Constrained Motion Estimation for a Multi-Camera System

by

Mehdi Mazaheri Tehrani

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN GEOMATICS ENGINEERING

CALGARY, ALBERTA

September, 2015

© Mehdi Mazaheri Tehrani 2015
Abstract

Compared to a single camera, multi-camera systems view a larger field and collect more data for motion estimation and 3D reconstruction. The constant rigid body transformation between the cameras mounted (mounting parameters) is valuable knowledge for outlier detection and estimation of the system motion in real scale. In this research, the mounting parameters of the cameras are estimated in single step through bundle adjustment. Taking advantage of the mounting parameters, a methodology is proposed to estimate the system motion between successive data acquisition epochs. In this method, several estimates for the system rotation are obtained by constraining the relative rotation of the individual cameras. The outliers within the rotation estimates are filtered out and the inliers are averaged. Then, the system translation is estimated by solving a system of linear equations. The experimental results show that the proposed methodology can successfully recover the multi-camera system motion.

Keywords: Motion Estimation, Rotation Averaging, Multi-camera systems, Structure from Motion
Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor Professor Ayman Habib for the continuous support of my MSc. study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

I must acknowledge the department head Prof. Derek Lichti for his supports and wonderful lectures, and all the department staff for helping me out during the program. A special thanks goes out to Steve Lam for his prompt helps.

I would like to thank the members of Digital Photogrammetry Research Group (DPRG). It was a pleasure working with you. A special thanks goes out to Ivan Detchev who helped me a lot in different stages of this research.

I express my gratitude to TECTERRA and The Natural Sciences and Engineering Research Council of Canada (NSERC). Without their financial support, this research would have not been possible.
To

Roshdiana
# Table of Contents

Abstract ................................................................................................................................. ii
Acknowledgements .............................................................................................................. iii
Dedication .............................................................................................................................. iv
Table of Contents ................................................................................................................ v
List of Tables ........................................................................................................................ vii
List of Figures ....................................................................................................................... viii
List of Abbreviations .......................................................................................................... x

Chapter 1 : Introduction ....................................................................................................... 1

1.1 Applicable sensors .......................................................................................................... 3
1.2 Problem statement ......................................................................................................... 6
1.3 Research objectives ....................................................................................................... 7
1.4 Scope and limitation ..................................................................................................... 8
1.5 Summary of contributions .......................................................................................... 9
1.6 Thesis outline .............................................................................................................. 10

Chapter 2 : Background ..................................................................................................... 11

2.1 Feature detection and description ................................................................................ 12
2.2 Motion estimation ......................................................................................................... 17
  2.2.1 Fundamental and Essential matrix ........................................................................ 17
  2.2.2 Image to image (2D to 2D) orientation ................................................................. 19
  2.2.3 Image resection upon existing 3D model (3D to 2D) ........................................... 20
  2.2.4 Model to model transformation (3D to 3D) ........................................................ 21
  2.2.5 Generalized Camera Model .................................................................................. 21
2.3 Rotation representation and averaging ......................................................................... 24
  2.3.1 Euler angles ............................................................................................................ 25
  2.3.2 Angle-axis representation ...................................................................................... 25
  2.3.3 Quaternions ........................................................................................................... 27
  2.3.4 Karcher mean of rotation ...................................................................................... 28
  2.3.5 Rotation averaging by quaternions ........................................................................ 29
  2.3.6 Motion estimation .................................................................................................. 30
2.4 Relating object space to image space in a multi-camera system ................................ 33
2.5 Summary ....................................................................................................................... 35
Chapter 3 : Methodology ........................................................................................................................................37

3.1 Feature extraction and matching ..................................................................................................................39
  3.1.1 Detection of outliers across the Epipolar lines in temporal matches .......... 42
  3.1.2 Detection of outliers in spatial matches .................................................................................................43
3.2 Motion estimation ...........................................................................................................................................45
  3.2.1 Rotation estimation .................................................................................................................................45
  3.2.2 Detection of incompatible rotation estimates .........................................................................................47
  3.2.3 Translation estimation ............................................................................................................................49
  3.2.4 Detection of outlier matches using the estimated system motion .......... 51
3.3 Sliding window bundle adjustment ..............................................................................................................52
3.4 Summary ......................................................................................................................................................55

Chapter 4 : Results ..............................................................................................................................................56

4.1 Hardware and software implementation ......................................................................................................56
4.2 System calibration .........................................................................................................................................58
4.3 Sources of error ............................................................................................................................................61
4.4 Datasets .......................................................................................................................................................62
  4.4.1 Geomatics corridor dataset ..................................................................................................................63
  4.4.2 Science corridor dataset ......................................................................................................................64
4.5 Feature extraction and matching ................................................................................................................66
4.6 Outlier detection ..........................................................................................................................................70
4.7 Recovery of the multi-camera system motion ............................................................................................71
  4.7.1 System rotation estimation and robust averaging ................................................................................72
  4.7.2 System translation estimation ..............................................................................................................73
  4.7.3 Refinement the estimated motion by bundle adjustment ..................................................................74
  4.7.4 Evaluation of the proposed methodology .........................................................................................77
  4.7.5 Accuracy check ....................................................................................................................................79
4.8 Summary ......................................................................................................................................................81

Chapter 5 : Conclusions and recommendations for future work .................................................................82

5.1 Recommendation for future works ............................................................................................................84

References ..........................................................................................................................................................87
List of Tables

Table 1.1: A comparison between 2D cameras and 3D cameras/mobile TLS sensors............6
Table 2.1: Popular corner and blob detectors and their characteristics (Fraundorfer and Scaramuzza, 2012)..........................................................13
Table 4.1: Multi-camera system calibration results..........................................................61
Table 4.2: Corner detection thresholds ........................................................................66
Table 4.3: Result of outlier detection of matched features for the two dataset ................71
Table 4.4: Outlier detection thresholds ........................................................................71
Table 4.5: Result of the system rotation estimation..........................................................73
Table 4.6: Average standard deviation of the translation estimation...............................74
Table 4.7: Result of sliding window and global BA to refine the system motion..............75
Table 4.8: Thresholds for sliding window bundle adjustment.........................................75
List of Figures

Figure 1.1: (a) a backpack indoor modelling system (Berkeley video and image processing lab) ................................................................................................................................................3

Figure 1.2: (a) TOF 3D camera (MESA SR4000), (b) Microsoft Kinect (c) Hokuyo sensor, (d) Velodyne TLS ..............................................................................................................................................5

Figure 2.1: FAST feature detector (www.edwardrosten.com) .................................................................................................................................14

Figure 2.2: A light ray can be defined by a vector and a point .................................................................................................................................23

Figure 2.3: Rotation of angle $\theta$ around axis $v$ .........................................................................................................................................................26

Figure 2.4: Motion of a non-overlapping stereo camera ......................................................................................................................................32

Figure 2.5: The vector summation relating an object point position to the reference and a non-reference camera (Kersting et al., 2011) ..............34

Figure 3.1: Flowchart of the proposed methodology for motion estimation ......................................................................................................39

Figure 3.2: Sub-pixel refinement of a corner .........................................................................................................................................................40

Figure 3.3: Spatial and temporal matches in a sequence of epochs ....................................................................................................................41

Figure 3.4: Distance to Epipolar lines .................................................................................................................................................................42

Figure 3.5: The reference camera coordinate system and two non-reference cameras ........44

Figure 3.6: Reference and a non-reference cameras at epochs $t_1$ and $t_2$, and all possible relative orientations parameters between the cameras ....................................................................................................................46

Figure 3.7: The shift caused by applying the rotation $\Delta R$ to a grid of image points ........48

Figure 3.8: Two types of triple sample checked for intersection error ...........................................................................................................52

Figure 3.9: Bundle adjustment with built-in outlier detection ..........................................................................................................................54

Figure 4.1: Data acquisition system .................................................................................................................................................................56

Figure 4.2: Images of the calibration test field captured at the same time. .....................................................................................................58

Figure 4.3: Image points residuals versus radius from principal point, for different sets of distortion coefficients .........................................................................................................................60

Figure 4.4: Sample images in which feature tracking was failed .....................................................................................................................63

Figure 4.5: Sample images of the Geomatics corridor dataset .........................................................................................................................63
Figure 4.6: Sample images of the Science corridor dataset. (a) Poor texture, (b) Repetitive texture (c) Strong texture, (d) 90 degree turn

Figure 4.7: Flowchart of the proposed methodology for motion estimation

Figure 4.8: Detected corner features in an indoor scene. Some of the corners are magnified at right. (a) Initially detect features (b) After purging the close features

Figure 4.9: A detected corner before and after sub-pixel refinement

Figure 4.10: Feature tracks in the images taken by the multi-camera system; length of the green lines represents the number of images in which the feature has been tracked, red lines are features tracked in only two images (weak feature tracks)

Figure 4.11: Mean feature track lengths seen by the cameras at each acquisition epoch for the Geomatics corridor dataset (a) and the Science corridor dataset (b). The sample images in which feature tracks are short (S) and long (L) are displayed in Figure 4.12

Figure 4.12: Sample images in which feature tracks are short (S) or long (L) for the Geomatics corridor dataset (left images) and the Science corridor dataset (right images)

Figure 4.13: CDF of the mean compatibility error of the rotation estimates at each epoch for the Geomatics corridor dataset (a) and the Science corridor dataset (b)

Figure 4.14: Recovered system motion by a) proposed method b) after refinement by sliding and global BA. (Visualized by Visual SfM software)

Figure 4.15: Recovered system motion in the Science corridor by a) proposed method

Figure 4.16: Angular error of system rotation estimation by the proposed method for the Geomatics corridor dataset (a), and the Science corridor dataset (b)

Figure 4.17: Error in system translation estimation by the proposed method for the Geomatics corridor dataset (a), and the Science corridor dataset (b)

Figure 4.18: Check distances from the scene and their error

Figure 4.19: Check distances for Science corridor dataset with their discrepancies
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BA</td>
<td>Bundle Adjustment</td>
</tr>
<tr>
<td>CDF</td>
<td>Cumulative Distribution Function</td>
</tr>
<tr>
<td>CenSurE</td>
<td>Center Surround Extremas</td>
</tr>
<tr>
<td>DPRG</td>
<td>Digital Photogrammetry Research Group</td>
</tr>
<tr>
<td>EOP</td>
<td>Exterior Orientation Parameters</td>
</tr>
<tr>
<td>FAST</td>
<td>Features from Accelerated Segment Test</td>
</tr>
<tr>
<td>GCM</td>
<td>Generalized Camera Model</td>
</tr>
<tr>
<td>GNSS</td>
<td>Global Navigation Satellite System</td>
</tr>
<tr>
<td>GSD</td>
<td>Ground Sampling Distance</td>
</tr>
<tr>
<td>ICP</td>
<td>Iterative Closest Point</td>
</tr>
<tr>
<td>IOP</td>
<td>Interior Orientation Parameters</td>
</tr>
<tr>
<td>LiDAR</td>
<td>Light Detection and Ranging</td>
</tr>
<tr>
<td>RANSAC</td>
<td>Random Sample Consensus</td>
</tr>
<tr>
<td>ROP</td>
<td>Relative Orientation Parameter</td>
</tr>
<tr>
<td>RSS</td>
<td>Received Signal Strength</td>
</tr>
<tr>
<td>SfM</td>
<td>Structure from Motion</td>
</tr>
<tr>
<td>SIFT</td>
<td>Scale-Invariant Feature Transform</td>
</tr>
<tr>
<td>SLAM</td>
<td>Simultaneous Localization And Mapping</td>
</tr>
<tr>
<td>SPR</td>
<td>Single Photo Resection</td>
</tr>
<tr>
<td>SURF</td>
<td>Speed Up Robust Feature</td>
</tr>
<tr>
<td>TLS</td>
<td>Terrestrial Laser Scanner</td>
</tr>
<tr>
<td>TOF</td>
<td>Time Of Flight</td>
</tr>
<tr>
<td>UWB</td>
<td>Ultra Wide Band</td>
</tr>
<tr>
<td>VO</td>
<td>Visual Odometry</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

A study by The National Human Activity Pattern Survey (NHAPS) shows that Americans and Canadians spend almost 90% of their time indoors (KLEPEIS et al., 2001). Therefore, improving Indoor Environment Quality (IEQ) through good design and regular maintenance is highly important. For large indoor environments like malls, hospitals, museums and similar public places, which may have hundreds of rooms and corridors, software packages are required for facility management, navigation, indoor Geographic Information System (GIS), or Building Information Modelling (BIM). In all of these applications, indoor 3D models play an important role in providing the necessary geometric information, and visualizing the indoor environment.

An accurate 3D model of indoor urban environments can be a valuable asset to cultural heritage documentation, generation of virtual environments, city planning, urban design, and fire & police planning (Chow, 2014). Precise cultural heritage documentation is essential for archiving purpose or scientific studies carried out during a restoration or renovation project. For a long time, 3D information of cultural heritage objects has been collected by imagery (Yastikli, 2007). Generating virtual environments for virtual tours or walk-through simulations requires realistic 3D models to reduce the gap between the real and virtual world (Shumaker, 2009). Virtual museums, Real State tours, and training simulators for firefighters are examples of virtual environments in which indoor 3D models are used.

The procedure of 3D modelling in indoor and outdoor environment is different in several aspects. In an outdoor environment, GNSS signals are available almost all the time except dense urban areas, under bridges or any construction that blocks the signal. Processing GNSS data together with Inertial Measurement Unit (IMU) facilitates accurate localization of the mobile mapping systems working in outdoor environments. The most important difference between the
outdoor and indoor environments is lack of GNSS signals for positioning inside the buildings. An alternative positioning method is using Wi-Fi signals, which are available in most of the buildings these days. A Wi-Fi enabled device is capable of identifying the surrounding access points and measure the Received Signal Strength (RSS). The signal strength can be translated to physical distance between the device and the Wi-Fi access point, using the related energy-distance formulas or other methods. Knowing the distance of the sensor to the access points, and the location where the access points are installed, the device position can be computed by trilateration. A distance to two or three access points would be enough for 2D or 3D positioning of the sensor, respectively. In addition to trilateration using Wi-Fi signal strength, there are various techniques such as Ultrasound or Bluetooth for positioning in indoor environment, which are surveyed in the work of Liu et al (2007). According to this survey, the accuracy of wireless positioning methods is several meters to sub-meter, and can be improved to 15 centimeters when high cost devices and complex methods are employed. Zwirrello et al. (2012) use Ultra Wide Band (UWB) signals for indoor positioning and reach to 2.5 cm accuracy. Due to the complexity and the need for moderate to high cost equipment, wireless positioning methods are not seriously considered in practice for high accuracy indoor poisoning.

Although outdoor navigation systems are quite mature (Han et al., 2014), indoor navigation systems are still under research and development. Ability to navigate indoors in an emergency situation where a part of a building is blocked, or vision is limited due to smoke, can save the lives of any trapped people or first responders. Though building 3D models are usually available from the design phase, in most cases they are too complicated for finding escape routes, and other forms of 3D models have to be used (Isikdag, 2014). In general, 3D models are more powerful than 2D
maps for indoor navigation, because they can provide accurate descriptions of indoor object locations such as doors, windows, tables, etc. (Xu et al., 2013).

There are various sensors applicable for creating an indoor 3D modelling system. The main sensors to acquire 3D data are 2D cameras, 3D cameras, and laser scanners, while auxiliary sensors such as Micro-Electro-Mechanical Systems (MEMS) and odometers can help refining the localization of the acquired 3D data. Data acquisition hardware could be a collection of these sensors in a backpack (Figure 1.1 - a), which makes it suitable for staircases, or mounted on a trolley (Figure 1.1 – b, c) to acquire long building corridors.

![Figure 1.1: (a) a backpack indoor modelling system (Berkeley video and image processing lab) (b) Trimble Indoor Mobile Mapping System (c) Viametris system](image)

In the next section, the major technologies applicable for indoor 3D modelling are briefly reviewed. As the goal of this research is to estimate 3D motion using a low-cost system, expensive sensors are not considered.

### 1.1 Applicable sensors

3D data acquisition sensors can be divided into active and passive ones. Active sensors like laser scanners, emit a light ray to the object surface and receive the reflected light ray. The sensor-object distance is then calculated by measuring the Time Of Flight (TOF) of the light ray or by
using phase shifting technique. Range of operation in active sensors is restricted by the fact that the energy of the emitted light ray diminishes with distance, so the reflected light rays from far objects would be too weak to be sensed. Passive sensors such as 2D cameras rely on the external light source and only receive the reflected light from the object surface. Theoretically, passive sensors can operate at any range, however, the spatial resolution of the passive sensors is limited and far objects are captured with less level of detail.

3D cameras or range cameras are active sensors that provide a range image in which every pixel have a depth value. 3D cameras can be divided in two categories, TOF based or triangulation based. A TOF camera (Figure 1.2-a) is a combination of an illumination source LED and an array of receiver pixels, which measures the distance by calculating the time of flight of the infra-red signals transmitted from the LED to the object and back. In order to find the range more accurately, some type of 3D cameras employ a phase difference technique. The most popular triangulation based 3D camera is the Microsoft Kinect (Figure 1.2-b). Microsoft Kinect emits a speckle pattern on an object using the emitter and records the reflected image using the IR sensor. A disparity map is then computed by correlating the emitted pattern and the captured image. Using the computed disparity map, the depths of the object points are computed by triangulation. The accuracy of the Kinect at short ranges (e.g. less than 3.5 m) is less than two centimeters (Khoshelham, 2011), which is suitable for limited range 3D modelling. However, 3D cameras have a low spatial resolution and a narrow field of view. For example, MESA SR4000 3D camera (Figure 1.2-a) has only 176×144 pixels.

