Relative Pose Estimation Using Non-overlapping Multicamera Clusters

by

Michael J. Tribou

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2014

Abstract

This thesis considers the Simultaneous Localization and Mapping (SLAM) problem using a set of perspective cameras arranged such that there is no overlap in their fields-of-view. With the known and fixed extrinsic calibration of each camera within the cluster, a novel real-time pose estimation system is presented that is able to accurately track the motion of a camera cluster relative to an unknown target object or environment and concurrently generate a model of the structure, using only image-space measurements. A new parameterization for point feature position using a spherical coordinate update is presented which isolates system parameters dependent on global scale, allowing the shape parameters of the system to converge despite the scale parameters remaining uncertain. Furthermore, a flexible initialization scheme is proposed which allows the optimization to converge accurately using only the measurements from the cameras at the first time step. An analysis is presented identifying the configurations of the cluster motions and target structure geometry for which the optimization solution becomes degenerate and the global scale is ambiguous. Results are presented that not only confirm the previously known critical motions for a two-camera cluster, but also provide a complete description of the degeneracies related to the point feature constellations. The proposed algorithms are implemented and verified in experiments with a camera cluster constructed using multiple perspective cameras mounted on a quadrotor vehicle and augmented with tracking markers to collect high-precision ground-truth motion measurements from an optical indoor positioning system. The accuracy and performance of the proposed pose estimation system are confirmed for various motion profiles in both indoor and challenging outdoor environments.

Acknowledgements

I would like to thank my supervisors, Dr. David Wang and Dr. Steven Waslander for their help and encouragement during my studies. I thank Professor Wang for offering his wisdom and guidance to make me a confident engineer and researcher, and for making me feel like part of the family. I thank Professor Waslander for his seemingly limitless enthusiasm and many stimulating brain-storming sessions, while showing great patience as I worked out the details. My supervisors have made my studies both productive and enjoyable, and they are a major reason for my success.

I would also like to thank the members of my committee for their helpful advice during this work. Their thoughtful comments and suggestions have improved the quality of this research.

I would like to thank all of my colleagues in both the Waterloo Aerial Vehicles Lab (WAVELab) and the UW Haptics Lab, for their friendship and support during my time at Waterloo. I especially appreciate their kindness, enthusiasm, and determination. In particular, many thanks to John Daly for countless interesting conversations, ranging from fruitful discussions regarding research ideas, to philosophies on life in general.

I would also like to thank Mr. Adam Harmat, Dr. Inna Sharf, and the members of the Aerospace Mechatronics Lab (AML) at McGill University. Adam's help was instrumental in being able to complete the software implementation of the theoretical ideas in this thesis so they could be verified in the experiments. I am especially grateful for this collaboration.

Dedication

For Julie,

and my family,

Dad, Mum, Katy, Joe, Ben, and Isaac.

Contents

List of Figures **xi**

1 Introduction **1**

1.1 Vision Research Areas 2

1.1.1 Computer Vision 2

1.1.2 Vision-based Robotics 3

1.2 Motivating Example 4

1.3 Proposed Solution 5

1.4 Contributions 7

1.5 Thesis Organization 7

2 Background **9**

2.1 Single Camera Estimation 9

2.1.1 Projective Geometry 10

2.1.2 Pin-hole Camera Model 11

2.1.3 Point Feature Target Object Model 13

2.1.4 Multiple View Geometry 14

2.1.5 Simultaneous Localization and Mapping 18

2.1.6 Ambiguities and Sensitivities 26

2.2 Multicamera Cluster Pose Estimation 27

2.2.1 Algorithm Classification Criteria 28

2.2.2 General Camera Model 32

2.2.3 Literature Review 34

3 Multicamera Cluster SLAM **43**

3.1 System Models 43

3.1.1 Calibrated Multicamera Cluster 43

3.1.2 Target Object Model 44

3.1.3 Camera Cluster Measurement Model 45

3.1.4 System Structure Graph 49

3.2 Optimization 51

3.2.1 State Representation Using $\boxplus$-Manifolds 51

3.2.2 $\boxplus$-Manifolds for 3D Poses and Points 53

3.2.3 Optimization on $\boxplus$-Manifolds 59

3.2.4 Jacobian Calculation 62

3.3 Algorithm 64

3.3.1 Bundle Adjustment 66

3.3.2 Pose Tracking 66

3.3.3 Initialization 68

4 Degeneracies of Multicamera Cluster SLAM **73**

4.1 Problem Formulation 74

4.1.1 System Parameterization 75

4.1.2 System Degeneracies 82

4.2 Observability of Scale 84

4.3 Observability of Motion and Structure 89

4.3.1 Previous Work 89

4.3.2 Identification of Degenerate Configurations 90

4.4 Summary 105

5 Experiments **107**

5.1 Software Implementation 107

5.1.1 Non-overlapping MCPTAM 107

5.2 Experimental Setup 108
5.2.1 Cluster FOV Configurations 109
5.3 Results 110
5.3.1 Indoor Tests 111
5.3.2 Outdoor Tests 126
5.4 Summary 131

6 Conclusions 133

6.1 Future Work 136
6.1.1 Multicamera Cluster Framework 137
6.1.2 Degenerate Configurations 138
6.1.3 Software Implementation 139

APPENDICES 141

A ⊞-Manifolds 142

B Previous Software Implementations 145

B.1 Parallel Tracking And Mapping (PTAM) 145
B.2 Multi-Camera PTAM (MCPTAM) 147

C System Calibration 148

C.1 Extrinsic Calibration of Camera Cluster 148
C.2 Calibration of IPS and MCPTAM Frames 149

References 153

List of Figures

1.1 Absolute and Relative Transformation Triangle 2

1.2 Example Camera Configurations for Robot 5

1.3 Camera Cluster Configuration 6

2.1 Pin-hole Camera Model 12

2.2 Feature Point-Based Target Model 13

2.3 Epipolar Geometry of Two Cameras 14

2.4 Inverse Depth Feature Point Parameterization 23

2.5 Keyframe-Based Bundle Adjustment Method 25

2.6 Types of General Cameras 33

2.7 Raxels of Camera Cluster 34

3.1 Camera Cluster Calibration 44

3.2 Target Model Parameterization 46

3.3 Camera Cluster Measurement Model Transformations 47

3.4 Structure Graph of Example System 50

3.5 Spherical Vector Update 58

3.6 Jacobian Block Structure for Absolute Keyframe Pose 65

3.7 Initial Target Model 70

3.8 Algorithm Timing Diagram 71

4.1 Degeneracy Problem Setup 75

4.2 Camera Cluster Calibrated Transformations 78

4.3 Measurements in First Camera 80
4.4 Measurements in Second Camera 81
4.5 Concentric Circle Degeneracy 88
4.6 Planar System Degeneracy 88
4.7 Structure of $\mathbf{J}$ 93
4.8 Structure of $\mathbf{J}_1$ 96
4.9 Structure of $\mathbf{J}_2$ 97
4.10 Structure of $\mathbf{K}$ 99
4.11 Structure of $\mathbf{K}_4$ 102
4.12 Quintic surface of degenerate cluster translations. 103

