the scale of the translation for the 6 DOF solution. The scale of translation can be determined by (7.4). However, the one point correspondence from a second camera is very essential to determine the scale of translation. Therefore, we incorporate RANSAC algorithm to select the best one point correspondence for estimating the scale of translation. In this RANSAC step, we select one point by checking a distance between the point and the corresponding epipolar line of the point. Finally, with the scale of translation, we get the motion of two-camera systems from 5 points from one camera and 1 point from another camera.

7.5 Experiments

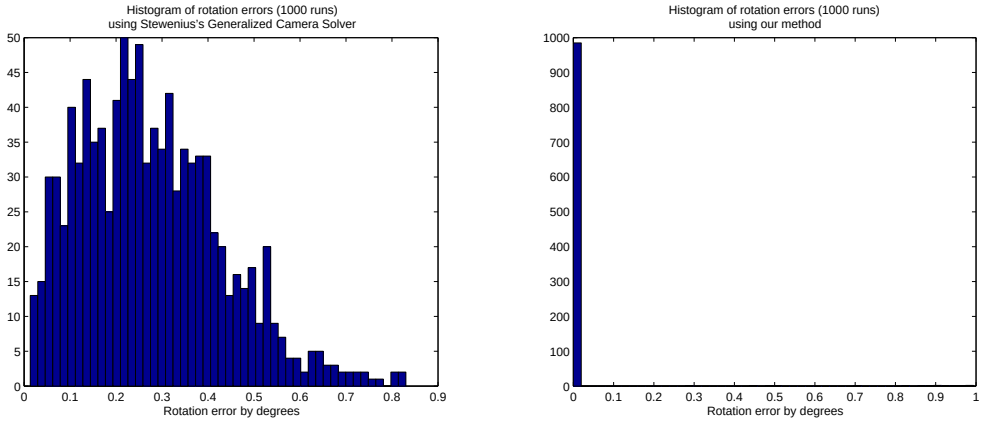
7.5.1 Synthetic data

Stewénius introduced a relative motion estimation algorithm for generalized cameras using a Gröbner basis [74]. It is possible to estimate the relative motion of two-camera systems using his method. In this section, we compare our method with Stewénius's method in the same configuration of synthetic data. Then, we examine which method gives better results of estimation.

First, we compute a relative rotation of a generalized camera model which consists of two central projection cameras. The synthetic data has two central projection cameras which form a generalized camera system, and they are located at random positions in the world coordinate system. Six points are placed randomly in space and they are projected onto each image plane of the two central projection cameras. Because we know the position of both centres of the central projection cameras and the six points in space, Plücker coordinates for six rays can be obtained. These six rays, represented by Plücker line coordinates, are used to estimate a relative motion of the generalized camera using Stewénius's method. For experiments with our method, the same set of data is used but note that Plücker line coordinates are not needed in our method. In this synthetic data, five points from the first camera are used to estimate an essential matrix, and then one point from the second camera is used to estimate the scale of translation. Let us call this configuration of the two-camera systems a "5+1 camera configuration".

The comparion between our method and Stewénius's method is shown in Figure 7.7(a) and Figure 7.7(b). Figure 7.7(a) shows a histogram of rotation error by Stewénius's method, and for 1,000 runs it gives less than 1.0 degree of rotation error in this 5+1 camera configuration. However, as shown in Figure 7.7(b), our method shows less than 0.1 degrees of rotation error with the same data, so our method gives better results of estimation than Stewénius's method for this 5+1 configuration. Note that our method is only applicable to this 5+1 configuration, not to all generalized camera models.

As shown in Figure 7.8, we show how sensitive our method is under the assumption of Gaussian measurement noise. The configuration of the generalized camera is the same as in



(a) (b)

Figure 7.7: *(a) Histogram of rotation error produced by Stewénius's method for a generalized camera having two centres where 5 rays meet on one centre and other ray meets on the other centre in two views. There are no noise on data. (b) Histogram of rotation error produced by our method for a generalized camera having two centres where 5 rays meet on one centre and other ray meets on the other centre in two views. There are no noise on data.*

the above experiment except the measurement noise. Experiments are carried out for various standard deviations of Gaussian measurement noise.

7.5.2 Real data

An experiment with real data is carried out in this section. An image sequence is captured by 8 cameras mounted on a vehicle. The vehicle is shown in Figure 7.9. All 8 cameras are firmly mounted on the vehicle, and 4 of them are assigned on the left side and the other 4 cameras are assigned on the right side of the vehicle to have wide field of view. The distance between a set of 4-camera on the left and a set of 4-camera on the right is about 1.9 metres. The position of 8 cameras is shown in Figure 7.10. These cameras have little field of overlapping views with each other. So, it is an example of a real implementation of a non-overlapping multi-camera systems. The size of the images is 1024×768 pixels, and the number of frames in the image sequence for each camera is about 1,000 frames. So, a total of 8,000 frames are dealt with in this real experiment. In Figure 7.9, a sample of captured images from 8 cameras is shown. Note that there is a very little overlapping field of view. In this experiment, only two cameras, one from the left side and another from the right side, are selected.

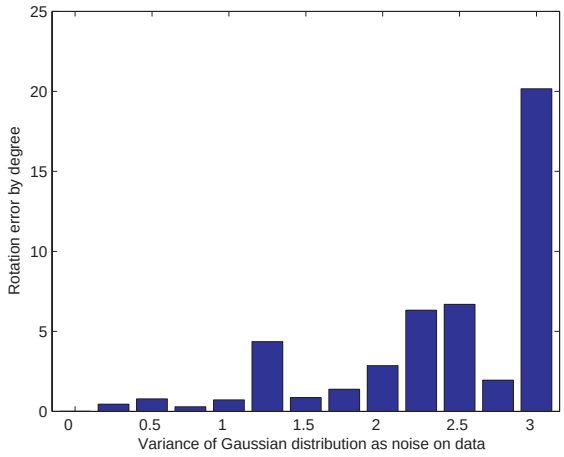


Figure 7.8: *Rotation error produced by our method for a generalized camera having two centres where 5 rays meet on one centre and other ray meets on the other centre in two views. Gaussian distribution of noise has been added to the data.*

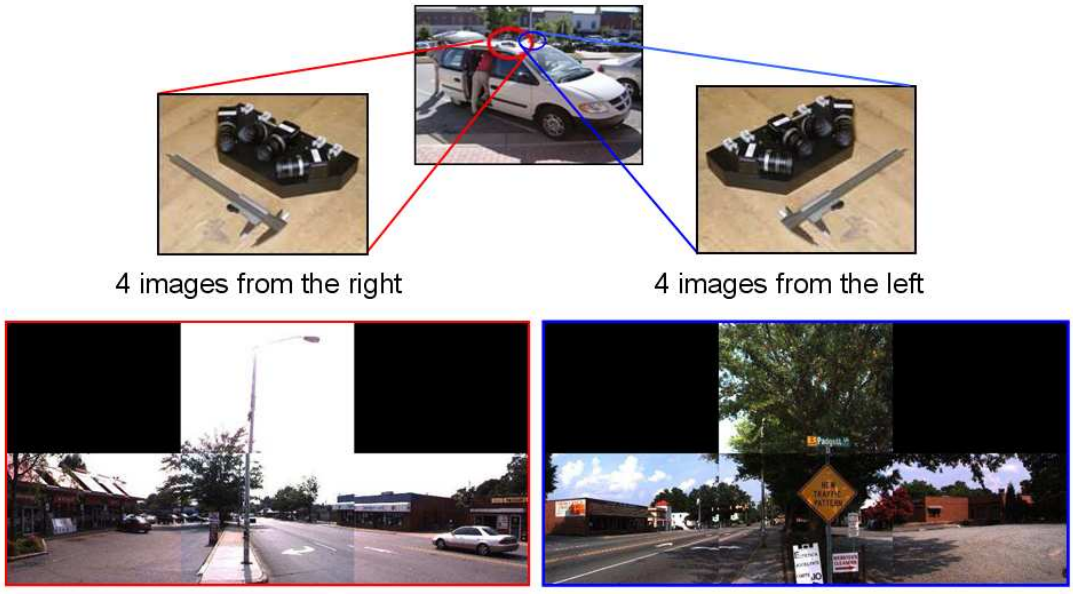


Figure 7.9: *An 8-camera system of non-overlapping multi-camera rigs on a vehicle and a sample of 8 images. (Images: Courtesy of UNC-Chapel Hill)*

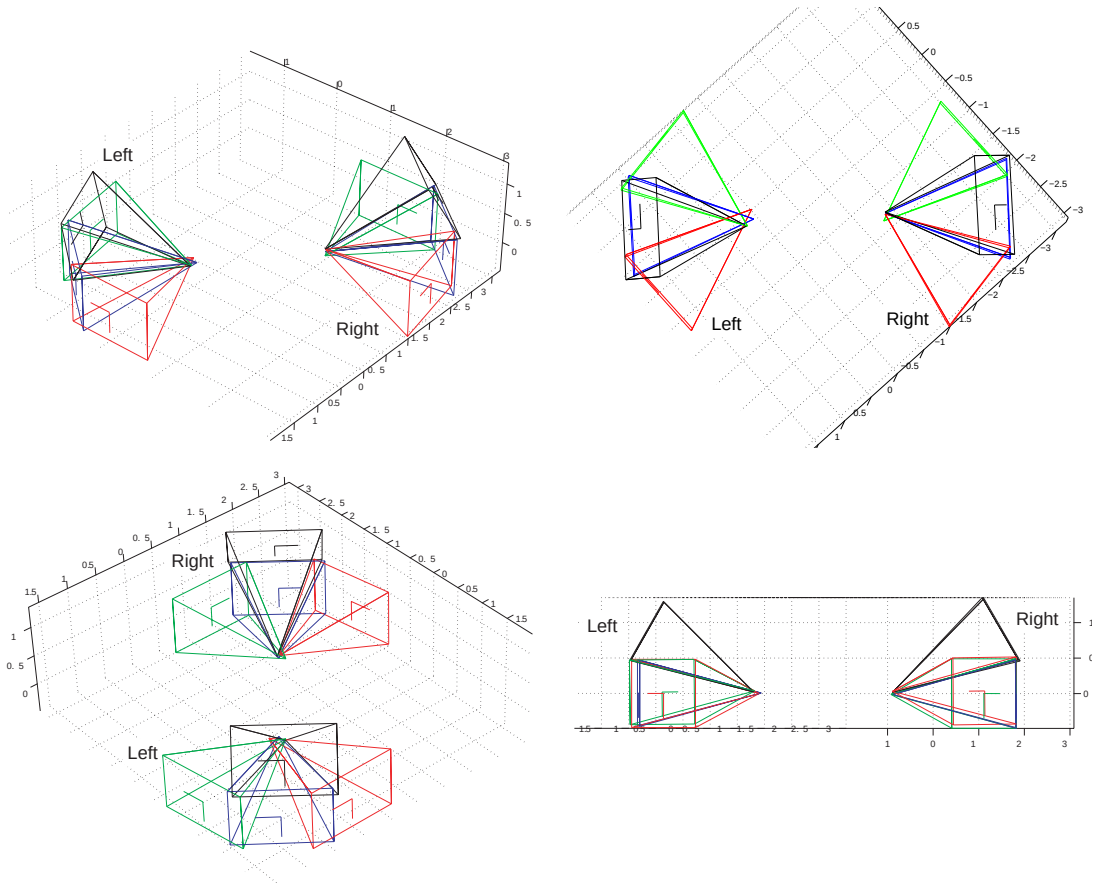


Figure 7.10: ***Position of 8 cameras in the system.*** *Red, green, blue and black colour indicate backward, side, forward and up direction of cameras, respectively. There are a little overlapping of field of view across cameras.*

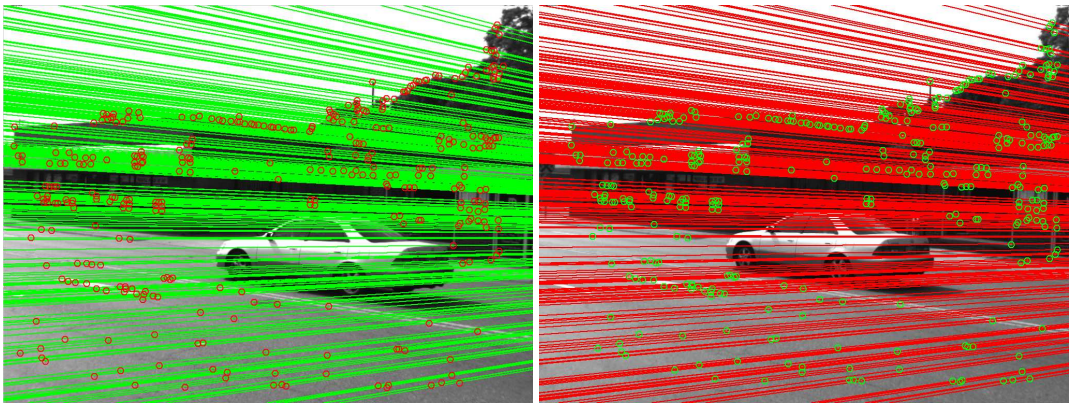
Figure 7.11: *Five points selected from the left-backward camera in two views, frame 120 and frame 125. Epipolar lines corresponding the five points are plotted. An essential matrix is estimated from the selected five points. The five points from the first view is indicated as red circles and the five epipolar lines corresponding to the five points are shown as red lines in the second image. In the same way, green circles for 5 points in the second view and green lines for the corresponding epipolar liners.*

First, features in image sequences are found and tracked across two views. We have used a commercial feature tracker, Boujou, to obtain robust feature tracks [1]. Then, an essential matrix from a camera on the left side of the vehicle (a backward camera is selected in this experiment) is estimated from five point correspondences using the five point minimal solution method [71]. The best five points are selected by the RANSAC algorithm and the estimated result is refined from inliers. In Figure 7.11, the five points and estimated epipolar lines are shown.

With the estimated essential matrix, the scale of translation direction is estimated from one point selected from the other camera on the right side of the vehicle. Like the five point method, the RANSAC approach is also used to find the best one point from the right side camera. For refinement of the scale estimation, all inliers on the right side camera are used to find a solution of the least-squares of liner equations, and non-linear optimization by minimizing the geometric reprojection errors is applied. In Figure 7.12(a) and Figure 7.12(b), the best one point and estimated epipolar lines from all inliers on the right side are shown, respectively.

For evaluation of the result, the ground truth for the position of the vehicle, in other words, the position of cameras, is provided from a global positioning system (GPS) and inertial mea-

(a)



(b)

Figure 7.12: *(a) One point selected from the right-backward camera in two views, frame 120 and frame 125 (indicated as red and green circles). This one point is used to estimate the scale of translation direction for multi-camera rigs. (b) All inliers used for the scale estimation and its epipolar lines. Note that there are no inliers found around a car in the image because the car in the image was moving and points on the car are identified as outliers. A total of 343 points out of 361 are found as inliers, and they contribute to find a solution of the scale by a refinement method. Red circles indicate the inliers in the first view and red lines show the epipolar lines corresponding to the red circles in the second view. Green circles indicate the inliers in the first view and green lines show the epipolar lines corresponding to the green circles in the second view.*

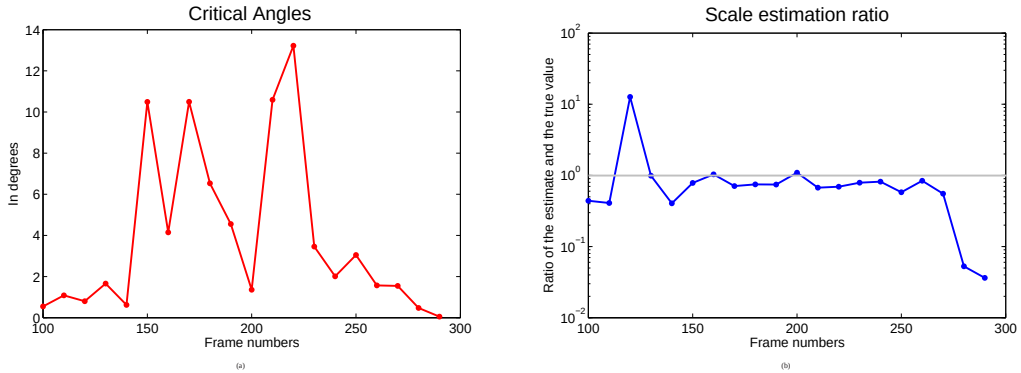


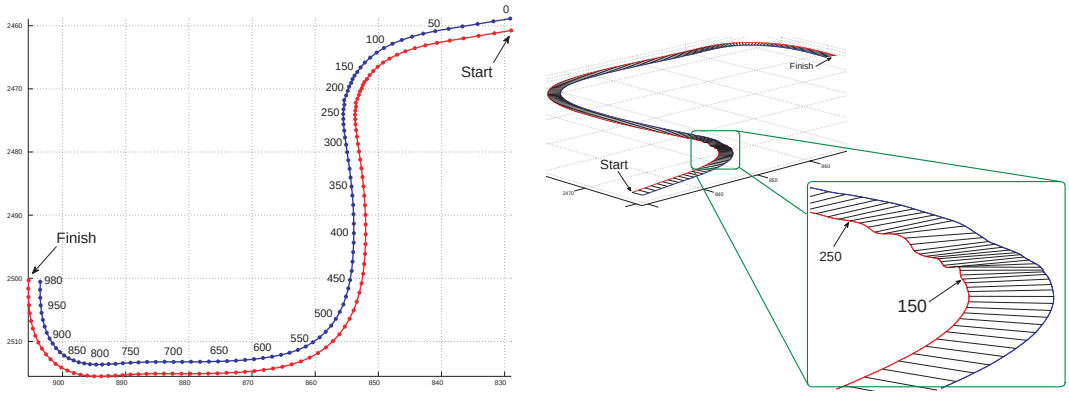
Figure 7.13: *(a) Critical angles in real experiments: From frame number 150 to 250, it is larger than 2 degrees of critical angles. (b) Scale estimation in real experiments: From frame number 150 to 250, the scale estimation shows values close to the ground truth.*

surement unit (IMU) device of POSLV, Applanix which is equipped in the vehicle system [2, 86].

From the geometric interpretation, we found that there is a critical configuration where our method cannot solve the scale of translation in non-overlapping multi-camera systems. Let us define critical angles as the angle between the translation vector of the first camera and the translation vector of the second camera. If the critical angle is equal to zero, it means that the motion of the multi-camera system is in a critical configuration. So, in this case, we cannot solve the scale of translation. Therefore, it is reasonable to examine how many times our 8-camera system on a vehicle has critical motions. In Figure 7.13(a), angles between the two translation vectors of two cameras are shown in each frame. From frame number 150 to 250, the angles are greater than about 2 degrees, and the rest of frames are less than 2 degrees. It means, unfortunately, most of motions of the vehicle are likely to be critical.

In Figure 7.14, the ground truth position of cameras is shown. The vehicles moved straight forward first, and then turned left and crossed over a speed bump. The speed bump mainly caused a large value of the critical angles and this motion corresponds to the frame numbers 150 to 250. Therefore, the scale of translation can be estimated correctly between these frame numbers.

In Figure 7.13(b), a ratio of scale estimation is shown. If the ratio is equal to one, then it tells us that the estimation of the scale for the translation is close to the correct solution.



(a) (b)

Figure 7.14: *(a) The ground truth positions of two cameras, left-backward (red) and right-backward side (blue), on the vehicle from frame number 0 to 980. (b) A zoomed part of the positions (from frame number 150 to 250) where the vehicle crosses over a speed bump: This part has enough large critical angles to estimate the motion of the vehicle.*

Otherwise, it fails to estimate the scale of translation. As shown in Figure 7.13(b), only frames between 150 and 250 give values close to one. These frame numbers have large critical angles.

In Figure 7.15, the rotation error and translation direction error in real experiments have been shown. The rotation part is usually estimated within less than about 0.1 degrees but the translation direction is estimated within about less than 2.0 degrees, which is mostly caused by the motion of the vehicle because the vehicle moves in a planar motion and the 8 cameras are mounted on the sides of the vehicle.

7.6 Conclusion

An algorithm solving the pose estimation for a multi-camera system having non-overlapping views is proposed.

Unlike Stewénius's method estimating the motion of generalized cameras, our proposed method does not need the 6-vector of rays represented by Plücker coordinates but use the 3-vector of points in homogeneous coordinates, and it needs five points from one camera and one point from another camera. In addition, our methods showed less residual error than Stewénius's method in the same experiment setup.

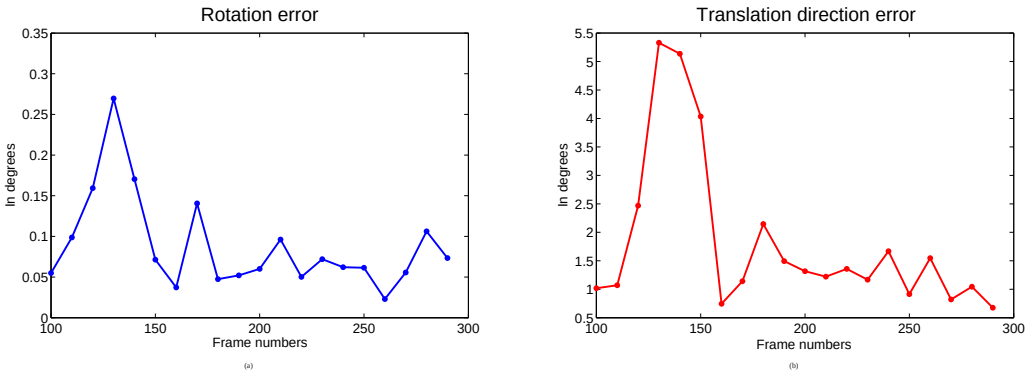


Figure 7.15: *(a) Rotation error in real experiments: From frame 150 to 250, the error is less than about 0.1 degrees; (b) Translation direction error in real experiments: From frame 150 to 250, the error is less than 2.0 degrees.*

A mathematical proof and geometric interpretation on the motion of non-overlapping multi-camera systems are presented. It is pointed out that there are critical motions, which prevent us from estimating the correct motion of multi-camera systems. For example, a parallel motion of the system is a degenerate case.

An algorithm solving the motion of non-overlapping multi-camera systems is introduced. The motion problem can be solved by estimating an essential matrix from five points in one image, decomposing the essential matrix to obtain rotation and translation direction of the multi-camera system, and then using one point from other image the scale of translation direction can be estimated. This straightforward method was very sensitive to the noise of point coordinates in images. Therefore, RANSAC approaches, kernel density approximation and non-linear refinement process have been applied to obtain robust estimation from an initial estimate.

From synthetic and real data experiments, it is observed that the most important part in the whole estimation is the estimation of an essential matrix. There are many algorithms to estimate essential matrices, however we used a minimal five point method by Stewénius's method because it provides minimum number of iterations for RANSAC algorithm. Mostly, in real experiments, the rotation part could be estimated very robustly less than about 0.1 degrees. However, the translation direction could be estimated within 2.0 degrees. This was a major

bottleneck to improve the result of scale estimation of the translation. The scale estimation could be achieved very robustly using RANSAC, kernel density approximation and non-linear optimization if the motion of multi-camera systems is not in critical motions. Unfortunately, most motion of vehicles are close to the case of critical motions such as moving forward and turning left or right on a flat ground unless they do a drift or cross over a speed bump.

With this approach, we could solve the 6 degrees of freedom motion of the multi-camera system with having non-overlapping views and it was also possible to solve 5 degrees of freedom when the system goes into the degenerate case.

For future research, it might be possible to improve the estimation result by using an geometrically optimal solution of essential matrix if we can achieve less than 0.1 degrees error of translation direction.

Chapter 8

A Linear Estimation of Relative Motion for Generalized Cameras

A generalized camera is a type of camera having no restrictions on mapping an incoming light ray to a photo-sensitive cell in image sensor arrays. Plücker coordinates were used by Pless to represent incoming light rays for generalized cameras in [60]. Pless formulated a generalized epipolar equation using the generalized essential matrix which is a 6×6 matrix describing the geometric relationships of the corresponding incoming light rays between two views.

The generalized essential matrix has 17 degrees of freedom. Therefore, given 17 points it is, in priciple, possible to construct a system of linear equations which estimate the generalized essential matrix. The linear system may be an overdetermined system of linear equations when more than 17 points are provided and the rank of the linear system is 17 or more. Hence, the linear system may be solvable by using the singular value decomposition (SVD). However, it applies only to a general configuration of generalized cameras. Unfortunately, in the most common type of multi-camera configurations, the rank of linear equations is less than 17. Consequently, the linear system cannot be solved linearly by using the SVD method.

Nevertheless, in this chapter, remarkably, we found that there is a linear approach to solve the generalized essential matrix in the case of common multi-camera configurations. The key idea is that a part of the solution is invariant when the generalized camera model is a locally-central, axial or locally-central-and-axial model. This constant part can be solved for linearly, so the rest of the solution can be obtained as well. Experiments on both synthetic and real data are conducted, and a reasonable accuracy is achieved by the proposed method.

8.1 Previous work

A general imaging model is introduced by Grossberg and Nayar in [18]. They described the general imaging model as a mapping of scene rays to pixels, and presented a concept of "raxels" which represent geometric, radiometric and optical properties. They also provided a calibration method for this general imaging model using structured light patterns. Pless used Plücker line coordinates to represent scene rays and derived an epipolar constraint equation for generalized cameras [60]. He predicted that 17 point correspondences are enough to solve a generalized essential matrix and solved the generalized essential matrix for non-central cameras using the Fisher Information Matrix. However, his method is not a linear approach to the problem. A hierarchy of generalized camera models and essential matrices for the different camera models are shown by Sturm in [77]. However, none of this research has shown any experiments with a linear method for estimating an essential matrix for generalized cameras. In this our research, we show and extensively verify a linear method for solving the relative motion problem for generalized cameras.

There exist many non-linear algorithms for solving for the generalized essential matrix for generalized cameras. Lhuillier used bundle adjustment for generalized cameras by using angular error instead of 3D errors [42]. Stewénius et al. used a Gröbner basis to solve for the generalized essential matrix and Byröd et al. improved the numerical accuracy of Stewénius's method [74, 7]. These methods based on a Gröbner basis approach solve polynomial equations to compute the generalized essential matrix, and apply to the minimal case only.

Mouragnon et al. [52] considered the rank of a matrix for the generalized epipolar equations when the generalized camera is a type of central camera, axial camera or non-axial camera. They confirmed that there are ambiguities in the solution for the generalized epipolar equations, and suggested a non-linear approach to address this problem. They carried out experiments with axial-type cameras only. They also introduced an incremental bundle adjustment method to refine their results. Schweighofer and Pinz gave an iterative algorithm for generalized camera to estimate its structure and motion by minimizing an object-space error which is the distance between a point in 3D and the projection of the point onto a scene

ray [67]. All these methods require a good initial estimate for their non-linear optimization process. However, none of the method actually used the linear 17-point algorithm for initialization.

There is some related work estimating relative motion for non-traditional cameras. Frahm et al. [14] proposed a pose estimation algorithm for multi-camera systems by putting a virtual camera into a multi-camera system. Alternate methods are discussed in chapters 9 and 10 of this thesis, and also in [40].