Terrestrial Laser Scanners (TLSs) are active sensors able to rapidly acquire 3D point clouds from the surrounding environment. This capability has made TLS the leading technology for capturing dense 3D information (Chow, 2014). Similar to 2D cameras, TLSs require calibration
and exterior orientation parameters of the sensor to generate an accurate geo-referenced point cloud. In outdoor applications, TLSs are usually geo-referenced with the aid of GNSS/IMU sensors, but for indoor applications, other methods such as Iterative Closest Point (ICP) (Besl and McKay, 1992) have to be used. ICP requires good initial values, and could be slow due to iterative nature of the algorithm. To speed up the process, closed form Point to Line ICP (Censi, 2008) can be employed.

TLS devices can be categorized into static and mobile scanners. Static scanners are usually mounted on a tripod, and automatically scan the surrounding environment in a few minutes, and outputs a list of X, Y, and Z coordinates. Mobile scanners are designed to be mounted on a moving platform, and sweep a narrow strip. As the platform moves through the scene, the point cloud is gathered from the scanned strips. In the world of mobile TLS, the Hokuyo sensor (Figure 1.2-c) has been employed in several mobile mapping systems such as the ones in Kua et al. (2012), Boose et al. (2012) and Achtelik et al. (2009) as well as the commercial mobile mapping systems such as Viametris (Figure 1.1-c). However, the Hokuyo sensor has limited range and it is not suitable for open indoor areas such as gyms or malls. The Velodyne sensor (Figure 1.2-d) can operate at ranges up to 120 meters, but it is not suitable for low-cost applications.

Figure 1.2: (a) TOF 3D camera (MESA SR4000), (b) Microsoft Kinect (c) Hokuyo sensor, (d) Velodyne TLS
2D cameras, or simply digital cameras, are passive sensors that take RGB images from the scene. Nowadays, off-the-shelf digital cameras can capture high quality images in which fine scene details are visible. Unlike 3D cameras, extensive post processing is required to extract 3D information from 2D images, but can be carried out in a timely manner by taking advantage of high processing power of current personal computers. Table 1.1 compare the 2D cameras and 3D cameras/mobile TLS for motion estimation and 3D reconstruction purpose. In this research, 2D cameras are employed for motion estimation and sparse 3D reconstruction, due to the low-cost, availability, and high accuracy.

Table 1.1: A comparison between 2D cameras and 3D cameras/mobile TLS sensors

<table>
<thead>
<tr>
<th></th>
<th>3D cameras/mobile TLS</th>
<th>2D camera</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>Medium (few hundreds to few thousands $)</td>
<td>Low (few hundred $)</td>
</tr>
<tr>
<td>Resolution</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Field of view</td>
<td>Narrow</td>
<td>Medium-Wide</td>
</tr>
<tr>
<td>Motion estimation</td>
<td>Medium complexity, slow, using ICP method</td>
<td>Complex, slow, using SfM procedure</td>
</tr>
<tr>
<td></td>
<td>Usually requires external sensor</td>
<td>External sensor is not essential</td>
</tr>
<tr>
<td></td>
<td>ICP is sensitive to flat scenes</td>
<td>Sensitive to poor textured areas</td>
</tr>
<tr>
<td>Point cloud extraction</td>
<td>Easy, directly provided by sensor</td>
<td>Complex, by extensive post processing</td>
</tr>
<tr>
<td>Texture mapping</td>
<td>Requires RGB sensor</td>
<td>No external sensor is required</td>
</tr>
<tr>
<td>Range of operation</td>
<td>Short, far range sensors are expensive</td>
<td>Far range</td>
</tr>
<tr>
<td>Accuracy</td>
<td>Centimeter level local accuracy</td>
<td>Sub-centimeter local accuracy</td>
</tr>
</tbody>
</table>

1.2 Problem statement

Using multiple cameras for data acquisition has many advantages compared to a single camera (Lee et al., 2014). First, larger area of the scene is captured as the field of view is expanded by using multiple cameras. Wider field of view increases the chance to find more reliable features and track them longer in an image sequence. Assume a long corridor captured by a data acquisition system with cameras looking to the both sides of the corridor, to the floor and up to the ceiling. In
case of poor textured side walls in a part of the corridor, floor or ceiling images can help continue tracking of features - and continue motion estimation accordingly. Expansion of field of view using multiple cameras does not distort the images unlike fish-eye lenses and spatial resolution is maintained. However, time synchronization between the cameras is critical in a multi-camera system.

The second advantage of a multi-camera systems is the ability to capture redundant data, which improves the precision of motion estimation and 3D reconstruction algorithms. For example, an object point is reconstructed by more image points in a multi-camera system, which improves the reconstruction precision. The last and most important benefit of a multi-camera system, is the constant rigid body transformation between the mounted cameras (mounting parameters), which is a valuable knowledge in every stage of the motion estimation and 3D reconstruction. The mounting parameters help in outlier detection and recovery of the 3D model in real scale.

According to the mentioned advantages of the multi-camera systems, this research aims at estimating 3D motion of a multi-camera system by incorporating the individual camera motions. Using the mounting parameters, motion of each camera can be related to the system motion. Therefore, depending on the number of employed cameras, several estimates for the system motion can be obtained and averaged for robustness. The developed methodology works for a low-cost multi-camera system without any need for auxiliary sensors or high cost component.

1.3 Research objectives

The main objectives in this research are:

- Develop a low-cost multi-camera data acquisition system to take synchronized images of building corridors. The system is calibrated in a single step for the interior
and the mounting parameters of the cameras, using a multi-camera calibration toolbox developed within DPRG group.

- Develop a processing framework for motion estimation of a multi-camera system within an indoor environment, by taking advantage of the constant mounting parameters of the cameras to combine the individual camera motion.

1.4 Scope and limitation

The scope of this research is:

- Estimating 3D motion of a multi-camera system (two or more cameras) in indoor environment.
- The cameras are mounted on a cart, in overlapping configuration.
- Only one side of the corridor is captured.
- The 3D motion of the multi-camera system is estimated from the captured images, no auxiliary sensor is used.

The limitations of this research are:

- A minimum of two sets of ROPs between similar/different cameras at successive epochs have to be estimated.
- Sufficient overlap between the successive images are required, to estimate the ROPs.
- In case of short base line (small motion), the ROPs are poorly estimated.
- The accuracy of the motion estimation is highly dependent of the scene texture.
- Auto-focus feature of the cameras are kept disabled, to avoid changes in the IOPs of the cameras.
The aperture and shutter speed are kept fixed, to avoid any delay caused by light tuning. In order to have synchronized images, a camera is supposed to take image immediately upon receiving the triggering command.

1.5 Summary of contributions

The contributions of this research are summarized as follows.

• A method to estimate the system rotation between successive epochs for an overlapping multi-camera system was introduced. This method is based on obtaining several estimates for the system rotation and averaging them.

• In practice, there exists incompatible rotation estimates that have to be filtered out. An iterative method was proposed to detect the outlier rotation estimates.

• A metric to describe the discrepancy of two rotation matrices was introduced. Using the proposed metric, the discrepancy of two rotation matrices is described in pixel unit, which is easier to interpret.

• A method was proposed to estimate the real-scale translation of a multi-camera system, using the estimated rotation.

• A procedure to detect outlier matches along the Epipolar lines using the estimated system motion was proposed.

• Implementation of sliding window bundle adjustment with built-in ROPs.

In practice, this method can be used for any multi-camera system for combining the individual camera motions. At least two individual camera motions are required to estimate the system motion in real scale. The results of motion estimation in a real challenging environment show that the proposed method is successful in practice.
1.6 Thesis outline

Chapter 2 of this thesis reviews the literature for motion estimation. In this chapter, first, the major feature detection and matching methods are reviewed. Then, the existing motion estimation techniques including the rotation averaging are explained. Chapter 3 discusses the methodology developed in this research for motion estimation. Chapter 4 presents the results of the system calibration and motion estimation, using couple of datasets captured from an indoor environment. Chapter 5 concludes with a summary and discussion, and provides recommendation for future works.
Chapter 2: Background

The problem of recovering three-dimensional structure of a scene and motion of a camera from a collection of images is called Structure from Motion (SfM) (Vedaldi et al., 2007). SfM is a close research line to Visual Odometry (VO) and Visual Simultaneous Localization and Mapping (VSLAM). VO focuses on sequential estimation of the camera motion - as a new frame arrives - in real time (Scaramuzza and Fraundorfer, 2011). SLAM is the problem of locating a robot within a map and updating the map simultaneously. VSLAM is a novel algorithm to solve the SLAM problem, which facilitate robust map-building and localization of the robot by fusing image and odometry data (Niklas Karlsson et al., 2005). Therefore, estimating the location/motion of the camera is a common task between SfM, VO, and VSLAM.

Structure and motion recovery proceeds by 1) extracting features, 2) and matching these features across the images (Torr and Zisserman, 2000). 3) Then, the camera motion in 3D space is estimated using the matched features, and finally 4) the estimated motion is refined by bundle adjustment (Triggs et al., 2000). This chapter reviews the existing methods for the structure and motion recovery. In the next section, the related works in feature detection and description are reviewed. Section 2.2 discusses the major existing methods for motion estimation. Section 2.3 explains different methods for rotation representation and averaging, which is used in the proposed methodology (Chapter 3) for estimating the 3D motion of a multi-camera system. Section 2.4 discusses a mathematical model that relates the object space to the image space, which is used in bundle adjustment to refine the estimated motion.
2.1 Feature detection and description

The first step for motion estimation is detecting features in the captured images and matching them. Feature detection is the procedure to find salient features such as points or lines, which are distinguishable from their neighborhood in an image. Feature description is referred to the procedure of finding appropriate descriptor for a detected feature.

According to Förstner (1986), a feature detection principle should fulfill the following requirements:

- Distinctness: the feature should be distinct from neighboring area.
- Invariance: the feature detection algorithm should not be affected by the radiometric changes such as illumination, or geometric changes such as scale (zoom), rotation, and affinity.
- Stability/Repeatability: the detected feature should be expected to appear in the other images.
- Seldomness: the feature should be globally distinct to avoid any confusion in the matching process. i.e., there should be no similar feature in its neighborhood or ideally in the entire image. This criterion is essential for images in which repetitive pattern is observed, where many similar features satisfying the local distinction criterion might be detected.

Corners and blobs are popular point features that can fulfill the mentioned criteria. A corner is intersection of two or more edges, where large intensity changes happen in more than one direction. A blob is an image pattern that differs from its immediate neighborhood in terms of intensity, color, and texture. A blob is neither an edge nor a corner (Fraundorfer and Scaramuzza, 2012).
Corner detector algorithms like Moravec (Moravec, 1980), Förstner (Förstner, 1986), Harris (Harris and Stephens, 1988), Shi-Tomasi (Shi and Tomasi, 1994), FAST (Rosten and Drummond, 2006), and blob detectors such as SIFT (Lowe, 1999), SURF (Bay et al., 2006), and CENSURE (Agrawal et al., 2008) have been used for image matching in different literature. Table 2.1 displays the characteristics of the commonly used feature detectors (Fraundorfer and Scaramuzza, 2012).

Table 2.1: Popular corner and blob detectors and their characteristics (Fraundorfer and Scaramuzza, 2012)

<table>
<thead>
<tr>
<th>Feature Detector</th>
<th>Corner Detector</th>
<th>Blob Detector</th>
<th>Rotation Invariant</th>
<th>Scale Invariant</th>
<th>Affine Invariant</th>
<th>Repeatability</th>
<th>Localization Accuracy</th>
<th>Robustness</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harris</td>
<td>x</td>
<td>x</td>
<td>++</td>
<td>+++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Shi-Tomasi</td>
<td>x</td>
<td>x</td>
<td>++</td>
<td>+++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>FAST</td>
<td>x</td>
<td>x</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>SIFT</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>SURF</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>CENSURE</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
</tbody>
</table>

Feature detection is followed by feature description, where a descriptor is evaluated for the feature from the neighboring area. The evaluated descriptor is used to find a match within the features detected in the other images. Similar to feature detectors, feature descriptors should be invariant to the illumination and geometrical changes that happen in a practical application. There are other descriptors developed for different applications, namely USURF/SURF (Bay et al., 2006), BRIEF (Calonder et al., 2010), ORB (Rublee et al., 2011), DAISY (Tola et al., 2010), BRISK (Leutenegger et al., 2011), and FREAK (Alahi et al., 2012).

Harris method detects corners by thresholding the Harris response ($H$) of every point in an image, using Equation (2.3), where $w$ is a window centered at the point in question, $k$ is a constant
(typically $k = 0.04$, which tunes the sensitivity of the operator), $I_x$ and $I_y$ are image gradient in the x and y directions, respectively. Shi-Tomasi made a small improvement to the Harris method, and proposed the minimum Eigen value of the matrix $G$ as the response function (OpenCV.org).

$$H = \det(G) - k(\text{trace}(G))^2, G = \sum_{x,y\in w} \begin{bmatrix} I_x^2 & I_xI_y \\ I_yI_x & I_y^2 \end{bmatrix}$$ \hspace{1cm} (2.1)$$

Features from Accelerated Segment Test (FAST) was introduced by Rosten et al.(2006). FAST examines whether or not a point $p$ is a feature, by considering a circle of 16 pixels around the $p$ (Figure 2.1). Now the pixel $p$ - with intensity $I(p)$ - is a corner if there exists a set of $N$ contiguous pixels in the circle (of 16 pixels) that are all brighter than $I(p) + t$, or all darker than $I(p) - t$, where $t$ is an appropriate intensity threshold (OpenCV.org). Figure 2.1 shows an example of $N=9$ contiguous pixels around a point $p$ (dash line). FAST is used for real time applications where limited computational resources are available. It is several times faster than other existing corner detectors but it is not robust to high level of noise and also dependent of a threshold ($t$).

![Figure 2.1: FAST feature detector (www.edwardrosten.com)](image)

Scale-invariant Feature Transform (SIFT) was proposed by Lowe (1999) to detect invariant descriptors in images. SIFT is rotation, illumination and view point invariant. Sinha (2010)
explains SIFT algorithm in detail with simple language in his webpage, which is briefly reviewed here. SIFT algorithm gets rid of details, by making different level of blurring for each level of image pyramid. Typically, a four level pyramid (each level of pyramid is called octave) is used and in each level, five blurring levels are applied. Therefore, 20 images are generated from the original image. The next step is to find Difference of Gaussian (DoG) in each octave. DoG approximates Laplacian of Gaussian (LoG) efficiently, which is proven to be scale invariant. The local maxima in DoG images are then calculated with sub-pixel accuracy, and considered as keypoints. The extracted key-points are filtered out if located on margins, or have low gradient value. In the next stage, the feature orientation is computed by collecting the directions and the magnitude of the gradients around the feature. The direction in which the gradients have higher magnitude is the feature orientation. In the next step, the feature descriptor is generated, which is a 128 vector containing magnitudes of gradients in eight direction in 16 grid cell around the feature (8×16=128). The descriptor is then subtracted from the feature orientation (already computed), which yields to a rotation invariant descriptor. Afterwards, the descriptor is normalized to obtain an illumination invariant feature.

Speeded Up Robust Features (SURF) (Bay et al., 2006) is based on Hessian blob detection. The SURF features are detected by non-maximal-suppression of the determinant of the Hessian matrices computed at each image point (Pedersen, 2011). Equation (2.2) expresses the Hessian matrix for a point \( p = (x, y) \) and a scale \( \sigma \), where \( L_{xx} \) is the convolution of the Gaussian second order derivative \( \frac{\partial^2}{\partial x^2} g(\sigma) \) with the image \( I \) at point \( p \), and similarly for \( L_{xy} \) and \( L_{yy} \) (Equation (2.3)).

\[
H(p, \sigma) = \begin{bmatrix} L_{xx}(p, \sigma) & L_{xy}(p, \sigma) \\ L_{xy}(p, \sigma) & L_{yy}(p, \sigma) \end{bmatrix} \tag{2.2}
\]
\[ L_{xx}(p, \sigma) = I(p) \ast \frac{\partial^2}{\partial x^2} g(\sigma) \] 

An image pixel is a SURF feature if determinant of the Hessian matrix on that pixel is high enough. For SURF description, a neighbourhood of 20x20 pixel is considered around the feature and divided to 4x4 sub regions, so each sub region has 5x5 pixel. The wavelet response in x and y direction is computed for each sub region, referred by \( dx \) and \( dy \), respectively. A four dimension vector \( v \) is constructed for each sub region by summation of the wavelet responses \( v = [\Sigma dx, \Sigma|dx|, \Sigma dy, \Sigma|dy|] \). Therefore, \( 4 \times 4 = 16 \) vectors are constructed in total, which forms a \( 16 \times 4 = 64 \) dimension descriptor. The descriptor is normalized to be contrast invariant.

Center Surround Extremas (CenSurE) was introduced by Agrawal et al. (2008). The main emphasis in CenSurE is to provide computationally feasible scale invariant local features by replacing the scale-space pyramid (e.g. in SIFT) with Center-Surround bi-level filters. In CenSurE detector, the Center-Surround filters are calculated at all locations and all scales. The original publication lists a few different options for filters, for boxes being the simplest. The filter shape defines how well the detector captures the rotation invariance. Similar to SIFT, CenSureE filters unstable features around the edged (Lankinen, 2014).