5.1 Quadrotor Aerial Vehicle 109
5.2 Camera Cluster Calibration 109
5.3 MCPTAM Software Running 110
5.4 Experimental Camera Cluster Configurations 111
5.5 Orientation Magnitudes in Large Rotation Motion 114
5.6 Trajectories with Large Rotation Motion 114
5.7 Pose Errors with Large Rotation Motion 115
5.8 Normalized Trajectories with Large Rotation Motion 116
5.9 Normalized Pose Errors with Large Rotation Motion 117
5.10 Scale Convergence with Large Rotation Motion 118
5.11 Initial Position Error Magnitude with Large Rotation Motion . . . 118
5.12 Orientation Magnitudes in Small Rotation Motion 119
5.13 Trajectories with Small Rotation Motion 120
5.14 Pose Errors with Small Rotation Motion 121
5.15 Normalized Trajectories with Small Rotation Motion 122
5.16 Normalized Pose Errors with Small Rotation Motion 123
5.17 Two Camera Cluster Configuration 123
5.18 Cluster Pendulum Motion Experiment Setup 124

5.19 Orientation Magnitudes in Pendulum Swing Motion 124

5.20 Normalized Trajectories with Pendulum Motion 125

5.21 Normalized Pose Errors with Pendulum Motion 125

5.22 Outdoor Roof Environment 127

5.23 Orientation Magnitudes for Outdoor Roof Flight 128

5.24 Trajectories for Outdoor Roof Flight 129

5.25 Pose Errors for Outdoor Roof Flight 129

5.26 Normalized Trajectories for Outdoor Roof Flight 130

5.27 Normalized Pose Errors for Outdoor Roof Flight 131

C.1 Trajectory Alignment Frames 150

Chapter 1

Introduction

Precise robotic motion and manipulation tasks with respect to unknown environments and objects require an accurate, real-time measurement of the relative position and orientation of the robot and target. When measurements of the absolute position and orientation of both the robot and target object are available in a common frame of reference, the relative pose between them can be found. However, any inaccuracies in these absolute measurements will combine to amplify the resulting uncertainty in the relative pose estimate. This is particularly true when the measurement uncertainty is of a similar magnitude to the actual separation between the robot and the target object. As a result, the calculated relative pose may be inaccurate, as shown in Figure 1.1, and a controller using these measurements as feedback may fail. This is problematic for precise tasks such as docking or grasping, and is aggravated when both the robot and target are in motion.

Since it is this relative position and orientation state estimate (pose) that is necessary for successful execution of tasks involving robot interaction with the target, it is advantageous to measure this relative pose directly. To this end, one or more sensors mounted on the robot can be used to estimate the relative pose between the robot and target. A popular sensor set used for this purpose is one or more cameras since they are inexpensive, light-weight, and passive devices capable of collecting a large amount of information in each image, which can be captured at high frame rates.

Many researchers across different research areas have investigated the use of cameras for the purpose of estimating motion and scene structure. As a result, a huge number of techniques using a variety of camera types and configurations have been detailed in the literature. All of the techniques grapple with the fundamental trade-off between computation speed and estimation accuracy.

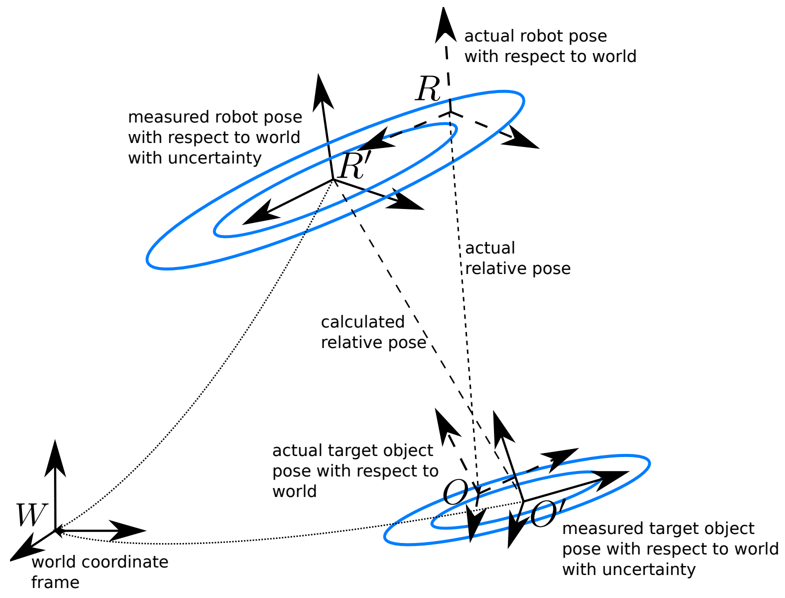


Figure 1.1: The measured poses of the robot R' and target object O' with respect to the world frame W with 1σ and 2σ error ellipses. The actual poses of the robot and target object are shown as R and O, respectively.

1.1 Vision Research Areas

The two main research areas investigating the use of visual feedback for environment mapping and camera localization are the vision-based robotics community principally in engineering, and the computer vision community within computer science. Each field approaches the problem with slightly different priorities, and as a result, the techniques employed in each community have traditionally been better-suited to these specific goals. Recently, the division has diminished as researchers have begun to leverage techniques from both areas to solve the common problem more effectively.

1.1.1 Computer Vision

The goal of geometric computer vision, or the geometry of multiple views [26], is to reconstruct the 3D geometry of a scene using a collection of 2D image measurements extracted from a set of camera images taken from different locations within the environment. This problem is known as Structure From Motion (SFM).

Using a model of a camera device derived from the laws of optics, the structure of the observed environment is inferred, as well as the position and orientation of the observing cameras. Typically, this involves measuring the image plane projections of sets of corresponding point features over multiple camera frames. When the 3D positions of these point features are rigidly fixed with respect to each other in a static environment, their location, as well as those of the camera, can be calculated. These estimations are typically performed by offline nonlinear optimizations or in a frame-to-frame manner where the priority is placed on accurate, dense reconstructions of the observed environment. Two excellent introductions to the problem from the computer vision perspective are found in [31] and [26].

1.1.2 Vision-based Robotics

In robotic control, the goal of any localization method is to provide high-quality real-time measurements of the robot location with respect to a reference frame using a set of on-board sensors and a physical model of the robot motion [82]. The motion of a camera can be tracked through an environment and therefore the motion of a robot with respect to a target object or environment can then be tracked by fixing a camera to the robot at a known position and orientation [89]. Subsequently, an algorithm can proceed by continually estimating the camera and robot pose with respect to the target using the latest set of camera image measurements. The task can be framed as an estimation problem tracking the relative pose of the robot and target through time using image measurements from one or more cameras.

To track the robot pose, a model of the environment is required. The accuracy of the relative pose estimate depends critically on the accuracy of the target model estimate and vice-versa. When a model of the environment is not available a priori, the system must create one as it progresses. As a result, both the structure of the target model and the relative pose of the robot must be estimated concurrently. In mobile robotics, this problem is called Simultaneous Localization and Mapping (SLAM). An introduction to the SLAM problem for general mobile robotic applications can be found in [82]. In the closely related field of industrial robotics, the analogous problem is known as Position-Based Visual Servoing (PBVS) [89].

One major difference between the structure and motion recovery problems as defined by the two communities is that time plays a much more critical role in robotics since the estimates may be used within the robot control loop. As a result, each application will place requirements and specifications on timing and accuracy. This trade-off between time and accuracy is a major consideration for each particular application.

1.2 Motivating Example

Between the two vision communities, there are many techniques proposed to work with a variety of different underlying assumptions regarding target environment or object structure (e.g. [89],[19]), and relative motion characteristics (e.g. [21],[36]). In an attempt to clarify and compare these techniques and qualify the resulting suitability to various control scenarios, it is helpful to consider an example robot task.

Consider an aerial robot platform capable of motion in all six degrees of freedom (DOF). The robot is equipped with an Inertial Measurement Unit (IMU), Global Positioning Satellite (GPS) receiver, and a model of its dynamics. As a result, the robot is capable of stabilizing itself and tracking trajectories in an Earth-fixed inertial frame.