8.2 Generalized essential matrix for multi-camera systems

In this section, we reintroduce Pless's generalized essential matrix and the notation of Plücker coordinates. We also give a brief introduction to Stewénius's method to solve the relative motion of generalized cameras.

Let us consider a light ray in the world coordinate system. If the light ray is incident on a photosensitive sensor such as films and CCDs, the sensor is activated and records the intensity of the light ray. Therefore, irrespective of the manner in which the light rays travel through some materials such lenses and mirrors, an image is captured by the camera system when they arrive at the CCD array. So, the model of propagation of the incoming light rays is determined by the materials that are present in the region between the world and the photosensitive sensor. If the incoming light rays pass through lenses, meet at one common point and hit the photosensitive sensor, then this model of propagation of the light rays is called the "central projection camera model" because all the incoming light rays meet at a single centre of projection. If the incoming light rays are reflected by materials such as mirrors and hit the photosensitive cell, then this camera model might not have a single centre of projection. This model is called the "non-central projection camera model." Therefore, the model of propagation, i.e, the manner of mapping from the incoming light rays to the photosensitive cells determines the type of camera model. This is the "generalized camera model" of Grossberg and Nayar in [18]. An illustration of the generalized camera model is shown in Figure 8.1.

From the original definition of the generalized camera model in [18], "raxel" is defined as

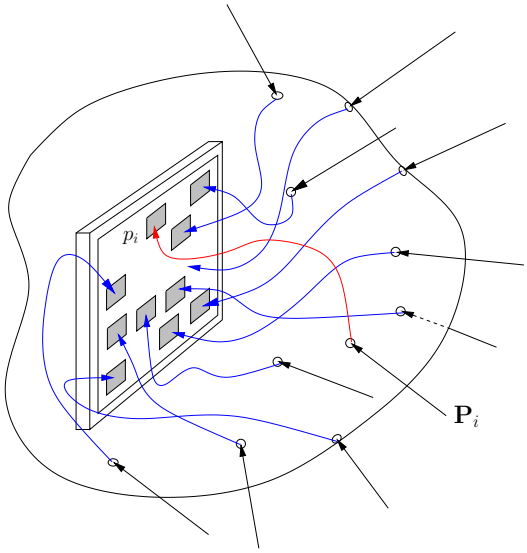


Figure 8.1: ***The generalized camera model.*** *An incoming light ray $\mathbf{P}_i$ in the generalized camera system is mapped to a pixel coordinate p_i. The incoming light rays are not restricted to a centre of projection and therefore they could have multiple centres of projection.*

an element of a light ray having geometric, optical and radiometric properties. However, in this thesis, we use a simplified representation of the generalized camera model as used by Pless in [60].

8.2.1 Plücker coordinates

Pless used Plücker vectors to describe a light ray in the world for generalized camera model in [60]. The Plücker vectors represent a line by a 6-vector that is a pair of 3-vectors, $\mathbf{q}$ and $\mathbf{q}'$, which are called the direction vector and moment vector, respectively. The direction vector $\mathbf{q}$ is a vector with the direction of the line. The moment vector $\mathbf{q}' = P \times \mathbf{q}$ has a direction that is perpendicular to the plane containing the line and the origin, and whose magnitude is equal to the area of the triangle that is defined by the direction vector and the origin. It is shown in Figure 8.2.

A property of the Plücker line coordinates is that $\mathbf{q}$ and $\mathbf{q}'$ are perpendicular to one another. Therefore, the inner product of them is equal to zero as $\mathbf{q}^\top \mathbf{q}' = 0$. The Plücker coordinates are homogeneous and therefore multiplying all the six coordinates by any real number gives new

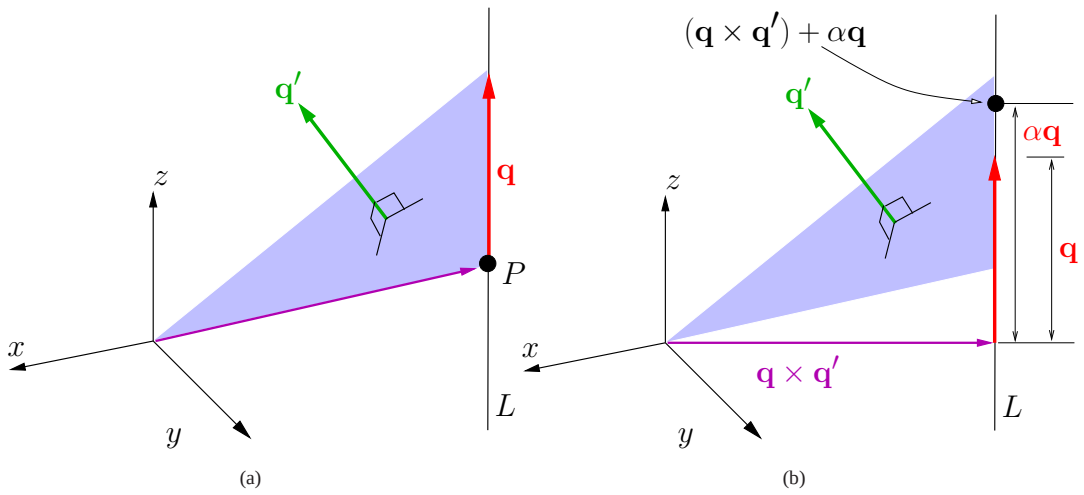


Figure 8.2: ***Plücker coordinates.*** *(a) A line L in 3-D space and two vectors $\mathbf{q}$ and $\mathbf{q}'$ for the Plücker line coordinates of L. The vector $\mathbf{q}$ represents the direction of the line and the vector $\mathbf{q}'$ is the moment vector which is depicted by a shaded triangle and a normal vector to the triangle. (b) All points on the line L are expressed by the two vectors $\mathbf{q} \times \mathbf{q}'$ and $\alpha\mathbf{q}$, where α is a scalar.*

Plücker coordinates for the same line. More details on the Plücker coordinates can be found in [75]. Assuming that the direction vector $\mathbf{q}$ is a unit vector, the set of all points that lie on the line L is written as follows:

$$(\mathbf{q} \times \mathbf{q}') + \alpha\mathbf{q} \,, \tag{8.1}$$

where α is a scale.

8.2.2 Pless equation

Let us derive the generalized epipolar constraint that we call the "Pless equation" or "Generalized Essential Matrix Constraint" for the generalized camera model as shown by Pless in [60].

Suppose there is a generalized camera model in the world coordinate. The incoming light rays are mapped to a pixel coordinate in the generalized camera model. If the generalized camera is subjected to a rigid motion, the rigid transformation is applied to the incoming light rays. Let $\mathtt{R}$ and $\mathbf{t}$ be the rotation and translation of the rigid transformation between two views in the generalized camera model. A light ray presented by the Plücker coordinates, 6-vector $\mathbf{L}$,

is written as follows:

$$\mathbf{L} = \begin{pmatrix} \mathbf{q} \\ \mathbf{q}' \end{pmatrix}. \tag{8.2}$$

If the light ray is transformed by a rigid motion, the line $\mathbf{L}$ in the Plücker coordinates after the rigid transformation becomes as follows:

$$\mathbf{L} \rightarrow \begin{pmatrix} \mathtt{R}\mathbf{q} \\ (\mathtt{R}P + \mathbf{t}) \times (\mathtt{R}\mathbf{q}) \end{pmatrix} = \begin{pmatrix} \mathtt{R}\mathbf{q} \\ \mathtt{R}\mathbf{q}' + \mathbf{t} \times (\mathtt{R}\mathbf{q}) \end{pmatrix}. \tag{8.3}$$

Considering a pair of matching light rays, $\mathbf{L} \leftrightarrow \mathbf{L}'$, where $\mathbf{L} = (\mathbf{q}_1^\top,\ \mathbf{q'}_1^\top)^\top$ and $\mathbf{L}' = (\mathbf{q}_2^\top,\ \mathbf{q'}_2^\top)^\top$. These two light rays intersect if and only if

$$\mathbf{q}_2^\top \mathbf{q'}_1 + \mathbf{q'}_2^\top \mathbf{q}_1 = 0\,. \tag{8.4}$$

From (8.3), $\mathbf{q}_1$ and $\mathbf{q}'_1$ become $\mathtt{R}\mathbf{q}_1$ and $\mathtt{R}\mathbf{q'}_1 + \mathbf{t} \times (\mathtt{R}\mathbf{q}_1)$, respectively. Therefore, in [60], Pless showed that (8.4) may be written as follows:

$$\mathbf{q}_2^\top (\mathtt{R}\mathbf{q}'_1 + \mathbf{t} \times (\mathtt{R}\mathbf{q}_1)) + \mathbf{q}_2'^\top (\mathtt{R}\mathbf{q}_1) = 0 \tag{8.5}$$

$$\mathbf{q}_2^\top \mathtt{R}\mathbf{q}'_1 + \mathbf{q}_2^\top [\mathbf{t}]_\times \mathtt{R}\mathbf{q}_1 + \mathbf{q}_2'^\top \mathtt{R}\mathbf{q}_1 = 0\,. \tag{8.6}$$

Let $\mathbf{L}_1 = (\mathbf{q}_1^\top, \mathbf{q}_1'^\top)^\top$ and $\mathbf{L}_2 = (\mathbf{q}_2^\top, \mathbf{q}_2'^\top)^\top$ be two Plücker lines. Equation (8.6) may be written with a 9×9 matrix $\mathtt{G}$ as follows

$$\mathbf{L}_2{}^\top \mathtt{G} \mathbf{L}_1 = \begin{pmatrix} \mathbf{q}_2 \\ \mathbf{q}'_2 \end{pmatrix}^\top \begin{bmatrix} [\mathbf{t}]_\times \mathtt{R} & \mathtt{R} \\ \mathtt{R} & 0 \end{bmatrix} \begin{pmatrix} \mathbf{q}_1 \\ \mathbf{q}'_1 \end{pmatrix} = 0\,. \tag{8.7}$$

Therefore, given the ray correspondence $\mathbf{L} \leftrightarrow \mathbf{L}'$, the generalized essential matrix $\mathtt{G}$ is written as follows:

$$\mathtt{G} = \begin{bmatrix} [\mathbf{t}]_\times \mathtt{R} & \mathtt{R} \\ \mathtt{R} & 0 \end{bmatrix}, \tag{8.8}$$

where $\mathtt{R}$ and $\mathbf{t}$ are the rotation and translation, respectively, of a rigid transformation between two views.

It is important to note that if the last three elements of the Plücker line are zero, i.e. if $\mathbf{q}_1' = \mathbf{0}$ and $\mathbf{q}_2' = \mathbf{0}$, then the form of the generalized essential matrix $\mathtt{G}$ is the same as the standard form of essential matrix $\mathtt{E}$ in (2.18). Owing to the use of the Plücker lines, the generalized essential matrix can represent relationships for the pair of matching light rays in multi-camera systems.

The two light rays $\mathbf{L}_1$ and $\mathbf{L}_2$ should intersect at one point in the world coordinate system. The point can be determined by finding the point of the intersection of the two rays. When $\mathtt{R}$ and $\mathbf{t}$ are known, from (8.1), the two light rays satisfy the following equality:

$$\mathtt{R}((\mathbf{q}_1 \times \mathbf{q}_1') + \alpha_1 \mathbf{q}_1) + \mathbf{t} = (\mathbf{q}_2 \times \mathbf{q}_2') + \alpha_2 \mathbf{q}_2 \ , \tag{8.9}$$

where α_1 and α_2 are scalars.

The reconstruction of the 3D point $\mathbf{X}$ is given by Pless in [60] as follows:

$$\mathbf{X} = (\mathbf{q}_1 \times \mathbf{q}_1') + \alpha_1 \mathbf{q}_1 \ , \tag{8.10}$$

where α_1 can be solved from the equation $\alpha_1 \mathtt{R}\mathbf{q}_1 - \alpha_2 \mathbf{q}_2 = (\mathbf{q}_2 \times \mathbf{q}_2') - \mathtt{R}(\mathbf{q}_1 \times \mathbf{q}_1') - \mathbf{t}$, which is derived from (8.9).

For continuous motion, Pless also derived the differential generalized epipolar constraint similar to the generalized epipolar constraint for discrete motion. He used the Fisher information matrix to solve the continuous motion equation in [60].

8.2.3 Stewénius's method

In [72], Stewénius obtained multiple solutions for the relative motion in multi-camera systems. Stewénius also used a generalized camera model to describe multi-camera systems, but he derived a more general form of Pless equation by allowing the first camera in a more general configuration.

Let $\mathtt{R}_1$ and $\mathtt{R}_2$ be the rotation for the first and second views, respectively, with respect to the world coordinate system in the generalized camera model. Similarly, let $\mathbf{t}_1$ and $\mathbf{t}_2$ be the translation for views 1 and 2 with respect to the world coordinate system. Then, the corresponding two light rays $\mathbf{L}_1$ and $\mathbf{L}_2$ may be transformed and expressed in the world coordinate as follows:

$$\hat{\mathbf{L}_1} = \begin{pmatrix} \mathtt{R}_1\mathbf{q}_1 \\ \mathtt{R}_1\mathbf{q}_1' + \mathbf{t}_1 \times (\mathtt{R}_1\mathbf{q}_1) \end{pmatrix} \tag{8.11}$$

$$\hat{\mathbf{L}_2} = \begin{pmatrix} \mathtt{R}_2\mathbf{q}_2 \\ \mathtt{R}_2\mathbf{q}_2' + \mathbf{t}_2 \times (\mathtt{R}_2\mathbf{q}_2) \end{pmatrix} . \tag{8.12}$$

From (8.4) the epipolar plane constraint gives us the standard form of the generalized essential matrix equation, and it may be written as

$$\mathbf{q}_2^\top \mathtt{R}_2^\top \mathtt{R}_1 \mathbf{q}_1' + \mathbf{q'}_2^\top \mathtt{R}_2{}^\top \mathtt{R}_1 \mathbf{q}_1 + \mathbf{q}_2^\top \mathtt{R}_2^\top [\mathbf{t}_1 - \mathbf{t}_2]_\times \mathtt{R}_1 \mathbf{q}_1 = 0 \,. \tag{8.13}$$

By choosing 6 rays in two cameras, Stewénius et al. showed a method to solve for the relative motion between two views of the generalized camera. In [72], a Gröbner basis is used to solve for the relative motion. Their method showed that there are 64 solutions to the problem, and they solved the problem by using the Gröbner basis.

8.3 Four types of generalized cameras

A generalized camera is a model for an imaging situation in which pixels in the image correspond to specified rays (straight lines) in space, but with no other limitation on how incoming light rays project onto an image. The image value at a pixel records the response (for instance colour) of some point along its associated ray. There can be multiple centres of projection, or indeed no centres of projection at all. This camera model is relatively general, and includes cameras such as perspective cameras, fish-eye cameras, central or non-central catadioptric cameras, linear or non-linear pushbroom cameras ([19]), whiskbroom cameras, panoramic

cameras ([20]) as well as multi-camera rigs and insect eyes. It is worth noting, however, that it does not cover certain important classes of cameras, such as synthetic aperture radar (SAR) images, and the rational cubic camera model ([26]) used in many surveillance images, or perhaps X-ray images.

Suppose that two images are taken by a generalized camera from two different positions and let 3D points $\mathbf{X}_i$ be projected in two images. Let $\mathbf{r}_{ij}$ be incoming light rays as a line-segment connecting from $\mathbf{X}_i$ to the centre $\mathbf{c}_j$ in the first view, and let $\mathbf{r}'_{ij}$ be incoming light rays from $\mathbf{X}_i$ to the centre $\mathbf{c}'_j$ in the second view. Then, let us consider the order of incoming light rays. If the position of all centres in a system is preserved, then all incoming light rays $\mathbf{r}_{ij}$ and $\mathbf{r}'_{ij}$ for two views have the same order. However, if the position of all centres in a system is not preserved, for example, if they have different position of centres, then all incoming light rays $\mathbf{r}_{ij}$ and $\mathbf{r}'_{ij}$ have different order of projection. This order of point correspondences is preserved in central projection cameras across views. However, in generalized camera models, an order of point correspondences can be different between two views. They are illustrated in Figure 8.3(a) and Figure 8.3(b).

In addition, it could have no centre of projections or multiple centre of projections. Specifically, projections of all image rays can lie in a single axis. Moreover, the order of light rays can be considered or not. In this section, we call these four types of generalized cameras as "the most-general case," "the locally-central case," "the axial case," and "the locally-central-and-axial case" as shown in Figure 8.3.

Let a light ray be described by a point $\mathbf{v}$ with the unit direction $\mathbf{x}$. The generalized epipolar equation with a corresponding light ray represented by Plücker vectors $\mathbf{L} = (\mathbf{x}^\top,\ (\mathbf{v} \times \mathbf{x})^\top)^\top$ and $\mathbf{L}' = (\mathbf{x}'^\top,\ (\mathbf{v}' \times \mathbf{x}')^\top)^\top$ may be written as follows

$$\mathbf{L}'^\top \mathtt{G} \mathbf{L} = \begin{pmatrix} \mathbf{x}' \\ \mathbf{v}' \times \mathbf{x}' \end{pmatrix}^\top \begin{bmatrix} \mathtt{E} & \mathtt{R} \\ \mathtt{R} & 0 \end{bmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{v} \times \mathbf{x} \end{pmatrix}^\top \tag{8.14}$$

and it can be rewritten as

$$\mathbf{x}'^{\top}\mathtt{E}\mathbf{x} + (\mathbf{v}' \times \mathbf{x}')^{\top}\mathtt{R}\mathbf{x} + \mathbf{x}'^{\top}\mathtt{R}(\mathbf{v} \times \mathbf{x}) = 0\,, \tag{8.15}$$

where $\mathtt{E}$ is a 3×3 matrix decomposed as $\mathtt{E} = [\mathbf{t}]_{\times}\mathtt{R}$, where $\mathtt{R}$ is a rotation matrix and $\mathbf{t}$ is a translation vector. This equation (8.15) may construct a system of linear equations as a form of $\mathtt{A}_i^{\top}\mathbf{y} = 0$ as follows:

$$\mathtt{A}_i^{\top}\mathbf{y} = \begin{bmatrix} \mathbf{x}'_1\mathbf{x}_1 \\ \mathbf{x}'_1\mathbf{x}_2 \\ \mathbf{x}'_1\mathbf{x}_3 \\ \mathbf{x}'_2\mathbf{x}_1 \\ \mathbf{x}'_2\mathbf{x}_2 \\ \mathbf{x}'_2\mathbf{x}_3 \\ \mathbf{x}'_3\mathbf{x}_1 \\ \mathbf{x}'_3\mathbf{x}_2 \\ \mathbf{x}'_3\mathbf{x}_3 \\ (\mathbf{v}'_2\mathbf{x}'_3 - \mathbf{v}'_3\mathbf{x}'_2)\mathbf{x}_1 + \mathbf{x}'_1(\mathbf{v}_2\mathbf{x}_3 - \mathbf{v}_3\mathbf{x}_2) \\ (\mathbf{v}'_2\mathbf{x}'_3 - \mathbf{v}'_3\mathbf{x}'_2)\mathbf{x}_2 + \mathbf{x}'_1(\mathbf{v}_3\mathbf{x}_1 - \mathbf{v}_1\mathbf{x}_3) \\ (\mathbf{v}'_2\mathbf{x}'_3 - \mathbf{v}'_3\mathbf{x}'_2)\mathbf{x}_3 + \mathbf{x}'_1(\mathbf{v}_1\mathbf{x}_2 - \mathbf{v}_2\mathbf{x}_1) \\ (\mathbf{v}'_3\mathbf{x}'_1 - \mathbf{v}'_1\mathbf{x}'_3)\mathbf{x}_1 + \mathbf{x}'_2(\mathbf{v}_2\mathbf{x}_3 - \mathbf{v}_3\mathbf{x}_2) \\ (\mathbf{v}'_3\mathbf{x}'_1 - \mathbf{v}'_1\mathbf{x}'_3)\mathbf{x}_2 + \mathbf{x}'_2(\mathbf{v}_3\mathbf{x}_1 - \mathbf{v}_1\mathbf{x}_3) \\ (\mathbf{v}'_3\mathbf{x}'_1 - \mathbf{v}'_1\mathbf{x}'_3)\mathbf{x}_3 + \mathbf{x}'_2(\mathbf{v}_1\mathbf{x}_2 - \mathbf{v}_2\mathbf{x}_1) \\ (\mathbf{v}'_1\mathbf{x}'_2 - \mathbf{v}'_2\mathbf{x}'_1)\mathbf{x}_1 + \mathbf{x}'_3(\mathbf{v}_2\mathbf{x}_3 - \mathbf{v}_3\mathbf{x}_2) \\ (\mathbf{v}'_1\mathbf{x}'_2 - \mathbf{v}'_2\mathbf{x}'_1)\mathbf{x}_2 + \mathbf{x}'_3(\mathbf{v}_3\mathbf{x}_1 - \mathbf{v}_1\mathbf{x}_3) \\ (\mathbf{v}'_1\mathbf{x}'_2 - \mathbf{v}'_2\mathbf{x}'_1)\mathbf{x}_3 + \mathbf{x}'_3(\mathbf{v}_1\mathbf{x}_2 - \mathbf{v}_2\mathbf{x}_1) \end{bmatrix}^{\top} \begin{bmatrix} \mathtt{E}_{11} \\ \mathtt{E}_{12} \\ \mathtt{E}_{13} \\ \mathtt{E}_{21} \\ \mathtt{E}_{22} \\ \mathtt{E}_{23} \\ \mathtt{E}_{31} \\ \mathtt{E}_{32} \\ \mathtt{E}_{33} \\ \mathtt{R}_{11} \\ \mathtt{R}_{12} \\ \mathtt{R}_{13} \\ \mathtt{R}_{21} \\ \mathtt{R}_{22} \\ \mathtt{R}_{23} \\ \mathtt{R}_{31} \\ \mathtt{R}_{32} \\ \mathtt{R}_{33} \end{bmatrix} = 0 \tag{8.16}$$

where $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)^{\top}$, $\mathbf{x}' = (\mathbf{x}'_1, \mathbf{x}'_2, \mathbf{x}'_3)^{\top}$, $\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3)^{\top}$, $\mathbf{v}' = (\mathbf{v}'_1, \mathbf{v}'_2, \mathbf{v}'_3)^{\top}$, and $\mathtt{E}_{ij}$ and $\mathtt{R}_{ij}$ are the (i, j)-th element of the matrix $\mathtt{E}$ and $\mathtt{R}$. By putting all 17 rays together, $\mathtt{A}_i^{\top}$

may construct a matrix $\mathtt{A}$ such as

$$\mathtt{A}\mathbf{y} = \begin{bmatrix} \mathtt{A}_1^\top \\ \mathtt{A}_2^\top \\ \vdots \\ \mathtt{A}_{17}^\top \end{bmatrix} (\mathtt{E}_{11},\ \mathtt{E}_{12},\ \cdots,\ \mathtt{E}_{33},\ \mathtt{R}_{11},\ \mathtt{R}_{12},\ \cdots,\ \mathtt{R}_{33})^\top \tag{8.17}$$

$$= \mathtt{A}(\mathrm{vec}(\mathtt{E})^\top,\ \mathrm{vec}(\mathtt{R})^\top)^\top = 0 \tag{8.18}$$

where $\mathrm{vec}(\mathtt{E})$ and $\mathrm{vec}(\mathtt{R})$ are 9-vectors whose elements are taken in column-major order from $\mathtt{E}$ and $\mathtt{R}$, respectively. For example, given a matrix $\mathtt{M} = [\mathbf{m}_1,\ \mathbf{m}_2,\ \mathbf{m}_3]$, where $\mathrm{vec}(\mathtt{M}) = (\mathbf{m}_1^\top,\ \mathbf{m}_2^\top,\ \mathbf{m}_3^\top)^\top$.

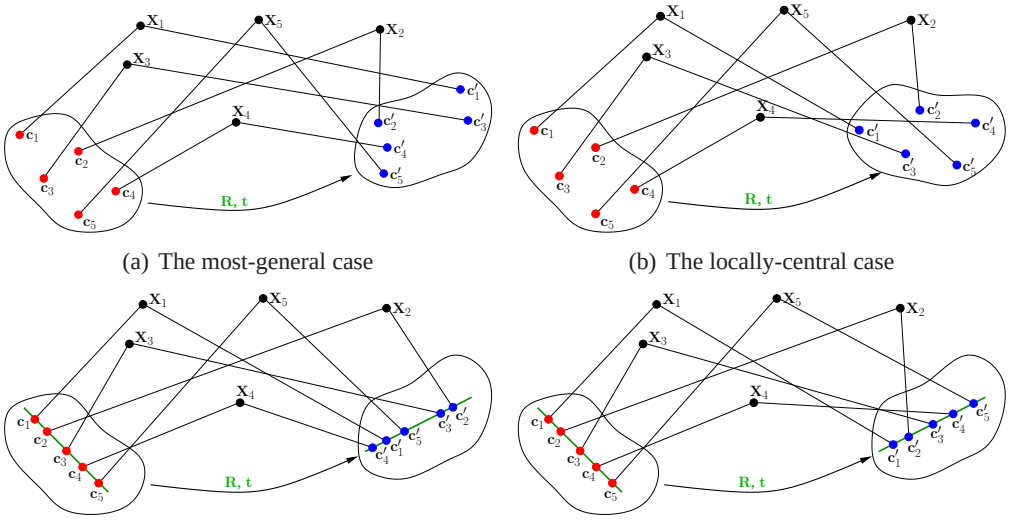
8.3.1 The most-general case

In the most-general case, the matrix $\mathtt{A}$ in (8.18) may have rank 17 given 17 unconstrained rays. In (8.18), the vector $\mathbf{X}$ contains the entries of two matrices $\mathtt{E}$ and $\mathtt{R}$, of the essential matrix and rotation matrix. This equation can be solved by using SVD. However, the matrix $\mathtt{A}$ for the two cases, the locally-central and axial case, as shown in Figure 8.3(b) and Figure 8.3(c), does not have a sufficient rank to solve the equation directly using the SVD method. In this thesis, this specific two cases are discussed and linear algorithms solving the problems for these cases are presented.

8.3.2 The locally-central case

As shown in Figure 8.3(b), the order of centres in a generalized camera is preserved throughout other views. A real camera setup for this locally-central case is possible such as using non-overlapping multi-camera systems consisting of multiple cameras physically connected to each other but possibly sharing little field of view.