After feature extraction (i.e., detection and description) each extracted feature is searched for within candidate images to find a match by comparing the descriptors. A set of matched features in different images is called a feature track. A feature track is generated by linking the matches using the common features. In other words, each feature track is the projection of an object point into different images. There are two strategies to generate feature tracks: a) Feature Matching, which is carried out by detecting features independently in all the images, and matching them by comparing their descriptors, b) Feature Tracking, is to detect features in an image, then, search for their corresponding matches in the following images. Feature tracking is suitable for
small movements as the apparent change between the adjacent frames is small (Fraundorfer and Scaramuzza, 2012).

The length of a feature track shows the ability of the employed feature matching algorithm to track a feature in the captured images. Longer feature tracks strengthen the scene geometry and improve the reconstruction accuracy. Using the feature tracks, camera/multi-camera system motion can be estimated by photogrammetric and computer vision algorithms. The related works in motion estimation by feature tracking are reviewed next.

2.2 Motion estimation

As mentioned in the beginning of this chapter, motion estimation is the major and the most challenging part of SfM/VO/VSLAM. The generated feature tracks are used to estimate the camera motion and 3D structure using the methods reviewed in this section. Then, the feature tracks are used in bundle adjustment to refine the estimated motion and 3D structure. Feature tracks are generated from raw images, which are usually distorted depending on the lens type and assembly. Lack of distortion compensation in the feature tracks impacts the accuracy of the motion estimation, especially when an amateur camera or a fish-eye lens is used. To avoid these problems, feature tracks are undistorted by the calibration information provided by the manufactures or estimated through a self-calibration procedure. The distortion-free feature tracks are then used to estimate the 3D motion parameters. In this section, the major motion estimation approaches are briefly reviewed.

2.2.1 Fundamental and Essential matrix

Before explaining the motion estimation methods, the Fundamental and the Essential matrices are reviewed. If $x$ and $x'$ represent the conjugate image points associated with an object
point $X$, then the image points satisfy the equation $x'^T F x = 0$. The matrix $F$ is a $3 \times 3$ matrix, rank two, which encapsulates the geometry of the two images taken by an un-calibrated camera. The matrix $F$ can be estimated using eight conjugate image points - called as the 8-point algorithm (Longuet-Higgins, 1981). Using the Fundamental matrix, the Epipolar lines passing the image points $x$ and $x'$ are formulated as $l = F^T x'$ and $l' = F x$, respectively (Hartley and Zisserman, 2003).

Essential matrix is a type of Fundamental matrix, or Fundamental matrix is generalization of the Essential matrix. The Essential matrix is a $3 \times 3$ matrix with zero determinant that encodes the geometry between the two images when the IOPs of the camera are known from the calibration. The relation $x'^T E x = 0$ is obtained by applying the co-planarity constraint between the conjugate light rays of $x$ and $x'$, and the translation vector between the cameras $T$. Therefore, it is formulated as $E = \hat{T} R$, where $R$ is the relative rotation between the cameras. The nine elements of the Essential matrix can be estimated with five conjugate points by taking the constraints $2 E E^T E = \text{Trace}(EE^T) E$ and $\text{det}(E) = 0$ into account (Hartley and Zisserman, 2003).

The classic way for the relative orientation is to estimate the Fundamental matrix ($F$) robustly using the 8-point algorithm (Longuet-Higgins, 1981), in conjunction with RANSAC (Fischler and Bolles, 1981). However, it is known that only a minimum of five points is required to solve for the ROPs. In this regard, the Nister 5-point algorithm (Nister, 2004) has become a standard closed-form solution to find the Essential matrix ($E$) between two images using five points. The Essential matrix is then decomposed to the rotation ($R$) and translation ($T$) between the cameras (Horn, 1990). In addition to Nister algorithm, there are other 5-point solvers such as Faugeras and Maybank (1990), Philip (1996), Heyden and Sparr (1997), Triggs (2000), Li and Hartley (2006) with available MATLAB code, Stewenius et al. (2006) with online MATLAB/Mex...
implementation, Batra et al. (2007), and Kukelova et al. (2008) along with MATLAB source code downloadable online. Botterill et al. (2011) proposes a Levenberg-Marquardt optimization to find the Essential matrix, and has published their C++ implementation together with Stewenius et al. (2006) algorithm code.

2.2.2 Image to image (2D to 2D) orientation

2D to 2D method is based on sequential relative orientation of new images to the previous one. In an image sequence, ROPs of a new image relative to the previous image are estimated first. It is known that using the matched points between two images, the ROPs are estimated up to a scale. Therefore, the 3D model constructed by the new and previous image – using the estimated ROPs - should be rescaled to be identical to the existing model scale. A rescaling factor for the new model is computed by comparing one or more distances between the points in the existing model and the same points in the new model. To compute the rescaling factor, existing 3D points that have projection in the new image are queried from the feature tracks. In other words, points in the new image that have at least two matches in the previous images are selected. Therefore, the 3D position of these points in the existing model is already known, and their 3D position in the new model is computed by intersection using the estimated ROPs. To compute the rescaling factor, a pair of these points is selected \((i,j)\) first. Then, the distance between the points \(i, j\) in the existing model \(d_{ij}\) and in the new model \(d'_{ij}\) is computed. By dividing the two distances, the rescaling factor \(r\) is obtained, \(r = d_{ij}/d'_{ij}\). In order to equalize the scale of the existing and the new model, the rescaling factor \(r\) is multiplied to the translation of the new image. For robustness, multiple point pairs (or all point pairs) are selected and the resulting rescaling factors are averaged. After rescaling the new model according to the existing 3D model, this procedure is repeated for the next image.
In practice, 2D to 2D method might be slow and complex in terms of implementation. The 5-point algorithm generates several hypothesis for the Essential matrix, each of which must be tested to find the best one - which minimizes the distance to Epipolar line for all the matched points. On top of that, the decomposition of the Essential matrix to rotation and translation is not unique. In other words, there exist two sets of rotation and translation that compose the same Essential matrix, \( E = \hat{T}_1 R_1 = \hat{T}_2 R_2 \). Therefore, each decomposition \((R_i, T_i)\) is used to intersect all the matched points, and the true decomposition is the one that reconstructs more 3D points in front of the cameras. Due to these complexities, a 3D to 2D method might be preferred, which is explained in the next section.

2.2.3 Image resection upon existing 3D model (3D to 2D)

In a 3D to 2D approach, the new image is resected upon the existing 3D points. The Single Photo Resection (SPR) requires at least three non-collinear image/object points, and can be combined with RANSAC to robustly estimate the EOPs. Three-point resection dates back to the work of Grunert (1841) and many SPR algorithms have been developed so far. Interested readers can refer to Mazaheri and Habib (2014) for more information about the existing resection algorithms.

The 3D to 2D method might be preferred to the 2D to 2D method, since the minimum requirement of three points (compared to the five points) facilitate a faster pose estimation by RANSAC. In addition, the EOPs of the resected image are already in real scale, and no rescaling procedure is required unlike the 2D to 2D method.
2.2.4 Model to model transformation (3D to 3D)

The 3D to 3D method is based on a rigid body transformation between the existing 3D points and the 3D points in the new model. For example, Roth (2001) uses a trinocular camera rig and intersect the matched points among the cameras to obtain a set of 3D points at each acquisition epoch. The rigid body transformation of the common 3D points between successive epochs is computed by an absolute orientation algorithm such as Helmert or Horn method (1990). Therefore, the 3D motion of the multi-camera system is sequentially estimated, by computing the rigid body transformation between the 3D points.

Nister et al. (2004) note that the 3D to 3D method can be greatly inferior to the 3D to 2D method. The reason is higher uncertainty of the intersected points in the depth direction. As the 3D to 3D method computes the motion using 3D points, the depth uncertainty have a devastating effect on the results. However, the 3D to 2D method is less affected by the uncertainty in depth. The reason is that the “new camera pose is not far from the pose from where the 3D points was originally triangulated”, so the depth uncertainty of 3D points does not cause much change in the reprojected image position.

So far, image to image, model to image, and model to model motion estimation methods have been reviewed. For motion estimation of a multi-camera system, the Generalized Camera Model (GCM) (Grossberg and Nayar, 2001) can also be adapted. The GCM is a framework to use many cameras as one, regardless of their position and orientation. In the next section, the Generalized Camera Model is explained.

2.2.5 Generalized Camera Model

Grossberg and Nayar (2001) introduced the Generalized Camera Model to serve non-conventional imaging sensors for which the traditional pinhole model is highly restrictive. “A
GCM 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. There can be multiple centers of projection, or indeed no center of projection at all. This model is relatively general and includes cameras such as perspective cameras, fish-eye cameras, central or non-central catadioptric cameras, linear or non-linear pushbrum cameras, whiskbroom cameras, panoramic cameras, as well as multi-camera rigs and insect eyes” (Kim et al., 2010).

Pless (2003) adapts the GCM to develop generalized formulas for common photogrammetric algorithms such as generalized point intersection or generalized Epipolar constraint. The generalized Epipolar constraint can be adapted for motion estimation of a multi-camera system between two epochs. Kim et al. (2010) explain the generalized Epipolar constraint as follows. A light ray \( L \) can be defined by a unit direction \( x \) and a point \( c \) on the ray (e.g. \( x \) could be 3D position in image coordinate system and \( c \) could be the perspective center coordinate). The Plucker coordinate (Plucker, 1865) of a light ray is a vector with six elements, \( L = (x^T, (c \times x)^T)^T \). Two lines \( L \) and \( L' \) intersect if the condition in Equation (2.4) is met. The line \( L \) transforms to the line \( L_{Rot} \) by Euclidean transformation \( R \) and \( T \), using Equation (2.5), where \( E = \hat{T}R \).

\[
L^T \begin{bmatrix} 0_{3\times3} & I_{3\times3} \\ I_{3\times3} & 0_{3\times3} \end{bmatrix} L = 0. \tag{2.4}
\]
\[
L_{Rot} = \begin{bmatrix} R & 0 \\ E & R \end{bmatrix} L \tag{2.5}
\]

Figure 2.2 shows a light ray \( L \) passes through a point \( c \) and a vector \( x \), and a light ray \( L' \) passes through a point \( c' \) and a vector \( x' \). In case \( x \) and \( x' \) are 3D coordinates of two conjugate points in image coordinate system, the Euclidean transformation required to intersect these two lines is expressed as the form in Equation (2.6). Equation (2.6) is the generalized Epipolar
constraint, which is satisfied for every conjugate pair of light rays $L$ and $L'$. This equation can be expanded to the form in Equation (2.7), which can be employed to estimate the unknown Essential matrix $E$ and the rotation $R$ using the conjugate image points $x$ and $x'$ and the points $c$ and $c'$ ($c$, $c'$ can be perspective center coordinates).

![Diagram of light rays and conjugate points](image)

Figure 2.2: A light ray can be defined by a vector and a point.

$$0 = L^T \begin{bmatrix} 0 & I \\ I & 0 \end{bmatrix} L_{Rot} = L^T \begin{bmatrix} 0 & I \\ I & E \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix} L = L^T \begin{bmatrix} E & R \\ R & 0 \end{bmatrix} L. \tag{2.6}$$

$$x^T Ex + (c' \times x')^T Rx + x'^T R (c \times x) = 0 \tag{2.7}$$

In order to estimate the unknowns $E$ and $R$ using Equation (2.7), a system of linear equations is formed, by taking advantage of Kronecker product. Kronecker product $\otimes$ of two matrix $A_{p,q} \otimes B_{r,s}$ is defined as the form in Equation (2.8). For an equation system $AXB = C$, where $X$ is unknown, Kronecker product is used to form a linear equation system as expressed in Equation (2.9), where $Vec$ is an operator that stacks columns of a matrix into a vector, $Vec(A_{p,q}) = (a_{11}, a_{21}, ..., a_{p1}, a_{12}, ..., a_{p1}, ..., a_{1q}, ..., a_{pq})^T$. By applying the Kronecker product to Equation (2.7), Equation (2.10) is obtained, which is one equation to 18 unknowns - $Vec(E)$ and $Vec(R)$. Therefore, at least 17 conjugate points are required to solve for the unknowns up to a scale (17-point algorithm).
\[ A_{p,q} \otimes B_{r,s} = \begin{bmatrix} a_{11}B & \cdots & a_{1q}B \\ \vdots & \ddots & \vdots \\ a_{p1}B & \cdots & a_{pq}B \end{bmatrix} \quad (2.8) \]

\[(B^T \otimes A)\text{Vec}(X) = \text{Vec}(C) \quad (2.9)\]

\[(x^T \otimes x'^T)\text{Vec}(E) + (x^T \otimes (c' \times x')^T + (c \times x)^T \otimes x'^T)\text{Vec}(R) = 0. \quad (2.10)\]

Equation (2.10) can be adapted to estimate a multi-camera motion between two epochs. For this purpose, this Equation is established in a way that the Essential matrix \(E\) encodes the system rotation and translation between two epochs as \(E = \hat{T}R\), \(c\) and \(c'\) be the coordinate of two similar/different cameras at the two epochs, \(x\) and \(x'\) be conjugate image points/vector. Therefore, using the conjugate points in the images taken by similar/different cameras at two epochs, the system motion can be estimated by Equation (2.10). The advantage of motion estimation by generalized Epipolar constraint is that all the cameras are considered as one, and they all contribute to the system motion in one step.

In the next section, different representations of a rotation in 3D space are explained. Then, the concept of rotation averaging is discussed, which is used later in the proposed methodology to estimate motion of a multi-camera system.

2.3 Rotation representation and averaging

A rotation in 3D space can be represented by Euler angles, angle-axis or quaternions, each of which has its own pros/cons. Euler angles are simple and intuitive, but suffer from the known Gimbal lock problem. Gimbal lock is a phenomenon that two rotation axes end up being aligned in the same direction, which causes loss of one degree of freedom (Perumal, 2011). In addition, interpolating a rotation represented by Euler angles is not straightforward. Another way to represent a rotation in 3D space is to define an axis and an angle of rotation around the axis. Using
angle-axis representation, a rotation matrix in 3D space can be interpolated by linear interpolation of the rotation angle. Quaternions are defined with four parameters and frequently used for rotation representation, interpolation, and averaging without the Gimbal lock problem.

2.3.1 Euler angles

Euler angles represent a rotation by three sequential rotations about the X, Y, and Z axes in 3D space as expressed in Equation (2.11). Euler angles are denoted with different symbols in literature. In photogrammetry \( \omega, \phi, \) and \( \kappa \) are used to represent the rotations around the X, Y, and Z axis, respectively.

\[
R = R_\omega R_\phi R_\kappa
\]

\[
= \begin{bmatrix}
1 & 0 & 0 \\
0 & \cos(\omega) & -\sin(\omega) \\
0 & \sin(\omega) & \cos(\omega)
\end{bmatrix}
\begin{bmatrix}
\cos(\phi) & 0 & \sin(\phi) \\
0 & 1 & 0 \\
-\sin(\phi) & 0 & \cos(\phi)
\end{bmatrix}
\begin{bmatrix}
\cos(\kappa) & -\sin(\kappa) & 0 \\
\sin(\kappa) & \cos(\kappa) & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

(2.11)

It is known that the Euler angles suffer from the Gimbal lock problem. On top of that, the sequential nature of the Euler angles makes them difficult for interpolation or averaging rotations. For example, half of a rotation of \( R = R_\omega (20^\circ) R_\phi (40^\circ) R_\kappa (60^\circ) \) applied to a 3D point is completely different with \( R' = R_\omega (10^\circ) R_\phi (20^\circ) R_\kappa (30^\circ) \), and averaging the two rotations \( R \) and \( R' \), is not \( R_\omega (15^\circ) R_\phi (30^\circ) R_\kappa (45^\circ) \). To avoid the Gimbal lock problem and facilitate interpolation and rotation averaging, the Euler angles are often converted to angle-axis representation or quaternions, which are explained in the next sections.

2.3.2 Angle-axis representation

Figure 2.3 depicts a rotation of magnitude \( \theta \) around a unit axis \( \mathbf{v} = [v_1, v_2, v_3]^T \), which equals to a rotation matrix in 3D space. Therefore, four parameters are required for angle-axis representation, which the unity condition of \( \mathbf{v} \) reduces one degree of freedom. One can say that a
rotation in 3D space can be represented by a 3D vector $V$, and decomposed to angle-axis representation as $\theta = \|V\|, v = V/\|V\|$.