The robot is tasked to perform an operation with respect to a ship – perhaps maintaining surveillance, inspecting the hull, or landing on the deck. The ship does not communicate its own pose in the Earth-fixed world frame to the robot, which must determine the relative motion using only its on-board sensors. There are two scenarios for the motion of the ship: (a) the ship is docked and therefore, stationary in the inertial frame; and (b) the ship is moving. For the latter case, the motion of the ship can be due by its own propulsion, as well as disturbances such as the motion of the water through currents and waves. As a result, the IMU, GPS, and dynamic model, which measure the robot's motion in world coordinates, do not provide a measure of the relative motion of the robot with respect to the moving ship and are therefore not directly applicable for the completion of the relative positioning task.

The robot is able to carry multiple cameras in a variety of configurations: (i) no cameras; (ii) a single camera (monocular); (iii) a stereo camera pair; or (iv) a cluster of cameras with non-overlapping field of view (FOV). These four configurations are shown in Figure 1.2. It is assumed that the robot's cameras can observe the ship and measure 2D locations of point features within the images.

With no camera (i) the robot would, at best, be able to perform the tasks upon the stationary ship provided that its location in the world frame is well-known. For the monocular camera case (ii), the tasks could be performed relative to the moving ship; however, the single optical centre means that the scale of the motion and structure recovered is ambiguous [10]. The scale metric, which is vital for feedback control, cannot be determined from these image measurements. The stereo camera

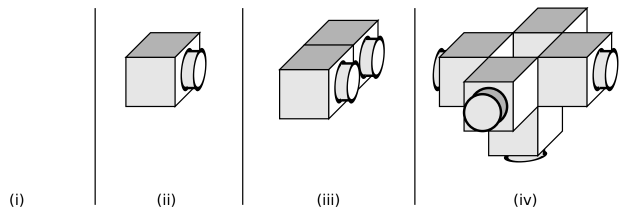


Figure 1.2: Four configurations for the cameras used in the robot and ship example: (i) no cameras; (ii) monocular camera; (iii) stereo camera; (iv) camera cluster with non-overlapping FOV.

(iii) is able to resolve the scale of the motion and structure using the known baseline length between the two cameras. However, the limited FOV from the overlapping cameras leads to ambiguities between small rotations and translations, which result in poor localization accuracy [25], similar to the monocular case.

The cluster (iv) is composed of any number of simple perspective cameras mounted rigidly with respect to each other and can be arranged such that their FOV are spatially disjoint. This arrangement makes effective use of the camera sensors to cover a large combined FOV with high resolution, and is able to overcome the limitations of other camera configurations, such as scale and translation-rotation motion ambiguities [63]. Additionally, by arranging the cameras to look in all directions, the pose estimation is made more robust since when certain cameras do not see any point features suitable for tracking, the other cameras in the cluster can maintain the localization. In this scenario, camera arrangements with a smaller FOV would become lost and the tracking operation would fail.

1.3 Proposed Solution

This thesis presents the design and analysis of a relative pose estimation system using a cluster of cameras which can be mounted on a robotic platform suitable for use in precision control applications. The system is able to estimate the current position and orientation of the cluster in real-time relative to an unknown target object or environment. The optimization can be initialized from the first set of camera images, even for clusters in which there is no overlap in the FOV of the component cameras, as in Figure 1.3. Furthermore, since only camera measurements are used, the target object or environment can be in motion and the system will estimate the relative pose of the robot and target.

Similar to [45], the task of pose estimation is run in parallel with a nonlinear

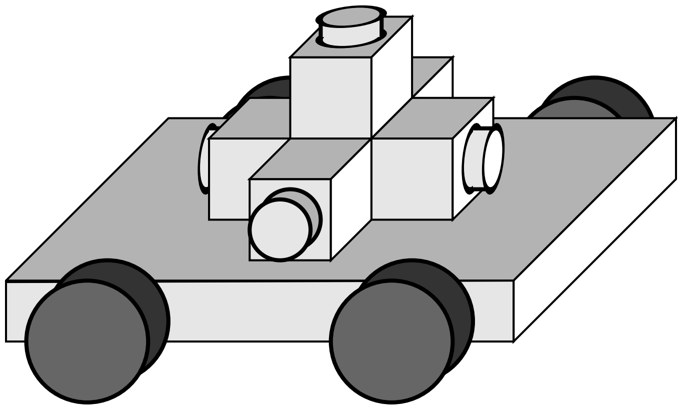


Figure 1.3: Multiple monocular cameras can be arranged to form a cluster such that they cover as large a FOV as possible, with no overlap.

target-modelling optimization which generates the target model as a set of point features parameterized within a subset of the camera trajectory poses selected to act as keyframes. The pose estimation proceeds at the frame rate of the cameras providing estimates with respect to the most up-to-date target model. The parameter space of the modelling optimization problem is formulated using the concept of $\boxplus$-manifolds [33] which are able to encapsulate and enforce the global topology of the spaces related to 3D motions and allow the system representation to avoid singularities. Additionally, a new point feature position parameterization is proposed that isolates the effect of global scale error to a single parameter and allows the solution to converge quickly despite large initial uncertainty in feature depth.

The proposed system is subsequently analyzed to determine the configurations of the cluster camera poses, relative motion, and target structure which lead to the system becoming under-constrained and degenerate. A set of conditions on the system parameters are detailed for the case of a two-camera cluster with non-overlapping FOV, allowing the estimator to determine when the system is close to degeneracy and the solution will be ill-conditioned.

Finally, a software implementation is presented and shown to produce accurate real-time relative pose estimates using a camera cluster mounted on a small aerial vehicle and compared against pose measurements from a motion capture system. The performance is demonstrated for camera clusters both with and without overlapping FOV, in both indoor and outdoor environments.

1.4 Contributions

The main contributions claimed within this thesis are as follows.

- A novel visual SLAM formulation, based on $\boxplus$-manifolds [33], for the specific case of calibrated multicamera clusters,
- A novel parameterization for point features in the target model, represented relative to an anchoring camera frame and updated using a spherical coordinate-based transformation isolating the effect of global scale error,
- A novel initialization scheme for the pose estimation system that converges even in the case of completely non-overlapping FOV camera clusters,
- A novel analysis of the configurations leading to solution degeneracy for the full motion and structure estimation system using an iterative nonlinear least-squares optimization method with a non-overlapping FOV camera cluster,
- A real-time implementation of the proposed algorithms, based on the MCPTAM software [30], mounted on an aerial robot platform, and shown to produce accurate pose estimates compared to high-precision ground truth measurements.

1.5 Thesis Organization

The remainder of this thesis is organized as follows:

Chapter 2 provides an overview of the vision-based position and orientation estimation problem. A summary of techniques from both the robotics and computer vision communities are presented for monocular camera systems to provide the fundamentals for the multicamera estimation methods. A thorough literature review of the state-of-the-art for pose estimation using calibrated multicamera clusters follows.

Chapter 3 presents the novel formulation of the multicamera cluster SLAM problem based on recent results in state manifold representations. A novel parameterization for point features anchored in an observing camera frame is detailed, in which the positions are updated using a spherical coordinate transformation to isolate the effects of scale error. Finally, an initialization method is provided for clusters with non-overlapping FOV to allow the optimization to operate from the first set of camera images with no prior knowledge of the target structure.

Chapter 4 presents an analysis of the degenerate configurations of the camera cluster, relative motion, and target model structure leading to an under-constrained system optimization when a two-camera cluster system with non-overlapping FOV observes a set of point features over two poses. A zero-determinant condition is identified to indicate when the system is degenerate and the solution is ambiguous.