Suppose that incoming rays are expressed in each camera's coordinate system and the camera is aligned with its own coordinate system. Then, a correspondence of rays, $\mathbf{L}^\top = (\mathbf{x}^\top,\ (\mathbf{v}\times\mathbf{x})^\top)^\top$ and $\mathbf{L'}^\top = (\mathbf{x'}^\top,\ (\mathbf{v'}\times\mathbf{x'})^\top)^\top$, will have the same centre $\mathbf{v} = \mathbf{v'}$. Therefore,



(a) The most-general case (b) The locally-central case

(c) The axial case (d) The locally-central-and-axial case

Figure 8.3: *(a) The most-general case; All incoming light rays project to the first camera and the order of corresponding rays in the second camera is different from the order of the first camera. (b) The locally-central case; The order of incoming light rays is consistent in their correspondence between two generalized cameras. However, there is no single common centre of projections. (c) The axial case; The order of incoming light rays in correspondence is not preserved and all light rays meet on an axis in each camera. (d) The locally-central-and-axial case; The order of incoming light rays in correspondence is preserved and all light rays meet on an axis in each camera. The ranks of generalized epipolar equations in each case are 17, 16, 16 and 14 for (a)-(d), respectively.*

the equation (8.15) becomes

$$\mathbf{x'}^{\top}\mathtt{E}\mathbf{x} + (\mathbf{v} \times \mathbf{x'})^{\top}\mathtt{R}\mathbf{x} + \mathbf{x'}^{\top}\mathtt{R}(\mathbf{v} \times \mathbf{x}) = 0 \,. \tag{8.19}$$

Given N rays, the size of the matrix $\mathtt{A}$ is N. Therefore, 17 points are enough to solve the equation because the vector $\mathbf{y}$ is represented in homogenerous coordinates. Unfortunately, in the locally-central case, the rank of the matrix $\mathtt{A}$ is not 17. Let us see one possible solution solution of $(\mathtt{E},\ \mathtt{R})$ is $(0,\ \mathtt{I})$ in (8.19). This solution makes the equation become zero as follows:

$$\begin{aligned}
&\mathbf{x'}^{\top}\mathtt{E}\mathbf{x} + (\mathbf{v} \times \mathbf{x'})^{\top}\mathtt{R}\mathbf{x} + \mathbf{x'}^{\top}\mathtt{R}(\mathbf{v} \times \mathbf{x}) && (8.20)\\
&= (\mathbf{v} \times \mathbf{x'})^{\top}\mathbf{x} + \mathbf{x'}^{\top}(\mathbf{v} \times \mathbf{x}) && (8.21)\\
&= \mathbf{x}^{\top}(\mathbf{v} \times \mathbf{x'}) + \mathbf{x}^{\top}(\mathbf{x'} \times \mathbf{v}) && (8.22)\\
&= \mathbf{x}^{\top}(\mathbf{v} \times \mathbf{x'}) - \mathbf{x}^{\top}(\mathbf{v} \times \mathbf{x'}) = 0 \,. && (8.23)
\end{aligned}$$

Assuming that the matrix $\mathtt{E}$ is not zero, the matrix $\mathtt{A}$ in (8.18) has rank 16 at least. Therefore, the solution has a two-dimensional linear family such as $(\lambda\mathtt{E},\ \lambda\mathtt{R} + \mu\mathtt{I})$ where λ and μ are scalar. In other words, the two-dimensional linear family gives us the rank 16 = 18 - 2. It is important to note that the $\mathtt{R}$ part of the solution may vary and the essential matrix $\mathtt{E}$ part of the solution is not changed. Therefore, the $\mathtt{E}$ part can be uniquely determined.

8.3.3 The axial case

In the axial case, there is a virtual single line in a generalized camera. All incoming light rays in the generalized camera intersect with the single line. This single line forms as an axis of all incoming light rays. In this special configuration of incoming light rays, the generalized epipolar equation may not have rank 17 to be solvable using the standard SVD method. However, a possible set of solutions can be found by analyzing the equation for this axial case. In this axial case, the order of projection centres of the incoming rays is not considered. When the order is preserved, another configuration, we call "the locally-central-and-axial case," can be considered.

To make the equation simpler, let us assume that the axis passes through the origin of the world coordinate system. Then, suppose that the axis is represented as $\mathbf{w}$, which is the direction vector. It means that all points in the axis will be expressed as a scalar value times the direction vector $\mathbf{w}$ such as $\mathbf{v} = \alpha\mathbf{w}$ and $\mathbf{v}' = \alpha'\mathbf{w}$. As seen before, the points $\mathbf{v}$ and $\mathbf{v}'$ are the points on a ray $\mathbf{L}$ and $\mathbf{L}'$, respectively. Therefore, the generalized epipolar equation (8.15) becomes as follows:

$$\mathbf{x}'^{\top}\mathtt{E}\mathbf{x} + \alpha'(\mathbf{w} \times \mathbf{x}')^{\top}\mathtt{R}\mathbf{x} + \alpha\mathbf{x}'^{\top}\mathtt{R}(\mathbf{w} \times \mathbf{x}) = 0\,. \tag{8.24}$$

Let $(\mathtt{E},\ \mathtt{R})$ be solutions for the equation, then other possible solution is $(\mathtt{0}, \mathbf{w}\mathbf{w}^{\top})$. It is verified as follows:

$$\begin{aligned}
&\mathbf{x}'^{\top}\mathtt{E}\mathbf{x} + \alpha'(\mathbf{w} \times \mathbf{x}')^{\top}\mathtt{R}\mathbf{x} + \alpha\mathbf{x}'^{\top}\mathtt{R}(\mathbf{w} \times \mathbf{x}) && (8.25)\\
&\quad = \alpha'(\mathbf{w} \times \mathbf{x}')^{\top}\mathbf{w}\mathbf{w}^{\top}\mathbf{x} + \alpha\mathbf{x}'^{\top}\mathbf{w}\mathbf{w}^{\top}(\mathbf{w} \times \mathbf{x}) && (8.26)\\
&\quad = \alpha'\mathbf{x}'^{\top}(\mathbf{w} \times \mathbf{w})\mathbf{w}\mathbf{w}^{\top}\mathbf{x} + \alpha\mathbf{x}'^{\top}\mathbf{w}\mathbf{x}^{\top}(\mathbf{w} \times \mathbf{w}) = 0\,. && (8.27)
\end{aligned}$$

So, for the axial case of generalized cameras, we have solutions $(\lambda\mathtt{E},\ \lambda\mathtt{R} + \mu\mathbf{w}\mathbf{w}^{\top})$, a two-dimensional linear family. Therefore, the rank of the matrix $\mathtt{A}$ for this axial case is 16 = 18 - 2. In particular, note that the $\mathtt{E}$ part of the solution is constant and the $\mathtt{R}$ part is involved with ambiguity on solutions.

8.3.4 The locally-central-and-axial case

This "locally-central-and-axial case" of generalized cameras is a special case of "the axial case" with preserving the order of incoming light rays. As the locally-central case has a solution of $(\mathtt{0},\ \mathtt{R})$, the locally-central-and-axial case also has the same solution of $(\mathtt{0},\ \mathtt{R})$. In addition, the locally-central-and-axial case has another possible solution because of its property of the axial case.

In the equation (8.24) for the axial case, α' may be substituted by α because the order of

the incoming light rays is consistent. Therefore, the equation becomes

$$\mathbf{x'}^\top \mathtt{E}\mathbf{x} + \alpha(\mathbf{w} \times \mathbf{x'})^\top \mathtt{R}\mathbf{x} + \alpha \mathbf{x'}^\top \mathtt{R}(\mathbf{w} \times \mathbf{x}) = 0 \,. \tag{8.28}$$

In this case, a possible solution is $(\mathtt{0},\ [\mathbf{w}]_\times)$. It can be proved as follows: [1]

$$\begin{aligned}
&\mathbf{x'}^\top \mathtt{E}\mathbf{x} + \alpha(\mathbf{w} \times \mathbf{x'})^\top \mathtt{R}\mathbf{x} + \alpha \mathbf{x'}^\top \mathtt{R}(\mathbf{w} \times \mathbf{x}) && (8.29)\\
&\quad = \alpha(\mathbf{w} \times \mathbf{x'})^\top [\mathbf{w}]_\times \mathbf{x} + \alpha \mathbf{x'}^\top [\mathbf{w}]_\times (\mathbf{w} \times \mathbf{x}) && (8.30)\\
&\quad = \alpha(\mathbf{w} \times \mathbf{x'})^\top (\mathbf{w} \times \mathbf{x}) + \alpha(\mathbf{x'} \times \mathbf{w})^\top (\mathbf{w} \times \mathbf{x}) && (8.31)\\
&\quad = \alpha(\mathbf{w} \times \mathbf{x'})^\top (\mathbf{w} \times \mathbf{x}) - \alpha(\mathbf{x'} \times \mathbf{w})^\top (\mathbf{w} \times \mathbf{x}) = 0 \,. && (8.32)
\end{aligned}$$

Therefore, the set of solutions for the locally-central-and-axial case is written in a four-dimensional family as follows:

$$(\alpha\mathtt{E},\ \alpha\mathtt{R} + \beta\mathtt{I} + \gamma[\mathbf{w}]_\times + \delta\mathbf{w}\mathbf{w}^\top) \,, \tag{8.33}$$

where α, β, γ and δ are scalars. Therefore, the rank of the matrix $\mathtt{A}$ in this locally-central-and-axis case is 14 = 18 - 4. It is significant to note that the matrix $\mathtt{E}$ part is not changed and can be uniquely determined up to scale.

8.4 Algorithms

8.4.1 Linear algorithm for generalized cameras

As seen so far, the generalized epipolar equations for the central case, the axial case and the locally-central-and-axial case have solutions in which only the $\mathtt{E}$ part is unchanged by the ambiguity. Therefore, we do not need to solve for the $\mathtt{R}$ part.

From this observation, solving the equation (8.18) may be rewritten as a problem of finding a solution minimizing

$$||\mathtt{A}(\mathrm{vec}(\mathtt{E})^\top,\ \mathrm{vec}(\mathtt{R})^\top)^\top|| \ \text{ subject to } \ ||(\mathrm{vec}(\mathtt{E})^\top,\ \mathrm{vec}(\mathtt{R})^\top)^\top|| = 1 \,, \tag{8.34}$$

[1] For any 3-vector $\mathbf{a}$ and $\mathbf{b}$, it satisfies $[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}$ and $\mathbf{a}^\top [\mathbf{b}]_\times = (\mathbf{a} \times \mathbf{b})^\top$

where the part of constraints can be changed and the equation can be written as follows:

$$||\mathtt{A}(\text{vec}(\mathtt{E})^\top,\ \text{vec}(\mathtt{R})^\top)^\top||\ \text{ subject to }\ ||\text{vec}(\mathtt{E})|| = 1\ . \tag{8.35}$$

This specific minimization problem can be solved by using the least-squares solution of homogeneous equations subject to a constraint, as discussed in Appendix A.5.4.2 on page 595, [27]. The details of the least-squares solution for our problem is explained in the following section section 8.4.2. Accordingly, finding the solution of the minimization problem in (8.35) is the same as getting the solution of the following minimization problem:

$$(\mathtt{A}_\mathtt{R}\mathtt{A}_\mathtt{R}^+ - \mathtt{I})\mathtt{A}_\mathtt{E}\text{vec}(\mathtt{E}) = \mathbf{0}\ , \tag{8.36}$$

where $\mathtt{A}_\mathtt{R}^+$ is the pseudo-inverse of $\mathtt{A}_\mathtt{R}$, and $\mathtt{A}_\mathtt{E}$ is the first 9 columns of the matrix $\mathtt{A}$, and $\mathtt{A}_\mathtt{R}$ is the last 9 columns of $\mathtt{A}$.

Algorithm 2: *A linear algorithm solving the generalized essential matrix in the cases of the locally-central, axial and locally-central-and-axial generalized camera model.*

Input: A set of corresponding rays $\mathbf{L} \leftrightarrow \mathbf{L}'$ in Plücker coordinates, where $\mathbf{L} = (\mathbf{x}^\top,\ (\mathbf{v} \times \mathbf{x})^\top)^\top$ and $\mathbf{L}' = (\mathbf{x}'^\top,\ (\mathbf{v}' \times \mathbf{x}')^\top)^\top$. For the locally-central case, $\mathbf{v} = \mathbf{v}'$. For the axial case, all $\mathbf{v}$ and $\mathbf{v}'$ should lie on a single line. For the locally-central-and-axial case, all $\mathbf{v}$ and $\mathbf{v}'$ should be the same point, and should lie on a single line.

Output: A 6×6 generalized essential matrix $\mathtt{G}$ including its components such as a 3×3 matrix $\mathtt{E}$, 3×3 rotation matrix $\mathtt{R}$ and the translation $\mathbf{t}$ with scale.

1 Normalization of rays: translate cameras by a centroid of given points, and scale them to lie in a unit distance.
2 Construct generalized epipolar equations: given corresponding rays, build a system of linear equations $\mathtt{A}_E\text{vec}(\mathtt{E}) + \mathtt{A}_R\text{vec}(\mathtt{R}) = \mathbf{0}$ using (8.15).
3 Compute the pseudo-inverse $\mathtt{A}_\mathtt{R}^+$ of $\mathtt{A}_\mathtt{R}$, build a system of linear equations $(\mathtt{A}_\mathtt{R}\mathtt{A}_\mathtt{R}^+ - \mathtt{I})\mathtt{A}_\mathtt{E}\text{vec}(\mathtt{E}) = \mathbf{0}$. Solve $\text{vec}(\mathtt{E})$ in (8.36) using SVD. Decompose the matrix $\mathtt{E}$ to get a rotation matrix $\mathtt{R}$, where $\mathtt{R}$ has two possible solutions.
4 Solve $\mathbf{t}$ with known $\mathtt{R}$ using (8.15).

8.4.2 Minimizing $||\mathtt{A}\mathbf{x}||$ subject to $||\mathtt{C}\mathbf{x}|| = 1$

An algorithm for the least-squares solutions to homogeneous equations with a constraint is summarized by Hartley and Zisserman in Appendix 5.4.2 on page 595, [27]. In this section, the algorithm is introduced and discussed how this algorithm fits to our problem. The algorithm of the least-squares solutions in [27] is rewritten as algorithm 3. This algorithm 3 can be modified to algorithm 4 by putting $(\mathrm{vec}(\mathtt{E})^\top,\ \mathrm{vec}(\mathtt{R})^\top)^\top$ into $\mathbf{x}$ and substituting $\mathtt{C}$ by $[\mathtt{I}_{9\times 9} \mid \mathtt{0}_{9\times 9}]$.

Algorithm 3: *Least-squares solution of homogeneous equations subject to the constraint* $||\mathtt{C}\mathbf{x}|| = 1$.

Input: $\mathtt{A}$ is a $m \times n$ matrix and $\mathtt{C}$ is a $k \times n$ matrix.
Output: $\mathbf{x}$ is n-dimensional vector for solution which minimizes $||\mathtt{A}\mathbf{x}||$ subject to $||\mathtt{C}\mathbf{x}|| = 1$.

1 Compute the SVD$(\mathtt{C}) = \mathtt{U}\mathtt{D}\mathtt{V}^\top$, and write $\mathtt{A}' = \mathtt{A}\mathtt{V}$.
2 Suppose rank$(\mathtt{D}) = r$ and let $\mathtt{A}' = [\mathtt{A}'_1 \mid \mathtt{A}'_2]$ where $\mathtt{A}'_1$ consists of the first r columns of $\mathtt{A}'$, and $\mathtt{A}'_2$ is formed from the remaining columns.
3 Let $\mathtt{D}_1$ be the upper $r \times r$ minor of $\mathtt{D}$.
4 Compute $\mathtt{A}'' = (\mathtt{A}'_2{\mathtt{A}'_2}^+ - \mathtt{I})\mathtt{A}'_1\mathtt{D}_1^{-1}$. This is an $m \times r$ matrix.
5 Minimize $||\mathtt{A}''\mathbf{x}''||$ subject to $||\mathbf{x}''|| = 1$ using the SVD.
6 Set $\mathbf{x}'_1 = \mathtt{D}_1^{-1}\mathbf{x}''$ and $\mathbf{x}'_2 = -{\mathtt{A}'_2}^+\mathtt{A}'_1\mathbf{x}'_1$.
7 Let $\mathbf{x}' = ({\mathbf{x}'_1}^\top,\ {\mathbf{x}'_2}^\top)^\top$.
8 The solution is given by $\mathbf{x} = \mathtt{V}\mathbf{x}'$.

Algorithm 4: *Modified least-squares solution for generalized epipolar equations.*

Input: $\mathtt{A}$ is a $m \times 18$ matrix constructed by a set of ray correspondences represented in Plücker coordinates.
Output: $\mathrm{vec}(\mathtt{E})$ is a 9-dimensional vector for the solution which minimizes $||\mathtt{A}(\mathrm{vec}(\mathtt{E})^\top,\ \mathrm{vec}(\mathtt{R})^\top)^\top||$ subject to $||(\mathrm{vec}(\mathtt{E}))|| = 1$.

1 Set $\mathtt{C} = [\mathtt{I}_{9\times 9} \mid \mathtt{0}_{9\times 9}]$.
2 Compute the SVD$(\mathtt{C}) = \mathtt{U}\mathtt{D}\mathtt{V}^\top$.
3 We have rank$(\mathtt{D}) = 9$ and let $\mathtt{A} = [\mathtt{A}_\mathtt{E} \mid \mathtt{A}_\mathtt{R}]$ where $\mathtt{A}_\mathtt{E}$ consists of the first 9 columns of $\mathtt{A}$, and $\mathtt{A}_\mathtt{R}$ is formed from the remaining 9 columns of $\mathtt{A}$.
4 Let $\mathtt{D}_1$ be the upper 9×9 minor of $\mathtt{D}$. Actually, $\mathtt{D}_1 = \mathtt{I}_{9\times 9}$.
5 Compute $\mathtt{A}'' = (\mathtt{A}_\mathtt{R}{\mathtt{A}_\mathtt{R}}^+ - \mathtt{I})\mathtt{A}_\mathtt{E}\mathtt{D}_1^{-1} = (\mathtt{A}_\mathtt{R}{\mathtt{A}_\mathtt{R}}^+ - \mathtt{I})\mathtt{A}_\mathtt{E}$. This is a 18×9 matrix.
6 Minimize $||\mathtt{A}''\mathbf{x}''||$ subject to $||\mathbf{x}''|| = 1$ using the SVD.
7 Set $\mathbf{x}'_1 = \mathtt{D}_1^{-1}\mathbf{x}'' = \mathbf{x}''$ and $\mathbf{x}'_2 = -{\mathtt{A}_\mathtt{R}}^+\mathtt{A}_\mathtt{E}\mathbf{x}'_1$.
8 Let $\mathbf{x}' = ({\mathbf{x}'_1}^\top,\ {\mathbf{x}'_2}^\top)^\top$.
9 The solution is given by $(\mathrm{vec}(\mathtt{E})^\top,\ \mathrm{vec}(\mathtt{R})^\top)^\top = \mathtt{V}\mathbf{x}' = \mathtt{I}_{18\times 18}\mathbf{x}' = \mathbf{x}'$.
10 Therefore, the solution $\mathrm{vec}(\mathtt{E})$ is equal to $\mathbf{x}''$.

8.4.3 Alternate method improving the result of the linear algorithm

Our proposed linear method gives a solution of $\mathbf{t}$ when a rotation $\mathtt{R}$ is known from the equation (8.15). In the same way, a solution of $\mathtt{R}$ can be obtained when a translation $\mathbf{t}$ is known. This fact gives us an alternative way of improving the solutions of $\mathtt{R}$ and $\mathbf{t}$, iteratively. The strategy is to first estimate $\mathbf{t}$ given $\mathtt{R}$, and re-estimate $\mathtt{R}$ given the $\mathbf{t}$, and repeat these until a reasonable residual error is achieved.

8.5 Experiments

8.5.1 Synthetic experiments

We carry out three experiments with synthetic data. The synthetic data simulates three commonly used generalized cameras which are (1) a general non-axial camera rig; (2) an axial camera rig; and (3) a non-overlapping stereo head. These three types of generalized cameras are shown in Figure 8.4. The image size for each camera is about $1,000 \times 1,000$ pixels. The three cases have the rank 16, 14 and 14, respectively, from the analysis of the generalized epipolar equations in the previous sections. Standard deviation 0.05 degrees of Gaussian distribution noise are added into the direction vector of Plücker line coordinates.

In Figure 8.5, we plotted an average convergence curve for 50 runs of the alternation method. As shown in Figure 8.5, the residual error for the alternation method decreases rapidly in less than 20 iterations. For the first two cases in Figure 8.4, 1,000 runs are carried out with random points and histograms of estimation errors are shown in Figure 8.6 and Figure 8.7. Graphs of the errors of the estimated rotation and the estimated translation from 1,000 trials are shown for the first two cases in Figure 8.8. and Figure 8.9. For the non-overlapping stereo head, errors of the estimated rotation and the estimated translation are shown in Figure 8.10. To see how much our method improves the result of estimation, another experiment with a monocular camera is carried out and the comparison between them is shown in Figure 8.10. As seen in Figure 8.10, our method gives better estimation results than the monocular camera system.

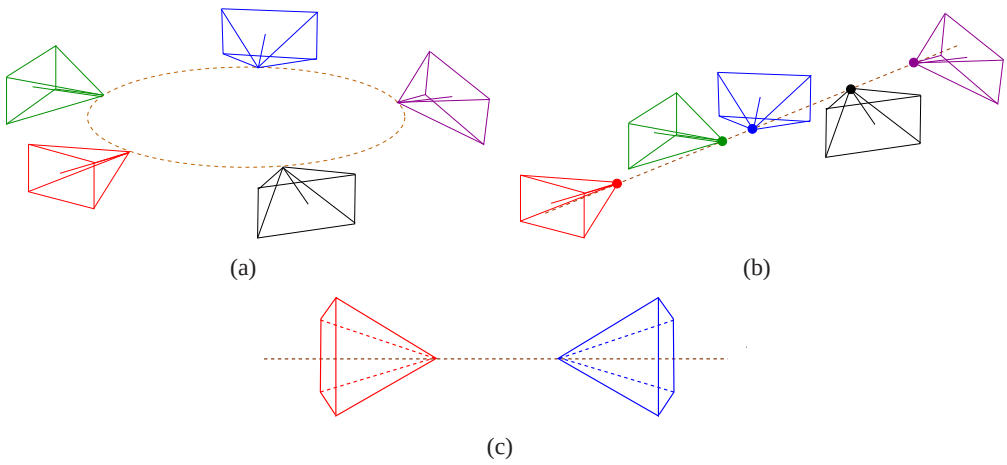


Figure 8.4: *Three types of generalized cameras used in the experiments with the synthetic data: (a) A general non-axial camera rig ("the locally-central case"), (b) an axial camera rig ("the locally-central-and-axial case") and (c) a non-overlapping stereo head ("the locally-central-and-axial case").*

8.5.2 Real experiments

An experiment with real data is carried out. The real data is obtained from a spherical imaging device, Ladybug™2 camera system [32]. The Ladybug™2 camera system consists of 6 cameras in the head unit. There are 5 cameras along the ring of the head unit and one camera on top of the head unit as shown in Figure 8.11. Although this camera system is mainly used to capture images of spherical or omnidirectional vision, the total 6 cameras are considered as a multi-camera system. Accordingly, the Ladybug™2 camera is a real example of the "locally-central" case of generalized cameras.

To acquire the ground truth, a trajectory of the Ladybug™2 camera is generated from a computer aided drawing tool (Xfig) as shown in Figure 8.12. This trajectory is a ∞-shape and it has marked positions for the Ladybug™2 camera to be aligned at every frame. As seen in Figure 8.11, the bottom of the Ladybug™2 camera is flat. So, one of the edges on the bottom of the head unit can be aligned with the marked positions in the experiment. For the alignment, a target point on the edge is marked with a label. Then, the trajectory is printed on a piece of A2-size paper and the printed trajectory is attached under a piece of half-transparent paper with 1mm grids. All the marked positions can be measured in millimetres in 2-dimensional coordinates, and they provide us the ground truth for the motion of the Ladybug™2 camera in

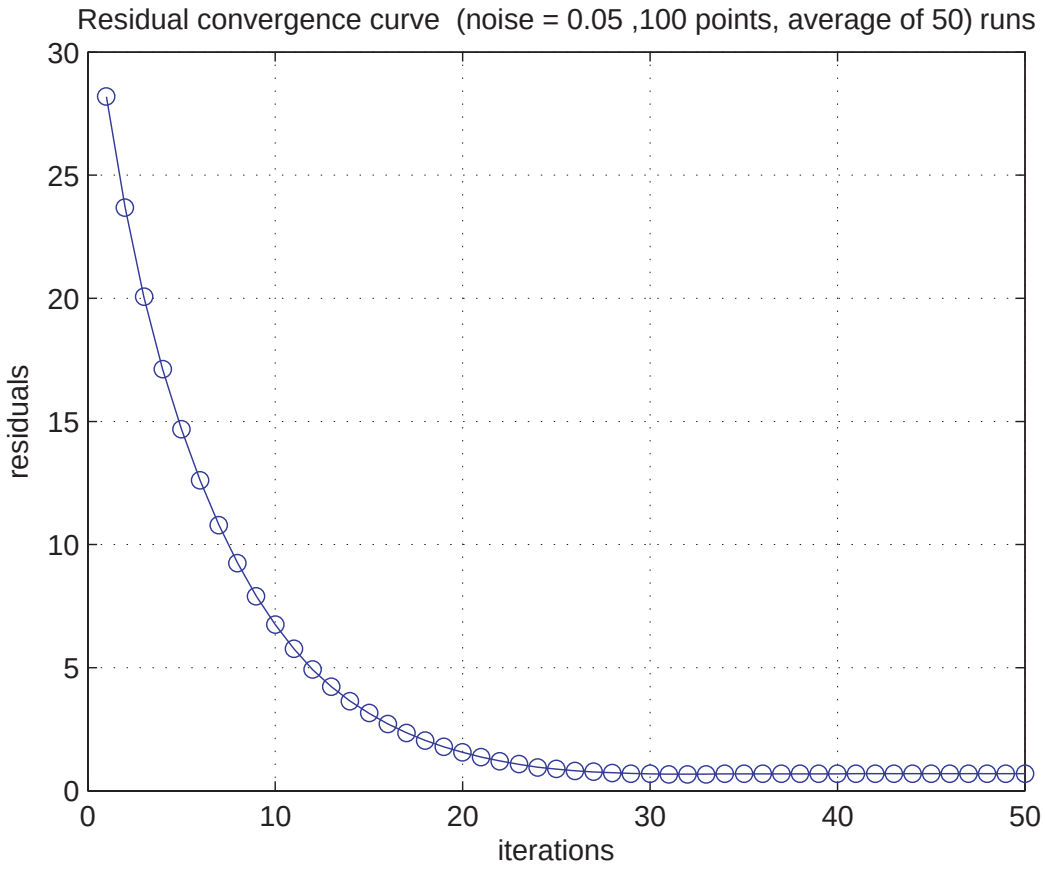


Figure 8.5: *An average convergence curve of the alternation procedure, i.e. residual error v.s. number of iterations. The curve was generated by averaging 50 runs with 0.05 degrees of the standard deviation noise.*