The Rodriguez formula (Rodrigues, 1816) relates the angle-axis representation to a rotation matrix as expressed in Equation (2.12), where $\tilde{v}$ is the skew symmetric matrix that represents the cross product as a matrix-vector product for any vector $w$ (Equation (2.13)),

$$R = I + \sin(\theta) \tilde{v} + (1 - \cos(\theta))\tilde{v}^2$$

(2.12)

$$v \times w = \tilde{v}w, \quad \tilde{v} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}.$$  

(2.13)

Figure 2.3: Rotation of angle $\theta$ around axis $v$

A rotation matrix $R$ equivalent to a rotation $\theta$ around an unit axis $v$, can also be expressed as exponential function $e^{\theta \tilde{v}}$ (Markley and Crassidis, 2014). The exponential representation can be proved using the Taylor expansion of the term $e^{\theta \tilde{v}}$, as shown in Equation (2.14). Given the fact that for a unit vector $v, \tilde{v}^3 = -\tilde{v}, \tilde{v}^4 = -\tilde{v}^2, \tilde{v}^5 = \tilde{v}, \tilde{v}^6 = \tilde{v}^2, \ldots$, Equation (2.14) can be written as the form in Equation (2.15). By comparing Equations (2.12) and (2.15), Equation (2.16) is obtained, which proves the exponential representation of a rotation matrix using the Rodrigues formula. Knowing the exponential representation, it is worthy to define the logarithm of a rotation matrix. Equation (2.17) expresses the logarithm of a rotation matrix, which equals to the multiplication of $\theta v$, where $R = e^{\theta \tilde{v}}$ (Hartley et al., 2013).
\[ e^{\theta \hat{v}} = \sum_{k=0}^{\infty} \frac{(\theta \hat{v})^k}{k!} = I + \theta \hat{v} + \frac{(\theta \hat{v})^2}{2!} + \frac{(\theta \hat{v})^3}{3!} + \cdots \] (2.14)

\[ e^{\theta \hat{v}} = I + \left( \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \cdots \right) \hat{v} + \left( \frac{\theta^2}{2!} - \frac{\theta^4}{4!} + \frac{\theta^6}{6!} - \cdots \right) \hat{v}^2 \] (2.15)

\[ = I + \sin(\theta) \hat{v} + (1 - \cos(\theta)) \hat{v}^2 \]

\[ R = e^{\theta \hat{v}} \] (2.16)

\[ \log(R) = \begin{cases} \arcsin(||y||_2) \frac{y}{||y||_2}, y \neq 0 \\ 0, y = 0 \end{cases}, y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}, \begin{bmatrix} 0 & -y_3 & y_2 \\ y_3 & 0 & -y_1 \\ -y_2 & y_1 & 0 \end{bmatrix} = \frac{1}{2}(R - R^T) \] (2.17)

2.3.3 Quaternions

Quaternions was first described by Hamilton (1845). A quaternion may be represented as a vector \( q = [q_0, q_1, q_2, q_3]^T = [q_0, q_{1:3}]^T \) along with a set of additional properties and operations. The conjugate of \( q \), is defined as \( q^* = [q_0, -q_{1:3}]^T \), and norm of \( q \) equals \( ||q|| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} \). A unit quaternion \( \hat{q} (||\hat{q}|| = 1) \), can represent a rotation in 3D space. A 3D vector \( z \) is rotated by a unit quaternion such as \( \begin{bmatrix} 0 \\ z_{rot} \end{bmatrix} = \hat{q} \begin{bmatrix} 0 \\ z \end{bmatrix} \hat{q}^* \), where symbol (.) stands for quaternion multiplication as expressed in Equation (2.18) (Diebel, 2006). The equivalent rotation matrix \( R \) of the unit quaternion \( \hat{q} \), is expressed in Equation (2.19), where \( z_{rot} = Rz \). Unit quaternions can also be converted to angle-axis representations; a rotation \( \theta \) around a unit vector \( v \) can be expressed by the unit quaternion as \( \hat{q}_{\theta v} = [\cos\left(\frac{\theta}{2}\right), v \sin\left(\frac{\theta}{2}\right)] \). As mentioned earlier, quaternions define a rotation in 3D space without the Gimbal lock problem. In addition, they can be used for interpolation of a rotation, and averaging a set of rotations as explained in the section 2.3.5.
\[ q, p = \begin{bmatrix} q_0p_0 - q_1^T p_{1:3} \\ q_0 p_{1:3} + p_0 q_{1:3} - q_{1:3} \times p_{1:3} \end{bmatrix} \] (2.18)

\[ R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2q_1q_2 + 2q_0q_3 & 2q_1q_3 - 2q_0q_2 \\
2q_1q_2 - 2q_0q_3 & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2q_2q_3 + 2q_0q_1 \\
2q_1q_3 + 2q_0q_2 & 2q_2q_3 - 2q_0q_1 & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \] (2.19)

In section 2.3.6 or the developed methodology (section 3.2), we will see procedures that produce several estimates for a rotation in 3D space. Angle-axis representation and quaternions can be used for averaging these rotation estimates, which are explained in the two following sections.

2.3.4 **Karcher mean of rotation**

For averaging a set of rotation in 3D space, a metric is required to measure the distance between two rotation matrices, such as angular distance. The angular distance \( d_\angle \) between two rotation matrices \( R_1 \) and \( R_2 \) is defined as the norm of the logarithm of \( R_1 R_2^T \) (Equation (2.20)) (Hartley et al., 2013). If \( R_1 = R_2 \), the product \( R_1 R_2^T \) equals identity. Therefore, the more different \( R_1 \) and \( R_2 \) are, the more matrix \( R_1 R_2^T \) would deviate from identity matrix. The deviation of \( R_1 R_2^T \) from the identity matrix – i.e. the angular distance between \( R_1 \) and \( R_2 \) – can be represented by the magnitude of the rotation angle (\( \theta \)) in the angle-axis representation of \( R_1 R_2^T \).

\[ d_\angle(R_1, R_2) = d_\angle(R_1 R_2^T, I) = \| \log(R_1 R_2^T) \| = \| \theta \mathbf{v} \| = \theta \] (2.20)

Using the angular distance, the Karcher mean (Grove et al., 1974; Karcher, 1977) of rotation matrices can be computed. For a set of \( n \) rotation matrices \( \{ R_1, R_2, ..., R_n \} \), the Karcher mean of rotation \( \bar{R} \) is defined as a rotation that satisfies Equation (2.21). In case of noise, this equation is not satisfied. Therefore, an algorithm is required to estimate a rotation \( \bar{R} \) that minimizes Equation (2.21).
\[ \frac{1}{n} \sum_{i=1}^{n} \log(\bar{R}^T R_i) = 0 \]  

(2.21)

Manton (2004) proposes the following iterative algorithm to find the Karcher mean of rotation \( \bar{R} \), which minimizes Equation (2.21). Assume \( n \) rotation matrices \( \{R_1, R_2, ..., R_n\} \).

- Set initial value \( \bar{R} = R_1 \)
- Loop
  - Compute \( r = \frac{1}{n} \sum_{i=1}^{n} \log(\bar{R}^T R_i) \)
  - If \( \|r\| < \text{Threshold} \), return \( \bar{R} \).
  - \( \bar{R} = \bar{R} \exp(r) \)

Therefore, this algorithm iteratively update the Karcher mean of rotation \( \bar{R} \), until no improvement is observed. In addition to the Karcher mean explained in this section, quaternions can also be employed to average a set of rotation as explained in the next section.

2.3.5 Rotation averaging by quaternions

Distance between two rotation matrices \( R_1 \) and \( R_2 \) may be defined as \( d(R_1, R_2) = \|\mathbf{q}_1 - \mathbf{q}_2\| \), where \( \mathbf{q}_1 \) and \( \mathbf{q}_2 \) are the unit quaternions corresponding to \( R_1 \) and \( R_2 \), respectively. But, the representation of a rotation matrix by quaternions is not unique as both \( \pm \mathbf{q} \) represent the same rotation matrix. Therefore, the distance between two quaternions \( d_{\text{quat}} \) must be computed by checking both signs of the quaternions, and choosing the one that minimizes the distance, as expressed in Equation (2.22) (Hartley et al., 2013).

\[ d_{\text{quat}}(R_1, R_2) = \min\{\|\mathbf{q}_1 - \mathbf{q}_2\|, \|\mathbf{q}_1 + \mathbf{q}_2\|\} \]  

(2.22)

Having defined the quaternion distance between two rotation matrices, the rotation averaging by quaternions is explained as follows. Given a set of \( n \) rotation matrices
\{R_1, R_2, ..., R_n\} and their equivalent unit quaternions \{\dot{q}_1, \dot{q}_2, ..., \dot{q}_n\}, the mean rotation quaternion \(\ddot{q}\) must minimize the summation in Equation (2.23), over all the choices of \(\varepsilon_i = \pm 1\). Hartley et al. (2013) proves that in case all of the rotations \(R_i\) lie within an angular distance of \(\frac{\pi}{2}\) of each other, the quaternion \(\ddot{q}\) that minimizes Equation (2.23) is computed from Equation (2.24). The quaternion \(\ddot{q}\) can then be converted to a rotation matrix, which is the mean of \{\(R_1, R_2, ..., R_n\)\}.

\[
\sum_{i=1}^{n} ||\ddot{q} - \varepsilon_i \dot{q}_i||^2 = \frac{1}{n} \sum_{i=1}^{n} \dot{q}_i^T \ddot{q}
\]

(2.23)

\[
\ddot{q} = \frac{\ddot{q}}{||\ddot{q}||}, \ddot{q} = \sum_{i=1}^{n} \dot{q}_i
\]

(2.24)

In the next section, a method is explained by which several rotations are estimated for the system rotation between two epochs. The rotation estimates are then averaged by a rotation averaging method already explained.

2.3.6 Motion estimation

For all overlapping image pairs sharing five points or more, ROPs can be estimated up to an arbitrary scale. Assume the rotation and translation estimated between three overlapping images \(I_1, I_2\) and \(I_3\) are denoted as \([R_{12}^2, r_{12}^2]\), \([R_{13}^2, r_{13}^2]\), and \([R_{23}^2, r_{23}^2]\). These rotations and translations are related/dependent to each other, or in other words, they must be compatible. For example, although the rotation \(R_{23}^3\) is estimated independently, but it must be compatible with the product \(R_{13}^2 (R_{12}^2)^T = R_{23}^3\). Therefore, there exist compatibility constraints between the ROPs estimated between the overlapping images. In a multi-camera system, the motion of individual cameras are also related/dependent to each other, as they are fixed on a platform and move together. Therefore, compatibility constraints can also be applied on the ROPs estimated between the cameras in a multi-camera system at two different epochs. Each relative rotation between a camera at epoch \(t_1\)
and a camera at epoch $t_2$ can provide an estimate for the rotation of the multi-camera system from epoch $t_1$ to $t_2$. The rotation estimates are then averaged to robustly compute the system rotation between the two epochs. After estimating the system rotation, the real scale translation of the system is estimated by solving a system of linear equations.

Depending on the configuration of the cameras in a multi-camera system, e.g. overlapping or non-overlapping, different number of ROPs between the cameras at two epochs can be estimated. Overlapping configuration of a multi-camera system allows us to estimate more ROPs between the cameras at two epochs. Therefore, more estimates for the system rotation are provided, which improves the rotation averaging accuracy. Moreover, the multi-camera system translation is solved with higher redundancy in an overlapping multi-camera system.

Kim et al. (2007) propose a solution for motion estimation of a non-overlapping multi-camera system by rotation averaging. After data acquisition with a non-overlapping multi-camera system, it is only possible to estimate the motion of each camera separately from time $t_1$ to $t_2$. The individual cameras ($c_i$) rotation and translation from time $t_1$ to $t_2$ are denoted as $R^c_{c_i(t_2)}, r^c_{c_i(t_2)}$, respectively. For simplicity, Kim et al. (2007) multiply the known rotation between the cameras in a multi-camera system to the image points, and then assume the mounted cameras are parallel to each other – i.e., there is only a translation between them. With this assumption, the individual camera rotations $R^c_{c_i(t_2)}$ are directly an estimate for the system rotation $R^{t_1}_{t_2}$ (Equation (2.25)). Therefore, in a non-overlapping multi-camera system, each camera (motion) provides an estimate for the system rotation $R^{t_1}_{t_2}$. They average the multiple rotation estimates using the quaternion method (section 2.3.5) for robust estimation of the system rotation.

$$R^{t_1}_{t_2} = R^c_{c_i(t_2)}$$

(2.25)
Figure 2.4 shows a non-overlapping stereo camera in two epochs. Using Figure 2.4, the estimated translation $r_{ci(t2)}^{cr}$ of each camera from time $t_1$ to $t_2$, can be related to one of the cameras (reference camera $c_r$), by applying the vector summation rule as expressed in Equation (2.26), where the factor $\lambda_i$ is used for scaling $r_{ci(t1)}^{cr}$ to real scale. The system rotation $R_{t_2}^{t_1} = R_{cr(t2)}^{cr(t1)} = R_{ci(t2)}^{cr(t1)}$ is already estimated by rotation averaging, and the translation of the cameras relative to the reference camera $r_{ci}^{cr}$ are known (from calibration or manufacture). The translation of the reference camera $r_{cr(t2)}^{cr}$ equals to the translation of the multi-camera system. Equation (2.26) is a set of three equations with four unknowns (three for $r_{ci}^{cr(t1)}$ and one for $\lambda_i$), and can be established for every camera in a non-overlapping multi-camera system.

$$r_{cr(t2)}^{cr(t1)} = r_{ci}^{cr} + \lambda_i r_{ci(t2)}^{ci(t1)} - R_{t_2}^{t_1} r_{ci}^{cr}$$

(2.26)

![Diagram showing motion of a non-overlapping stereo camera.](image)

Figure 2.4: Motion of a non-overlapping stereo camera

Kim et al. (2007) see Equation (2.26) as an intersection problem, where $r_{cr(t2)}^{cr}$ is assumed as an object point coordinate and $\lambda_i$ is the unknown depths of the object point relative to the camera $i$. They solve this problem using the norm-infinity intersection approach introduced in Hartley and Schaffalitzky (2004). “Norm-infinity optimization can be regarded as something between the
norm-2 and linear algorithms. The norm-infinity framework inherits good properties from both of these alternative approaches. For example global estimates are guaranteed with a geometrically meaningful cost function and at a reasonable computation cost. A potential disadvantage is that norm-infinity is not robust to outliers” (Kahl, 2005).

So far, the related work in feature extraction and motion estimation were reviewed. In the next section, a mathematical model to relate the object space to the image space in a multi-camera system is explained. This model is used in bundle adjustment for multi-camera calibration as well as refinement of the estimated motion.

2.4 Relating object space to image space in a multi-camera system

Assume a multi-camera system in which a set of cameras are fixed on a platform. In this case, the cameras are expected to translate and rotate together with the same magnitude. To describe the motion of the multi-camera system, two coordinate systems are defined. The first one is the world coordinate system where EOPs of all the images and 3D coordinates of the object points are described in this coordinate system. The second one is a local coordinate system defined by the multi-camera system. The constant rotation and translation of the mounted cameras relative to each other are defined in the local coordinate system. Frahm et al. (2004) defines the origin of the local coordinate system on a virtual camera. Then, the rotations and translations of the actual cameras are defined w.r.t this virtual camera. Habib et al. (2011) defines the local coordinate system on one of the cameras in the system - referred to a reference camera.

The rotation and translation of the mounted cameras $c_i$ relative to the reference camera $c_r$ are denoted as $R_{c_i}^{c_r}, t_{c_i}^{c_r}$, respectively, and are referred to the mounting parameters in this work. The rotation and translation of a camera $c_i$ relative to the world coordinate system, $R_{c_i}^{w}, t_{c_i}^{w}$, are related
to those of the reference camera \( c_r \) using the mounting parameters \( h_{c_l}^{c_r}, r_{c_l}^{c_r} \). Therefore, it would be enough to estimate the EOPs of the reference camera at each acquisition epoch by which EOPs of the other cameras can be computed using the mounting parameters (Habib et al., 2011). Figure 2.5 shows the vector summation that relates the position of an object point \( P \) in the world coordinate system \( r_p^w \), to the position of the reference camera in world coordinate system \( r_{c_r}^w \) and the position of the camera \( i \) relative to the reference camera \( r_{c_l}^{c_r} \). The point/vector \( r_p^{c_i} \) is the projection of the object point \( P \) into the camera \( i \).