Chapter 5 demonstrates the performance of the proposed algorithm implemented in software for a cluster mounted on a small aerial robot. The system is shown to be capable of real-time relative pose tracking using camera cluster configurations both with and without FOV overlap. The pose estimates are compared to high-precision optical motion capture measurements to demonstrate that when the degenerate configurations from Chapter 4 are avoided, the system estimates the cluster motion accurately in both indoor and outdoor environments.

Chapter 6 draws conclusions for the thesis and provides several suggestions for future research directions in this area.

Chapter 2

Background

The idea of extracting scene structure from images is well-established and is an extremely popular topic of research. A great deal of research has looked at how to derive both camera motion and environment structure from an image sequence captured by a single camera. By comparison, a smaller body of work has considered using calibrated multicamera clusters with non-overlapping fields-of-view. Many of the multicamera techniques are motivated by ideas from the monocular case. This chapter will present a brief review of important single-camera techniques for estimating camera motion and rigid scene structure, and then provide a detailed review of the use of multicamera clusters for pose and structure estimation.

2.1 Single Camera Estimation

Extracting structure and motion from a single moving camera is appealing because the hardware required is simple, inexpensive, and most images and video sequences are captured in this format. One frame alone does not contain enough information to extract structure without making significant assumptions about features in the image [31] so a number of images viewing the target from different locations are used.

The type of individual cameras considered in this thesis are all of the central projection class [81], in which all of the observed rays pass through a single point in the camera sensor. Only the bearing to a point can be measured using a single camera of this type. That is, the 3D position of a point measured by a central projection camera in one image is constrained only in two dimensions and is free to move in the degree-of-freedom along the measured ray.

2.1.1 Projective Geometry

The projective space, $\mathbb{P}^n$, consists of the real vector space $\mathbb{R}^n$, with the addition of points at infinity [31]. The projective space representation provides a convenient way of representing the camera measurement system in terms of homogeneous transformations and also provides a mechanism for dealing with points which are, for practical purposes, infinitely far from the camera, such as those on the horizon line. Only a very brief description of the projective space is presented here and the reader is referred to [31] for a more thorough introduction.

A point in the projective space is represented by the $n+1$ homogeneous coordinates,

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x}_1 & \tilde{x}_2 & \dots & \tilde{x}_{n+1} \end{bmatrix}^\top \in \mathbb{P}^n. \tag{2.1}$$

The points at infinity in $\mathbb{R}^n$ are represented by those with coordinate $x_{n+1} = 0$. For finite points in $\mathbb{R}^n$ – when $x_{n+1} \neq 0$ – the coordinates of the corresponding point $\mathbf{x} \in \mathbb{R}^n$ are determined by,

$$\begin{aligned} \mathbf{x} &= \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^\top & (2.2) \\ &= \begin{bmatrix} \frac{\tilde{x}_1}{\tilde{x}_{n+1}} & \frac{\tilde{x}_2}{\tilde{x}_{n+1}} & \dots & \frac{\tilde{x}_n}{\tilde{x}_{n+1}} \end{bmatrix}^\top. & (2.3) \end{aligned}$$

Notice that there is no way of mapping a point at infinity back to $\mathbb{R}^n$ since it would require division by zero.

Each ray in the projective space maps to the same point in the real vector space. As a result, the points $\tilde{\mathbf{x}}$ and $\lambda\tilde{\mathbf{x}}$, for $\lambda \in \mathbb{R}$, map to the same point $\mathbf{x} \in \mathbb{R}^n$. Not surprisingly, there is an extra degree of freedom in the projective vectors using $n+1$ coordinates to represent a n-dimensional space. Finally, it is possible to represent any point $\mathbf{x} \in \mathbb{R}^n$ in the corresponding projective space $\mathbb{P}^n$ simply by augmenting the coordinates,

$$\tilde{\mathbf{x}} = \begin{bmatrix} \mathbf{x}^\top & 1 \end{bmatrix}^\top. \tag{2.4}$$

The work in this thesis will make use of the projective spaces $\mathbb{P}^2$ and $\mathbb{P}^3$ to represent points in $\mathbb{R}^2$ and $\mathbb{R}^3$ plus the points on the line or plane at infinity, respectively. Specifically, these spaces allow for projective and coordinate transformations to be represented as linear matrix operations. It will sometimes be necessary to move between the respective real and projective spaces, and the following promotion and demotion operators are provided. The projective promotion operator $\tilde{\boldsymbol{\rho}} : \mathbb{R}^n \to \mathbb{P}^n$ maps a point $\mathbf{x}$ in the real vector space to its representation in the projective space,

$$\tilde{\boldsymbol{\rho}}\left(\mathbf{x}\right) = \begin{bmatrix}\mathbf{x}^\top & 1\end{bmatrix}^\top. \tag{2.5}$$

The projective demotion operator $\boldsymbol{\pi}_n : \mathbb{P}^n \to \mathbb{R}^n$ maps a point $\tilde{\mathbf{x}}$ in the projective space back to the corresponding point $\mathbf{x}$ the real vector space,

$$\begin{aligned}\mathbf{x} &= \boldsymbol{\pi}_n\left(\tilde{\mathbf{x}}\right) && (2.6)\\ &= \begin{cases}\text{undefined} & \text{if } x_{n+1} = 0\\ \begin{bmatrix}\dfrac{\tilde{x}_1}{\tilde{x}_{n+1}} & \dfrac{\tilde{x}_2}{\tilde{x}_{n+1}} & \dots & \dfrac{\tilde{x}_n}{\tilde{x}_{n+1}}\end{bmatrix}^\top & \text{if } x_{n+1} \neq 0.\end{cases} && (2.7)\end{aligned}$$

Note that the result of this operator is undefined for points at infinity.

For the remainder of this thesis, unless it is ambiguous from the context, the promotion and demotion operators will be implied by the vector notion. The homogeneous coordinates for a given vector $\mathbf{x} \in \mathbb{R}^n$ will simply be written as $\tilde{\mathbf{x}} \in \mathbb{P}^n$, but implicitly, $\tilde{\mathbf{x}} \equiv \tilde{\boldsymbol{\rho}}\left(\mathbf{x}\right)$, and likewise, $\mathbf{x} \equiv \boldsymbol{\pi}_n\left(\tilde{\mathbf{x}}\right)$ assuming $\tilde{x}_{n+1} \neq 0$.

2.1.2 Pin-hole Camera Model

An individual perspective camera is modelled as a simple pin-hole imaging device, which maps 3D points onto a 2D plane called the image plane [53]. An example is shown in Figure 2.1. A 3D point $\tilde{\mathbf{p}}^{C_i} = \begin{bmatrix}\tilde{x}^{C_i} & \tilde{y}^{C_i} & \tilde{z}^{C_i} & \tilde{w}^{C_i}\end{bmatrix}^\top$, represented in the projective space $\mathbb{P}^3$, and expressed with respect to the ith camera coordinate frame, C_i, is mapped to a particular camera pixel on the image plane. The intersection of the point feature ray $\tilde{\mathbf{p}}^{C_i}$, through the optical centre, $\mathbf{o}_i$, with the image pixel plane D_i, occurs at the point, $\begin{bmatrix}\dfrac{-f}{P_x}\dfrac{\tilde{x}^{C_i}}{\tilde{z}^{C_i}} + o_x^{D_i} & \dfrac{-f}{P_y}\dfrac{\tilde{y}^{C_i}}{\tilde{z}^{C_i}} + o_y^{D_i}\end{bmatrix}^\top \in \mathbb{R}^2$. The intrinsic camera parameters include, f, the focal length, P_x and P_y, the inter-pixel spacing of the camera sensor in the respective axes, and $(o_x^{D_i},\ o_y^{D_i})$, the coordinates on the image plane at the intersection with the optical axis. It is assumed that each camera has been intrinsically calibrated using one of the many existing offline techniques [31], and therefore all of the above parameters are known. If nonlinearities such as spherical distortion are present in the camera system, they can be identified and rectified through the calibration process so that the image measurements match those of the pin-hole camera model detailed here, and the subsequent results suffer no loss of generality.