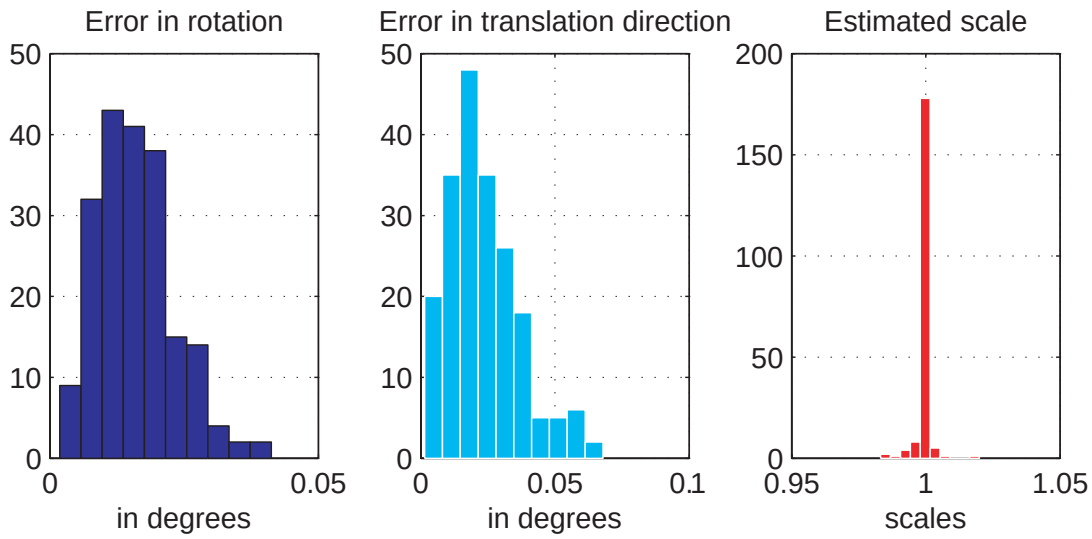


Figure 8.6: *Histograms of estimation accuracy based on 1,000 randomly simulated tests for non-axial multi-camera rig. In all these tests, we introduce angular noise at the level of standard deviation 0.05 degrees. The number of rays is 100.*

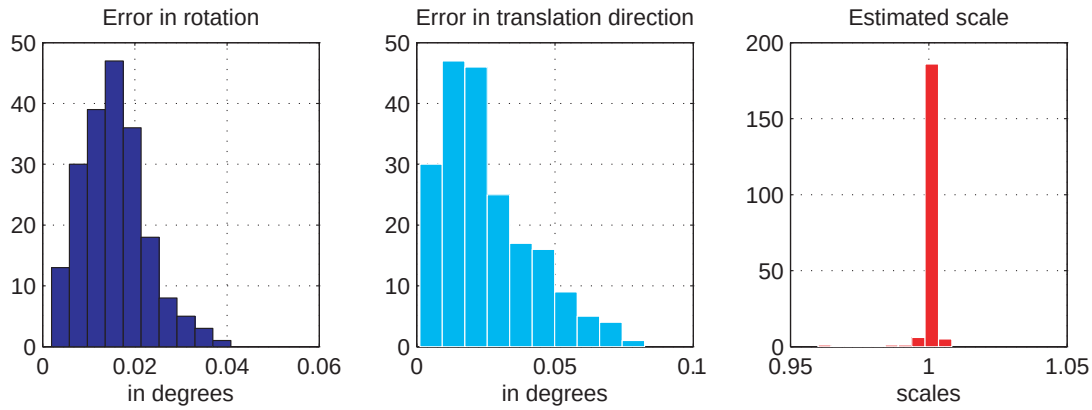


Figure 8.7: *Histograms of estimation accuracy based on 1,000 randomly simulated tests for an axial camera rig. In all these tests, we introduce angular noise at the level of standard deviation 0.05 degrees. The number of rays is 100.*

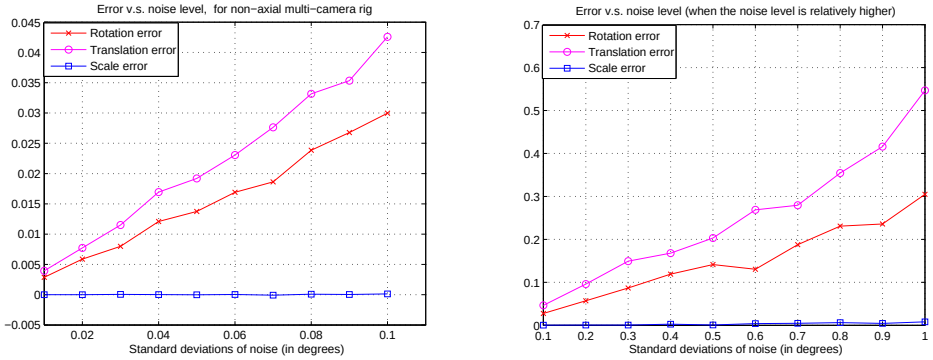


Figure 8.8: *This figure shows estimation accuracy (in rotation, translation, scale) as a function of noise level. The error in scale estimate is defined as* $\|1 - \frac{||\mathbf{t}||}{||\hat{\mathbf{t}}||}\|$. *Results for simulated non-axial camera rigs.*

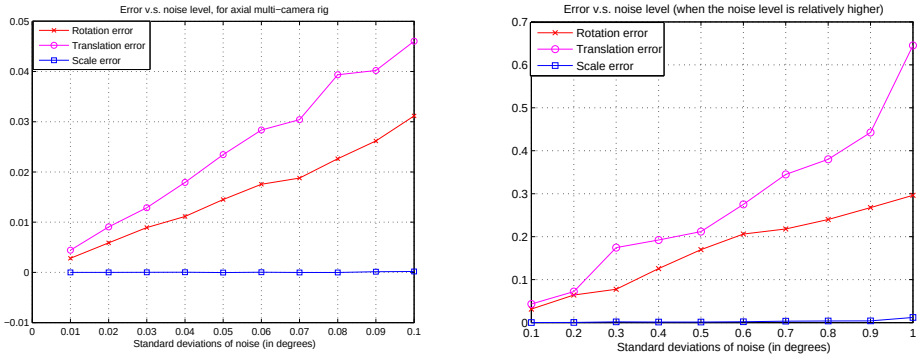


Figure 8.9: *This figure shows estimation accuracy (in rotation, translation, scale) as a function of noise level. The error in scale estimate is defined as* $\|1 - \frac{||\mathbf{t}||}{||\hat{\mathbf{t}}||}\|$. *Results for simulated axial camera rigs.*

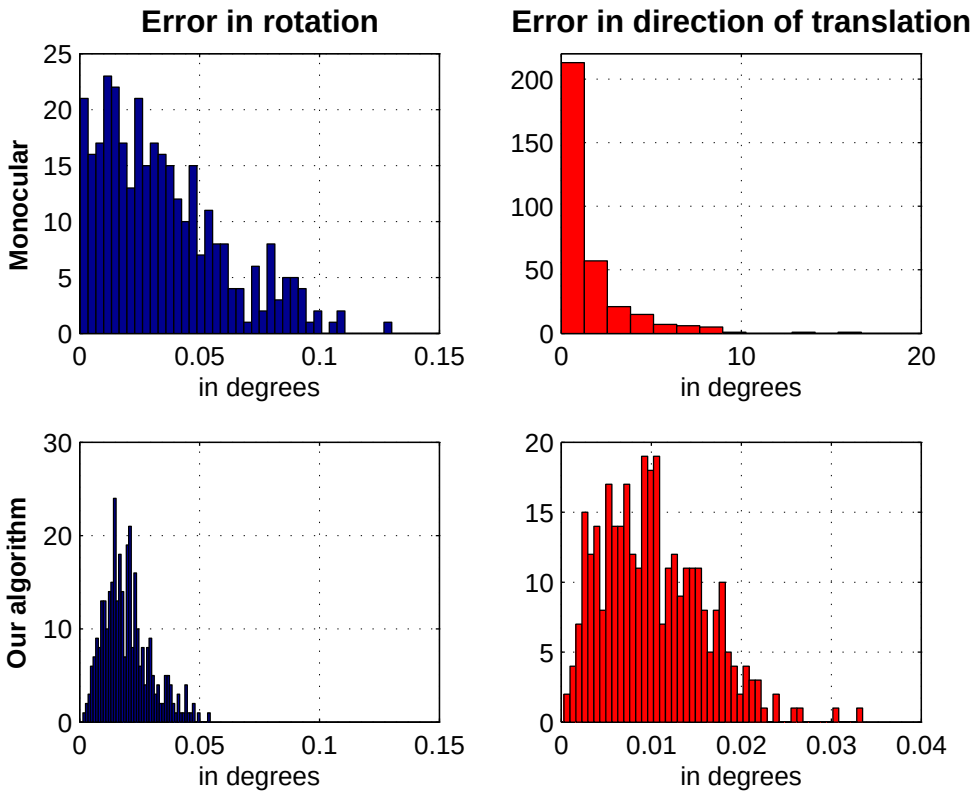
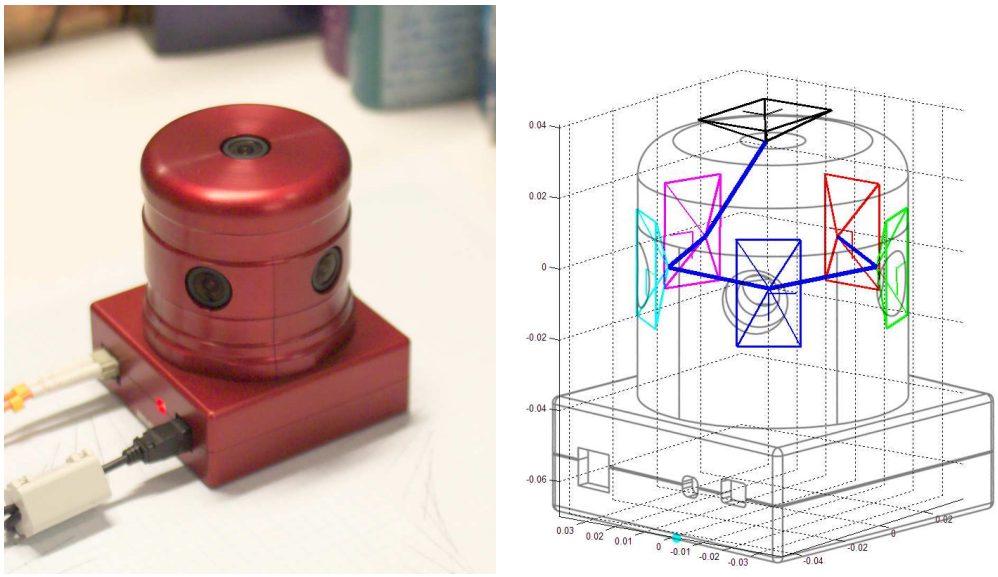


Figure 8.10: ***Experiment results for a 2-camera stereo system.*** *Top row: estimation errors in rotation and translation direction by using one camera only (i.e., monocular). Bottom row: estimation errors obtained by the proposed method.*



(a) (b)

Figure 8.11: *(a) Ladybug^{TM}2 camera system consisting of 5 cameras on the side and 1 camera on the top of the head unit. A label is attached on the left-side edge of the bottom of the head unit, which is just under the red light-emitting diode (LED). The label is used to align the camera with a trajectory printed on a piece of paper. (b) Positions of the 6 cameras in Ladybug^{TM}2 camera. The positions are retrieved from calibration information provided by Point Grey Inc. The order of cameras is indicated as colour red, green, blue, cyan, magenta and black, respectively. The label for the alignment is indicated as a cyan dot at the bottom of the head unit. (All copyrights of the original CAD drawing are reserved to Point Grey Inc. Modified and reprinted with permission from* `http://www.ptgrey.com`*)*

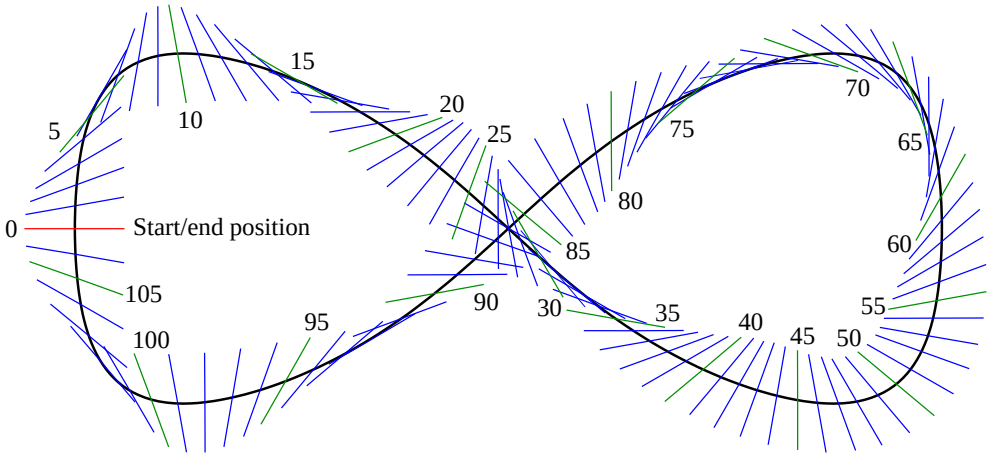


Figure 8.12: *A ∞-shape trajectory produced by a drawing tool. The trajectory is printed on a piece of paper and is used for the path of the Ladybug^{TM}2 camera in the experiment. The trajectory is a closed-loop and has 108 positions. A starting position and end position are shown as a red line segment, and the frame numbers are shown.*

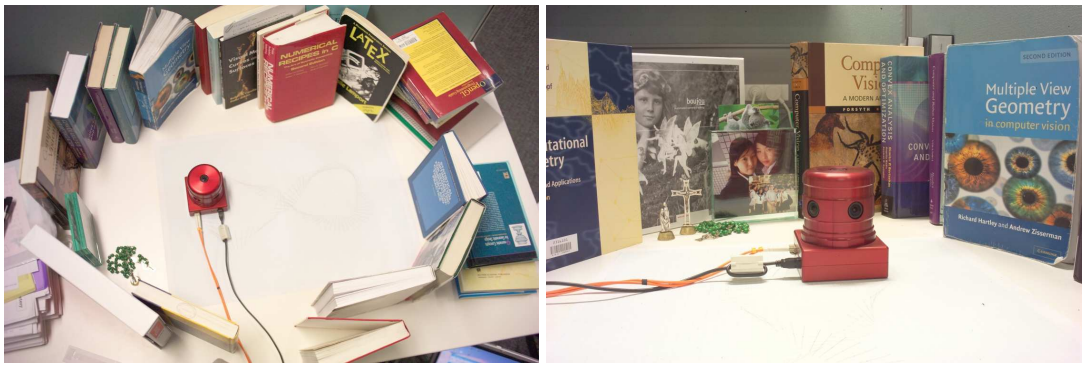


Figure 8.13: *Experiment setup with a Ladybug^{TM}2 camera and books surrounding the camera. The Ladybug^{TM}2 camera is placed on a piece of A2-size paper on which the trajectory of 108 positions of cameras is printed.*

Figure 8.14: *A sample of 6 images taken by the Ladybug^{TM}2 camera placed on a piece of paper and surrounded by books in the experiment. The first 5 images from the left are from the camera id number 0 to 5, which are on a ring of the head unit, and the last picture is from the camera id 6, which is on the top of the head unit.*

this experiment.

For features to track in this real experiment, static objects such as books and boxes are placed around the Ladybug^{TM}2 camera, as shown in Figure 8.13. Then, the Ladybug^{TM}2 camera is manually moved and aligned with the marked positions at every frame.

A set of six images is captured by the Ladybug^{TM}2 camera at each marked position. The number of the marked positions is 108, so a total of 648 images are captured in this experiment. The size of the captured images is 1024×768 pixels and all calibration information is provided by Point Grey Inc [32]. A sample of 6 images captured by the Ladybug^{TM}2 camera in the experiment is shown in Figure 8.14.

Features in the images are detected, and tracking of the features is performed throughout

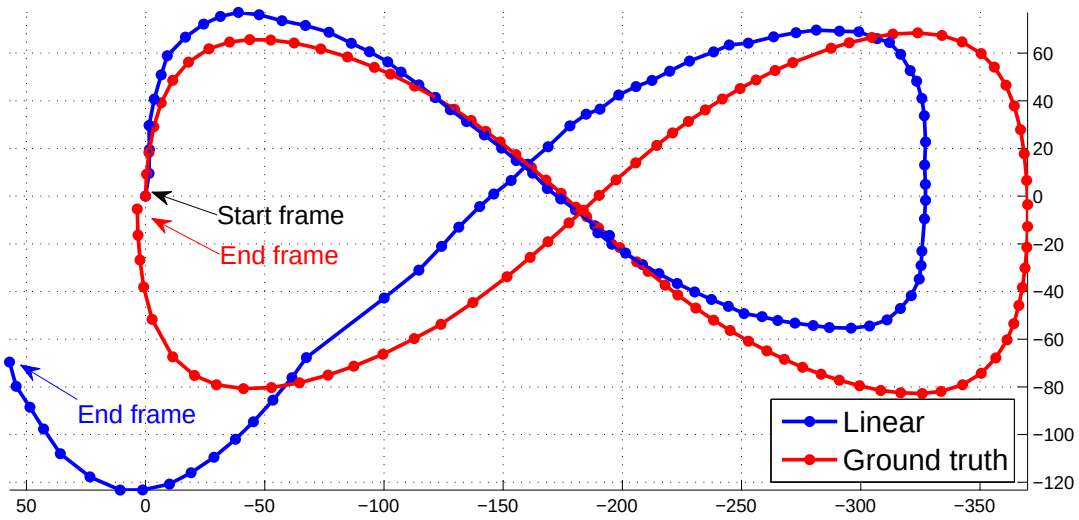


Figure 8.15: *Estimated motion of the Ladybug™2 camera in the real experiment using our proposed "linear method" which is indicated as blue dots and lines. The ground truth of the motion is superimposed as red dots and lines. All the estimated positions go well until the frame number 92 out of total 108 frames. At the moment of the frame number 93, the linear method gives a large amount of displacement error. However, after that frame, the estimation goes well again until the last frame. The estimated loop would be closed if there were no large error at the frame 93. It tells us our linear method needs to find some other ways or non-linear estimation using bundle adjustment to improve the result. Accordingly, the linear method serves as a good initial estimate for the bundle adjustment. The measurement unit in this figure is millimetre.*

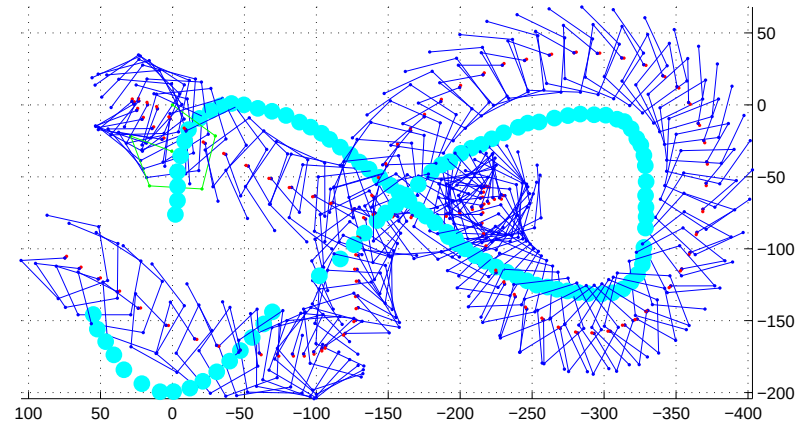
6 image sequences by Boujou 2d3 software [1]. Because of the wide-angle lenses of the Ladybug™2 camera –2.5mm focal length high quality micro lenses– there is a large amount of radial distortion in the captured images. So, radial distortion correction is applied to the coordinates of the features. After the radial distortion correction, a RANSAC algorithm is used to get rid of outliers from the features [13].

Given all inliers at every frame and camera calibration information, Plücker line coordinates for the inliers are represented in a local coordinate system. One of the six cameras in the Ladybug™2 camera system is selected and aligned with the origin of the local coordinate system. With all these real data, the estimated motion of the Ladybug™2 camera and its comparison with the ground truth are shown in Figure 8.15. We showed a 3D view of the estimated motion and positions of all 6 cameras of the Ladybug™2 camera system in Figure 8.16. Specifically, note that the trajectory is a closed loop and the estimated positions of the cameras

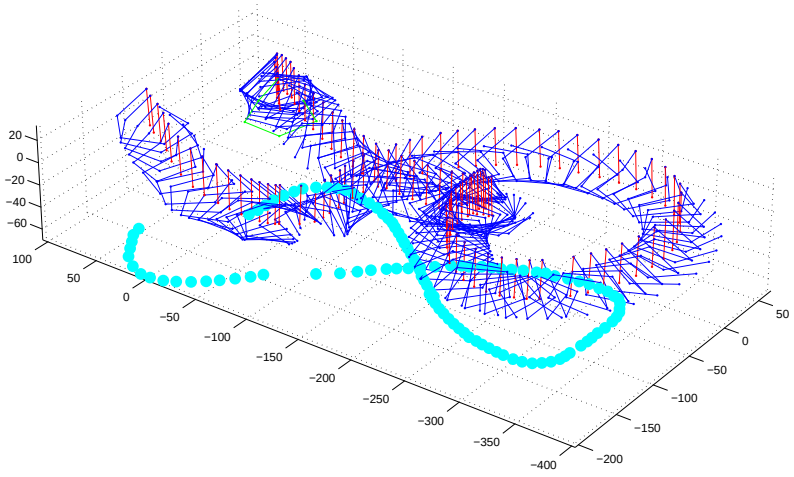
accumulates errors at every frame. Therefore, examining how well the trajectory is closed at the last frame can be one of criteria to verify the result. In this experiment, the estimation seems fine throughout all frames. However, there is a large displacement in the estimation at the moment of the frame number 93. It tells us the linear method is fairly applicable and gives good result, but in terms of robustness we need a better way of minimizing residual errors in motion estimation.

8.6 Conclusion

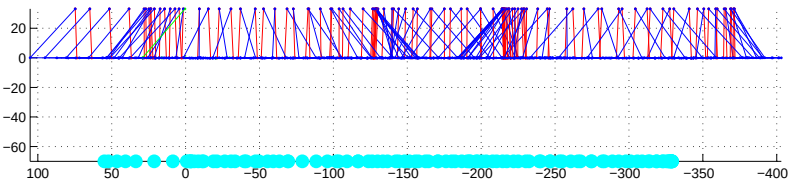
In this chapter, a linear approach to estimate motion of generalized cameras is proposed. This linear method, especially, deals with four types of generalized cameras such as the most-general case, the locally-central case, the axial case and the locally-central-and-axial case. For these four cases, our proposed linear algorithm obtains a solution of the generalized epipolar equations using a constrained minimization method based on the singular value decomposition, and it gives an estimate of the 6×6 generalized essential matrix. Our method extracts a 3×3 rotation matrix and a translation 3-vector with scale from the estimated generalized essential matrix. Because it is a linear method, practically, it is fast and easy to implement compared to non-linear methods. Furthermore, this linear method could be a good initial estimate for non-linear optimization methods such as bundle adjustment.



(a)



(b)



(c)

Figure 8.16: *The estimated motion and position of 6 cameras of the Ladybug^{TM}2 camera are plotted. The 6-camera is indicated as blue dots and lines. The axis of the Ladybug^{TM}2 camera is shown as red lines. The marked position of the label attached on the head unit, which is aligned with the pre-defined trajectory, is shown as cyan dots. (a) Top view of the estimated motion and positions of 6 cameras; (b) Perspective view of the estimated motion and positions of 6 cameras; (c) Side view of the estimated motion and positions of 6 cameras.*

Chapter 9

Visual Odometry in Non-Overlapping View Using Second-order cone programming

We present a further solution of motion estimation for a set of cameras firmly mounted on a head unit not having overlapping views in each image. We have found that this is related to solving a triangulation problem which finds a point in space from multiple views. The optimal solution of the triangulation problem in L_∞ norm is solved using second-order cone programming (SOCP) lately in computer vision research. Consequently, with the help of the optimal solution for the triangulation, we can solve visual odometry by using SOCP.

In this chapter, we propose a solution to estimate 6 degree of freedom motion of a set of multiple cameras with non-overlapping views, based on L_∞ triangulation.

9.1 Problem formulation

Consider a set of n calibrated cameras with non-overlapping fields of view. Since the cameras are calibrated, we may assume as before that they are all oriented in the same way just to simplify the mathematics. This is easily done by multiplying an inverse of the rotation matrix to the original image coordinates. This being the case, we can also assume that they all have camera matrices originally equal to $\mathtt{P}_i = [\mathtt{I}| - \mathbf{c}_i]$. We assume that all $\mathbf{c}_i$ are known.

The cameras then undergo a common motion, described by a Euclidean matrix

$$\mathtt{M} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\,\mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}.$$

where $\mathtt{R}$ is a rotation, and $\mathbf{t}$ is a translation of a set of cameras. Then, the i-th camera matrix changes to

$$\mathtt{P}'_i = \mathtt{P}_i\,\mathtt{M}^{-1} = [\mathtt{I} \mid -\mathbf{c}_i] \begin{bmatrix} \mathtt{R}^\top & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} = [\mathtt{R}^\top \mid \mathbf{t} - \mathbf{c}_i] \tag{9.1}$$

which is located at $\mathtt{R}(\mathbf{c}_i - \mathbf{t})$.

Suppose that we compute all the essential matrices of the cameras independently, then decompose them into rotation and translation. We observe that the rotations computed from all the essential matrices are the same. This is true only because all the cameras have the same orientation. We can average them to get an overall estimate of rotation. Then, we would like to compute the translation. This is a triangulation problem as will be demonstrated.

9.1.1 Geometric concept

First, let us look at a geometric idea derived from this problem. An illustration of the motion of a set of cameras is shown in Figure 9.1. A bundle of cameras is moved by a rotation $\mathtt{R}$ and translation $\mathbf{t}$. All cameras at $\mathbf{c}_i$ are moved to $\mathbf{c}'_i$. The first camera at position $\mathbf{c}'_1$ is a sum of vectors $\mathbf{c}_i$, $\mathbf{c}'_i - \mathbf{c}_i$ and $\mathbf{c}'_1 - \mathbf{c}'_i$ where $i = 1...3$. Observing that the vector $\mathbf{v}_i$ in Figure 9.1 is the same as the vector $\mathbf{c}'_i - \mathbf{c}_i$ and the vector $\mathbf{c}'_1 - \mathbf{c}'_i$ is obtained by rotating the vector $\mathbf{c}_1 - \mathbf{c}_i$, the first camera at position $\mathbf{c}'_1$ can be rewritten as a sum of three vectors $\mathbf{c}_i$, $\mathtt{R}(\mathbf{c}_1 - \mathbf{c}_i)$ and $\mathbf{v}_i$. Therefore, the three vectors $\mathbf{v}_i$, colored solid arrows in Figure 9.1 meet in one common point $\mathbf{c}'_1$, the position of the centre of the first camera after the motion. It means that finding the motion of the set of cameras is the same as solving a triangulation problem of translation direction vectors derived from each view.