Figure 2.5: The vector summation relating an object point position to the reference and a non-reference camera (Kersting et al., 2011)

The vector summation depicted in Figure 2.5, can be expressed mathematically in Equation (2.27), which relates the position of the object point \( P \) in the world coordinate system \( r_p^w \) to the position of the object point \( P \) relative to the coordinate system of the camera \( i, r_p^{c_i} \) (Rau et al., 2011). By re-arranging the terms in Equation (2.27), Equation (2.28) is obtained. The vector \( r_p^{c_i} \) is a factor (\( \lambda \)) of the vector \( r_p^{c_i} \) as expressed in Equation (2.29), where \( x_p \) and \( y_p \) are the image coordinate of the projection of the object point \( P \) into the camera \( i \). The terms \( \Delta x \) and \( \Delta y \) compensate the lens distortions, \( x_p^{pp} \) and \( y_p^{pp} \) are the coordinate of the principal point, and \( c^{c_i} \) is the principal distance of the camera \( i \). By substituting the right side of Equation (2.28) with \( N \), the
scale factor $\lambda$ can be computed as Equation (2.30). By comparing Equations (2.28), (2.29), and (2.30), the collinearity equations for a multi-camera system is obtained (Equation (2.31)). The mathematical model to compensate radial and de-centric lens distortion is expressed in Equation (2.32), where $K_1$ and $K_2$ are radial lens distortion coefficients, $P_1$ and $P_2$ are de-centric lens distortion coefficients, and $A_1$ and $A_2$ compensate for any non-orthogonality of the frame axes. The collinearity equations are used for calibrating the multi-camera system and refining the system motion through bundle adjustment.

\[
\begin{align*}
r_p^w &= r_c^w + R_c^w r_c^l + R_c^w R_c^l r_p^c_i \tag{2.27} \\
r_p^c_i &= R_c^c_i (R_w^c (r_p^w - r_c^w) - r_c^c_l) \tag{2.28} \\
r_p^c_i &= \lambda r_p^c_i, \quad r_p^c_i = \begin{bmatrix} x_p - x_{pp}^c - \Delta x \\
y_p - y_{pp}^c - \Delta y \\
- c_{ci} \end{bmatrix} \tag{2.29} \\
N &= [N_x, N_y, N_z]^T = R_c^c_i (R_w^c (r_p^w - r_c^w) - r_c^c_l) = \lambda r_p^c_i \Rightarrow \lambda = \frac{N_z}{-c_{ci}} \tag{2.30} \\
x_p &= x_{pp}^c - c_{ci} \frac{N_x}{N_z} + \Delta x \\
y_p &= y_{pp}^c - c_{ci} \frac{N_y}{N_z} + \Delta y \tag{2.31} \\
\Delta x &= K_1 r^2 \bar{x} + K_2 r^4 \bar{x} + P_1 (r^2 + 2 \bar{x}^2) + 2 P_2 \bar{x} \bar{y} - A_1 \bar{x} + A_2 \bar{y} \\
\Delta y &= K_1 r^2 \bar{y} + K_2 r^4 \bar{y} + P_2 (r^2 + 2 \bar{y}^2) + 2 P_1 \bar{x} \bar{y} + A_1 \bar{y} \tag{2.32} \\
\bar{x} &= x_p - x_{pp}^c, \quad \bar{y} = y_p - y_{pp}^c, \quad r^2 = \bar{x}^2 + \bar{y}^2
\end{align*}
\]

### 2.5 Summary

In this chapter, the existing methods for feature detection and description were reviewed first. Processing of 2D images to detect stable features and match them robustly is essential to recover 3D motion/structure of the scene accurately. Afterwards, the major motion estimation methods were discussed, focusing on the rotation averaging. Then, the mathematical model
relating the object space to the image space in a multi-camera system was described in detail, which is the core of the bundle adjustment employed in this research for multi-camera system calibration and motion refinement. In the next chapter, the proposed methodology in this research for multi-camera motion estimation is explained.
Chapter 3: Methodology

In this chapter, the proposed methodology for motion estimation of a multi-camera system in indoor environments is explained. As mentioned in section 1.2, utilizing multiple cameras has many advantages compared to a single camera. The overall field of view of the system is increased, redundant data is collected, and the necessary information to recover the 3D model in real scale is provided. The number and configuration of the cameras in a multi-camera system depend on the application and the usage environment. For example, a multi-camera system could have two cameras (stereo cameras) such ATOS (GOM Co.) and Bumblebee (Point Grey Co.), or could have hundreds of cameras mostly used in film industries. In a stereo camera system, the ROPs of the cameras might be provided by the manufacture or estimated through a calibration procedure. Using the available ROPs, the Epipolar geometry can be established to verify the matched features between the two cameras, by checking their distance to the Epipolar lines. Unfortunately, the outlier matches along the Epipolar lines are not detectable by Epipolar geometry, as they can pass the threshold for the distance to the Epipolar line and labeled as inliers. The majority of the outlier matches along the Epipolar lines happen when the data acquisition system moves parallel to a direction in which a particular texture is repeated. For example, assume a camera mounted on a cart is being pushed straight through a corridor, while taking images of a side wall. Repetitive pattern of wall bricks might result in many outliers in feature matching. Since the bricks are repeated parallel to the moving direction, the number of outlier matches along this direction could be significant. As the Epipolar lines in adjacent images are almost parallel to the moving direction, the outlier matches in the direction of motion/Epipolar lines are not detectable by the Epipolar geometry. To eliminate the potential outliers along the Epipolar lines and strengthen the geometry,
three or more cameras can be used. The geometry of three or more cameras allows us to verify the matched features along and across the Epipolar lines, by looking into their intersection error.

To reconstruct an imaged scene, the system motion parameters at each data acquisition epoch have to be estimated accurately. Figure 3.1 shows the flowchart of the methodology proposed for estimating 3D motion of a multi-camera system. The proposed methodology can be divided into three major steps. In the first step, salient features in each image are extracted and matched with neighboring images that have been captured at the same epoch (spatial matching) or across the epochs (temporal matching). The spatial and temporal matching procedure is explained in section 3.1. Spatial matches are checked for outliers along and across the Epipolar lines using the known mounting parameters. For temporal matches, RANSAC in conjunction with the 5-point or 8-point algorithm is used to detect outliers across the Epipolar lines.

In the second and major step of the proposed methodology, the system motion is estimated (section 3.2). To estimate the system motion, a method is proposed that estimates the system rotation first and then the system translation between successive epochs is computed. In this method, several estimates for the system rotation are obtained by relating the ROPs between the cameras across the epochs to the system rotation using the mounting parameters. Then, incompatible rotation estimates are filtered out, and compatible ones are averaged for robustness. Then, the system translation between successive epochs is estimated by solving a system of linear equations.

In the third step, the estimated system motion is refined. First, outliers along the Epipolar lines – which are not detectable in the first step – are filtered out using the estimated system motion. Then, bundle adjustment in a sliding fashion is used to refine the 3D structure and motion.
Depending on the system resources, bundle adjustment can be executed globally to optimize the entire system motion and the recovered 3D scene.

![Flowchart of the proposed methodology for motion estimation](image)

**Figure 3.1: Flowchart of the proposed methodology for motion estimation**

### 3.1 Feature extraction and matching

The first step in structure and motion recovery is feature extraction and matching. Indoor environments are rich in corner features, such as brick corners, windows, doors, bulletin boards, tables, and flooring. Therefore, we choose to detect corners using the well-known Harris operator. First, the Harris response $(H)$ of every point in an image is computed using Equation (2.3). Then, corners are detected by thresholding the Harris response of the image points. Simple thresholding of the Harris response may result in accumulation of corners in rich textured area of the image. To avoid this problem and ensure good distribution of features across the image, close features should be purged down. A simple purging algorithm is to create a list of features sorted by their quality (Harris response), and starting from top feature in the list, remove all the features below that fall within a given spatial distance (Ferruz and Ollero, 2000; Ma et al., 2004; Pollefeys, 2002).

To detect stable and repeatable corners, Harris operator can be applied at higher level of image pyramid, which has less details and noise compared to the original one. Detected features at the higher level of image pyramid are transferred down to the original level, followed by a sub-
pixel refinement. Sub-pixel refinement of corners detected by the Harris operator is quite important for accurate motion and structure recovery, as shown in the work of Sroba (2015). In this research, the iterative sub-pixel refinement algorithm available in OpenCV is used (opencv.org), which is based on the fact that a vector from a corner $c$ to any neighboring point $p$, is perpendicular to the intensity gradient at $p$ ($\overrightarrow{cp}.\nabla I_p = 0$). Figure 3.2 shows a corner and examples of the $\overrightarrow{cp}$ vector. As seen in that figure, the equation $\overrightarrow{cp}.\nabla I_p = 0$ is valid only for a corner point $c$, whether $p$ is located on a flat region on which $\nabla I_p = 0$, or on an edge that $\overrightarrow{cp}.\nabla I_p = 0$. The corner refinement algorithm iteratively update the location of $c$ to minimize the summation of $\overrightarrow{cp}.\nabla I_p$ over all the points $p$ in a given neighboring area (e.g. 7 pixels).

![Figure 3.2: Sub-pixel refinement of a corner](image)

After detecting corner features, the SIFT descriptor is evaluated for each detected feature. SIFT descriptor is a normalized 128-dimension vector, which is invariant to scale, affine deformation, and illumination changes of an image. The Euclidean distance of two feature descriptors is considered as the similarity measure between the two features. To match the features between two images, for all the features in the first image, the closest features in the second image are found by computing the Euclidean distance of the descriptors (i.e. forward matching). In addition, for all the features in the second image, the closest features in the first image are found
(i.e. backward matching). The common matches between the forward and backward matching procedures are considered as reliable ones.

Features of each image are matched to the neighboring images at the same acquisition epoch (spatial matching) and to the neighboring images at the other epochs (temporal matching). Figure 3.3 shows an example of spatial and temporal matches of point $P$ in a sequence images, captured by three cameras ($c_1, c_2, c_3$) at three epochs ($t_1, t_2, t_3$).

![Figure 3.3: Spatial and temporal matches in a sequence of epochs](image)

Although forward-backward descriptor matching discards many incorrect matches, a significant number of outliers still remain. Outliers within temporal and spatial triple/double matches are detected using the procedures explained in sections 3.1.1 and 3.1.2. One should note that a spatial double match connected to a temporal match can be treated as a triplet (spatial-temporal triple match), and verified by the intersection error (like a spatial triple match). But it is not possible since the system motion between the two epochs is unknown at this stage. Therefore, verifying triplets of spatial-temporal matches is the first thing to be carried out after the estimating the system motion (section 3.2.4).
3.1.1 Detection of outliers across the Epipolar lines in temporal matches

To verify the temporal matches, RANSAC in conjunction with the 5-point algorithm is used to estimate the Essential matrix \((E)\) between the two images. Knowing the Essential matrix, the Epipolar constraint can be defined as \(x_2^T E x_1 = 0\), where \(x_1 = [x_1, y_1, -c_1]^T\) and \(x_2 = [x_2, y_2, -c_2]^T\) are the coordinates of the conjugate points in the image coordinate system. To verify the points \(x_1\) and \(x_2\), the symmetric Epipolar distance of \(x_1\) and \(x_2\) is computed as follows.

Figure 3.4 shows the Epipolar lines for two conjugate image points. The Epipolar lines in the left and right images are defined as \(l_1 = E^T x_2\) and \(l_2 = E x_1\), respectively. It can be seen when there is no noise, the points \(x_1\) and \(x_2\) are on the Epipolar lines, i.e., \(x_1.l_1 = 0\), and \(x_2.l_2 = 0\), respectively. In presence of noise, the symmetric Epipolar distance is computed by first defining \(l_1 = [i_1, j_1, k_1]^T\), \(l_2 = [i_2, j_2, k_2]^T\). Then, the Epipolar line in the 2D space of the left and right images are formulated as \(i_1x + j_1y - k_1c_1 = 0\) and \(i_2x + j_2y - k_2c_2 = 0\), respectively. The distance of the points \(x_1\) and \(x_2\) to the Epipolar lines are computed by Equation (3.1) and Equation (3.2), respectively, where the operator \((\ )_i\) refers to the \(i\)-th element of the vector. The root of sum of squared distances to the Epipolar lines is called the symmetric Epipolar distance \(d\), as expressed in Equation (3.3) (Hartley and Zisserman, 2003). The distance to Epipolar \(d\) is used in RANSAC iterations to find the number of inliers/outliers in the Epipolar geometry established by an Essential matrix computed from the random sample of five points.

![Figure 3.4: Distance to Epipolar lines](image-url)
\[
d_1 = \frac{|x_1l_1|}{\sqrt{a_1^2 + b_1^2}} = \frac{|x_1^TE^Tx_2|}{\sqrt{(E^Tx_2)_1^2 + (E^Tx_2)_2^2}} \Rightarrow d_1^2 = \frac{(x_1^TE_1E_2)^2}{(E^Tx_2)_1^2 + (E^Tx_2)_2^2} \quad (3.1)
\]

\[
d_2 = \frac{|x_2l_2|}{\sqrt{a_2^2 + b_2^2}} = \frac{|x_2^TE_1E_2x_1|}{\sqrt{(E^Tx_1)_1^2 + (E^Tx_1)_2^2}} \Rightarrow d_2^2 = \frac{(x_2^TE_1E_2x_1)^2}{(E^Tx_1)_1^2 + (E^Tx_1)_2^2} \quad (3.2)
\]

\[
d^2 = d_1^2 + d_2^2 = (x_2^TE_1E_2x_1)^2\left(\frac{1}{(E^Tx_2)_1^2 + (E^Tx_2)_2^2} + \frac{1}{(E^Tx_1)_1^2 + (E^Tx_1)_2^2}\right) \quad (3.3)
\]

Although RANSAC and the 8-point algorithm is faster than the 5-point one for outlier detection, the 5-point algorithm is more accurate than the 8-point algorithm. The 5-point algorithm is more accurate because the constraints on the Essential matrix, \(\text{det}(E) = 0\) and \(2EE^TE = \text{trace}(EE^T)E\) are enforced (Ma et al., 2004), which is not the case in the 8-point algorithm. But the 5-point algorithm is slower because several hypothesis for \(E\) are generated. Each of which should be tested to find the best \(E\) - which minimizes \(x_2^TE_1E_2x_1\) over all the conjugate points. Since the ROPs between the images are required later in this research (for section 3.2), RANSAC and the 5-point algorithm are used for outlier detection, which estimates robust Essential matrices to be decomposed later to ROPs.

3.1.2 Detection of outliers in spatial matches

The ROPs between the images at the same epoch, i.e., mounting parameters, can be used to detect outliers of spatial matches. In this regard, spatial triple matches (see Figure 3.3) can be verified by multiple-ray intersection (Slabaugh et al., 2001). If the back projection error of the intersected point into the images is higher than a given threshold, the spatial triple match is labeled as outlier. To verify double spatial matches, the symmetric Epipolar distance is used (Equation (3.3)), using the Essential matrix composed from the mounting parameters. The Essential matrix can be composed from the mounting parameters as follows. Figure 3.5 depicts a reference camera
and two non-reference cameras. The vectors $r_p^{ci}$ and $r_p^{cj}$ are the light rays corresponding to the image vector $r_p^{ci}$ and $r_p^{cj}$, respectively. Vector $B$ is defined as the base line between the cameras $c_i$ and $c_j$ in the reference camera coordinate system (Equation (3.4)). Considering the co-planarity condition for the three vectors $B$, $r_p^{cj}$, and $r_p^{ci}$, the cross product $B \times R_{c_j}^{cr} r_p^{cj}$ must be perpendicular to the vector $R_{c_j}^{cr} r_p^{ci}$. In other words, the dot product in Equation (3.5) must be zero. Then, the Essential matrix is defined as $E = R_{cr}^{cl} \hat{B} R_{c_l}^{cr}$, which relates the conjugate points in the two cameras $r_p^{ci}, r_p^{cj}$ as expressed in Equation (3.6).

\[
B = r_{c_i}^{cr} - r_{c_j}^{cr}
\]  
\[
B \times R_{c_j}^{cr} r_p^{cj} \perp R_{c_i}^{cr} r_p^{ci} \Rightarrow \hat{B} R_{c_j}^{cr} r_p^{cj} \cdot R_{c_i}^{cr} r_p^{ci} = (r_p^{cj})^T R_{c_i}^{cr} \hat{B} R_{c_i}^{cr} r_p^{ci} = 0
\]  
\[
(r_p^{cj})^T E r_p^{ci} = (r_p^{cj})^T E r_p^{ci} = 0, \quad E = R_{cr}^{cl} \hat{B} R_{c_l}^{cr}
\]  

Figure 3.5: The reference camera coordinate system and two non-reference cameras

At this stage, the outliers within temporal and spatial matches have been filtered out. In the next section, the proposed method for motion estimation of the multi-camera system is explained. In this method, the system rotation is estimated first, then the translation is estimated by solving a system of linear equations.
3.2 Motion estimation

In the feature matching and outlier detection stage, the Essential matrix between the images across the epochs are estimated using RANSAC and the 5-point algorithm. The Essential matrix between two images can be decomposed to the ROPs of the cameras (e.g. $c_i$ and $c_j$) between two epochs $t_1$, and $t_2$, $R_{c_j(t_2)}^{c_i(t_1)}, r_{c_j(t_2)}^{c_i(t_1)}$. Taking advantage of the known mounting parameters, the ROPs between the cameras can be related to the reference camera $c_r$ rotation $R_{c_r(t_2)}^{c_r(t_1)}$, $r_{c_r(t_2)}^{c_r(t_1)}$, which describe the system rotational and translational motion, respectively. In the proposed method, the system rotation $R_{c_r(t_2)}^{c_r(t_1)}$ is estimated first, then the system translation $r_{c_r(t_2)}^{c_r(t_1)}$ is estimated by solving a system of linear equations.

3.2.1 Rotation estimation

Assuming $n$ cameras are mounted on the system platform with an overlapping field of view. A maximum number of $n^2$ relative rotations can be estimated between the reference/non-reference cameras at two epochs. Each relative rotation of the reference/non-reference cameras can estimate the reference camera rotation between two epochs $R_{c_r(t_2)}^{c_r(t_1)}$, by taking advantage of the mounting parameters. Therefore, $n^2$ estimates for the rotation $R_{c_r(t_2)}^{c_r(t_1)}$ are obtained and can be averaged using the algorithms explained in section 2.3.