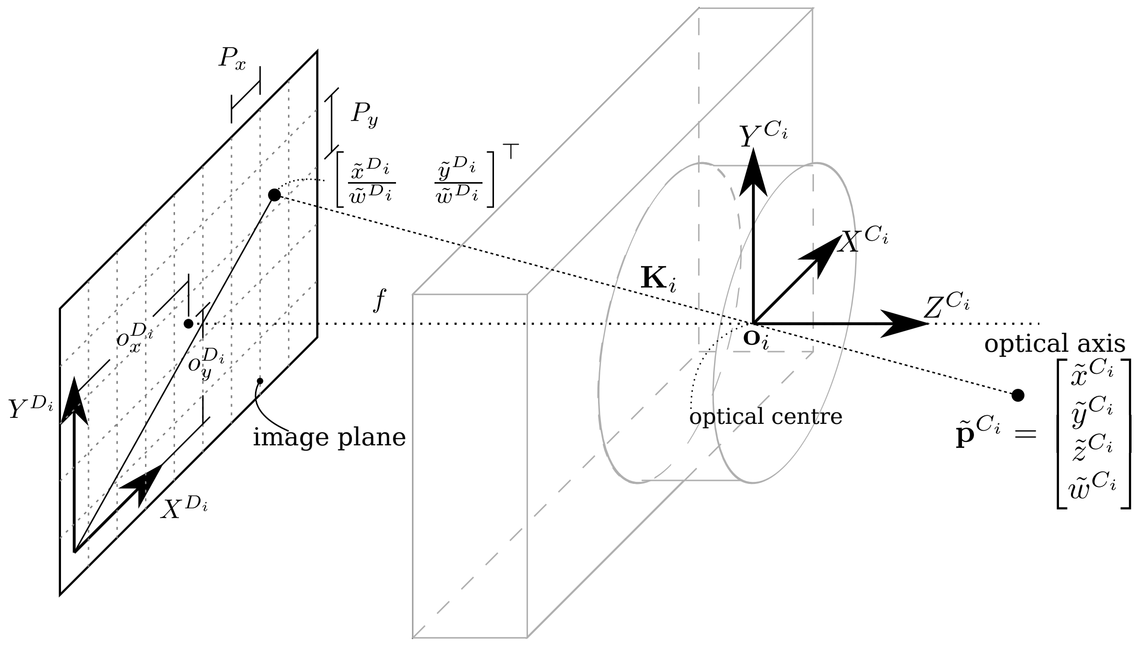

P_x
P_y
$\begin{bmatrix} \frac{\tilde{x}^{D_i}}{\tilde{w}^{D_i}} & \frac{\tilde{y}^{D_i}}{\tilde{w}^{D_i}} \end{bmatrix}^\top$
Y^{C_i}
X^{C_i}
$\mathbf{K}_i$
f
Z^{C_i}
$o_x^{D_i}$
$o_y^{D_i}$
$\mathbf{o}_i$
optical axis
Y^{D_i}
image plane
optical centre
$\tilde{\mathbf{p}}^{C_i} = \begin{bmatrix} \tilde{x}^{C_i} \\ \tilde{y}^{C_i} \\ \tilde{z}^{C_i} \\ \tilde{w}^{C_i} \end{bmatrix}$
X^{D_i}

Figure 2.1: A simple pin-hole camera measurement model is used to relate the camera frame coordinates to the camera image plane coordinates for a feature point.

The camera projection matrix, $\mathbf{K}_i$, maps a point in $\mathbb{P}^3$ into $\mathbb{P}^2$ on the image plane using the intrinsic calibration parameters,

$$\tilde{\mathbf{p}}^{D_i} = \mathbf{K}_i \tilde{\mathbf{p}}^{C_i} \tag{2.8}$$

$$\begin{bmatrix} \tilde{x}^{D_i} \\ \tilde{y}^{D_i} \\ \tilde{w}^{D_i} \end{bmatrix} = \begin{bmatrix} -\dfrac{f}{P_x} & 0 & o_x^{D_i} & 0 \\ 0 & -\dfrac{f}{P_y} & o_y^{D_i} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \tilde{x}^{C_i} \\ \tilde{y}^{C_i} \\ \tilde{z}^{C_i} \\ \tilde{w}^{C_i} \end{bmatrix}, \tag{2.9}$$

which is subsequently mapped to the actual image pixel plane coordinates through the demotion function for $\mathbb{P}^2$,

$$\boldsymbol{\pi}_2(\tilde{\mathbf{p}}^{D_i}) = \begin{bmatrix} \dfrac{\tilde{x}^{D_i}}{\tilde{w}^{D_i}} \\ \dfrac{\tilde{y}^{D_i}}{\tilde{w}^{D_i}} \end{bmatrix} = \begin{bmatrix} -\dfrac{f}{P_x} \dfrac{\tilde{x}^{C_i}}{\tilde{z}^{C_i}} + o_x^{D_i} \\ -\dfrac{f}{P_y} \dfrac{\tilde{y}^{C_i}}{\tilde{z}^{C_i}} + o_y^{D_i} \end{bmatrix}, \quad \tilde{w}^{D_i} \neq 0. \tag{2.10}$$

A physical camera of this type is only able to observe points in front of the lens so every point is constrained to have,

$$\left|\tilde{z}^{C_i}\right| > 0, \quad \text{and} \quad \tilde{z}^{C_i}\tilde{w}^{C_i} \geq 0 \tag{2.11}$$

which satisfies (2.10) since

$$\tilde{w}^{D_i} = \tilde{z}^{C_i} \neq 0. \tag{2.12}$$

2.1.3 Point Feature Target Object Model

The tracked target object or environment, henceforth referred to simply as the target, is assumed to be a rigid body which contains a set of visible point features. A point feature is a visually distinguishable point on the tracked physical target that corresponds to a unique 3D position in a local target coordinate frame and is measureable in a set of camera images through a relative motion sequence.

Image measurements of these point features are extracted from the images using image processing techniques, including feature extraction algorithms like the FAST corner detector [65, 66], the Scale-Invariant Feature Transform (SIFT) [52], or Speeded-Up Robust Features (SURF) [6]. An example of a moving target, a ship, with a set of corner point features is shown in Figure 2.2.