Secondly, let us derive detail of equations for this problem from the geometry concept we have described above. Let $\mathtt{E}_i$ be the essential matrix for the i-th camera. From $\mathtt{E}_1$, we can

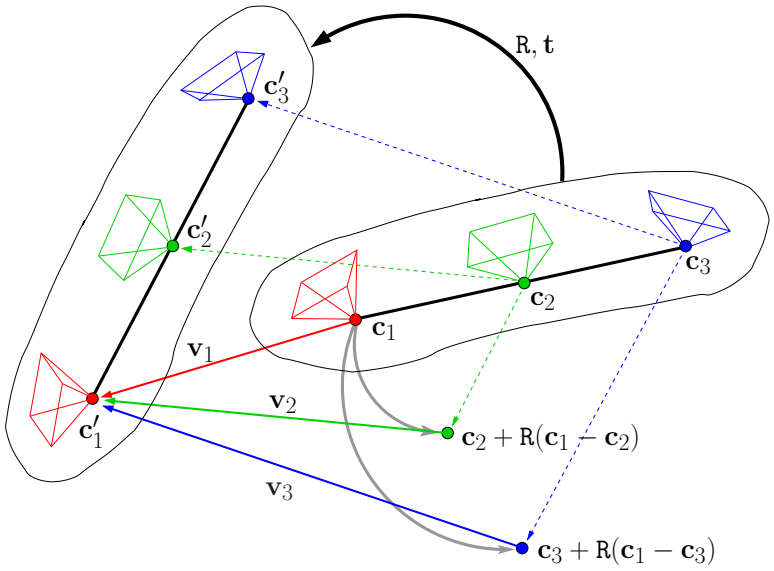


Figure 9.1: *A set of cameras is moved by a Euclidean motion of rotation* $\mathtt{R}$ *and translation* $\mathbf{t}$*. The centre of the first camera* $\mathbf{c}_1$ *is moved to* $\mathbf{c}'_1$ *by the motion. The centre* $\mathbf{c}'_1$ *is a common point where all translation direction vectors meet. The translation direction vectors are indicated as red, green and blue solid arrows which are* $\mathbf{v}_1$*,* $\mathbf{v}_2$ *and* $\mathbf{v}_3$*, respectively. Consequently, this is a triangulation problem.*

compute the translation vector of the first camera, $\mathtt{P}_1$, in the usual way. This is a vector passing through the original position of the first camera. The final position of this camera must lie along this vector. Next, we use $\mathtt{E}_i$, for $i > 1$ to estimate a vector along which the final position of the **first** camera can be found. Thus, for instance, we use $\mathtt{E}_2$ to find the final position of $\mathtt{P}_1$. This works as follows. The i-th essential matrix $\mathtt{E}_i$ decomposes into $\mathtt{R}_i = \mathtt{R}$ and a translation vector $\mathbf{v}_i$. In other words, $\mathtt{E}_i = \mathtt{R}[\mathbf{v}_i]_\times$. This means that the i-th camera moves to a point $\mathbf{c}_i + \lambda_i \mathbf{v}_i$, the value of λ_i being unknown. This point is the final position of each camera $\mathbf{c}'_i$ in Figure 9.1. We transfer this motion to determine the motion of the first camera. We consider the motion as taking place in two stages, first rotation, then translation. First the camera centre $\mathbf{c}_1$ is rotated by $\mathtt{R}$ about point $\mathbf{c}_i$ to point $\mathbf{c}_i + \mathtt{R}(\mathbf{c}_1 - \mathbf{c}_i)$. Then it is translated in the direction $\mathbf{v}_i$ to the point $\mathbf{c}'_i = \mathbf{c}_i + \mathtt{R}(\mathbf{c}_1 - \mathbf{c}_i) + \lambda_i \mathbf{v}_i$. Thus, we see that $\mathbf{c}'_i$ lies on the line with direction vector $\mathbf{v}_i$, based at point $\mathbf{c}_i + \mathtt{R}(\mathbf{c}_1 + \mathbf{c}_i)$.

In short, each essential matrix $\mathtt{E}_i$ constrains the final position of the first camera to lie along a line. These lines are not all the same, in fact unless $\mathtt{R} = \mathtt{I}$, they are all different. The problem

now comes down to finding the values of λ_i and $\mathbf{c}'_i$ such that for all i:

$$\mathbf{c}'_1 = \mathbf{c}_i + \mathtt{R}(\mathbf{c}_1 - \mathbf{c}_i) + \lambda_i \mathbf{v}_i \quad \text{for } i = 1, \ldots, n \,. \tag{9.2}$$

Having found $\mathbf{c}'_1$, we can get $\mathbf{t}$ from the equation $\mathbf{c}'_1 = \mathtt{R}(\mathbf{c}_1 - \mathbf{t})$.

9.1.2 Algebraic derivations

Alternatively, it is possible to show an algebraic derivation of the equations as follows. Given $\mathtt{P}_i = [\mathtt{I}| - \mathbf{c}_i]$ and $\mathtt{P}'_i = [\mathtt{R}^\top \mid \mathbf{t} - \mathbf{c}_i]$ (See (9.1)), an essential matrix is written as

$$\mathtt{E}_i = \mathtt{R}^\top [\mathbf{c}_i + \mathtt{R}(\mathbf{t} - \mathbf{c}_i)]_\times \mathtt{I} = [\mathtt{R}^\top \mathbf{c}_i + (\mathbf{t} - \mathbf{c}_i)]_\times \mathtt{R}^\top \,.$$

Considering that the decomposition of the essential matrix $\mathtt{E}_i$ is $\mathtt{E}_i = \mathtt{R}_i [\mathbf{v}_i]_\times = [\mathtt{R}_i \mathbf{v}_i]_\times \mathtt{R}_i$, we may get the rotation and translation from (9.3) such as $\mathtt{R}_i = \mathtt{R}^\top$ and $\lambda_i \mathtt{R}_i \mathbf{v}_i = \mathtt{R}^\top \mathbf{c}_i + (\mathbf{t} - \mathbf{c}_i)$. As a result, $\mathbf{t} = \lambda_i \mathtt{R}^\top \mathbf{v}_i + \mathbf{c}_i - \mathtt{R}^\top \mathbf{c}_i$ which is the same equation derived from the geometric concept.

9.1.3 Triangulation problem

Equation (9.2) gives us independent measurements of the position of point $\mathbf{c}'_1$. Denoting $\mathbf{c}_i + \mathtt{R}(\mathbf{c}_1 - \mathbf{c}_i)$ by $\mathbf{C}_i$, the point $\mathbf{c}'_1$ must lie at the intersection of the lines $\mathbf{C}_i + \lambda_i \mathbf{v}_i$. In the presence of noise, these lines will not meet, so we need to find a good approximation to $\mathbf{c}'_1$. It is important to note that this problem is identical with the triangulation problem studied in [27]. We adopt the approach of [23] of solving this under L_∞ norm. The derived solution is the point $\mathbf{c}'_i$ that minimizes the difference between $\mathbf{c}'_1 - \mathbf{C}_i$ and the direction vector $\mathbf{v}_i$. In the presence of noise, the point $\mathbf{c}'_1$ will lie in the intersection of cones based at the vertex $\mathbf{C}_i$, and with axis defined by the direction vectors $\mathbf{v}_i$.

In particular, note that the points $\mathbf{c}_i$ and vectors $\mathbf{v}_i$ are known, having been computed from the known calibration of the camera geometry, and the computed essential matrices $\mathtt{E}_i$.

9.2 Second-order cone programming

In the previous section, the problem of estimating the motion of a set of cameras with non-overlapping fields of view is redefined as a triangulation problem. We provide the mathematical equations for the triangulation problem solving the motion estimation of the set of cameras.

Here instead of $\mathbf{c}'_1$, we write $\mathbf{X}$ as the final position of the first camera where all translations decomposed from each essential matrix meet together. As we have explained in the previous section, we have n cones, one on each line of the translation directions. Therefore, finding the overlapping of the cones is the solution we need to get the motion of cameras. Then, our original motion estimation problem is formulated as the following minimization problem:

$$\min_{\mathbf{X}} \max_{i} \frac{||(\mathbf{X} - \mathbf{C}_i) \times \mathbf{v}_i||}{(\mathbf{X} - \mathbf{C}_i)^\top \mathbf{v}_i} \ . \tag{9.3}$$

Specifically, note that the quotient is equal to $\tan(\theta_i)$ where θ_i is the angle between $\mathbf{v}_i$ and $(\mathbf{X} - \mathbf{C}_i)$. This problem can be solved as an SOCP using a bisection algorithm [35].

9.3 Summarized mathematical derivation

From the previous sections, we summarize the previous section in the following lemma and theorem.

Lemma 3. *Let* $\mathtt{P}_i = [\mathtt{I} \mid -\mathbf{c}_i]$ *be a camera matrix and* $\mathtt{P}'_i = \mathtt{P}_i \mathtt{M}^{-1} = [\mathtt{R}^\top \mid \mathbf{t} - \mathbf{c}_i]$ *be the camera matrix after a Euclidean motion* $\mathtt{M}$ *defined by*

$$\mathtt{M} = \begin{bmatrix} \mathtt{R} & -\mathtt{R}\,\mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} ,$$

where $\mathtt{R}$ *is a rotation and* $\mathbf{t}$ *is a translation of the motion. Let* $\mathtt{R}_i$ *and* $\mathbf{v}_i$ *be an orientation and translation vector which are decomposed from an essential matrix* $\mathtt{E}_i$ *corresponding to the pair of camera* $\mathtt{P}_i$ *and* $\mathtt{P}'_i$*. Then, the rotation and translation of the motion,* $\mathtt{R}$ *and* $\mathbf{t}$*, are determined*

by

$$\mathtt{R} = \mathtt{R}_i^\top \quad \textit{and} \quad \mathbf{t} = (\mathtt{I} - \mathtt{R}^\top)\mathbf{c}_i + \lambda_i \mathtt{R}^\top \mathbf{v}_i \ ,$$

where λ_i is non-zero scale of the translation $\mathbf{v}_i$.

Theorem 4. *Given n cameras, the centre of the camera $\mathtt{P}'_1$ is a point where all vectors $\mathbf{q}_i$ meet together for $i = 1...n$. The vector $\mathbf{q}_i$ is defined as $\mathbf{q}_i = \mathbf{C}_i + \mathbf{v}_i$ where $\mathbf{C}_i = \mathbf{c}_i + \mathtt{R}(\mathbf{c}_1 - \mathbf{c}_i)$ is a starting point of the vector $\mathbf{q}_i$ and $\mathbf{v}_i$ is a direction vector of the translation.*

Remark. The centre of the first camera can be found in L_∞ norm using SOCP as a solution of a triangulation problem.

9.4 Algorithm

The algorithm to estimate motion of cameras having non-overlapping views is as follows:

Objective: Given point correspondences $\mathbf{x}_{ij}$ in non-overlapping views, determine the motion of the cameras, $\mathtt{P}_i = [\mathtt{R}_i \mid -\mathtt{R}_i\mathbf{c}_i]$.

Algorithm:

1. Express the image points in the coordinate frame of the first camera by setting $\hat{\mathbf{x}}_{ij} = \mathtt{R}_i^\top \mathbf{x}_{ij}$ and also $\hat{\mathtt{P}}_i = [\mathtt{I} \mid -\mathbf{c}_i]$.

2. Compute each essential matrix $\mathtt{E}_i$ in terms of $\hat{\mathbf{x}}_{ij}$.

3. Decompose as $\mathtt{E}_i = \mathtt{R}_i[\mathbf{v}_i]_\times$ and set $\mathbf{C}_i = \mathbf{c}_i + \mathtt{R}(\mathbf{c}_1 - \mathbf{c}_i)$.

4. Solve the L_∞ triangulation problem to find $\mathbf{X} = \mathbf{c}'_1$ minimizing $\max_i[||((\mathbf{X} - \mathbf{C}_i) \times \mathbf{v}_i)||/((\mathbf{X} - \mathbf{C}_i)^\top \mathbf{v}_i)]$.

5. Compute $\mathtt{R}$ and $\mathbf{t}$ from $\mathbf{t} = \mathbf{c}_1 - \mathtt{R}^\top \mathbf{c}'_1$.

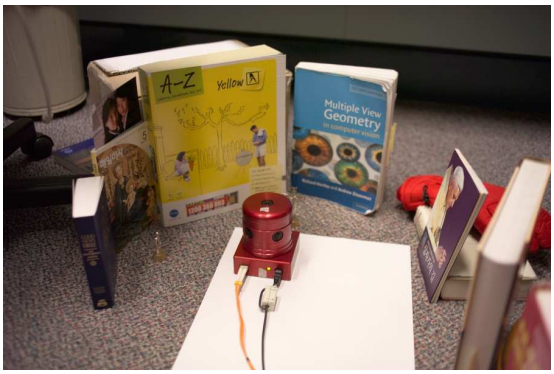


Figure 9.2: *An experimental setup of the Ladybug™2 camera on an A3 size paper surrounded by books. The camera is moved on the paper by hands, and each position of the camera at frames is marked on the paper to provide the ground truth of experiments.*

9.5 Experiments

We have used SeDuMi and Yalmip toolbox for optimization of SOCP problems [76, 45]. We have also used a five point solver to estimate an essential matrix [71, 44].

9.5.1 Real data

We used a Ladybug™2 camera as an example of our problem [32]. Calibration information provided by Point Grey Research Incorporated is used to get intrinsic and relative extrinsic parameters of all six cameras. The camera coordinate system of the Ladybug™2 uses a ZYX convention of Euler angles for the rotation matrix, so the rotation matrix of the extrinsic parameters from the calibration information needs to be converted to a XYZ convention for our mathematical notation.

A piece of paper is positioned on the ground, and the camera is placed on the paper. Some books and objects are randomly located around the camera. The camera is moved manually while the positions of the camera at some points are marked on the paper as edges of the camera head unit. These marked edges on the paper are used to get the ground truth of relative motion of the camera for this experiment. The experimental setup is shown in Figure 9.2. A panoramic image stitched in our experimental setup is shown in Figure 9.3.

In the experiment, 139 frames of image are captured by each camera. Feature tracking is

Figure 9.3: *A panoramic image is obtained by stitching together all six images from the Ladybug™2 camera. This image is created by LadybugPro, the software provided by Point Grey Research Inc.*

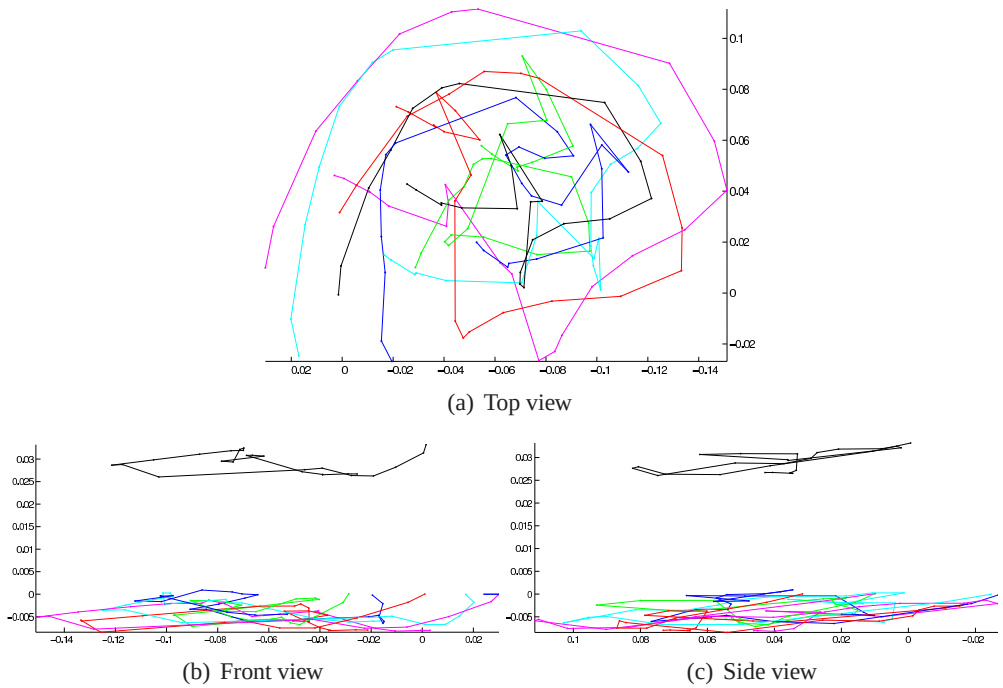
performed on the image sequence by the Kanade-Lucas-Tomasi (KLT) tracker [47]. Since there is lens distortion in the captured image, we correct the image coordinates of the feature tracks using lens distortion parameters provided by the Ladybug™ software development kit (SDK) library. The corrected image coordinates are used in all the equations we have derived. After that, we remove outliers from the feature tracks by the random sample consensus (RANSAC) algorithm with a model of epipolar geometry in two view and trifocal tensors in three view [13].

There are key frames where we marked the positions of the camera. They are frames 0, 30, 57, 80, 110 and 138 in this experiment. An estimated path of the cameras over the frames is shown in Figure 9.4. After frame 80, the essential matrix result was badly estimated and subsequent estimation results were erroneous.

Rotation pair	True rotation		Estimated rotation	
	Axis	Angle	Axis	Angle
$(\mathtt{R}_0, \mathtt{R}_1)$	[0 0 -1]	85.5°	[-0.008647 -0.015547 0.999842]	85.15°
$(\mathtt{R}_0, \mathtt{R}_2)$	[0 0 -1]	157.0°	[-0.022212 -0.008558 0.999717]	156.18°
$(\mathtt{R}_0, \mathtt{R}_3)$	[0 0 -1]	134.0°	[0.024939 -0.005637 -0.999673]	134.95°

Table 9.1: *Experimental results of rotations at key frames 0, 30, 57 and 80, which correspond to the position number 0–3, respectively. For instance, a pair of rotation* $(\mathtt{R}_0, \mathtt{R}_1)$ *corresponds to a pair of rotations at key frame* 0 *and* 30*. Angles of each rotation are represented by the axis-angle rotation representation.*

A summary of the experimental results is shown in Table 9.1 and 9.2. As can be seen, we have acquired reasonable good estimation of rotations from frame 0 up to frame 80 within



(a) Top view

(b) Front view

(c) Side view

Figure 9.4: *An estimated path of the Ladybug^{TM}2 camera in the view of (a) top, (b) side, and (c) front. The camera number 0, 1, 2, 3, 4 and 5 are indicated as red, green, blue, cyan, magenta and black color, respectively.*

approximately less than 1 degree of accuracy. Adequate estimation of translations is reached up to frame 57 within less than 0.5 degrees. We have successfully tracked the motion of the camera through 57 frames. Somewhere between frame 57 and frame 80 an error occurred that indicated the computation of the position of frame 80. This was probably due to an critical configuration that made the translation estimation invalid. Therefore, we have shown the critical configurations, frame-to-frame rotations, over frames in Figure 9.5-(a) and (b). As can be seen, there are some frames having less than 5 degrees at frames from 57 to 62, from 67 to 72 and from 72 to 77.

In Figure 9.5-(c), we have shown the difference between the ground truth and estimated position of the cameras in this experiment. As can be seen, the position of the cameras are accurately estimated up to 57 frames. However, the track went off at frame 80. A beneficial feature of our method is that we can avoid such bad condition for the estimation by looking at the angles between frames and residual errors on the SOCP, and then we try to use other frames for the estimation.

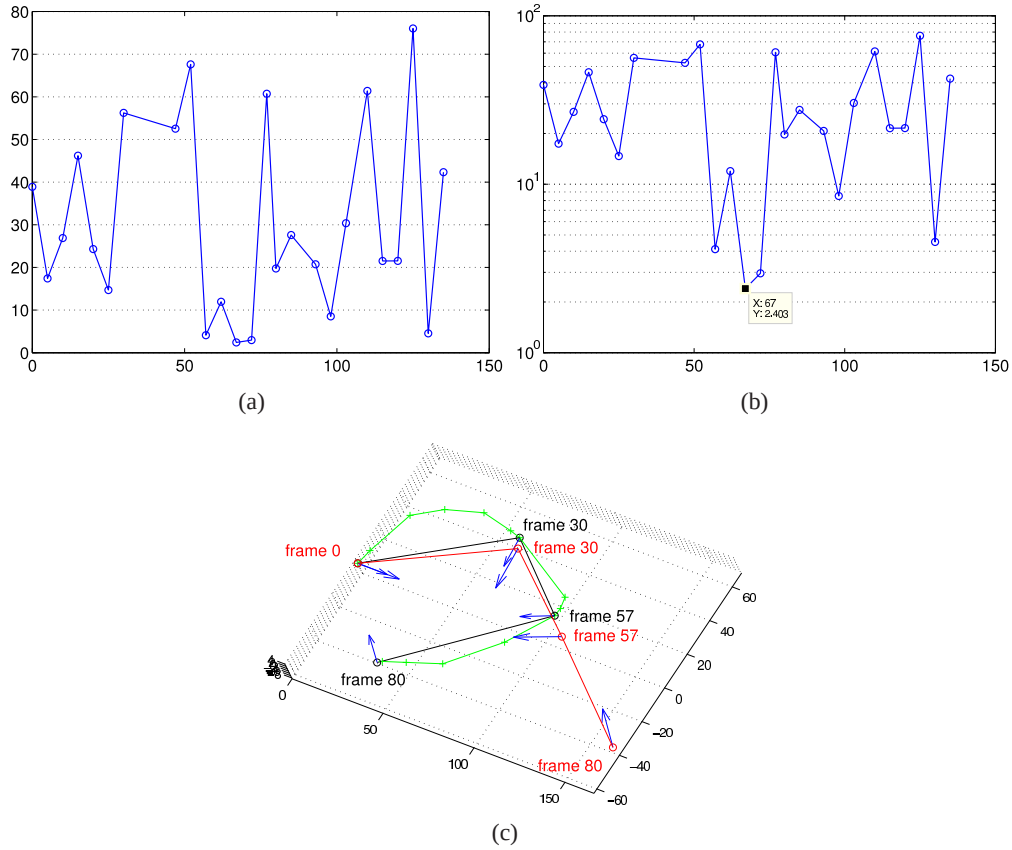


Figure 9.5: *Angles of the critical configuration are shown in degrees (a) and in log-scale (b). In particular, note that zero rotation means critically impossible condition to estimate the motion of the cameras given frames. (c) Ground truth of positions (indicated as red lines) of the cameras with orientations at key frames 0, 30, 57 and 80, and Estimated positions (indicated as black lines) of the cameras with its orientations at the same key frames. Orientations of the cameras are marked as blue arrows. Green lines are the estimated path through all 80 frames.*

Translation pair	Scale ratio		Angles	
	True value	Estimated value	True value	Estimated value
$(\mathbf{t}_{01}, \mathbf{t}_{02})$	0.6757	0.7424	28.5°	28.04°
$(\mathbf{t}_{01}, \mathbf{t}_{03})$	0.4386	1.3406	42.5°	84.01°

Table 9.2: *Experimental results of translation between two key frames are shown in scale ratio of two translation vectors and in angles of that at the two key frames. The translation direction vector* $\mathbf{t}_{0i}$ *is a vector from the centre of the camera at the starting position, frame number 0, to the centre of the camera at the position number* i*. For example,* $\mathbf{t}_{01}$ *is a vector from the centre of the camera at frame 0 to the centre of the camera at frame 30.*

9.6 Discussion

We have presented a solution to find motion of cameras which are firmly fixed and have little overlap of their field of view. This method works equally well for any number of cameras, not just two, and will therefore most likely avoid some of the critical configurations that the two-view method suffers. The method requires a non-zero frame-to-frame rotation. Probably because of this, the estimation of motion through a long image sequence significantly went of track.

The method geometrically showed good estimation result real experiments. However, the accumulated errors in processing long sequences of images made the system produce bad estimations over long tracks. A robust and accurate estimation algorithm of the essential matrix is very critical to obtain correct estimation of motions of the set of cameras.

Chapter 10

Motion Estimation for Multi-Camera Systems using Global Optimization

In this chapter, we would like to present a ***geometrically optimal*** L_∞ solution for 6 DOF motion for multi-camera systems from image point correspondences without any 3D point reconstruction. Hartley and Kahl recently showed that it is possible to find an optimal solution of the essential matrix for a single camera under L_∞ using a branch-and-bound algorithm, by searching for the optimal rotation over the rotation space [21]. Here we extend that algorithm to make it solve the 6 DOF motion for multiple cameras as well.

The method relies on the observation that if the rotation of the rigid multi-camera setup is known, then the optimal translation may be found using second-order cone programming (SOCP), as shown in chapter 9. As in [21], we use a branch-and-bound search over rotation space to find the optimal rotation. This allows the optimal translation to be computed at the same time. Instead of using SOCP, we improve the speed of computation by using linear programming (LP) which speeds up the computation enormously. In addition, a preemptive feasibility test allows us to speed up the branch-and-bound computation. In our experiments, the LP method with the feasibility-test showed 90 times faster convergence of errors than the pure LP method.

Multi-camera systems. Let us suppose that there are m cameras in the multi-camera system. We assume that the complete calibration of the camera system is known. The system of m cameras is moved rigidly and point correspondences are obtained between two points seen

before and after the motion. Given this camera and motion configuration, we would like to estimate the 6 DOF motion, namely the rotation and translation with scale, of the multi-camera system.

For multi-camera systems, there is an algorithm to estimate motion of the multi-camera systems using SOCP, as shown in chapter 9. In that chapter, it is shown that the motion problem is the same as a triangulation problem, once the rotation is known. SOCP was applied to obtain an optimal solution for translation in the multiple camera system. However, that method uses an unstable initial estimate of rotation which is extracted from an essential matrix from a single camera. Although, that method tries to obtain good initial estimates by averaging the selected rotations, the initial estimates come from each camera not from all cameras. Therefore, the rotation that is estimated from a single camera is still not an optimal solution for the whole system in terms of global optimization. Surely, it can be improved if we could estimate the initial rotation from all cameras.

In this chapter, we introduce a way of using all cameras to estimate the motion – rotation and translation – from the optimal essential matrix for the multi-camera system.

10.1 The L_∞ method for a single camera

In this section, we describe the method for obtaining an essential matix which is an optimal solution in a single-camera system using a branch-and-bound algorithm.

Hartley and Kahl performed a study to obtain a global solution for the essential matrix in terms of the geometric relations between two views [21]. There were no algorithms before this that proved a geometrical optimality for the essential matrix in L_∞ norm minimization.

Hartley and Kahl introduced a technique to rapidly search the rotation space in order to estimate an optimal solution for the essential matrix in L_∞ norm. However, their method is not an exhaustive search method. It does not examine all possible rotations, but attempts to minimize the maximum of L_∞ error for the essential matrix using convex optimization techniques, which have recently become popular among computer vision researchers. The convex optimization technique can be used to solve two pose problems – derivation of a camera

matrix for given 3D points and projected 2D points, and derivation of the relative pose of two views for given 2D points. In this thesis, we are concerned with the the second pose problem.