Figure 3.6 shows the reference camera and a non-reference camera at two data acquisition epochs ($c_r(t_1), c_r(t_2), c_i(t_1), c_j(t_2)$), and the ROPs between them. The ROPs between the cameras at the same epoch (mounting parameters) are not time dependent as the cameras are tightly fixed on the platform, therefore the time symbols $t_1$ or $t_2$ are not used for them.
Figure 3.6: Reference and a non-reference cameras at epochs $t_1$ and $t_2$, and all possible relative orientations parameters between the cameras

From Figure 3.6, one can note that the rotation of the reference camera between two epochs (system rotation) $R_{cr(t_1)}^{cr(t_2)}$ can be estimated under four different scenarios:

1- Rotation between a pair of non-reference cameras at different epochs $R_{cj(t_2)}^{ci(t_1)}$, can be related to the rotation of the reference camera $R_{cr(t_1)}^{cr(t_2)}$ as expressed in Equation (3.7), where $R_{ci}$ and $R_{cr}$ are the known mounting parameters. Therefore, Equation (3.7) provides an estimate for the system rotation $R_{cr(t_1)}^{cr(t_2)}$.

$$R_{cr(t_1)}^{cr(t_2)} = R_{ci}^{cr} R_{cj(t_2)}^{ci} R_{cr}^{cj}$$  \(3.7\)

2- Rotation between the reference camera at epoch $t_1$ and a non-reference camera at $t_2$, $R_{cr(t_1)}^{cr(t_2)}$, is related to the system rotation $R_{cr(t_1)}^{cr(t_2)}$ as expressed in Equation (3.8).

$$R_{cr(t_1)}^{cr(t_2)} = R_{cj(t_2)}^{cr(t_1)} R_{cr}^{cj}$$  \(3.8\)

3- Rotation between the reference camera at $t_2$, and a non-reference camera at $t_1$, $R_{cr(t_2)}^{ci(t_1)}$, is related to the system rotation $R_{cr(t_2)}^{cr(t_1)}$ using Equation (3.9).

$$R_{cr(t_2)}^{cr(t_1)} = R_{ci}^{cr} R_{cr(t_2)}^{ci}$$  \(3.9\)
4- The estimated rotation between the reference cameras at two epochs $t_1$ and $t_2$, is a direct estimate for the system rotation $R_{c_r(t_1)}^{c_r(t_2)}$.

Ideally, the rotation estimates $R_{c_r(t_1)}^{c_r(t_2)}$ computed by any of the four equations above should be compatible. However, the accuracy of the rotation estimates is limited by the accuracy of the employed relative orientation method (e.g., Nister 5-point), and also affected by the mounting parameters accuracy. Therefore, there might be incompatible rotations within the estimates of $R_{c_r(t_1)}^{c_r(t_2)}$, which can have a devastating effect on the averaged rotation. To avoid this problem, incompatible rotation estimates are filtered out before averaging the rotation estimates. In this research, an iterative method is proposed to detect incompatible rotation estimates, which is explained in the next section.

3.2.2 Detection of incompatible rotation estimates

This part of the research aims at finding incompatible rotations within the estimated rotations in the previous section. For this purpose, a metric for compatibility error between two rotation matrices is introduced, which is in pixel unit. Having two rotation matrices $R_1$ and $R_2$, the matrix $\Delta R$ is defined as the product $R_1 R_2^T = \Delta R$. The more closer $R_1$ and $R_2$ are, the closer the matrix $\Delta R$ is being to identity matrix. Inspired by Habib et al. (2014), a syntactic grid is defined on the image to evaluate the closeness of $\Delta R$ to identity. The rotation $\Delta R$ is multiplied to the grid of image points and the resulting shift relative to their original position is measured. Figure 3.7 depicts the shift caused by applying a rotation $\Delta R$ to the grid of image points. Therefore, the magnitude of this shift indicates the closeness of $\Delta R$ to identity, or in other words, indicates the degree of similarity between $R_1$ and $R_2$. The more closer $R_1$ and $R_2$ are, the smaller is the shift. Using this fact, the distance between a pair of rotations $R_1$ and $R_2$ is defined as the Root Mean
Squared Error (RMSE) after applying $\Delta R$ to the $k$ corners of a grid $G$ imposed on an image (Figure 3.7). Equation (3.10) expresses the formula for the distance $D$ between two rotation matrices, where $x$ and $y$ are the image coordinates the grid points, and $c$ is the principal distance.

$$D^2(R_1, R_2) = \frac{1}{k} \sum_{x, y \in G} \left\| \Delta R \begin{bmatrix} x \\ y \\ -c \end{bmatrix} - \begin{bmatrix} x \\ y \\ -c \end{bmatrix} \right\|^2$$  \hspace{1cm} (3.10)

Figure 3.7: The shift caused by applying the rotation $\Delta R$ to a grid of image points

Using the introduced distance function between two rotation matrices in Equation (3.10), the compatibility error $e_i$ of a $i$-th rotation estimate $R^{cr(t_2)}_{i^{cr(t_1)}}$ is defined as the root mean squared error/distance of $R^{cr(t_1)}_{i^{cr(t_2)}}$ to the other rotation estimates. Equation (3.11) expresses the compatibility error $e_i$, where $n^2$ is the number of rotation estimates, and the denominator for the averaging equals $n^2-1$, since one of the distances is always zero, $D \left( R^{cr(t_1)}_{i^{cr(t_2)}}, R^{cr(t_1)}_{j^{cr(t_2)}} \right) = 0$.

$$e_i^2 = \frac{1}{n^2 - 1} \sum_{j=1 \& j \neq i}^{n^2} D^2 \left( R^{cr(t_1)}_{i^{cr(t_2)}}, R^{cr(t_1)}_{j^{cr(t_2)}} \right)$$  \hspace{1cm} (3.11)

Using the introduced formula (Equation (3.11)), the compatibility error of each rotation estimate is computed first. Then, an iterative algorithm is employed to detect the incompatible
ones. In case the standard deviation of the compatibility errors are above a given threshold, the rotation matrix that is most incompatible – i.e. has the highest compatibility error – is thrown out. Then, the compatibility errors are re-computed for the remaining rotations. This procedure continues until compatibility errors of all the rotations become almost similar, i.e., the standard deviation of the compatibility errors becomes smaller than the given threshold. This algorithm is summarized as follows:

- Define a set of compatibility errors \( e = \{e_i, i = 1, \ldots, n^2\} \). Initially label all of them as compatible.
- While standard deviation of \( e > \text{Threshold} \),
  - Throw out the rotation estimate \( R_{k c_r(t_2)}^{c_r(t_1)} \) as incompatible, if \( e_k = \max(e) \)
  - Re-compute \( e_i \) using compatible ones by Equation (3.11).

After throwing the incompatible rotations out, the Karcher mean of rotation estimates \( \bar{R}_{c_r(t_2)}^{c_r(t_1)} \) can be computed (section 2.3.4). Then, the system translation \( \bar{r}_{c_r(t_2)}^{c_r(t_1)} \) is estimated using a method explained next.

3.2.3 Translation estimation

According to Figure 3.6, the ROPs can be estimated between all the cameras at two epochs by RANSAC and 5-point algorithm. Therefore, \( n^2 \) translations are estimated between the cameras, each of which is up to a scale. Using the constant translation between the mounted cameras (e.g. \( r_{c_i}^{c_r} \)) and the estimated system rotation \( \bar{R}_{c_r(t_2)}^{c_r(t_1)} \), the translations of the reference/non-reference cameras at two epochs \( t_1 \) and \( t_2 \), can be related to the real scale translation of the system \( \bar{r}_{c_r(t_2)}^{c_r(t_1)} \). One should note that since the estimated translation between the cameras across the epochs is up
to an arbitrary scale, for each a rescaling factor needs to be estimated. Similar to the rotation estimation, there are four scenarios under which the system translation can be estimated:

1- Translation between a pair of non-reference cameras at different epochs, \( r_{ci}^{c}(t) \), can be related to the real scale translation of the system \( \overline{r}_{cr}^{c}(t) \), as expressed in Equation (3.12), where \( \lambda_{ci}^{c}(t) \) is the unknown rescaling factor.

\[
\lambda_{ci}^{c}(t)R_{ci}r_{ci}^{c}(t) - \overline{r}_{cr}^{c}(t) = -r_{ci}^{c} + \overline{R}_{cr}^{c}(t)\overline{r}_{cr}^{c} 
\]  

Equation (3.12)

2- Translation between the reference camera at \( t_1 \) and a non-reference camera at \( t_2 \), \( r_{cj}^{c}(t) \), can be related to the system translation \( \overline{r}_{cr}^{c}(t) \) through the form in Equation (3.13),

\[
\lambda_{cj}^{c}(t)R_{cj}r_{cj}^{c}(t) - \overline{r}_{cr}^{c}(t) = \overline{R}_{cr}^{c}(t)\overline{r}_{cj}^{c} 
\]  

Equation (3.13)

3- Translation between the reference camera at \( t_2 \) and a non-reference camera at \( t_1 \), \( r_{ci}^{c}(t) \), can be related to the system translation using Equation (3.14).

\[
\lambda_{ci}^{c}(t)R_{ci}r_{ci}^{c}(t) - \overline{r}_{cr}^{c}(t) = -r_{ci}^{c} 
\]  

Equation (3.14)

4- Translation estimated for the reference camera from epochs \( t_1 \) to \( t_2 \), \( r_{cr}^{c}(t) \), is a factor of system translation \( \overline{r}_{cr}^{c}(t) \) (Equation (3.15)).

\[
\lambda_{cr}^{c}(t)R_{cr}r_{cr}^{c}(t) - \overline{r}_{cr}^{c}(t) = 0 
\]  

Equation (3.15)

Similar to the rotation estimates, for \( n \) overlapping cameras, a maximum number of \( n^2 \) translation estimates are obtained using the four equations mentioned above. Each of which is three equations to solve for the three parameters of the system translation between the two epochs.
\( \bar{r}_{cr(t_2)} \) and a rescaling factor \((\lambda)\). Therefore, a linear system with a maximum of \(3n^2\) equations to solve for \(n^2 + 3\) unknowns is formed, which robustly solves for \( \bar{r}_{cr(t_1)} \).

At this stage, the outliers in the feature matches across the Epipolar lines for both the spatial and temporal matches are already detected (sections 3.1.1 and 3.1.2). Outliers along the Epipolar lines for the spatial-temporal matches can be detected after estimating the system motion, using a procedure explained in the next section.

3.2.4 Detection of outlier matches using the estimated system motion.

As explained in sections 3.1.1 and 3.1.2, right after the feature matching two types of outliers are filtered. 1) Outliers along and across the Epipolar lines in spatial triple matches and 2) outliers across and the Epipolar lines in the temporal and double spatial matches. In order to detect other outliers along the Epipolar lines for spatial-temporal matches, knowledge of the system motion is required. Using the system motion estimated in the previous stage, the EOPs of all the images can be calculated. The EOPs of the images can be used for detecting the spatial-temporal outlier matches, which is explained as follows.

The outlier detection algorithm is based on selecting samples of three points from a feature track, and checking their intersection error. Figure 3.8 shows two possible types of samples: 1) a spatial match connected to a temporal match and 2) two connected temporal matches. Due to the error accumulation in the epoch by epoch motion estimation, samples with time difference of more than two epochs are not considered.

All the samples of three points are intersected, and the intersection error for each point in the sample is computed. A point \( p \) in a feature track might appear in several samples, therefore, the intersection error of point \( p \) in all the samples is averaged and denoted as \( e_p \). Then, the point \( p \)
is considered as outlier if $e_p$ is higher than a given threshold. So, this method checks the compatibility of each point to all matched points in its neighborhood, which improves robustness of the outlier detection algorithm. This algorithm is computationally expensive as it checks every sample of three point within a feature track, and usually thousands of feature tracks exits in a practical application. To improve speed, this algorithm can be executed in parallel, where each thread detects outliers in a feature track.

![Diagram of two types of triple sample checked for intersection error](image)

Figure 3.8: Two types of triple sample checked for intersection error

At this stage, the system motion is estimated, and the majority of the matching outliers has been filtered out either at the feature matching stage or after estimating the system motion parameters. Now, the estimated system motion can be refined by a bundle adjustment in a sliding window fashion, which is explained in the next section.

### 3.3 Sliding window bundle adjustment

A Bundle Adjustment (BA) is a non-linear minimization of a cost function, which is the sum of squared back projection error of the object points (Triggs et al., 2000). The object points are projected to the images through the collinearity equations explained in section 2.4. BA is the
standard procedure to achieve the highest accuracy in a photogrammetric task, and it is employed in this research to optimize the recovery of the system motion parameters.

BA can be executed in a sliding window fashion (Mouragnon et al., 2009), which means a part of the scene that falls inside a window is refined, and the window slides through to refine the entire scene. The reason is that the contribution of a particular image to the EOPs of the following images reduces as the overlapping area decreases. Therefore, presence of the earlier images would not be useful as BA grows, and BA can slide and pass over them. Therefore, the scene structure and system motion is refined faster while less memory is consumed. The appropriate window size depends on the overlap between the successive images, and could be adapted as the overlapping percentage changes. One should note that the datum and 3D model scale should be maintained in a sliding window BA. In this research, the datum is maintained by fixing the EOPs of the first image in the window and the scale is automatically maintained by keeping the mounting parameters fixed in the BA.

Sliding window BA starts by a model generated by two images, and grows until reach the window size, and then begins to slide. Figure 3.9 is the flowchart of the sliding window BA employed in this research. The BA window slides by reading the approximate EOPs of the new images as well as the image points and the corresponding object points. The approximate EOPs are computed from the system motion estimated in section 3.2. Using the approximate EOPs and the existing 3D points in the adjusted model, gross errors (if any) of the new image points are filtered out by looking to their residuals. Since the EOPs are approximate, only the gross errors – which have high residual – are filtered at this stage. Then, BA is executed to minimize the residuals for all the image points, by updating the EOPs and the object point coordinates. After BA converges, the image point residuals are checked for outliers. For this purpose, image points are
sorted in descending order based on their residual, and the points with residuals above a given threshold $t$ are assumed as outliers. However, since outliers affect inliers, only a given percentage of the points with residual above $t$ are removed from the top of the sorted list, and BA is executed again. This procedure is repeated until no points with residual above the threshold $t$ remains. This threshold is very important as it controls the expected accuracy for the sliding BA. A strict threshold can remove many points and even cause failure in the sliding BA. In contrast, a relaxed threshold may cause deformation in the scene structure, as many outliers may pass the threshold $t$.

After removing the outliers, bad image/object points should be eliminated from the 3D model (Snavely et al., 2006). Bad points have small $x$ parallax (far depth), or reconstructed behind the cameras. Strictly speaking, bad points are not necessarily outliers, as they already passed the outlier detection threshold. Far points are just weak observations and do not improve the geometry, but the reconstructed points behind the cameras are wrong results.

Figure 3.9: Bundle adjustment with built-in outlier detection
An optional global bundle adjustment can be executed using all the images and object points. Global bundle adjustment is not always possible as it require a large amount of memory and takes a long time. In addition, it could be ill-conditioned especially for strip-wise scenes like what we have in this research. For a large bundle adjustment, efficient sparse matrix libraries must be employed to store the normal matrix and solve the normal equation (Davis, 2006).

3.4 Summary

In this chapter, the proposed methodology to estimate the 3D motion of a multi-camera system in an indoor environment was presented. The methodology is based on matching the features between images acquired at the same or different epochs, and rejecting the outlier matches by robust techniques. Then, the rotation of the system between successive epochs is computed by providing several estimates for the system rotation and averaging them after removing the incompatible ones. Then, the system translation is estimated by solving a system of linear equations. After all, the estimated system motion is refined by a bundle adjustment that takes advantage of an efficient mathematical model for relating the object space to the image space in a multi-camera system. In the next chapter, the result of motion estimation by the proposed methodology using real data is presented.
Chapter 4: Results

In this chapter, the developed methodology is tested over datasets acquired at the University of Calgary, and the results of motion estimation and refinement are presented.

4.1 Hardware and software implementation

For data acquisition, three cameras are mounted on a cart to capture images from one side of building corridors (This is to be extended for both sides, floor, and ceiling for future work). To improve the geometry for system calibration and the 3D reconstruction, the mounted cameras are slightly tilted inward to increase overlap of the images taken at the same epoch. Figure 4.1 shows the data acquisition system created for this research, which uses three Canon T3 Rebel cameras. Each Canon T3 Rebel is a 12.2 mega pixel DSLR camera with 18-35 mm lens, which takes 4272×2848 pixel images. The cameras are connected to a laptop, and controlled through CamControl software developed in Digital Photogrammetry Research Group (DPRG) that sends trigging commands simultaneously to the cameras and records the captured images.

![Data acquisition system](image)

Figure 4.1: Data acquisition system

The multi-camera system can be calibrated before or after data acquisition, or in both situations to verify the stability of the system during a practical work (Habib et al., 2014). Then,
the multi-camera system is pushed through the corridors and synchronized images are taken every 1.5 seconds. Currently the CamControl software is able to capture images at a fixed rate, and the interval of 1.5 seconds guarantees that the captured images are downloaded properly to the laptop and the cameras are ready to capture again. Using these images, the system motion is estimated by which the 3D structure of the imaged environment can be reconstructed.

The processing software for feature extraction and matching as well as motion estimation and refinement was entirely coded in C/C++ using visual studio 2010. The multi-camera system was calibrated using a software developed by DPRG (Detchev et al., 2014) that identifies the targets on the test field and prepares them to be used in Multi Sensor Aerial Triangulation software (Kersting et al., 2011), which estimates the IOPs and the mounting parameters of the cameras through bundle adjustment.