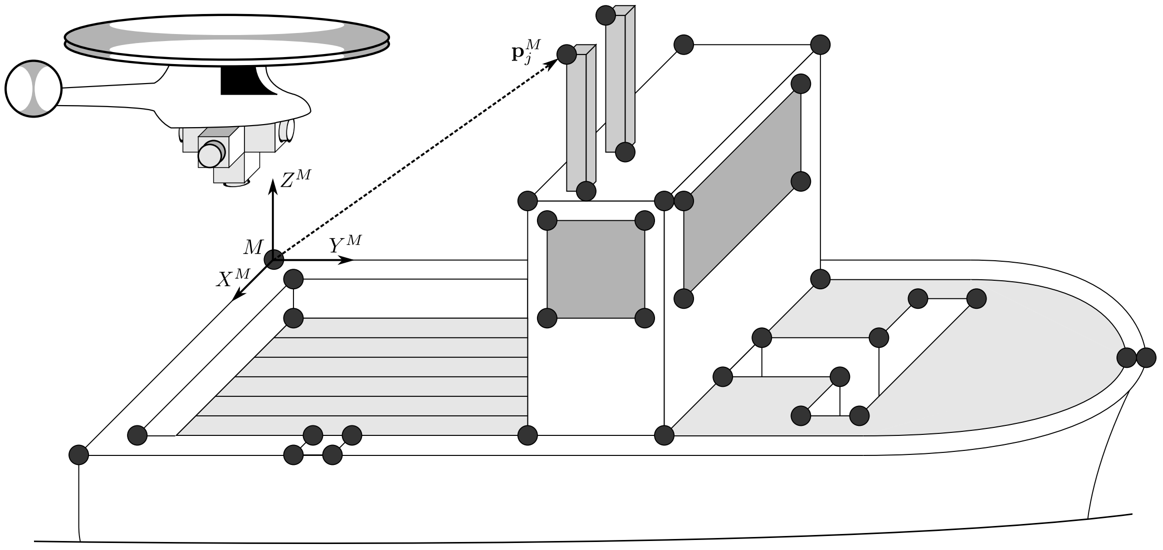


Figure 2.2: The target object is a rigid body consisting of a set of point features, $\mathbf{p}_j^M$, at fixed locations in a local target model coordinate frame, M. Point features can be identified and measured in the image plane of any observing cameras. In this example, the cameras on the aerial vehicle measure the image coordinates of visible corners on the moving ship.

The locations of the point features are constrained to be fixed with respect to each other. This allows for the relative target pose to be fully characterized by a single homogeneous transformation matrix in the Special Euclidean group $SE(3)$ [59] representing the position and orientation of the local target model frame, M, with respect to a reference coordinate frame, W,

$$\mathbf{T}_M^W = \begin{bmatrix} \boldsymbol{\mathcal{R}}_M^W & \mathbf{t}_M^W \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in SE(3), \tag{2.13}$$

where $\boldsymbol{\mathcal{R}}_M^W \in SO(3)$, $\mathbf{t}_M^W \in \mathbb{R}^3$, and $\mathbf{0}$ is the zero matrix with the specified dimensions.

2.1.4 Multiple View Geometry

Traditionally, the focus of the structure and motion techniques developed in computer vision is to extract as much information as possible about the scene structure from a small number of camera images collected at various positions, in a batch process run all at once [26, 31].

Epipolar Geometry

This section gives a brief introduction to epipolar geometry [26]. Consider the case of two cameras observing a point $\mathbf{p} \in \mathbb{R}^3$ from different poses as shown in Figure 2.3. The pair of image measurements from the cameras related to the same point feature is called a correspondence. For a single camera moving through time, consider the two cameras to be the same camera at different positions through a motion sequence.

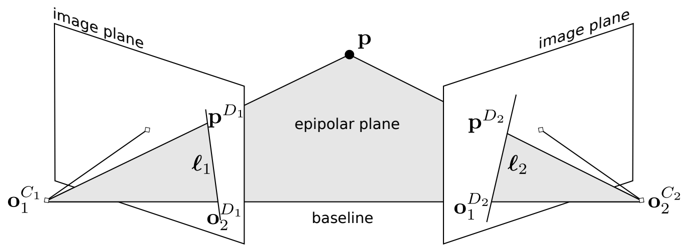


Figure 2.3: The epipolar constraint for a pair of calibrated cameras dictates that the observed point $\mathbf{p}$, the camera centres, $\mathbf{o}_1$ and $\mathbf{o}_2$, and the images of the observed point, $\mathbf{p}^{D_1}$ and $\mathbf{p}^{D_2}$, are all coplanar and lie on the epipolar plane [26]. Note that the image planes are drawn in front of the camera centre instead of behind, as previously shown. The geometry is unchanged, but convenient to visualize with the image in front.

It will be assumed in this section that the projection matrices $\mathbf{K}_i$ for all of the cameras have the form,

$$\mathbf{K}_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2.14}$$

The projection matrices can be made to fit this structure if the intrinsic parameters are known from prior calibration, by multiplying the points by the left-inverse of the projection matrix,

$$\begin{bmatrix} -\frac{P_x}{f} & 0 & \frac{P_x}{f} o_x^{D_i} \\ 0 & -\frac{P_y}{f} & \frac{P_y}{f} o_y^{D_i} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -\frac{f}{P_x} & 0 & o_x^{D_i} & 0 \\ 0 & -\frac{f}{P_y} & o_y^{D_i} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2.15}$$

The optical centres of the cameras are located at $\mathbf{o}_1^{C_1}$ and $\mathbf{o}_2^{C_2}$, each of which are known points in their respective camera coordinate frames, C_1 and C_2.

The coordinate frames of the two cameras are separated by a rigid motion consisting of a rotation followed by a translation. With respect to the first camera frame, the position and orientation of the second camera frame can be represented by a translation vector $\mathbf{t}_{C_2}^{C_1} \in \mathbb{R}^3$ and a rotation matrix $\boldsymbol{\mathcal{R}}_{C_2}^{C_1} \in SO(3)$. Together, these define a homogeneous coordinate transformation $\mathbf{T}_{C_2}^{C_1} \in SE(3)$ which maps points in C_2 into C_1,

$$\mathbf{T}_{C_2}^{C_1} = \begin{bmatrix} \boldsymbol{\mathcal{R}}_{C_2}^{C_1} & \mathbf{t}_{C_2}^{C_1} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}. \tag{2.16}$$

The images of each camera's centre on the other's image plane are called the epipoles [26]. The epipole related to the second camera centre in the first camera image plane occurs at,

$$\tilde{\mathbf{o}}_2^{D_1} = \mathbf{K}_1 \mathbf{T}_{C_2}^{C_1} \tilde{\mathbf{o}}_2^{C_2} \in \mathbb{P}^2. \tag{2.17}$$

Similarly the epipole for the first camera centre in the second camera image plane is at,

$$\tilde{\mathbf{o}}_1^{D_2} = \mathbf{K}_2 \mathbf{T}_{C_1}^{C_2} \tilde{\mathbf{o}}_1^{C_1} \in \mathbb{P}^2. \tag{2.18}$$

where

$$\mathbf{T}_{C_1}^{C_2} = \left(\mathbf{T}_{C_2}^{C_1}\right)^{-1} \tag{2.19}$$

$$= \begin{bmatrix} \left(\boldsymbol{\mathcal{R}}_{C_2}^{C_1}\right)^\top & -\left(\boldsymbol{\mathcal{R}}_{C_2}^{C_1}\right)^\top \mathbf{t}_{C_2}^{C_1} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}. \tag{2.20}$$

In each camera image, the line through the epipole and the image of the observed point is known as the epipolar line, $\boldsymbol{\ell}_i$, for that particular point. The camera centres and the observed point form a plane known as the epipolar plane. It can be seen in Figure 2.3 that the epipoles and point images fall on this plane as well. Accordingly, the epipolar constraint dictates that the camera translation vector, and the two point images are coplanar when all are written in the first camera coordinate frame. This can be expressed using the scalar triple product as,

$$\tilde{\mathbf{p}}^{D_1} \cdot \left[\mathbf{t}_{C_2}^{C_1} \times (\boldsymbol{\mathcal{R}}_{C_2}^{C_1} \tilde{\mathbf{p}}^{D_1})\right] = 0, \tag{2.21}$$

or more compactly as

$$\left(\tilde{\mathbf{p}}^{D_1}\right)^{\top} \mathbf{E} \tilde{\mathbf{p}}^{D_2} = 0, \tag{2.22}$$

where $\mathbf{E} = \left[\mathbf{t}_{C_2}^{C_1}\right]_{\times} \boldsymbol{\mathcal{R}}_{C_2}^{C_1} \in \mathbb{R}^{3\times 3}$, and $[\mathbf{a}]_{\times}$ is the skew-symmetric matrix such that $[\mathbf{a}]_{\times} \mathbf{x} = \mathbf{a} \times \mathbf{x}$, with $\mathbf{a}, \mathbf{x} \in \mathbb{R}^3$ [26]. The matrix $\mathbf{E}$ is called the essential matrix [51], and is a rank-2 matrix with two equal non-zero singular values [31].