Given a 3D point $(\mathbf{X},\ 1)^\top$, where the value of the last coordinate is set as one for convenience, a projected point $\mathbf{x}$ in an image can be written as

$$\mathbf{x} = \mathtt{K}\mathtt{R}[\mathtt{I} \mid -\mathbf{c}] \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} \tag{10.1}$$

and its image vector $\mathbf{v}$ is written as

$$\mathbf{v} = \mathtt{K}^{-1}\mathbf{x} = \mathtt{R}[\mathtt{I} \mid -\mathbf{c}] \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} \tag{10.2}$$

$$\mathbf{v} = \mathtt{R}(\mathbf{X} - \mathbf{c})\ . \tag{10.3}$$

By representing the image vector $\mathbf{v}$ as a unit vector, equation (10.3) may be rewritten as

$$\mathbf{v} = \frac{\mathtt{R}(\mathbf{X} - \mathbf{c})}{\|\mathtt{R}(\mathbf{X} - \mathbf{c})\|}\ . \tag{10.4}$$

Let the two camera matrices be $\mathtt{P} = [\mathtt{I} \mid \mathbf{0}]$ and $\mathtt{P}' = [\mathtt{R} \mid -\mathtt{R}\mathbf{c}]$ by assuming that calibration matrices are all identity matrices. For a set of image correspondences as image vectors $\mathbf{v}_i \leftrightarrow \mathbf{v}'_i$, where $\mathbf{v}_i$ and $\mathbf{v}'_i$ are points in the first and second image, respectively, then the L_∞ optimization problem of the estimation of the relative orientation and baseline may be written as follows:

$$\min_{\mathtt{R},\mathbf{X}_i,\mathbf{c}} \left\{ \max_i \left\{ \angle(\mathbf{v}_i, \mathbf{X}_i),\ \angle(\mathbf{v}'_i, \mathtt{R}(\mathbf{X}_i - \mathbf{c})) \right\} \right\}\ , \tag{10.5}$$

where i is the index of the i-th point correspondences and $\angle(\cdot,\cdot)$ is an operator of the angle difference between two vectors. Equation (10.5) represents minimizing maximum errors of all angles between the measured image vectors ($\mathbf{v}_i$ and $\mathbf{v}'_i$) and the estimated image vectors ($\mathbf{X}_i$ and $\mathtt{R}(\mathbf{X}_i - \mathbf{c})$), which are determined by the rotation and the centre of the camera. If the rotation is known in (10.5), then it becomes optimally solvable in L_∞ norm using second-

order cone programming (SOCP) [23] [35]. Hartley and Kahl proposed a branch-and-bound method to perform fast searching over all rotations in order to efficiently solve the relative pose problem for two views.

In their method, the rotations in $\mathbb{R}^3$ are expressed as angle-axis rotations and the parameter space is divided into cubic blocks, which represent a set of similar rotations. This representation can be considered as projecting a quaternion (hemi-)sphere on a plane as an azimuthal equidistant projection. The azimuthal equidistant projection is a particular type of map projection, where all the distances measured from the centre of the map along any longitudinal line are accurate.

By introducing a block D in rotation space, equation (10.5) becomes a restricted optimization problem, which finds the optimal solution in a restricted parameter space. It may be written as follows:

$$\min_{D(\mathtt{R}),\mathbf{X}_i,\mathbf{c}} \left\{ \max_i \left\{ \angle(\mathbf{v}_i, \mathbf{X}_i),\ \angle(\mathbf{v}'_i, \mathtt{R}(\mathbf{X}_i - \mathbf{c})) \right\} \right\} , \tag{10.6}$$

where $D(\mathtt{R})$ is a cubic block representing simliar rotations around $\mathtt{R}$.

Given the block D, we can calculate the minimum error for the cost function. We divide the block D into smaller blocks and examine the minumum errors of each divided smaller block. If there exists a small block that has an error less than the current error of D, the small block is selected as the best candidate and is subdivided into smaller blocks to search over the rotation space. This process is repeated until eventually the size of the block results in a rotation of the desired resolution. This is a simple description of the branch-and-bound algorithm. Therefore, the feasibility of he problem stated in (10.6) is tested as follows:

$$\begin{array}{ll} \textbf{Do there exist} & D(\mathtt{R}), \mathbf{c} \textbf{ and } \mathbf{X}_i \\ \textbf{such that} & \angle(\mathbf{v}_i, \mathbf{X}_i) < \epsilon_{\min} \\ \textbf{and} & \angle(\mathbf{v}'_i, \mathtt{R}(\mathbf{X}_i - \mathbf{c})) < \epsilon_{\min} , \end{array} \tag{10.7}$$

where $\epsilon_{\min}$ is the L_∞ error. The above equation (10.7) cannot be solved instantly. Therefore, by fixating the rotation and with a weaker bound, the weaker but solvable problem can be

defined as follows:

$$\begin{aligned} &\textbf{Do there exist} && \mathbf{c} \textbf{ and } \mathbf{X}_i \\ &\textbf{such that} && \angle(\mathbf{v}_i, \mathbf{X}_i) < \epsilon_{\min} \\ &\textbf{and} && \angle(\mathbf{v}'_i, \bar{\mathtt{R}}(\mathbf{X}_i - \mathbf{c})) < \epsilon_{\min} + \sqrt{3}\sigma \ , \end{aligned} \tag{10.8}$$

where $\bar{\mathtt{R}}$ is the rotation at the centre of cube D and σ is the half-side length of D. Equation (10.8) uses a zero-th order approximation for the rotations in the region D of the rotation space. The details of the term $\sqrt{3}\sigma$ in the last constraint are discussed in [21]. The algorithms for this branch-and-bound method are described in algorithm 5 and function 6. The proof of the feasibility test will be discussed in detail later in chapter 10.

Algorithm 5: Search optimal rotation in L_∞ across two views.

Input: Matched image vectors $\mathbf{v}$ and $\mathbf{v}'$ and initial rotation matrix R
Output: Estimated rotation R

```
// An initial minimum error is obtained
```
Given an initial rotation matrix R*, find the minimum error* ϵ_{best} *by testing the feasibility. In order to do this, refer to the algorithm* `FindMinError`

```
// Search over the rotation space with the minimum error
   ε_best
```
Subdivide the rotation space into a few cubes (for instance, $5 \times 5 \times 5$ cubes), and place them in a queue
repeat
    Get a rotation cube $D(R)$ from the queue
    Test the feasiblity given the rotation cube $D(R)$ using `TestFeasibility`
    if *feasible* **then**
        if *the rotation cube* $D(R)$ *is too small to be subdivided* **then**
            continue
        else
            Subdivide the rotation cube $D(R)$ into smaller cubes and put them into the queue for the next search
            Try for a better solution by testing the feasibility calling a function `TestFeasibility` with the current rotation cube $D(R)$
            if *feasible* **then**
                The best rotation is determined so far by calling a function `FindMinError`
until *no more rotation cubes are available*

Function `FindMinError`

Input: Matched image vectors $\mathbf{v}$ and $\mathbf{v}'$, rotation matrix $\mathtt{R}$, and errors ϵ_0 and ϵ_1
Output: Minimum error

```
while (maxError − minError) > Resolution do
    midError = (maxError + minError)/ 2.0
    Test feasibility by calling TestFeasibility with the current error midError
    if feasible then
        maxError = midError
    else
        minError = midError
Return maxError as the minimum error
```

10.2 Branch-and-bound algorithm

The branch-and-bound algorithm is used to find an optimal solution in L_∞ norm [21, 43, 36]. As given by Hartley and Kahl in [21], the branch-and-bound algorithm for essential matrix estimation finds the optimal rotation by dividing the space of all rotation into several blocks and testing them one by one to find which one gives the best solution. Rotation space is represented as a 3-dimensional space using the angle-axis representation of a rotation. As the algorithm progresses, the blocks may need to be subdivided into smaller blocks in order to get a more accurate answer. Ultimately after a finite number of steps, one can find the optimal rotation, and hence translation within any required degree of accuracy.

The key to the branch-and-bound technique is a method of bounding the cost associated with the rotations within a block. Let $\hat{\mathtt{R}}_0$ be the rotation represented by the centre of a block in rotation space, and let r represent the maximum radius of the block (measured in radians). Since the translational part of the motion may be computed optimally (in L_∞ norm) once the rotation is known, we might find this optimal solution assuming the rotation $\hat{\mathtt{R}}_0$, and compute the best residual δ (namely the maximum reprojection error, also measured in radians) over all possible choices of translation. Now the key point is that for all other rotations $\mathtt{R}$ in the rotation block of radius r, the best residual is bounded below by $\delta + r$ (see [21]).

Now, suppose that $\delta_{\min}$ is a best residual found so far in the search, we ask the following question. Is it possible to find a solution with rotation assumed equal to $\hat{\mathtt{R}}_0$ that has residual

less than $\delta_{\min} + r$. If the answer is no, it means that no rotation inside the current rotation block can beat the best residual $\delta_{\min}$. In this case, we do not need to consider the current block any further. If on the other hand the answer is yes, or possibly, then the result is inconclusive. In this case, we subdivide the rotation block by dividing into 8 subblocks, and keep them for future consideration. This method is guaranteed to find the optimal rotation, and hence translation within any desired bound within a finite number of steps.

The main computation in the method just described is, for each block we need to answer a feasibility question: is it possible with rotation $\hat{\mathtt{R}}_0$ to find a solution with residual less than $\epsilon = \delta_{\min} + r$? We will see that this feasibility problem can be answered very efficiently using LP.

This LP problem arises in the following way. It will be shown that each point correspondence (before and after the motion) must constrain the translation vector of the motion to lie in a wedge of space bounded by a pair of planes. The placement and angle of this wedge depends on the value of ϵ just defined. The feasibility problem has a positive answer if the set of all these wedges (one wedge for every point correspondence) has a common intersection. This is a standard LP problem, and may be solved quickly and efficiently.

10.3 Theory

We now give more details of the method given above. We assume a rotation $\hat{\mathtt{R}}$ is given, and our task is to find whether there exists a solution to the motion problem with residual less than a given value ϵ.

Single camera constraints. Let $\mathbf{x} \leftrightarrow \mathbf{x}'$ be a pair of matched points observed in one of the cameras. These represent direction vectors expressed in a coordinate frame attached to the camera rig. Knowing (or rather hypothesizing) the rotation, we may transform one of the vectors so that they are both in the same coordinate system. Therefore, define $\mathbf{v} = \hat{\mathtt{R}}\mathbf{x}$ and $\mathbf{v}' = \mathbf{x}'$. These two vectors and the translation vector must now satisfy the coplanarity condition $\mathbf{t}^\top(\mathbf{v} \times \mathbf{v}') = 0$ which specifies that the three vectors involved are coplanar. This

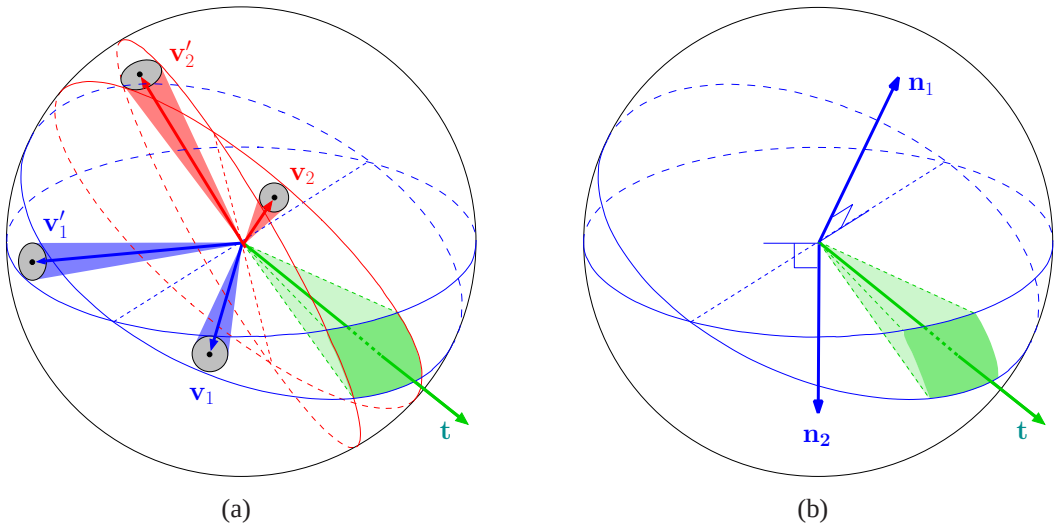


Figure 10.1: *(a) Translation direction* $\mathbf{t}$ *exists in a region of intersections (shaded as green) of half-spaces bounded by planes which are tangent to two cones having axes* $\mathbf{v}_i$ *and* $\mathbf{v}'_i$*. Two matched pairs of points* $\mathbf{v}_1 \leftrightarrow \mathbf{v}'_1$ *and* $\mathbf{v}_2 \leftrightarrow \mathbf{v}'_2$ *give the two intersections of two wedges. The intersection of the two wedges is a polyhedron containing the translation direction* $\mathbf{t}$*. (b) The two normals of the two half-spaces.*

obviously places a constraint on the vector $\mathbf{t}$.

However, we do not expect this constraint to be satisfied exactly for all point correspondences. Rather, we wish to know if it may be satisfied within a given error bound ϵ. A technical detail discussed in [21] allows us to specify different bounds ϵ and ϵ' on the two points. This is not necessary to follow the argument further, but we will assume that $\mathbf{v}$ and $\mathbf{v}'$ are allowed different error bounds ϵ and ϵ'. If we allow $\mathbf{v}$ and $\mathbf{v}'$ to be perturbed in this way, then this means they must lie inside cones of radius ϵ and ϵ' respectively as shown in Figure 10.1(a).

The translation direction $\mathbf{t}$ must lies inside a wedge bounded by planes tangent to the two cones. The two normals of these planes are shown in Figure 10.1(b). For several matched points, the translation direction must lie inside all such wedges.

To solve the feasibility problem, we need to express the normals to the planes in terms of $(\mathbf{v}, \epsilon)$, and $(\mathbf{v}', \epsilon')$. Then answering the feasibility problem is equivalent to solving the LP problem. We give the formulas for the normals below, without full details.

As shown in Figure 10.2, let us assume that angles α, β and ϵ are the angle between two

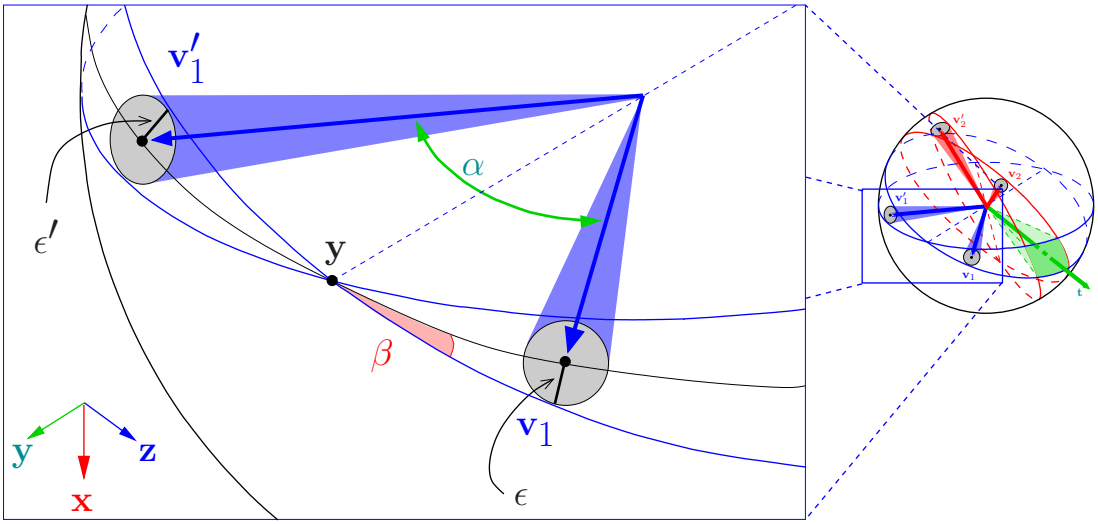


Figure 10.2: *The angle β, between the planes which are bi-tangent to two cones and the plane containing the axes $\mathbf{v}_1$ and $\mathbf{v}'_1$ of the two cones, is determined by the angle α, ϵ and ϵ' where α is the angle between $\mathbf{v}_1$ and $\mathbf{v}'_1$, and both ϵ and ϵ' are the angle errors at measured image point coordinates of matched points. The vectors $\mathbf{x}$ and $\mathbf{z}$ are given by $\mathbf{v}_i \times \mathbf{v}'_i$, and $\mathbf{y} \times \mathbf{x}$, respectively, and the vectors $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ construct a basis of a coordinate system.*

axes of cones, the angle between bi-tangent planes and the cones, and radius error of matched points, respectively. Let $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ be vectors given by two cones $\mathbf{v}$ and $\mathbf{v}'$ as shown in Figure 10.2.

The vectors $\mathbf{x}$ and $\mathbf{z}$ are determined by the axes of two cones $\mathbf{v}$ and $\mathbf{v}'$, and by the vector $\mathbf{y}$ where two great circles meet as shown in Figure 10.2. The vector $\mathbf{y}$ is derived as follows:

$$\mathbf{y} = \frac{\sin(\epsilon)\mathbf{v}' + \sin(\epsilon')\mathbf{v}}{\sin(\beta)\sin(\alpha)} , \tag{10.9}$$

where β is the angle between the planes bi-tangent to two cones and the plane containing the axes of the two cones as illustrated in Figure 10.2. This angle β is given by (see Appendix)

$$\sin^2\beta = \frac{\sin^2(\epsilon) + 2\sin(\epsilon)\sin(\epsilon')\cos(\alpha) + \sin^2(\epsilon')}{\sin^2(\alpha)} , \tag{10.10}$$

where α, ϵ and ϵ' are shown in Figure 10.2.

The vectors $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ form a basis for a coordinate system and serve to build equations

of normals for the two half-spaces. From the work of [21], given a pair of matched cones on $\mathbf{v}_i \leftrightarrow \mathbf{v}'_i$, we derive the two normals $\mathbf{n}_1$ and $\mathbf{n}_2$ of half-spaces as follows:

$$\mathbf{n}_1 = \sin(\beta)\mathbf{z} + \cos(\beta)\mathbf{x} \tag{10.11}$$

$$\mathbf{n}_2 = \sin(\beta)\mathbf{z} - \cos(\beta)\mathbf{x}\ . \tag{10.12}$$

These equations provide two normals $\mathbf{n}_1$ and $\mathbf{n}_2$ for planes from a pair of matched points $\mathbf{x} \leftrightarrow \mathbf{x}'$, and eventually will be used to get an intersection of all half-spaces from all matched pair of points. This is an intersection from only one camera, and the existence of the intersection tells us whether a problem is feasible for the optimal essential matrix in one camera. In this chapter, we would like to deal with multiple cameras instead of a single camera to find the optimal rotation and translation.

Multiple cameras. We represent each camera by a sphere centred at the camera centre. Therefore, we have m spheres for an m-camera system. Associated with each sphere, as in Figure 10.1 there is a polyhedral cone with apex positioned at the centre of each camera, formed as the intersection of wedges defined by the point correspondences for that camera. These cones represent the direction of motion of each of the cameras. A correspondence of points in the k-th camera generates a constraint of the form

$$\mathbf{n}^\top(\mathbf{c}'_k - \mathbf{c}_k) \geq 0\ , \tag{10.13}$$

where $\mathbf{c}_k$ is the centre of k-th camera and $\mathbf{c}'_k$ is the centre of k-th camera after the motion. The constraints from different cameras involve different variables, however. To get a set of consistent constraints, we need to transform these cones so that they constrain the final position of a specific chosen one of the cameras, let us say the final position $\mathbf{c}'_1$ of the first camera.

This problem is the same as the triangulation problem considered in [40]. We will see how the cones given by the linear constraints are transformed by the assumed rotation of the camera. This is illustrated in Figure 10.3.

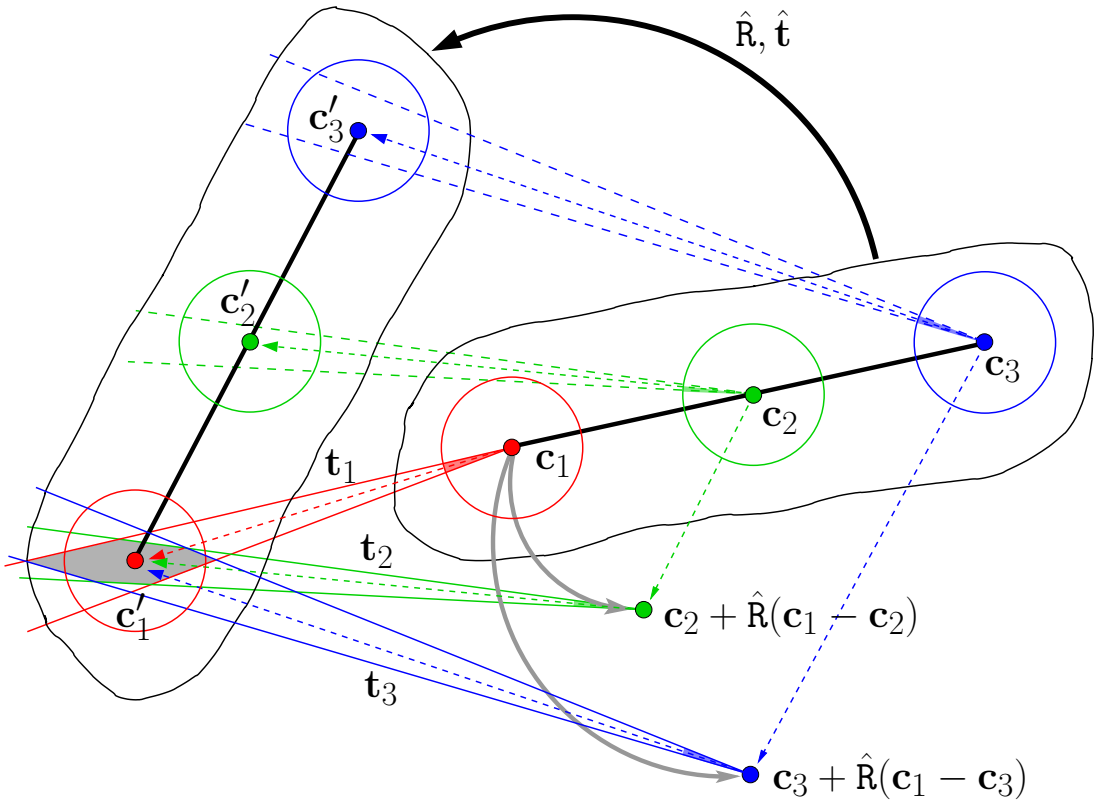


Figure 10.3: *The shaded region is the intersection of three polyhedra located on where each camera sees, $\mathbf{c}'_1$, the centre of the first camera after a rigid motion. The shaded region is a feasible solution of the translation of this multi-camera system.*

To express (10.13) in terms of $\mathbf{c}'_1$ instead of $\mathbf{c}'_k$, we use the following relationship, which may be easily read from Figure 10.3:

$$\mathbf{c}'_1 = \mathbf{c}_k + \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k) + (\mathbf{c}'_k - \mathbf{c}_k)$$

By substituting for $(\mathbf{c}'_k - \mathbf{c}_k)$ in (10.13), we obtain the inequality for multiple camera systems as follows:

$$\begin{aligned} 0 &\leq \mathbf{n}^\top(\mathbf{c}'_k - \mathbf{c}_k) \\ &= \mathbf{n}^\top(\mathbf{c}'_1 - \mathbf{c}_k - \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \\ &= \mathbf{n}^\top\mathbf{c}'_1 - \mathbf{n}^\top(\mathbf{c}_k + \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \ . \end{aligned}$$

This is the specific inequality involving $\mathbf{c}'_1$ after the transformation. Finding a solution satisfying all these inequalities is the same as finding an intersection of all half-spaces.

We find the centre of the first camera after the final motion by an intersection of all wedges defined by all pairs of matched points. In other words, we find a solution to a set of linear constraints by linear programming. More precisely, this feasibility problem is described as follows:

$$\begin{aligned} &\text{Does there exist} && \mathbf{c}'_1 \\ &\quad\text{Satisfying} && \mathbf{n}_{i1}^\top\mathbf{c}'_1 - \mathbf{n}_{i1}^\top(\mathbf{c}_k + \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \geq 0 \\ &&& \mathbf{n}_{i2}^\top\mathbf{c}'_1 - \mathbf{n}_{i2}^\top(\mathbf{c}_k + \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \geq 0 \\ &\quad\text{for} && i = 1, \ldots, N \ , \end{aligned}$$

where $\mathbf{n}_{i1}$ and $\mathbf{n}_{i2}$ are the two normals derived from matched point i and k is the appropriate index of the camera generating the matched point i.

The feasible region is the region of space satisfying all these inequalities. In this problem, it is not important to know the entire polygon, but only to find one particular point of interest. Solving this feasibility problem tells us the position of the centre of the first at the final motion,

and finally it gives us the optimal solution of translation direction vector and its scale value in multi-camera systems.

Feasibility test. All half-spaces from matched pairs serve as inequalities in this LP problem. Given a total of N matched points in m cameras, the number of inequalities is $2N$. Generally, for 5 cameras with 100 points, LP requires to find an intersection of 1,000 half-spaces. If we use only LP to solve this problem, it will take too much computation time.

We introduce a way to reduce the time of computation for LP in this particular problem by testing the feasibility at an earlier stage before solving a full LP problem. The feasibility for multi-camera systems depends on the feasibility of a single camera. If any feasibility observed for one single camera fails, then we do not need to look at feasibilities of other cameras. This observation gives a method to reduce the computation time greatly.

Testing a feasibility for a single camera is done by reducing the number of variables for the translation direction vector to two variables as shown in [21]. This feasibility test for a single camera can be adopted for greater speed of LP in multi-camera systems.

The order of matched points also affect the speed of the feasibility test. A larger angle α between two matched points leads to a narrower wedge in which the translation direction must lie, and gives more chance to finish the feasibility test earlier. Thus, these points should be tested first. In our experiments, using a preemptive feasibility test makes the algorithm 90 times faster than an algorithm without this feasibility test.

Degeneracy. It is important to note that if the motion from one frame to the next has no rotation, then the scale of the translation can not be computed. Because of the independence of the different cameras, there is an overall scale ambiguity, despite having known distances between the cameras. If the rotation is close to zero, the translation will be less reliable.

10.4 Algorithm

Given m calibrated cameras with a total of N matched points in each image, we can transform the matched points into vectors on the surface of a sphere by multiplying the inverse of

the calibration matrix and the inverse of the rotation matrix of each camera. An example of these vectors is illustrated in Figure 10.6. With these simplified image vectors, the problem becomes easier to describe. The algorithm to find the optimal solution of motion of multi-camera systems is written in algorithm 7.

Algorithm 7: Optimal L_∞ Motion in Multi-Camera.

Input: Given m calibrated cameras with N matched points, $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$.
Output: Estimated optimal rotation and translation with scale.

1 Obtain an initial estimate for the motion by any means (a random guess if necessary) and compute an initial estimate $\delta_{\min}$ for the minimal residual. Then carry out a branch-and-bound algorithm over rotation space, with the following steps.
2 Select a rotation block and consider its centre as an initial estimate of rotation $\hat{\mathtt{R}}$ in rotation space.
3 Multiply $\hat{\mathtt{R}}$ by $\mathbf{x}$ to get axes of two cones $\mathbf{v} = \hat{\mathtt{R}}\mathbf{x}$ and $\mathbf{v}' = \mathbf{x}'$.
4 Let $\epsilon = \delta_{\min} + r$, where r is the radius of the rotation block. Next determine whether there is a solution with rotation $\hat{\mathtt{R}}$ and residual less than ϵ by the following steps.
5 From the two cones about $\mathbf{v}$ and $\mathbf{v}'$ with half vertex-angle errors ϵ, compute two normals $\mathbf{n}_1$ and $\mathbf{n}_2$ from (10.12). Do this for all correspondences $\mathbf{v} \leftrightarrow \mathbf{v}'$.
6 Transform the two half-spaces to obtain inequality equations $\mathbf{n}_i^\top \mathbf{c}'_1 - \mathbf{n}_i^\top(\mathbf{c}_k + \hat{\mathtt{R}}(\mathbf{c}_1 - \mathbf{c}_k)) \geq 0$.
7 Solve linear programming with the constraints.
8 If it is a feasible problem, then divide the selected rotation block into subblocks, and queue for further processing; otherwise discard the rotation block.
9 Repeat until we meet a desired error, then return the estimated rotation and translation

10.5 Experiments

Two experiments are conducted on synthetic and real data to show robustness and applications. A comparison with other method is presented to show improved accuracy of our proposed method.