For corner detection, the Goodfeaturestotrack function in OpenCV (opencv.org) was used, followed by CornerSubPix function to iteratively refine the corners. The SIFT descriptor was evaluated by SIFTGPU library (Wu, 2007), and again OpenCV was used to match the descriptors. To detect outliers, the RANSAC/5-point algorithm in Bundler software was used (Snavely et al., 2006). By default, the Bundler software takes advantage of Sparse Bundle Adjustment (Lourakis and Argyros, 2009) to execute bundle adjustment. In order to process a large dataset and to improve the processing speed, Parallel Bundle Adjustment (Wu et al., 2011) can be adapted easily in Bundler. However, in order to implement the multi-camera bundle adjustment using the collinearity equation explained in section 2.4, the Google Ceres Solver (Agarwal et al., n.d.) was employed. The results of motion estimation and sparse reconstruction was visualized using Visual SfM software (Wu, 2013).
4.2 System calibration

The multi-camera system (Figure 4.1) is calibrated by taking 67 images per camera of a test field designed for multi-camera system calibration. Figure 4.2 shows three images taken of the test field at the same time. The dimensions of the test field are 110×80 cm and contains 12 coded targets through which the other 84 targets can be identified, even for the images in which the test field is partially visible. So, the test field targets can be identified in all the images taken by the multi-camera system. Using the identified targets, the IOPs of each camera as well as the mounting parameters are estimated in single step through bundle adjustment. The mathematical model used for bundle adjustment has been explained earlier in section 2.4. This mathematical model is an extension of the collinearity equations that relates the object space to the image space through the IOPs and the mounting parameters of the cameras. Interested readers can refer to Lari et al. (2014a, 2014b) for more information about theory and implementation of the multi-camera calibration using this test field.

![Figure 4.2: Images of the calibration test field captured at the same time.](image)
Careful estimation of the distortion coefficients through system calibration is essential to recover the system motion accurately. Radial and de-centric lens distortions are compensated by a set of correction polynomials as explained in section 2.4. The polynomial degree is usually selected by investigating the pattern of image residuals against different choices of the polynomial degree. The lowest polynomial degree by which lens distortions are compensated is the optimum degree, which is explained in the following.

Figure 4.3 shows four plots of image points residuals versus distance to the principal point (r), using different choices of polynomial degrees. \( K_1 \) and \( K_2 \) are the coefficients corresponding to the second and fourth order radial distortion polynomial, and \( P_1 \) and \( P_2 \) are the coefficients for de-centric distortion model. The plots a, b, and c in Figure 4.3 show that the magnitude of the residuals grows by moving towards the image edges, which indicates poor compensation of lens distortions by the set of coefficients used. But, it can be clearly seen in Figure 4.3-d that the residual pattern disappears when \( K_1, K_2, P_1 \) and \( P_2 \) coefficients are used in the distortion model. Therefore, \( K_1, K_2, P_1 \) and \( P_2 \) are estimated in the calibration bundle adjustment, and are used for undistorting any observed image points.
Figure 4.3: Image points residuals versus radius from principal point, for different sets of distortion coefficients

The bundle adjustment for multi-camera system calibration is executed with 201 images (3×67) including 14,708 image points, and converged to a-posteriori standard deviation of 0.32 pixels. Table 4.1 shows the calibration result including IOPs, and mounting parameters relative to the reference camera (Camera 1), followed by their standard deviations. The accuracy of principal point position \((x_p, y_p)\) and principal distance \(c\) is less than half a pixel as seen in the first three rows of the table. The next four rows show the estimation error of the distortion coefficients, which is less than 1\% of their value. The Euler rotation angles between the reference camera and the other cameras are estimated with an accuracy less than 25 arc seconds, and the translation of the cameras are estimated by an accuracy about 0.1 mm, as seen in the last six rows of the table.
Table 4.1: Multi-camera system calibration results

<table>
<thead>
<tr>
<th></th>
<th>Camera 1</th>
<th>Camera 2</th>
<th>Camera 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_p$ (pixel)</td>
<td>-22.3 ± 0.3</td>
<td>16.1 ± 0.3</td>
<td>23.6 ± 0.4</td>
</tr>
<tr>
<td>$y_p$ (pixel)</td>
<td>-31.25 ± 0.3</td>
<td>-46.3 ± 0.2</td>
<td>11.1 ± 0.3</td>
</tr>
<tr>
<td>$c$ (pixel)</td>
<td>3718.62 ± 0.2</td>
<td>3582 ± 0.2</td>
<td>3694.2 ± 0.2</td>
</tr>
<tr>
<td>$K_1$ (pixel)$^2$</td>
<td>-1.29e-8 ± 1.5e-11</td>
<td>-1.4e-8 ± 1.4e-11</td>
<td>-1.3e-8 ± 1.3e-11</td>
</tr>
<tr>
<td>$K_2$ (pixel)$^4$</td>
<td>8.17e-16 ± 2e-18</td>
<td>9.3e-16 ± 2.5e-18</td>
<td>8e-16 ± 2.4e-18</td>
</tr>
<tr>
<td>$P_1$ (pixel)$^1$</td>
<td>2e-7 ± 9.2e-9</td>
<td>2.4e-8 ± 9.6e-9</td>
<td>2.3e-7 ± 7.3e-9</td>
</tr>
<tr>
<td>$P_2$ (pixel)$^1$</td>
<td>-3.6e-7± 8.1e-9</td>
<td>-1.4e-6 ± 7.4e-9</td>
<td>-7.5e-7 ± 3.2e-9</td>
</tr>
<tr>
<td>$\omega$ (Deg)</td>
<td>0</td>
<td>14.5 ± 17”</td>
<td>13.7 ± 17”</td>
</tr>
<tr>
<td>$\varphi$ (Deg)</td>
<td>0</td>
<td>12.5 ± 23”</td>
<td>-14.1 ± 23”</td>
</tr>
<tr>
<td>$k$ (Deg)</td>
<td>0</td>
<td>-1.3 ± 6”</td>
<td>2.8 ± 6”</td>
</tr>
<tr>
<td>$X$ (meter)</td>
<td>0</td>
<td>0.427 ± 0.1e-3</td>
<td>-0.363 ± 0.1e-3</td>
</tr>
<tr>
<td>$Y$ (meter)</td>
<td>0</td>
<td>-0.355 ± 0.09e-3</td>
<td>-0.356 ± 0.1e-3</td>
</tr>
<tr>
<td>$Z$ (meter)</td>
<td>0</td>
<td>-0.334 ± 0.1e-3</td>
<td>-0.310 ± 0.1e-3</td>
</tr>
</tbody>
</table>

In this research, the multi-camera system was calibrated before data acquisition. One can also calibrate the system after the data acquisition, and evaluate the stability of the IOPs and the mounting parameters of the cameras during data acquisition (Habib et al., 2014).

### 4.3 Sources of error

There are various sources of error affecting the accuracy of the motion estimation such as feature detection accuracy, instability of the platform (vibration, shake), and the system calibration error. In the developed data acquisition hardware in this research, it was observed that the tripod heads used to mount the cameras on the cart are not really stable for such a mobile platforms. In addition, even though the synchronization of the cameras was already tested to be under 10 milliseconds, but there is no guarantee that the cameras remain accurately synchronized during data acquisition, where each camera takes hundreds of images continuously.
The pixel size and the principal distance of the employed cameras are \( s = 5 \) micron and \( c = 18 \) mm, respectively. Considering the average camera to object distance \( Z = 2 \) meters, the Ground Sampling Distance (GSD) would be approximately 0.5 mm. The average base-line between the images is approximately \( B = 0.5 \) meter. Therefore, the vertical accuracy \( \sigma_z \) of the intersected points can be computed by the formula \( \sigma_z = \frac{Z^2 \sigma_p}{Bc} \) in which \( \sigma_p \) is the accuracy of the x parallax between conjugate points. Using this formula, the vertical accuracy is expected to be 2.2 mm considering one pixel error for x parallax. By multiplying the system calibration error (0.32 pixel) into GSD, a rough estimate of the calibration error in object space can be found (0.32*0.5 = 0.16 mm). Similarly, effect of the random errors minimized by the global bundle adjustment (explained later in Table 4.7) is ~2 pixels in image space, equals to 1 mm in object space. Assuming errors are independent, the root sum squared of the mentioned errors can be computed as the overall expected accuracy, which equals 2.5 mm in object space.

### 4.4 Datasets

The proposed methodology was tested over two datasets acquired by the multi-camera system from the corridors of two buildings at University of Calgary. The Geomatics corridor dataset is a small dataset with strong texture and without any turn. The Science corridor dataset is a large challenging dataset containing poor textures and turns. The other captured datasets failed in the middle of process due to lack of features. Figure 4.4 shows two sample images where feature tracking has failed. The left image is a glassy metal object, without enough feature to detect and track. The right one is a blank door close to the camera. This problem can be mitigated by mounting more cameras to look at the both sides of the corridors as well as floor and ceiling, which is the plan for future works.
Figure 4.4: Sample images in which feature tracking was failed

4.4.1 Geomatics corridor dataset

A dataset was collected from one side of the corridor of the department of Geomatics, which is almost 30 meters long with a strong texture. The data acquisition cart (Figure 4.1) was pushed through the corridor, and 150 images were taken at 50 epochs. Figure 4.5 shows sample images of this dataset. As seen in this figure, the scene texture is strong and rich in corners especially around the bricks. However, brick corners are repetitive features and can be simply matched to an incorrect brick.

Figure 4.5: Sample images of the Geomatics corridor dataset
4.4.2 Science corridor dataset

The data acquisition cart was pushed through the Science building corridor in a rectangular loop. The loop circumference is approximately 150 meters, which was captured by 1656 images at 552 acquisition epochs. Figure 4.6 shows sample images from this dataset, including poor (a), repetitive (b), and strong texture (c) as well as a 90 degree turn (d). In poor textured areas few reliable features can be detected, which might cause failure in the motion estimation. In an area with repetitive texture, many similar features are extracted that weaken the feature matching. In addition to the difficulties caused by poor or repetitive texture, the scene length in this dataset is much larger than the width (strip-wise), which is a poor geometry to recover. In a strip-wise scene, the angle of intersection between the light rays in side overlapping images is small that contribute weakly to the global geometry. Therefore, presence of poor and repetitive texture in this dataset as well as a weak geometry, highly challenge the research methodology. In the next section, the results of feature extraction and matching over the two datasets are presented.
Figure 4.6: Sample images of the Science corridor dataset. (a) Poor texture, (b) Repetitive texture (c) Strong texture, (d) 90 degree turn

Figure 4.7 shows the flowchart of the proposed methodology to estimate the system motion in indoor environment. The results of 2D image processing are presented in sections 4.5 and 4.6. Then, the results of motion estimation and refinement are presented in section 4.7.

Figure 4.7: Flowchart of the proposed methodology for motion estimation
4.5 Feature extraction and matching

As mentioned in section 3.1 and seen in Figures 4.5 and 4.6, indoor environments are generally rich in corner feature, which can be accurately localized. Figure 4.8-a shows a sample image with corner features detected by the Harris operator. The corners are detected in the second level of the image pyramid and then transferred to the original level, followed by the sub-pixel refinement algorithm explained in section 3.1. Figure 4.8-b shows the detected features after purging the ones closer than 1% of image diagonal. Purging the spatially close features reduces the mistakes in feature matching due to the similarity of descriptors in neighboring features. Besides, close image points do not contribute significantly to the geometry. Therefore, close features just slow down the processing speed and might cause numerical ill-conditioning or singularity.

It can be observed in Figure 4.8 that few features have been detected on the floor. It is due to a limiting factor that discards the features for which Harris response is lower than a threshold - 0.0001 of the quality of the best corner (OpenCV.org). Table 4.2 lists the thresholds used for corner detection by the OpenCV library. These thresholds have been set by trial and error. Figure 4.9 shows a sample corner before and after sub-pixel refinement. It can be seen that the corner is accurately localized after applying the sub-pixel refinement.

Table 4.2: Corner detection thresholds

<table>
<thead>
<tr>
<th>Operation</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>The layer of image pyramid in which features are detected and transferred to the original level</td>
<td>2</td>
</tr>
<tr>
<td>Harris response window size</td>
<td>3 pixels</td>
</tr>
<tr>
<td>Minimum feature distance (purging distance)</td>
<td>1% of image diagonal</td>
</tr>
<tr>
<td>Minimum acceptable feature quality (Harris response)</td>
<td>0.0001 of maximum quality within the detected features</td>
</tr>
</tbody>
</table>
Figure 4.8: Detected corner features in an indoor scene. Some of the corners are magnified at right. (a) Initially detect features (b) After purging the close features

Figure 4.9: A detected corner before and after sub-pixel refinement

The detected features in an image are matched to the features in the neighboring images, by comparing their SIFT descriptors evaluated by SiftGPU library (Wu, 2007). So for each feature, a number of matches can be found in the other images (feature track). A feature track is the projections of an object point into the images. Figure 4.10 shows the feature tracks extracted in the images captured by three cameras \(c_1, c_2, c_3\) at four epochs \(t_1 - t_4\). The length of green lines
represents the number of images in which the feature is tracked, so the longer the better. The red lines represent the features tracked in only two images (weak feature tracks). Generating such a graph helps us to evaluate the quality of feature matching before executing the next step - motion estimation.

Figure 4.10: Feature tracks in the images taken by the multi-camera system; length of the green lines represents the number of images in which the feature has been tracked, red lines are features tracked in only two images (weak feature tracks)

Figure 4.11 shows the average feature track lengths seen in the cameras at each acquisition epoch. As seen in this figure, the average feature tracks length – i.e., quality of feature matching - changes throughout the scene. It is due to the fact that in strong textured area, more salient features can be detected and tracked. In addition, the lighting condition and moving speed affect the quality of feature matching. According to this figure, the quality of feature matching is similar for the cameras. Figure 4.12 shows the sample images indicated in Figure 4.11 by symbols (S) and (L) in which feature tracks are short (S) or long (L). In the top images of Figure 4.12 feature tracks are short. It is due to reflections on the glass and the white column in the top left image taken at the
Geomatics corridor, and the blank texture of the door in the top right image captured at the Science corridor. The two bottom images are the areas in which feature tracks are long. The variation of texture in the Geomatics corridor dataset is not as much as the Science corridor dataset. Therefore, the mean feature track length fluctuate more in the Science corridor.

Figure 4.11: Mean feature track lengths seen by the cameras at each acquisition epoch for the Geomatics corridor dataset (a) and the Science corridor dataset (b). The sample images in which feature tracks are short (S) and long (L) are displayed in Figure 4.12.
4.6 Outlier detection

Table 4.3 presents the numerical result of outlier detection after feature matching. The first and second rows show the time and percentage of outlier matches detected by stereo view geometry, using RANSAC in conjunction with the 5-point algorithm implemented in Bundler software (Snavely et al., 2006). One can see in the first row that the outlier detection time for a large dataset like the Science corridor is considerable. It is due to the fact that the 5-point algorithm is relatively slow (see section 3.1.1). The number of outliers to be detected at this stage is controlled by the distance to Epipolar line threshold (15 pixels in this research). This threshold should be tuned with the outlier detection threshold in the sliding bundle adjustment executed later. A relaxed
threshold in this stage might overload the outlier detection in the bundle adjustment. In contrast, a strict threshold might break the feature tracks, and even cause failure in the motion estimation. The third row shows the percentage of outliers in spatial triple matches (Figure 3.3) detected by the intersection error using the mounting parameters. An image point in a spatial triple match is labeled as outlier, if the distance of this point to the back projection of the intersected point is more than a given threshold (e.g. 5 pixels here). Since the mounting parameters are accurately known, this threshold could be strict. Table 4.4 shows the thresholds used for outlier detection, set by trial and error.

Table 4.3: Result of outlier detection of matched features for the two datasets

<table>
<thead>
<tr>
<th>Datasets/Process</th>
<th>Geomatics corridor</th>
<th>Science corridor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time for outlier detection across Epipolar lines (min)</td>
<td>3.5</td>
<td>45</td>
</tr>
<tr>
<td>Outlier matches across the Epipolar lines</td>
<td>36%</td>
<td>33%</td>
</tr>
<tr>
<td>Outlier matches detected by spatial multi-view intersection</td>
<td>6.7%</td>
<td>10%</td>
</tr>
</tbody>
</table>

Table 4.4: Outlier detection thresholds

<table>
<thead>
<tr>
<th>Operation</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance to Epipolar line for spatial double and temporal matches</td>
<td>15 pixel</td>
</tr>
<tr>
<td>Back projection error for spatial triple matches</td>
<td>5 pixel</td>
</tr>
</tbody>
</table>

After detecting the outlier matches, the inlier matches are manipulated by the next stage to estimate system motion while recovering the 3D structure of the scene.

4.7 Recovery of the multi-camera system motion

The developed methodology explained in Chapter 3 was tested over the two datasets introduced in section 4.4, to evaluate the feasibility of motion estimation of the multi-camera system in an indoor environment. The tests were carried out in two stages, which are explained in this section:
1- Recovering the multi-camera system rotation by providing several rotation estimates and averaging them robustly. The multi-camera system translation is then estimated by solving a system of linear equations.

2- Refining the estimated system motion, using a sliding window BA that uses the mathematical model explained in section 2.4. A global BA can then be performed to optimize the entire scene at one step.

4.7.1 System rotation estimation and robust averaging

To estimate the system rotation, the Essential matrices that already estimated by RANSAC and the 5-point algorithm (section 4.6) are decomposed to ROPs. Having the ROPs between the cameras at two successive epochs, the rotation and translation of the multi-camera system between the two epochs are estimated, using the proposed method in section 3.2. Table 4.5 shows the result of rotation estimation by the proposed method. In this method, a maximum of nine (3^2) estimates for the system rotation are obtained between successive epochs. The incompatible rotation estimates are detected by the method introduced in section 3.2.2. This method iteratively removes incompatible rotation estimates, until the standard deviation of the compatibility errors falls under a threshold (10 pixel). As seen in the second row of this table, the incompatible rotation estimates are significant for both datasets. After throwing out the incompatible rotations, the compatibility error of each rotation estimate to the other (eight or less) estimates are computed again. The third row of the table shows the average compatibility error of the rotation estimates. Ideally, the average compatibility error should be zero, which means all the estimates for the system rotation between successive epochs would be perfectly compatible. But due to noises in computing the ROPs of the cameras between successive epochs, and the constant error of mounting parameters, the rotation estimates are not perfectly compatible. Figure 4.13 shows the Cumulative Distribution Function
for the average compatibility error of the system rotation estimates between successive epochs. As seen in this figure, 80% and 85% of the inlier rotation estimates at each epoch (nine or less) have compatibility error less than 25 pixels in the Geomatics and the Science corridor datasets, respectively.