It is important to note that the global scale of any solution estimated using a single perspective camera is not unique regardless of the method used [26]. In fact, for any non-zero scale factor $\lambda \in \mathbb{R}$, the camera translation and point locations can be scaled without changing the image plane measurements,

$$\tilde{\mathbf{p}}^{D_1} = \mathbf{K}_1 \begin{bmatrix} \boldsymbol{\mathcal{R}}_{C_2}^{C_1} & \lambda \mathbf{t}_{C_2}^{C_1} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} \lambda \mathbf{p}^{C_2} \\ 1 \end{bmatrix} = \mathbf{K}_1 \begin{bmatrix} \lambda(\boldsymbol{\mathcal{R}}_{C_2}^{C_1} \mathbf{p}^{C_2} + \mathbf{t}_{C_2}^{C_1}) \\ 1 \end{bmatrix}, \tag{2.23}$$

resulting in

$$\mathbf{p}^{D_1} = \begin{bmatrix} \frac{\lambda \tilde{x}^{D_1}}{\lambda \tilde{w}^{D_1}} & \frac{\lambda \tilde{y}^{D_1}}{\lambda \tilde{w}^{D_1}} \end{bmatrix}^T = \begin{bmatrix} \frac{\tilde{x}^{D_1}}{\tilde{w}^{D_1}} & \frac{\tilde{y}^{D_1}}{\tilde{w}^{D_1}} \end{bmatrix}^T, \tag{2.24}$$

which are the same as the unscaled solution. As a result, a solution will be said to have been recovered *up-to-scale*, if it can be multiplied by a scale factor in this manner without affecting the measurements.

Bundle Adjustment

Bundle Adjustment (BA) is a full nonlinear least squares optimization on the image-space reprojection error of a set of point feature correspondences through a sequence of camera frames, minimizing the associated Mahalanobis distance cost function [31], $c : \mathbb{R}^n \to \mathbb{R}$,

$$c(\mathbf{x}) = \sum_{j,k} \bar{\mathbf{z}}_{j,k}^T (\mathbf{R}_{j,k})^{-1} \bar{\mathbf{z}}_{j,k}, \tag{2.25}$$

$$\bar{\mathbf{z}}_{j,k} = \mathbf{z}_{j,k} - \mathbf{g}_{j,k}(\mathbf{x}), \tag{2.26}$$

where $\mathbf{z}_{j,k} \in \mathbb{R}^2$ are the measurements of point feature j at camera pose k, $\mathbf{R}_{j,k} \in \mathbb{R}^{2\times 2}$ is the measurement noise covariance, $\mathbf{g}_{j,k} : \mathbb{R}^n \to \mathbb{R}^2$ maps the parameter set to the predicted image point for the j^{th} point feature predicted measurement at camera pose k, and $\mathbf{x} \in \mathbb{R}^n$ is the vector containing all of the estimated pose and feature parameters to be optimized.

BA solves an optimization problem by iterating on a solution for the camera motion and environment structure parameters with all available feature correspondences over all camera frames at the same time until convergence is achieved. The optimization takes place over the set of camera pose parameters for each frame, as well as all of the position parameters for the observed point features.

Bundle Adjustment can been implemented using any optimization method, but it is commonly solved using the Levenberg-Marquardt (LM) algorithm, as shown in [31]. Some efficient semantic optimizations to the naive formulation can be realized by close examination of the resulting sparse structure of the matrices in the parameter update equations [31] based on the Schur complement [1], and the sparse secondary structure [46]. They can be used to reduce the computational complexity of calculating a state estimate mean to being linearly proportional to the number of feature points. However, the computation is still approximately cubic in the number of camera frames over which the optimization is performed.

Typically, BA is used to estimate a large number of environment feature points from a small number of spatially well-separated frames. The computational requirements are therefore suited to this scenario since it is expensive to optimize over a large number of frames, but a large number of feature points across a few frames can be estimated accurately and efficiently [58]. Recent results in large-scale BA are reported in [1] and [2] using Internet picture databases and recreating entire city blocks using supercomputers over a period of several hours. The problem is parallelized and good results are shown for reconstructions with hundreds of thousands of points.

Bundle Adjustment offers high accuracy of reconstruction but suffers from some disadvantages: the feature correspondence problem is difficult to solve since the camera frames may have been captured at locations far apart in space; the initial estimates must be in the neighbourhood of the true solution to achieve convergence and avoid local minima; and the computation time required is cubic in the number of frames being considered. Algebraic object-space error methods like the eight [51] and five-point [61] algorithms are commonly used to seed BA since they do not need initial estimates to find a solution. However, the correspondence problem remains difficult to solve. Additionally, due to the high computational requirements with respect to the number of camera frames included in the optimization, the standard BA approach is not suitable to real-time robot tracking tasks in which the current pose of a camera or robot is tracked through a long sequence of frames.

2.1.5 Simultaneous Localization and Mapping

Within the robotics community, the Simultaneous Localization and Mapping (SLAM) problem involves a mobile robot being placed at an unknown location in an unknown environment and incrementally building a consistent map of that environment while concurrently localizing itself within it [23]. For a more complete introduction to the SLAM problem, the reader is referred to [23, 4, 82].

Relative Pose Estimation

This section considers the estimation-theoretic formulation of SLAM identified in [22]. In this framework, the solution to the navigation problem can be found recursively at each time step and the uncertainties in the system and resulting estimates are represented as random variables.

In contrast with the camera motion estimation techniques from computer vision where the solution provides discrete camera poses for each image, SLAM treats the relative motion as a dynamic system with the point feature 3D positions as parameters of that system. A robotic platform moving through the environment is able to make measurements of the relative location between the landmarks and the robot itself [22]. By making successive observations over time at different locations, an algorithm concurrently builds a statistical map of the feature locations and tracks the robot location with respect to it.

This type of SLAM system can be represented by a nonlinear discrete-time state-space system by choosing an appropriate set of states, $\mathbf{x} \in \mathbb{R}^n$, as well as suitable process and measurement models to represent the system dynamics and outputs, $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^m$,

$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t) + \boldsymbol{\eta}_t, \tag{2.27}$$

$$\mathbf{z}_t = \mathbf{g}(\mathbf{x}_t) + \boldsymbol{\gamma}_t, \tag{2.28}$$

where $\boldsymbol{\eta}_t \sim \mathcal{N}\left(\mathbf{0}_{n\times 1}, \mathbf{Q}_t\right)$ and $\boldsymbol{\gamma}_t \sim \mathcal{N}\left(\mathbf{0}_{m\times 1}, \mathbf{R}_t\right)$ are vectors of zero-mean Gaussian disturbance and measurement noise, respectively, with noise covariance matrices

$\mathbf{Q}_t \in \mathbb{R}^{n\times n}$ and $\mathbf{R}_t \in \mathbb{R}^{m\times m}$. The measurements of the image point features in the camera images are modelled using the pin-hole camera model from Section 2.1.2.