10.5.1 Synthetic data experiments

A synthetic data set has four cameras with 50 image points randomly located in space. A total of 200 points are projected onto four image planes, and the system of four cameras is moved by a rigid motion of rotation and translation. The 200 points are also projected onto another four

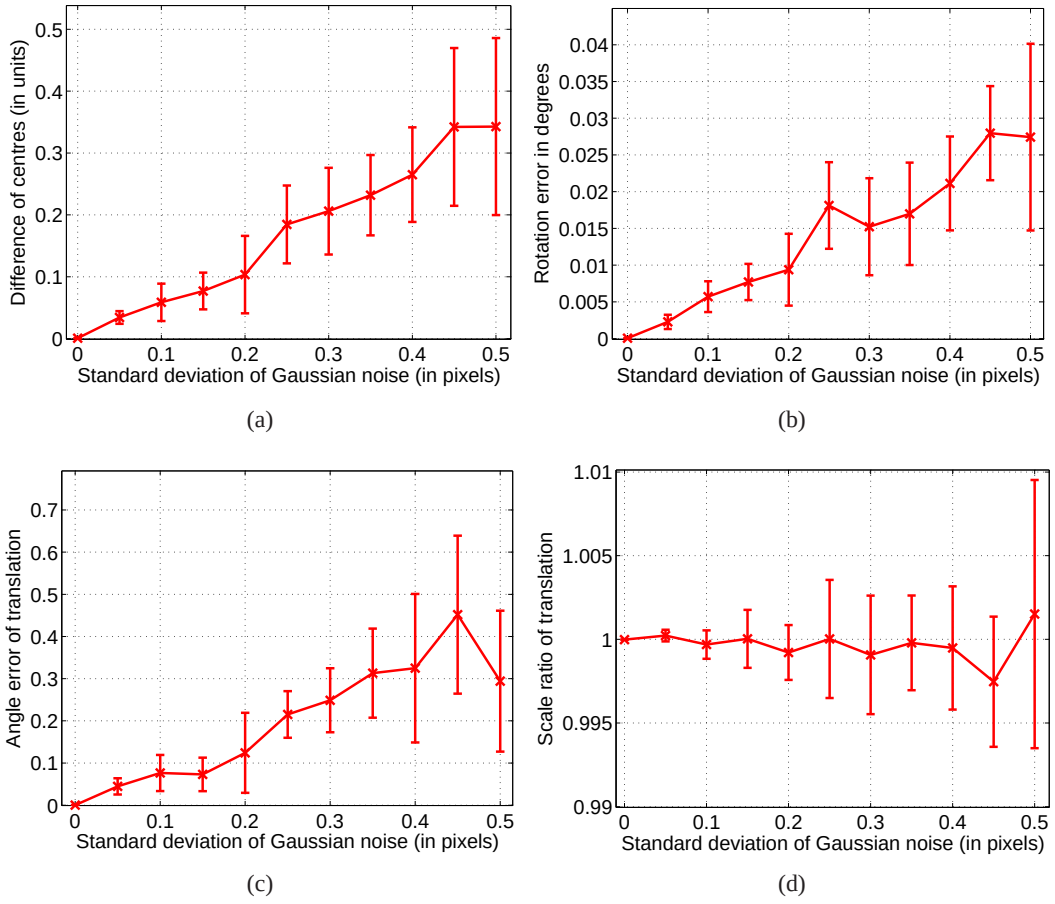


Figure 10.4: ***Result of the synthetic data experiments.*** *Normally distributed noise with standard deviation parameter σ is added to image coordinates in pixel units. (a) The angle difference between the estimated rotation and the true rotation of cameras, (b) The angle difference between the estimated translation direction and the true translation direction of cameras, (c) The distance between the estimated centres and the true centre of cameras and (d) the scale ratio between the estimated translation and the true translation are compared by varying noise parameters σ from 0 to 0.5 which means about 99.7% of the image points have errors from 0 to ± 1.5 pixels because of 3σ.*

image planes of cameras at the final motion. When we process this synthetic data to estimate the motion by using our method, the central processing unit (CPU) time of computation is about 3.5 seconds in a standard Intel Core 2 CPU PC based on 32-bit instructions and a single process. The implementation is written in C++ with GNU Linear Programming Kit (GLPK) [15]. As shown in Figure 10.4, several experiments are conducted 10 times on the same synthetic data by increasing noise parameters in pixels, and the distance error of centres is compared with the ground truth and its mean values are shown.

We have examined the performance comparison with the method of chapter 9, which we call "E+SOCP" in this chapter, which uses a single essential matrix and SOCP to estimate the motion of multi-camera systems. As seen in Figure 10.5, our proposed method gives a better estimation for rotation and translation than E+SOCP.

10.5.2 Real data experiments

As a real example of multi-camera systems, we have used Point Grey's Ladybug^{TM}2 [32]. Six images are captured at each camera, and feature points on the images are extracted and tracked by the Kanade-Lucas-Tomasi (KLT) tracker [47] through image sequences. Outliers in the tracked features are removed using RANSAC [13]. We transform these tracked features to image vectors on a sphere by multiplying the inverse calibration matrix and the inverse rotation matrix in each camera. The image vectors are shown in Figure 10.6. They are used in our algorithm to obtain the optimal solution of the rotation and translation in the 6-camera system of LadyBug^{TM}. It is important to note that we are not dealing with omnidirectional cameras but a multi-camera system.

10.5.2.1 First real data set

The data is collected in the same way as for chapter 9. The 6-camera system is moved on a piece of paper and the position is marked on the piece of paper. The motion of the 6-camera system, LadyBug, is a circular-like motion for 95 frames. We have selected key-frames every 5 frames from the image sequences. The estimated motion of the 6-camera system using our

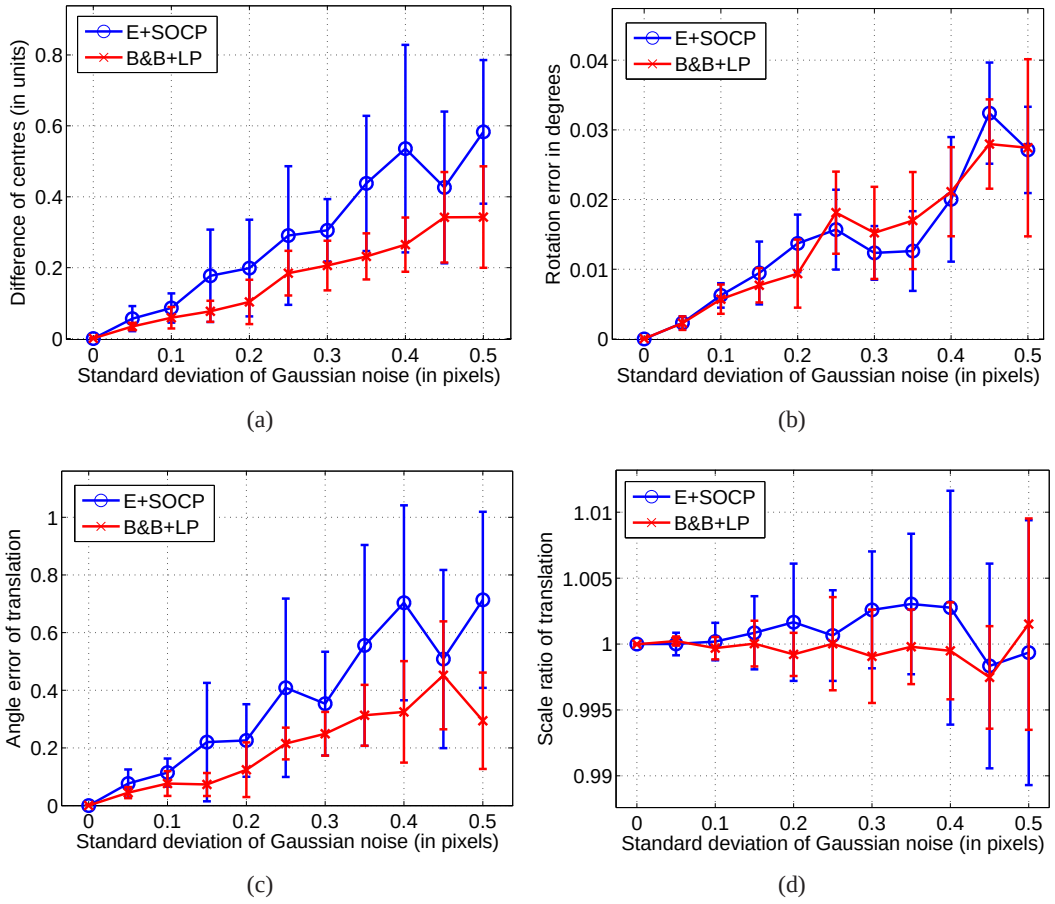


Figure 10.5: *Comparison of two methods which are the SOCP based on the single essential matrix method by [40]. (indicated as blue lines, "E+SOCP") and our proposed method based on branch-and-bound algorithm with LP (indicated as red lines, "B&B+LP"). (a) The difference between the true position of camera and the estimated position of the camera at the final motion. (b) Angle error of estimated rotation. (c) Angle error of estimated translation direction. (d) Scale error of estimated translation. The "B&B+LP" method gives more accurate position of camera though it has under-estimation of rotation and translation direction compared with the "E+SOCP" method. The difference of the errors is less than 1 degrees, so it is minimal. The less scale error of translation in the "B&B+LP" method shows why it estimates better position of cameras at the final position.*

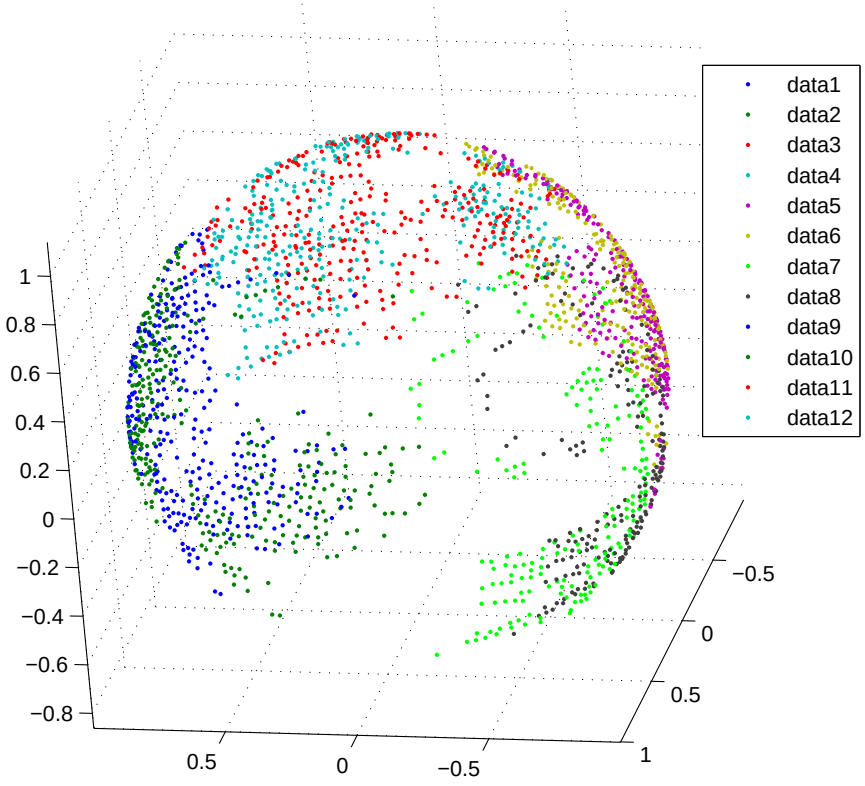


Figure 10.6: ***Image vectors on a sphere from LadyBug^{TM} camera.*** *These image vectors represent matched points which are transformed by the inverse of calibration matrix and the inverse of rotation matrix for our simplified model. Data 1 and 2 are from the first camera, data 3 and 4 are from the second camera, data 5 and 6 are from the third camera, and so on.*

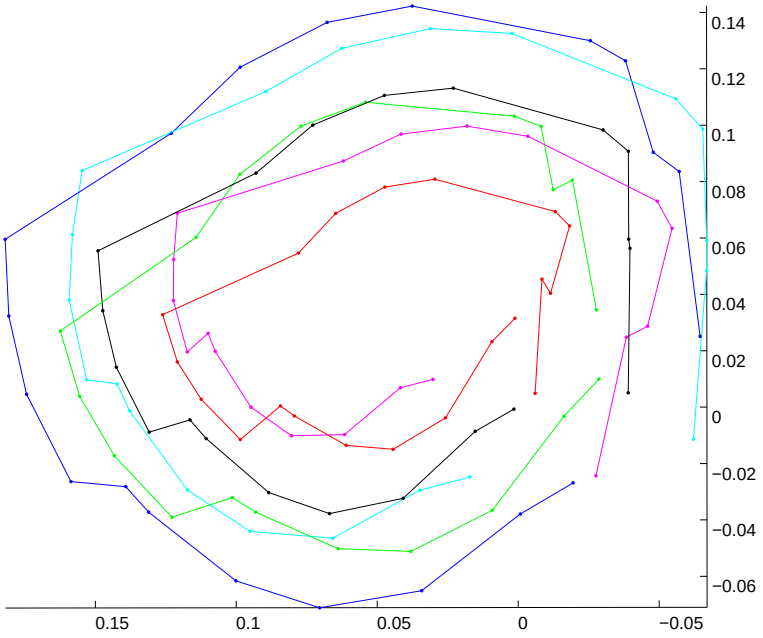


Figure 10.7: ***Path of cameras from a top view.*** *Each point in coloured lines represents the centre of six cameras in the system.*

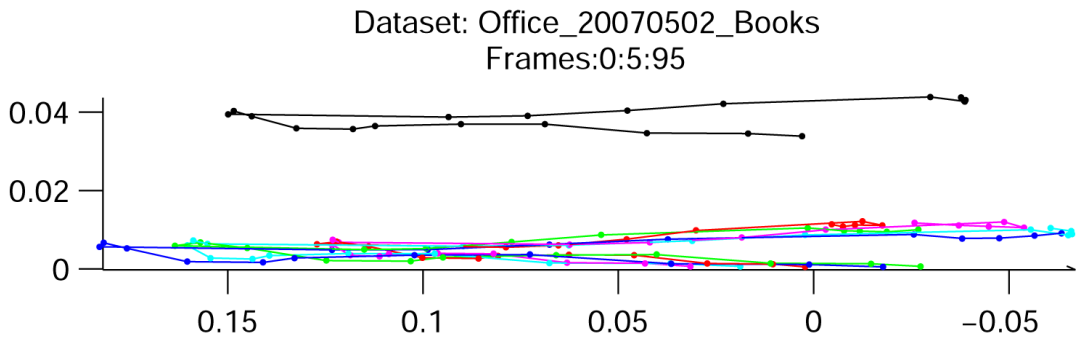


Figure 10.8: ***Path of cameras from a side view.*** *Each point in coloured lines represents the centre of six cameras in the system. To see this, note that a black coloured line is the camera on top of the base unit.*

proposed method is shown in Figure 10.7 and Figure 10.8. The purpose of this experiment is to see how estimated motion is similar to the circular-like motion because the camera is moved randomly and the ground truth for this motion is not measured. In the next experiment, we will look at how the motion is accurately estimated by locating the cameras at the pre-determined path.

10.5.2.2 Second real data set

We test the algorithm also on the data described in chapters 8 and 9. The configuration of the camera setup is shown in Figure 10.9, and the images taken by the six cameras are shown in Figure 10.10.

Analysis of accuracy. Before we proceed with this particular "∞-shape" like motion of cameras, first, we would like to analyze how much pixel errors in images affect the accuracy of estimation for rotations and translations. For a better analysis and simulation of the experimental environment, we used the same data set which has all the measured trajectories of the Ladybug™2 camera with rotations and translations, and we also used the camera calibration information of the Ladybug™2 camera. With this measured ground truth and the Ladybug™2 camera calibration information from the real experimental setup, the estimation of translations and rotations is simulated. The computed motion of cameras is shown in Figure 10.11.

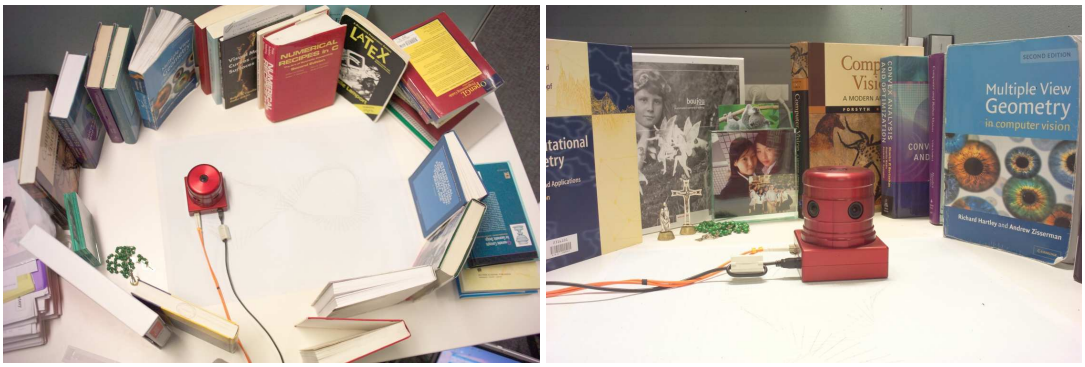


Figure 10.9: *Experiment setup. A LadyBug camera is placed on a piece of paper which has 1mm grids and it is surrounded by books. A trajectory of cameras is marked on the paper. Total 108 positions of ground truth are measured from the marked positions.*

Figure 10.10: ***Six images captured at each camera of LadyBug.*** *Five cameras (camera id 0 to 4) are placed to look horizontally view, and the last one (camera id 5) is located for a top view (From left to right order). There are only small overlapped fields of view across cameras.*

Results. For 108 images, the motion of the 6-camera system is estimated and the results are shown and compared with the results of the "E+SOCP" method in Figure 10.14. The graph in Figure 10.14 shows that the estimated rotation and translation by our proposed method are more accurate than the estimated motion by the method uses SOCP with essential matrix from a single camera. The estimated trajectories of cameras are superimposed the ground truth of the measured trajectories of the cameras in Figure 10.12. Histograms of translation and rotation errors of the simulated motion are shown in Figure 10.13. These analysis shows that the translation direction is sensitive to noise on image coordinates. The estimated trajectories of the Ladybug^{TM}2 camera and its consisting 6 cameras with the marker are shown in Figure 10.15. It shows the "∞-shape" path from the positions of the marker.

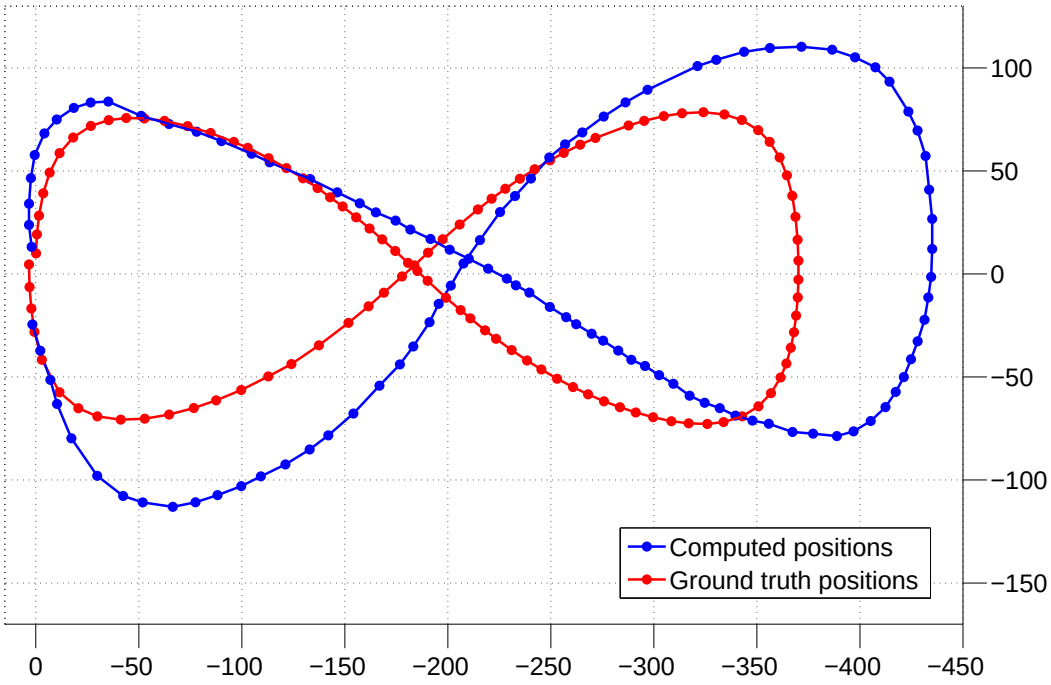


Figure 10.11: *Computed motion of cameras from synthetic data with the Ladybug^{TM}2 camera calibration information and the ground truth positions. The computed motion is indicated as blue lines and the ground truth positions of cameras are drawn with red lines. The computed motion is generated with 0.1 standard deviation of the normal distribution for noises in image coordinates by pixel units. The overall scale of the computed motion is expanded compared with ground truth, perhaps largely due to the scale ambiguity caused by small rotation between frames. Nevertheless, note that the computed path almost closes accurately. This suggests a systematic bias towards overestimating translation characteristic of Maximum Likelihood estimation.*

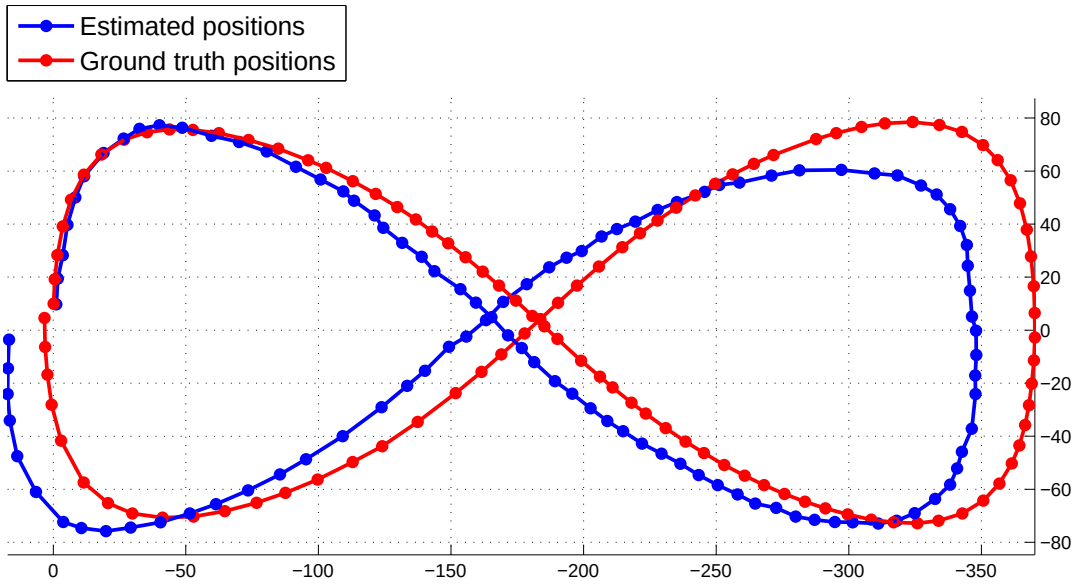
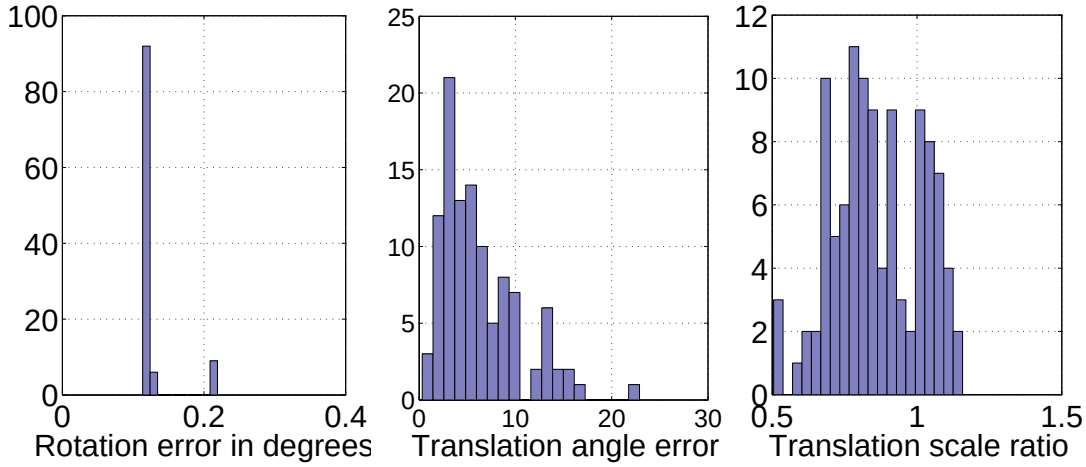
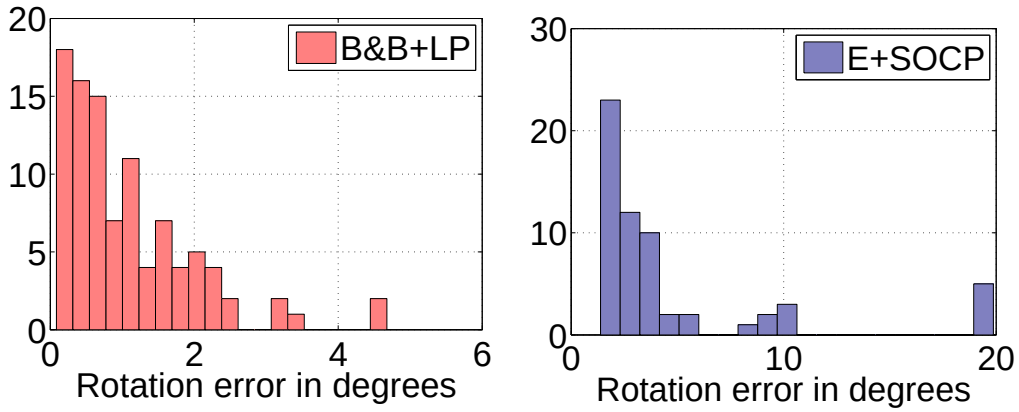