Table 4.5: Result of the system rotation estimation.

<table>
<thead>
<tr>
<th>Datasets/Process</th>
<th>Geomatics corridor</th>
<th>Science corridor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of rotation estimates between successive epochs</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Incompatible rotation estimates</td>
<td>20.3%</td>
<td>11.1%</td>
</tr>
<tr>
<td>Average compatibility error of inlier rotation estimates (pixel)</td>
<td>24.1</td>
<td>19.4</td>
</tr>
</tbody>
</table>

Figure 4.13: CDF of the mean compatibility error of the rotation estimates at each epoch for the Geomatics corridor dataset (a) and the Science corridor dataset (b)

4.7.2 System translation estimation

After averaging the system rotation, the system translation is estimated by solving a system of linear equations (section 3.2.3). Table 4.6 shows average of standard deviation of the translation components estimated by the least-square optimization. The standard deviation of x component is higher than the others as the relative movement of the system happens in x direction. One source of error is propagation of the system rotation estimated earlier into the translation estimation.
(Equations (3.12) and (3.13)). Moreover, it is known that the system translation is highly correlated with the system rotation in small movements.

Table 4.6: Average standard deviation of the translation estimation

<table>
<thead>
<tr>
<th>Average standard deviation of the translation estimation</th>
<th>Geomatics corridor</th>
<th>Science corridor</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma_x$ (cm)</td>
<td>12.3</td>
<td>7.8</td>
</tr>
<tr>
<td>$\sigma_y$ (cm)</td>
<td>4.2</td>
<td>3.5</td>
</tr>
<tr>
<td>$\sigma_z$ (cm)</td>
<td>4.1</td>
<td>3.4</td>
</tr>
</tbody>
</table>

4.7.3 Refinement the estimated motion by bundle adjustment

Knowing the system motion between two successive epochs, the EOPs of all the images can be computed. Then, feature tracks are intersected using the computed EOPs to find the 3D coordinates of their corresponding object point. The EOPs of all the images and the 3D coordinates of the points are refined through a sliding window BA. The approximate EOPs can be used for gross error detection before BA (section 3.3). Since the EOPs are approximate, gross errors are filtered by checking the back projection error with a given threshold (e.g. above 50 pixel). The rest of outliers (e.g. residuals between 5 to 50 pixels) are detected iteratively by BA.

Table 4.7 shows the result of sliding window BA followed by a global BA. The first row shows the processing time for the sliding window BA, which depends on the window size. A larger window size is safer to deal with different situations in the scene, while smaller one is faster and might be suited for real time applications. A window size of 10 epochs - which equals to 30 images - was used for the sliding window BA in this research. The datum of the sliding window BA is maintained by fixing the EOPs of the first reference camera at the beginning of the window; the model scale is maintained by keeping the mounting parameters fixed in the BA. In case a long enough window size is used for the sliding window BA, a global BA could be skipped as there is no room for improvement.
The second row of Table 4.7 shows the percentage of outlier image points, detected by sliding and global BA. The outlier image points within the BA window are detected by checking their residual with a given threshold, and are thrown out in multiple steps as explained in section 3.3. Therefore, the percentage of outliers depends on the given threshold. In the sliding window BA for these datasets, points with residuals more than 5 pixels are rejected in 5 steps/iterations. The high percentage of the detected outliers indicates that the given threshold is strict for majority of the features. This might not be a problem as long as enough number of inlier points remain in the scene. The last row of Table 4.7 shows the a-posteriori standard deviation after the global BA and Table 4.8 summarizes the threshold used for sliding window BA, set by trial and error.

Table 4.7: Result of sliding window and global BA to refine the system motion

<table>
<thead>
<tr>
<th>Datasets/Process</th>
<th>Geomatics corridor</th>
<th>Science corridor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sliding window BA execution time (min)</td>
<td>7.1</td>
<td>150</td>
</tr>
<tr>
<td>Outlier image points removed by sliding window BA</td>
<td>63%</td>
<td>57%</td>
</tr>
<tr>
<td>Total number of image points</td>
<td>38,804</td>
<td>823,984</td>
</tr>
<tr>
<td>Total number of object points</td>
<td>12,259</td>
<td>148,952</td>
</tr>
<tr>
<td>Global BA a-posteriori STD (pixel)</td>
<td>2.3</td>
<td>2.2</td>
</tr>
</tbody>
</table>

Table 4.8: Thresholds for sliding window bundle adjustment

<table>
<thead>
<tr>
<th>Operation</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>BA window size</td>
<td>10 epochs = 30 images</td>
</tr>
<tr>
<td>Gross error detection before BA</td>
<td>50 pixel</td>
</tr>
<tr>
<td>Outlier residual threshold</td>
<td>5 pixel</td>
</tr>
</tbody>
</table>

Figure 4.14 depicts the multi-camera system motion and the recovered motion/scene of the Geomatics corridor estimated by the proposed methodology (a) and after refinement with sliding and global bundle adjustment (b). As it can be seen in this figure, the proposed method has successfully recovered the trajectory, which can be further optimized by bundle adjustment.
Figure 4.15 shows the Science corridor recovered by the proposed method (a), and refined by sliding and global bundle adjustment (b). As seen in this figure, the accumulated error in the proposed method (Figure 4.15-a) has been minimized by bundle adjustment, and the rectangular loop is reconstructed (Figure 4.15-b). The side view of the Science corridor trajectory is depicted in Figure 4.15-c.

Figure 4.14: Recovered system motion by a) proposed method b) after refinement by sliding and global BA. (Visualized by Visual SfM software)
Figure 4.15: Recovered system motion in the Science corridor by a) proposed method, b) Refined after sliding and global BA, c) Side view of the trajectory after sliding and global BA (Visualized by Visual SfM software)

In the next section, the estimated motion by the proposed method is evaluated. The refined motion by bundle adjustment is considered as true values, and compared epoch by epoch to the motion estimated by the proposed method.

4.7.4 Evaluation of the proposed methodology

To evaluate the methodology, the system motion estimated between successive epochs using the proposed methodology \( \overline{R}_{cr(t_1)}^{cr(t_2)}, \overline{r}_{cr(t_1)}^{cr(t_2)} \), is compared to the refined motion after bundle adjustment \( \hat{R}_{cr(t_1)}^{cr(t_2)}, \hat{r}_{cr(t_1)}^{cr(t_2)} \). In other words, the refined EOPs of the reference camera after bundle
adjustment are hypothesised as true values, and used to compute the (true) relative motion of the system between successive epochs. The true relative motion of the system between successive epochs $\mathbf{R}_{c_r(t_1)}c_r(t_1)$, $\mathbf{R}_{c_r(t_2)}c_r(t_2)$ is then used to evaluate the motion estimated by the proposed method.

First, the system rotation between successive epochs computed by the proposed method $\mathbf{R}_{c_r(t_1)}c_r(t_1)$ are compared to the true rotation of the system $\mathbf{R}_{c_r(t_1)}c_r(t_1)$. For this purpose, the angular distance (in degree) between true rotation $\mathbf{R}_{c_r(t_1)}c_r(t_1)$ and the estimated rotation $\mathbf{R}_{c_r(t_1)}c_r(t_1)$ is considered as the (angular) error. Figure 4.16 shows the angular error of system rotation $\mathbf{R}_{c_r(t_1)}c_r(t_2)$ for the Geomatics (a) and the Science (b) corridor datasets at each epoch $t_1$. It can be seen for both datasets that the estimated system rotation deviates less than one degree from the true rotation (bundle adjusted one).

![Figure 4.16: Angular error of system rotation estimation by the proposed method for the Geomatics corridor dataset (a), and the Science corridor dataset (b).](image)

The translation of the system between successive epochs $\mathbf{r}_{c_r(t_1)}c_r(t_1)$ can also be evaluated by comparing to the refined system translation after bundle adjustment $\mathbf{r}_{c_r(t_1)}c_r(t_1)$. Therefore, the error of system translation $\mathbf{r}_{c_r(t_1)}c_r(t_2)$ estimated by the proposed method is defined as the norm
\[
\left\| \hat{r}_{\mathbf{p}}(t_2) - \hat{r}_{\mathbf{p}}(t_1) \right\| .
\]

Figure 4.17 shows the Cumulative Distribution Function (CDF) of the translation error for the Geomatics (a) and Science (b) corridor datasets. It can be seen that for 80% of the epochs in the Geomatics corridor dataset and almost 90% of the epochs in the Science corridor dataset, the translation error is below 10 cm.

![Figure 4.17](image)

**Figure 4.17**: Error in system translation estimation by the proposed method for the Geomatics corridor dataset (a), and the Science corridor dataset (b).

### 4.7.5 Accuracy check

In order to evaluate the accuracy of the recovered system motion, some check distances were measured with a surveying grade nylon, and compared to the distances measured over the 3D model. Figure 4.18 shows some of the check distances (D) in the Geomatics corridor dataset. The error (\(\Delta\)) in the measured bulletin board and poster dimensions indicate relative (local) accuracy of the reconstructed scene, which turned out to be several millimeters. The long distance measured between two points at the both ends of the scene shows a 4.6 cm error, which is almost 0.16% of the measured distance.

Figure 4.19 shows some of the check distances for the Science corridor dataset. As seen in this figure, similar to the Geomatics corridor, the check distances over the bulletin boards show
several millimeters error. The longest possible distances in the Science corridor were also measured, which shows 4.5 and 9.0 centimeters error - around 0.15% of the measured length.

Figure 4.18: check distances from the scene and their error

Figure 4.19: Check distances for Science corridor dataset with their discrepancies
In addition to the error sources present in the motion estimation (section 4.3), there is usually 2-3 pixel ambiguity in identifying the point of interest, which equals to 1-1.5 mm error. This error is present at the both ends of the measured distances in this section. Therefore, the measured distances over the 3D model are at least 2-3 mm biased due to the identification error, in addition to the 2.5 mm motion estimation error (section 4.3). Therefore, an error of at least 3.5-4 mm is expected on the measured distances. The average error of the short measured distances is 5.5 mm, which indicates that we have been slightly optimistic in our error analysis. For long measured distances, the error is accumulated and reaches to several centimetres, which is still below 0.16% of the measured distances. This accuracy has been achieved by a low-cost multi-camera system.

4.8 Summary

In this chapter, the multi-camera motion was estimated in the Geomatics and Science corridors, while sparse 3D structure of the scene is reconstructed. Corner features were detected by the Harris operator, and matched to the neighboring images by the SIFT descriptor. Outliers in the spatial matches were detected by employing the mounting parameters. Outliers of temporal matches were filtered out by RANSAC and the 5-poing algorithm, while estimating the Essential matrix between the cameras at successive epochs. The Essential matrices were decomposed to the ROPs, and the system rotation and real scale translation were estimated by the proposed method. To refine the estimated system motion, bundle adjustment were executed in sliding window fashion. Then, all the system motion parameters and the entire 3D structure of the scene were optimized by a global bundle adjustment. Afterwards, the accuracy of the recovered system motion was evaluated with check distances.
Chapter 5: Conclusions and recommendations for future work

The objective of this research was accurate motion estimation of a multi-camera system in indoor environment, to be used later for 3D reconstruction. The multi-camera system consists of three off-the-shelf cameras mounted on a cart, which takes synchronized images at a given interval. Multiple cameras allow us to cover larger area of the scene and estimate the system motion and 3D model in real scale, by taking advantage of the constant ROPs of the cameras (mounting parameters). Mounting parameters is a valuable knowledge for outlier detection and motion estimation, and we took advantage of them at all stages of the motion estimation.

The procedure of motion estimation starts by feature extraction and matching. Corner features at the second level of image pyramid were detected by Harris operator and transferred to the original level, followed by sub pixel refinement. For the detected corners, SIFT descriptor was evaluated to find their match in the neighboring images. Using the matched features, a methodology was proposed to estimate the system motion.

In order to estimate 3D motion of a multi-camera system, a methodology was proposed that relates the ROPs between the cameras across the epochs to the reference camera motion, taking advantage of the mounting parameters. The system motion was estimated in two stages:

1) By relating the relative rotation between the reference/non-reference cameras to the rotation of the reference camera at successive epochs, one estimate for the system rotation between successive epochs is obtained. Outliers within the rotation estimates are detected first, and the inlier rotation estimates are then averaged to compute a robust system rotation between successive epochs.

2) Using the robust system rotation estimated previously, the real scale translation of the system is estimated by solving a system of linear equations.
After estimating the system motion, bundle adjustment was executed to refine the estimated trajectory. The mathematical model to relate the image and the object space while considering the mounting parameters was explained in this thesis. The mathematical model is an extension of the collinearity equation, which is used for the system calibration as well as the sliding window and global bundle adjustment. The refined motion of the system after bundle adjustment was considered as true values, by which the motion estimated by the proposed methodology was evaluated.

The accuracy of the recovered system motion was evaluated by check distances over the scene. The relative (local) accuracy – measurements in a small area of the scene – turned to be accurate within the range of milliliters. The error in long check distances is less than 0.16% of the measured distances. This accuracy has been achieved by a low-cost data acquisition system costs approximately 1500$.

Contributions of this research are summarized as follows.

- A method to estimate the system rotation between successive epochs for an overlapping multi-camera system was introduced.
- A method to find the incompatible rotations within a set of rotation estimates.
- A method was proposed to estimate the translation of the multi-camera system, using the estimated rotation.
- A metric to describe the discrepancy of two rotation matrices was proposed.
5.1 Recommendation for future works

In each stage of motion recovery, there is a room to make improvements in order to increase the accuracy. For feature extraction, there are various state-of-the-art feature detectors and descriptors - as named in section 2.1- that can be employed to improve the overall performance. For example, Harris operator is relatively slow, as Harris response is computed for all the pixels in the image. FAST corner detector can improve the efficiency of corner detection, by extracting potentially corner features to be checked by their Harris response. Moreover, auxiliary sensors such as odometer or IMU can be used to improve feature matching. By employing the auxiliary sensors, system motion is approximately available at the feature matching stage. Therefore, the search area to find a match for a feature can be narrowed down, which improves the feature matching speed and quality.

It is known that when the baseline of motion approaches to zero, the Epipolar geometry is no longer valid. But Homography between the two images holds even there is no translation (Ma et al., 2004). Therefore, for short baseline - which frequently happens in acquired data - Homography can be used instead of the Epipolar geometry to find the ROPs between the images. To choose whether Homography or Epipolar geometry fit the matched points, the Geometric Information Criterion (GIC) for model selection can be employed (Kanatani, 1998). An alternative approach is to purging the close images (short-baseline) and working only on the images with appropriate base-line (key-frames).

Scene information can be used to constrain motion estimation. For example, loop closure can be detected and considered in the bundle adjustment to minimize the accumulation of error. Flatness of corridors can be used as a constraint for feature extraction and motion estimation. Since the movement in upward direction is almost zero, the matched feature in this direction can be
discarded. Moreover, the system motion can be constrained to a plane parallel to the corridors floor.

The basic pose estimation tasks such as relative orientation or resection have been heavily studied and are quite mature for single camera. For multi-camera systems, the Generalized Camera Model (GCM) can be employed to incorporate all the cameras for motion estimation. The concepts of GCM (section 2.2.5) and the generalized Epipolar constraint (Equation (2.10)) were already discussed. Kim et. al (2010) propose a solution to estimate the system motion using generalized Epipolar constraint, while considering the constraints on the Essential matrix. This solution could be evaluated as a future work.

GCM can be adapted for state-of-the-art single photo resection methods, to use all the cameras in single step for pose estimation of the multi-camera system. For example, the closed form single photo resection algorithms such as RPnP (Li et al., 2012) or EPnP (Fiore, 2001), can be extended for resection of the multi-camera systems – i.e., resection of the reference camera using all the cameras. Then, the 3D to 2D method (section 2.2.3) can be followed to estimate the system motion between successive epochs by incorporating all the cameras at once.

In this research, only one side of the corridor is reconstructed. By adding more cameras to cover the both sides as well as the floor and ceiling, the entire scene can be reconstructed. In addition, mounting more cameras facilitate recovery of the trajectory in poor textured area of the scene. For example, if the side wall is blank, the feature tracking on the floor or ceiling would help to estimate the system motion.

The scene depth could significantly vary in indoor environments. Therefore, keeping the cameras at fixed focal length (by disabling the automatic focus) might result in slightly blur images. By enabling the auto-focus feature, sharper images can be taken by the cameras. However,
auto-focus can change the internal orientation parameters of the cameras. This trade-off can be investigated as future work.

In this research many thresholds have been used, which needs to be manually tuned. Poor choice of a threshold (e.g. outlier detection threshold) may cause failure in the middle of motion estimation. The separate thresholds can be related to each other, and all tuned by only one threshold. For example, the threshold for outlier detection in image space, can be related to the one in object space. On top of that, thresholds can be set automatically by analyzing the mathematical models, looking into the characteristics of the acquired data, and specification of the employed hardware.
References