Figure 10.12: *Top view of the estimated trajectories of cameras and the ground truth of the cameras from frame 0 to 108. The estimated trajectories are indicated as red lines with dots on their positions of the cameras. The ground truth is illustrated as blue lines with its positions of the cameras. The starting position of the cameras is the left middle point which is (0, 0, 0) in the coordinates. There is a jittering or drift movement in the estimated motion because of accumulated errors over frames.*



(a) (b) (c)

Figure 10.13: ***Histograms of rotation and translation errors on the simulated motion.*** *The simulated motion is generated with 0.1 standard deviation of normal distribution as noises on the image coordinates. (a) Histogram of rotation errors. (b) Histogram of translation direction errors. (c) Histogram of translation scale errors. These shows the translation direction errors are sensitive to the noises.*



(a) (b)

Figure 10.14: *(a)Histogram of rotation error by our proposed method "B&B+LP" method. It shows 1.08 degrees of the mean and 0.83 degrees of the variance. (b)Histogram of rotation error by the "E+SOCP" method which is based on the essential matrix from a single camera and SOCP by [40]. It shows 4.73 degrees of the mean and 25.61 degrees of the variance. The proposed "B&B+LP" method estimates the rotation better than the "E+SOCP" method in real data experiments.*

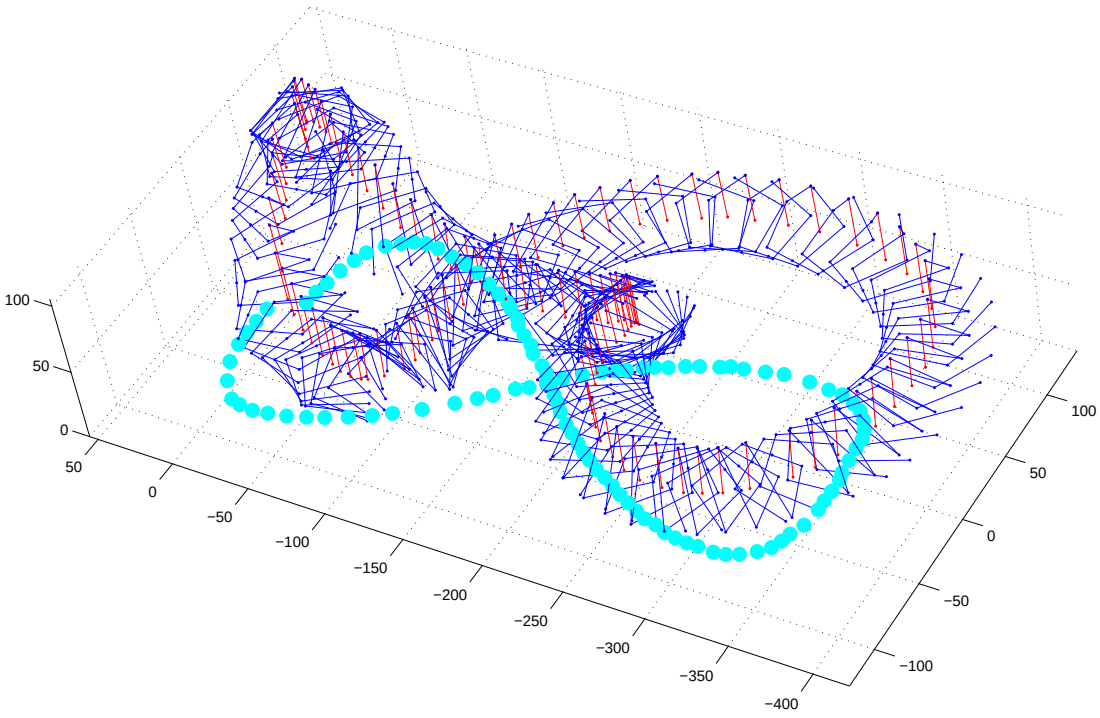


Figure 10.15: *The top-side view of the path of the 6 cameras (blue and red lines) and marker (cyan dots).*

10.6 Conclusion

An optimal solution of motion for multi-camera systems under L_∞ norm is presented, and a feasibility test of linear programming for the multi-camera systems reduced the computation time of the problem significantly. The algorithm is optimal under L_∞ through all steps of the algorithm. Analysis of simulated motion showed that this algorithm is robust to estimate rotation angles and translation scale values (at least when the rotation is not too small) when there is noise in the image coordinates. However, we found that the estimate of the direction of translation is sensitive to the noise in the images.

Chapter 11

Conclusions and discussions

Camera motion estimation for multi-camera systems is studied for an omnidirectional camera, non-overlapping multi-camera rigs and general imaging models.

An omnidirectional camera is first used to estimate the relative motion of the omnidirectional camera, which is an example of a multi-camera system. The translational motion of an omnidirectional camera is estimated and the result is improved by constrained minimization across three views.

As a second example, general imaging models are used to estimate the relative motion of a generalized camera using a linear method. To our knowledge, this linear method and its experiments have been studied and performed for the first time in this thesis. The results show that this linear method is capable of estimating the relative motion in real time and is used as an initial estimate for other nonlinear methods.

Third, linear methods for non-overlapping multi-camera rigs are presented to estimate the relative motion of multi-camera systems. We used an 8-camera system that uses 8 cameras on a vehicle to estimate the motion of an 8-camera system. A linear method using 6 points is presented and critical motions for which the estimation cannot be obtained are studied.

Finally, nonlinear methods for multi-camera systems are presented using SOCP and LP with a branch-and-bound algorithm. These methods give an optimal solution under L_∞ norm error. The SOCP method was the first method provided a global solution to the motion estimation for multi-camera systems. We showed that the motion estimation problem of multi-camera systems is the same as the triangulation problem of multiple views. The second LP with a branch-and-bound algorithm provides a global solution to the motion estimation of

multi-camera systems. The branch-and-bound algorithm is used to search a rotation over the rotation space, and it reduces the time to search rotation by testing the feasibility of the LP problem.

All the six degrees of freedom of the rigid motion of multi-camera systems can be estimated. Particularly, the scale of translation is able to be obtained. In this work, we gave a new direction to camera motion estimation for multi-camera systems.

In this work, we have found that the best method estimating the relative motion of multi-camera systems is LP+B&B method, which is described in chapter 10. This method gives the most accurate estimated position of cameras compared to other two methods (linear method and E+SOCP method) shown in chapter 8 and chapter 9. Because, in this LP+B&B algorithm, an error term to be minimized is based on the convex optimization techinques, it gives us a guarantee to obtain a globally optimal solution under L_∞ norm. It is the main reason that this LP+B&B method gives better results than the linear method and E+SOCP mehod.

The shortcoming of the LP+B&B method, including E+SOCP method, is the time of computation. They usually run slower than the linear method. Because they rely on LP or SOCP, the complexity of the algorithm depends on the number of points and the number of cameras. However, the linear method is generally faster and easy to implement, so it is a good method to be used in real-time applications or to provide an initial estimate for non-linear methods. Another thing is that the relative motion of multi-camera systems cannot be estimated if the motion is critical, as described in chapter 7. In the case of the critical motion, no methods provided in this thesis can estimate all the six degrees of freedom of the motion. Only five (except the scale of the translation) can be estimated.

Future works may include many studies on feature matching for multi-camera systems, self-calibration of non-overlapping multi-camera rigs and real-time implementation of the motion estimation of multi-camera systems using graphic processor unit (GPU) programming. In particular, GPU programming may reduce the time of computations to solve the motion of multi-camera systems.

There exists an unsolved problem such as investigating the motion estimation of multi-

camera systems across three views. Using the trifocal tensor and the global rotation-space-searching method, the results of the motion estimation of multi-camera systems may be significantly improved.

Appendix

A.1 Proofs

We re-introduce the proof of the equation (10.9) given in section 10.3, which is shown in [21]. By symmetry, $\mathbf{y}$ is coplanar with $\mathbf{v}$ and $\mathbf{v}'$. We write $\mathbf{y} = a\mathbf{v} + b\mathbf{v}'$ where $a > 0$ and $b > 0$. Taking cross products with vectors $\mathbf{v}$ and $\mathbf{v}'$ and expressing the length of the resulting vector in two ways leads to

$$\sin(\gamma) = \|\mathbf{y} \times \mathbf{v}\| = \|b\mathbf{v} \times \mathbf{v}'\| = b\sin(\alpha)$$
$$\sin(\gamma') = \|\mathbf{y} \times \mathbf{v}'\| = \|a\mathbf{v} \times \mathbf{v}'\| = a\sin(\alpha)$$

where γ and γ' are the angles separating $\mathbf{y}$ from $\mathbf{v}$ and $\mathbf{v}'$ respectively. From this we obtain

$$\mathbf{y} = \frac{\sin(\gamma')}{\sin(\alpha)}\mathbf{v} + \frac{\sin(\gamma)}{\sin(\alpha)}\mathbf{v}' \tag{11.1}$$

We do not yet know the angles γ and γ'. At this point, we need an elementary result from spherical trigonometry (see 11.1).

Lemma 4. *Let ABC be a spherical triangle in which C is a right-angle, and the edges be arcs of length a, b and c respectively, on a unit sphere. Then* $\sin B = \sin(b)/\sin(c)$.

This compares with the formula for a Euclidean triangle in which $\sin B = b/c$. We do not intend to prove this lemma.

Now, applying this to the triangles shown in Figure 11.2 we see that

$$\sin(\beta) = \frac{\sin(\epsilon)}{\sin(\gamma)} = \frac{\sin(\epsilon')}{\sin(\gamma')}$$

Substituting for $\sin(\gamma)$ and $\sin(\gamma')$ in (11.1) gives the required formula (10.9) for $\mathbf{y}$.

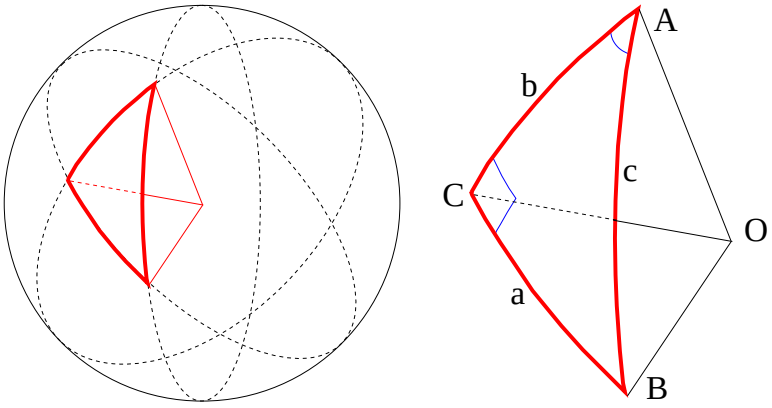


Figure 11.1: *The formula for the* sin *of an angle in a right-angled spherical triangle formed by arcs of great circles is* $\sin(B) = \sin(b)/\sin(c)$ *where* b *and* c *are the lengths of the arcs on the surface of the unit sphere.*

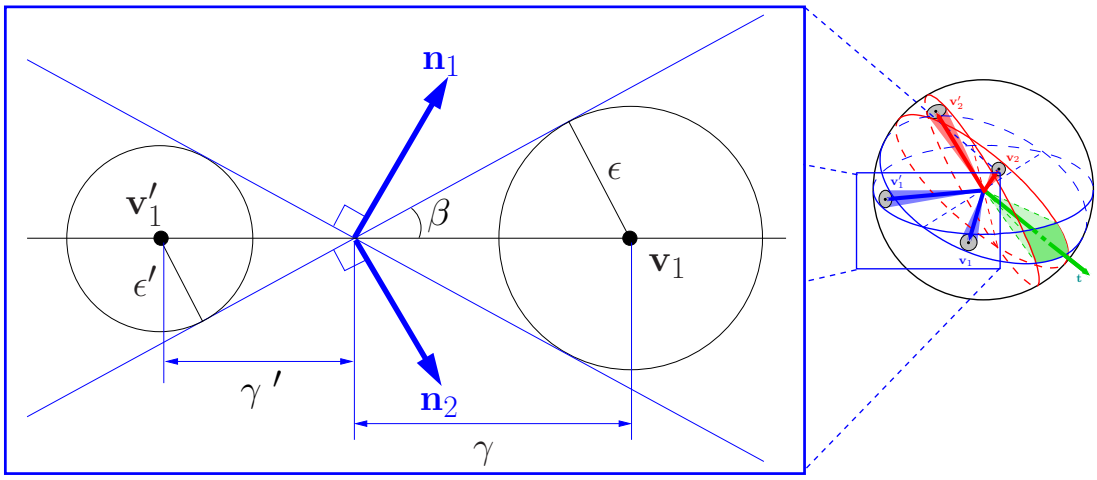


Figure 11.2: *Computing the angle between the plane bi-tangent to two cones and the plane containing the axes of the two cones. See the text for the computation.*

Next we wish to prove the formula (10.10) for β. This is simply a result of the fact that $\mathbf{y}$ is a unit vector. Computing the norm of $\mathbf{y}$ given by (10.9) yields

$$\|\mathbf{y}\|^2 = \mathbf{y}^\top \mathbf{y} = \frac{\sin^2(\epsilon) + 2\sin(\epsilon)\sin(\epsilon')\cos(\alpha) + \sin^2(\epsilon')}{\sin^2(\alpha)\sin^2(\beta)} \ .$$

from which the result follows:

$$\begin{aligned} \frac{\sin^2(\epsilon) + 2\sin(\epsilon)\sin(\epsilon')\cos(\alpha) + \sin^2(\epsilon')}{\sin^2(\alpha)\sin^2(\beta)} &= 1 \\ \frac{\sin^2(\epsilon) + 2\sin(\epsilon)\sin(\epsilon')\cos(\alpha) + \sin^2(\epsilon')}{\sin^2(\alpha)} &= \sin^2(\beta) \ . \end{aligned}$$

Finally, the equation (10.12), namely $\mathbf{n}_i = \sin(\beta)\mathbf{z} \pm \cos(\beta)\mathbf{x}$ is simply a statement that the angle between the tangent plane and the z-axis is β.

A.2 Skew-symmetric matrix

For a 3-vector $\mathbf{t} = (t_1, t_2, t_3)^\top$, a skew-symmetric matrix is defined as

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

For any 3-vector $\mathbf{a}$ and $\mathbf{b}$, the cross product of $\mathbf{a}$ and $\mathbf{b}$ satisfies

$$\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_\times \mathbf{b} \qquad \text{and} \qquad (\mathbf{a} \times \mathbf{b})^\top = \mathbf{a}^\top [\mathbf{b}]_\times \ .$$

For any non-singular 3×3 matrix $\mathtt{R}$ and a 3-vector $\mathbf{t}$, we have the following equalities:

$$\begin{aligned} \mathtt{R}[\mathbf{t}]_\times &= [\mathtt{R}\mathbf{t}]_\times \mathtt{R} \\ [\mathtt{R}\mathbf{t}]_\times &= \mathtt{R}[\mathbf{t}]_\times \mathtt{R}^\top \\ [\mathbf{t}]_\times \mathtt{R} &= \mathtt{R}[\mathtt{R}^\top \mathbf{t}]_\times \ . \end{aligned}$$

Bibliography

1. 2d3 Limited. 2d3 boujou. `http://www.2d3.com`, 2005.

2. Applanix: A Trimble Company. POSLV (position and orientation system land vehicles). `http://www.applanix.com`.

3. F. Baeschlin and M. Zeller. *Lehrbuch der Stereophotogrammetrie, mit 2 beitrgen von Heinr (Text-book of Stereophotogrammetry, with 2 contributions of Heinr)*. Zürich, Orell Füssli, 1934.

4. S. Baker and S. K. Nayar. Single viewpoint catadioptric cameras. In R. Benosman and S. B. Kang, editors, *Panoramic Vision: Sensors, Theory, Applications*. Springer-Verlang, 2001.

5. S. Boyd and L. Vanderberghe. *Convex Optimization*. Cambridge University Press, 2004.

6. Breeze Systems. `http://www.breezesystems.com`.

7. M. Byröd, K. Josephson, and K. Åström. Improving numerical accuracy of gröbner basis polynomial equation solvers. In *IEEE International Conference on Computer Vision*, 2007.

8. W. D. Curtis, A. L. Janin, and K. Zikan. A note on averaging rotations. In *IEEE Virtual Reality Annual International Symposium*, pages 377–385, 1993.

9. Digital Air. `http://www.digitalair.com`.

10. O. Faugeras and Q.-T. Luong. *The Geometry of Multiple Images*. MIT Press, 2001.

11. O. D. Faugeras. *Three-Dimensional Computer Vision: a Geometric Viewpoint.* MIT Press, 1993.

12. F. Ferrari, E. Grosso, G. Sandini, and M. Magrassi. A stereo vision system for real time obstacle avoidance in unknown environment. *Intelligent Robots and Systems '90. 'Towards a New Frontier of Applications', Proceedings. IROS '90. IEEE International Workshop on*, pages 703–708 vol.2, Jul 1990.

13. M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

14. J.-M. Frahm, K. Köser, and R. Koch. Pose estimation for Multi-Camera Systems. In *DAGM*, 2004.

15. GNU Project. GNU Linear Programming Kit version 4.9. `http://www.gnu.org/software/glpk/`.

16. G. H. Golub and C. F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

17. V. M. Govindu. Lie-algebraic averaging for globally consistent motion estimation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2004.

18. M. D. Grossberg and S. K. Nayar. A general imaging model and a method for finding its parameters. In *IEEE International Conference on Computer Vision*, pages 108–115, 2001.

19. R. Gupta and R. I. Hartley. Linear pushbroom cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):963 – 975, Sept. 1997.

20. R. Hartley. Photogrammetric techniques for panoramic cameras. In *SPIE93-photogrammetry*, pages 127–139, April 1993.

21. R. Hartley and F. Kahl. Global optimization through searching rotation space and optimal estimation of the essential matrix. In *IEEE International Conference on Computer Vision*, 2007.

22. R. Hartley and F. Kahl. Optimal algorithms in multiview geometry. In *Asian Conference on Computer Vision*, volume 1, pages 13 – 34, Nov 2007.

23. R. Hartley and F. Schaffalitzky. L_∞ minimization in geometric reconstruction problems. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume I, pages 504–509, Washington DC, USA, 2004.

24. R. I. Hartley. Estimation of relative camera positions for uncalibrated cameras. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 579–587, London, UK, 1992. Springer-Verlag.

25. R. I. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593, Oct 1997.

26. R. I. Hartley and T. Saxena. The cubic rational polynomial camera model. In *IUWS*, pages 649 – 653, 1997.

27. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

28. G. Hauck. Neue constructionen der perspective und photogrammetrie. (theorie der trilinearen verwandtschaft ebener systeme, i. artikel.). *Journal für die reine und angelwandte Mathematik*, 95:1–35, 1883.

29. B. Hendys. Hendy's law: Pixels per dollars. `http://en.wikipedia.org/wiki/Image:Hendys_Law.jpg`, 2007.

30. B. K. P. Horn. Relative orientation. *International Journal of Computer Vision*, 4:59–78, 1990.

31. B. K. P. Horn. Relative orientation revisited. *Journal of the Optical Society of America A*, 8:1630–1638, 1991.

32. P. G. R. Inc. Ladybug^{TM}2 camera. `http://www.ptgrey.com`, 2006.

33. S. E. Inc. Sony icx204ak, diagonal 6mm (type 1/3) progressive scan ccd image sensor with square pixel for color cameras. `http://products.sel.sony.com/semi/PDF/ICX204AK.pdf`.

34. K.-H. Jeong, J. Kim, and L. P. Lee. Biologically inspired artificial compound eyes. *Science Magazine*, 312(5773):557–561, April 2006.

35. F. Kahl. Multiple view geometry and the L_∞-norm. In *IEEE International Conference on Computer Vision*, pages 1002–1009, Beijing, China, 2005.

36. F. Kahl, S. Agarwal, M. Chandraker, D. Kriegman, and S. Belongie. Practical global optimization for multiview geometry. *International Journal of Computer Vision*, 2008.

37. R. Kaucic, R. I. Hartley, and N. Dano. Plane-based projective reconstruction. In *IEEE International Conference on Computer Vision*, pages 420–427, 2001.

38. R. Kaucic, R. I. Hartley, and N. Y. Dano. Plane-based projective reconstruction. In *IEEE International Conference on Computer Vision*, pages 420–427, 2001.

39. Q. Ke and T. Kanade. Quasiconvex optimization for robust geometric reconstruction. In *IEEE International Conference on Computer Vision (ICCV 2005)*, volume 2, pages 986 – 993, October 2005.

40. J.-H. Kim, R. Hartley, J.-M. Frahm, and M. Pollefeys. Visual odometry for non-overlapping views using second-order cone programming. In *Asian Conference on Computer Vision*, pages 353–362, 2007.

41. K. Konolige. Small vision system: Hardware and implementation, 1997.

42. M. Lhuillier. Effective and generic structure from motion using angular error. In *International Conference on Pattern Recognition*, pages 67–70, 2006.

43. H. Li and R. Hartley. The 3d-3d registration problem revisited. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 14-21 Oct. 2007.

44. H. Li and R. Hartley. Five-point motion estimation made easy. In *ICPR (1)*, pages 630–633. IEEE Computer Society, 2006.

45. J. Löfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the Computer Aided Control System Design*, Taipei, Taiwan, 2004.

46. H. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

47. B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.

48. L. Matthies, A. Kelly, T. Litwin, and G. Tharp. Obstacle detection for unmanned ground vehicles: a progress report. In *Proceedings of IEEE Intelligent Vehicles '95 Conference*, pages 66 – 71, September 1995.

49. L. Matthies and S. Shafer. Error modeling in stereo navigation. *IEEE Journal of Robotics and Automation*, RA-3(3):239 – 250, June 1987.

50. N. Molton and M. Brady. Practical structure and motion from stereo when motion is unconstrained. *Int. J. Comput. Vision*, 39(1):5–23, 2000.

51. H. Moravec. The stanford cart and the cmu rover. *Proceedings of the IEEE*, 71(7):872–884, July 1983.

52. E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyster, and P. Sayd. Generic and real-time structure from motion. In *British Machine Vision Conference*, 2007.

53. D. Murray and J. J. Little. Using real-time stereo vision for mobile robot navigation. *Auton. Robots*, 8(2):161–171, 2000.

54. P. Narayanan, P. Rander, and T. Kanade. Constructing virtual worlds using dense stereo. *Computer Vision, 1998. Sixth International Conference on*, pages 3–10, 4-7 Jan 1998.

55. S. K. Nayar. Catadioptric omnidirectional camera. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 482, Washington, DC, USA, 1997. IEEE Computer Society.

56. D. Nistér. An efficient solution to the five-point relative pose problem. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages II: 195–202, 2003.

57. D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–777, 2004.

58. D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 652–659, 2004.

59. J. Philip. A non-iterative algorithm for determining all essential matrices corresponding to five point pairs. *The Photogrammetric Record*, 15(88):589–599, Oct 1996.

60. R. Pless. Using many cameras as one. In *CVPR03*, pages II: 587–593, 2003.

61. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 1992.

62. S. Ramalingam, P. Sturm, and S. Lodha. Towards complete generic camera calibration. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 1:1093–1098 vol. 1, 20-25 June 2005.

63. C. Rother and S. Carlsson. Linear multi view reconstruction and camera recovery. In *IEEE International Conference on Computer Vision*, pages 42–51, 2001.

64. C. Rother and S. Carlsson. Linear multi view reconstruction and camera recovery using a reference plane. *International Journal of Computer Vision*, 49(2-3):117–141, 2002.

65. J. S. A. Salt. The scope of stereographic survey: Review. *The Geographical Journal*, 84(3):254–255, September 1934.

66. P. H. Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, March 1966.

67. G. Schweighofer and A. Pinz. Fast and globally convergent structure and motion estimation for general camera models. In *British Machine Vision Conference*, 2006.

68. S. Se and M. Brady. Stereo vision-based obstacle detection for partially sighted people. In *ACCV '98: Proceedings of the Third Asian Conference on Computer Vision-Volume I*, pages 152–159, London, UK, 1997. Springer-Verlag.

69. K. Sim and R. Hartley. Recovering camera motion using l_∞ minimization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1230–1237. IEEE Computer Society, 2006.

70. C. C. Slama, C. Theurer, and S. W. Henriksen, editors. *Manual of Photogrammetry*. America Society of Photogrammetry, fourth edition, 1980.

71. H. Stewénius, C. Engels, and D. Nistér. Recent developments on direct relative orientation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 60:284–294, June 2006.

72. H. Stewénius, F. Kahl, D. Nistér, and F. Schaffalitzky. A minimal solution for relative pose with unknown focal length. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 789–794, San-Diego, USA, June 2005. Chapter 8 of my thesis.

73. H. Stewénius and D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–777, 2004.

74. H. Stewénius, D. Nistér, M. Oskarsson, and K. Åström. Solutions to minimal generalized relative pose problems. In *Workshop on Omnidirectional Vision*, Beijing China, Oct. 2005.

75. J. Stolfi. *Oriented projective geometry*. Academic Press Professional, Inc., San Diego, CA, USA, 1991.

76. J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999. Special issue on Interior Point Methods (CD supplement with software).

77. P. Sturm. Multi-view geometry for general camera models. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 206–212, Jun 2005.

78. P. Sturm and S. Ramalingam. A generic concept for camera calibration. In *Proceedings of the European Conference on Computer Vision, Prague, Czech Republic*, volume 2, pages 1–13. Springer, May 2004.

79. P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *The 4th European Conference on Computer Vision*, pages 709–720, 1996.

80. S. Tariq and F. Dellaert. A Multi-Camera 6-DOF Pose Tracker. In *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2004.

81. E. H. Thompson. A rational algebraic formulation of the problem of relative orientation. *The Photogrammetric Record*, 3(14):152–159, October 1959.

82. B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - A modern synthesis. In *Vision Algorithms: Theory and Practice*, LNCS, pages 298–372, London, UK, 2000. Springer-Verlag.

83. M. Uyttendaele, A. Criminisi, S. B. Kang, S. A. J. Winder, R. Szeliski, and R. Hartley. Image-based interactive exploration of real-world environments. *IEEE Computer Graphics and Applications*, 24(3):52–63, 2004.

84. H. von Sanden. *Die Bestimmung der Kernpunkte in der Photogrammetrie*. PhD thesis, Georg-August-Universität Göttingen, 1908.

85. C. Wheatstone. Contributions to the physiology of vision. part the first. on some remarkable, and hitherto unobserved, henomena of binocular vision. *Philosophical Transactionsof the Royal Society of London*, 128:371–394, 1838.

86. W. Whittaker and L. Nastro. Utilization of position and orientation data for preplanning and real time autonomous vehicle navigation. *Position, Location, And Navigation Symposium, 2006 IEEE/ION*, pages 372–377, April 25-27, 2006.

87. G.-S. J. Young and R. Chellappa. 3-d motion estimation using a sequence of noisy stereo images: Models, estimation, and uniqueness results. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(8):735–759, 1990.

88. M. Zeller, E. A. Miskin, and R. Powell. *Text Book of Photogrammetry*. London H. K. Lewis & Co. Ltd., 1952.

89. Z. Zhang and O. Faugeras. Estimation of displacements from two 3d frames obtained from stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(12):1141–1156, Dec 1992.

Index