The full SLAM system state vector is formed by augmenting the position and orientation states, $\mathbf{w}$, of the robot in the world frame, with the parameters for the map point features, $\mathbf{m}$, represented in the target model coordinate frame M,

$$\mathbf{x} = \begin{bmatrix} \mathbf{w} \\ \mathbf{m} \end{bmatrix}. \tag{2.29}$$

Typically, SLAM solutions assume that the observed environment is stationary and the relative motion can be modelled by the kinematics or dynamics of the robot alone [82]. When both the robot and the target object or environment are free to move independently, the relative motion dynamics are commonly assumed to be well-approximated by the constant velocity or constant acceleration process models [19, 89]. This model assumes that the six pose parameter velocities or accelerations, respectively, are subject to random walk [27], and integrates them into the position states accordingly.

The ability of the constant velocity process model to approximate the true dynamics of a physical relative motion system depends on the magnitude of the sampling period. Decreasing the sampling period reduces the apparent change in velocity between frames and the constant velocity model provides a better approximation of the motion. This implies that using cameras with high frame rates will result in better relative pose estimates. However, for camera-based estimation, there is a trade off since smaller camera motions lead to poor point triangulation over small baselines and the accumulation of round-off error. Furthermore, the constant velocity and constant acceleration dynamic models create a low-pass filter which introduces some lag into the estimation system. As a result, careful selection of the process noise parameters is essential if the estimates are to be used in a real-time control system.

Recursive Estimation

Recursive filters, such as the Extended Kalman Filter (EKF) and its derivatives, are used extensively to provide estimates of a system, evolving in time, given a dynamic model and noisy measurements of the system outputs [82]. A recursive filter is able to integrate each new set of measurements as they arrive online, and maintain an up-to-date estimate of the system. This is ideal for a mobile robot using the estimated pose for feedback control purposes. The recursions require computation time that is linear in the number of camera frames. However, due to the dense structure of the covariance matrix that is maintained for the estimate, the computational requirements at each step is approximately cubic in the number of features. This complexity can be contrasted with BA, which grows linearly in the number of features, but cubically in the number of poses.

After initialization, the EKF proceeds recursively in two steps: a prediction step using the process model of the system dynamics; and a measurement update step that adjusts the predicted states based on the measured outputs and relative magnitudes of the disturbance and measurement noise covariances. Both of these steps linearize the process and measurement models about the current operating point to compute the evolution of the covariance matrix through the motion trajectory. The repeated marginalization of the measurement information into the error covariance matrix at each time step can introduce a systematic error into the estimate when the process or measurement models are poorly approximated by the linearization about the current state estimate.

Literature Review

While the SLAM process applies most generally to mobile robots equipped with any number of different types of sensors, the use of cameras as the primary sensor has been investigated by many researchers. Techniques differ not only in the optimization method chosen but also whether they use other sensors or prior knowledge to resolve global scale.

Deans and Hebert [20] investigate 2D SLAM using a bearing-only sensor mounted on a mobile robot operating in a planar environment. After establishing the connection with computer vision techniques, they suggest using a hybrid approach applying a combined EKF and BA technique. More significant than the proposed algorithm itself is the insight offered by Deans and Hebert into the structure of the bearing-only SLAM problem. Specifically, a single image from a monocular camera is able to only determine the ray on which a feature point lies and in the two dimensional case, there are four gauge freedoms in the estimation. That is, the solution can be translated, rotated or scaled and the resulting measurements will be unchanged [53]. To overcome this limitation, an absolute coordinate frame and scale must be imposed as a set of constraints for the solution to be unique. Further, they assert that odometric data is sufficient to disambiguate the scale of the environment, but a bias in the odometry cannot be corrected by the bearing-only measurements.

The first significant application of the recursive SLAM framework using a single monocular camera was by Davison in [17]. The full 3D position and orientation of a hand-held camera with respect to an (almost) unknown environment is estimated in real-time, as well as the locations of a large set of feature points within the environment. The estimations are generated by an EKF, and the algorithm includes the ability to add or remove feature points from the map as new landmarks become available or others are no longer stable during the estimation. The use of a camera with a wide-angle lens is suggested in [18] and the process is further refined in [19].

Accomplishing a similar goal, Eade and Drummond also developed an early monocular camera SLAM algorithm [24] based on the FastSLAM filter proposed by Montemerlo *et al.* [57]. The FastSLAM algorithm is a variation of the Particle Filter [82] which capitalizes on the structure of the SLAM problem to decrease the computational requirements while maintaining accurate estimates. As a result, maps with large numbers of landmarks can be maintained at reduced computational cost compared with EKF-based solutions. The difficulty with this method comes when trying to maintain a sufficient number of particles to adequately represent the probability distributions over the entire state space. Even if there are a large number of particles, there may not be any particles near the correct state, a phenomenon known as particle deprivation [82]. Despite this, Eade and Drummond show good results for large maps of features with their system operating in real-time.

An improvement to Davison's framework is detailed in Civera *et al.* [10]. The significant contribution of this work is the explicit realization that the global scale of the map and robot pose cannot be recovered from monocular camera measurements alone. More importantly, it is demonstrated that the estimation can proceed with no a priori information about the environment or initial robot location and the resulting solution will be consistent in an up-to-scale manner. In other words, with only the monocular camera measurements, the global scale parameter will not converge to the correct value.

Feature Parameterization

The target model point feature positions can be modelled directly as the three Cartesian coordinates within the target model coordinate frame M. However, when there is a large amount of uncertainty associated with the 3D position of the point feature, the incorrect position estimate may cause the recursive filter or BA process to fail due to the poor-quality linearization that results, and its use in updating the system states. This is problematic for SLAM systems using vision since the initial depths to the point features from the camera are impossible to determine from the first observations. This has lead researchers to formulate point feature position parameterizations which accommodate for this uncertainty and allow the estimators to converge to the solution despite poor initial estimates.

Civera *et al.* [12] present the Inverse Depth Parameterization (IDP) for feature points measured using bearing-only sensors, which effectively encapsulates the relative uncertainty arising from these measurement systems in a single inverse depth parameter instead of distributing it over all parameters used to define a feature location.

The j^{th} feature point position is represented in the local target model frame, M, as the sum of an initial observation point and an observation ray,

$$\mathbf{p}_j^M = \begin{bmatrix} x_o^M \\ y_o^M \\ z_o^M \end{bmatrix}_j + \frac{1}{\rho_j} \begin{bmatrix} \cos\alpha\sin\beta \\ -\sin\beta \\ \cos\alpha\cos\beta \end{bmatrix}_j , \tag{2.30}$$

where $[\ x_o^M\ y_o^M\ z_o^M\ ]_j^\top \in \mathbb{R}^3$ is the position of the camera centre in the object frame when the feature point was first observed, $\alpha_j,\ \beta_j \in \mathbb{R}$ are the azimuth and altitude angles respectively, to the feature point from this first observation point, and $\dfrac{1}{\rho_j}$ is the distance along those bearings to the feature point. It is important to note that the azimuth and altitude angles are specified with respect to the target coordinate frame. This parameterization is shown in Figure 2.4.

By using this parameterization, the feature point can be more effectively estimated using the EKF, since the resulting measurement model has better linearity properties than when the parameters are the Cartesian coordinates. Since the model linearizations are valid over a larger region, propagating the Gaussian state estimates through the process and measurement equations results in more Gaussian-like distributions and the filter is able to provide more accurate estimates and converge over a larger set of initial conditions [12].

As a consequence of parameterizing the feature point with the inverse depth, points at infinity, those which are infinitely far away from the camera, can be approximated with $\rho \to 0$. When a feature point at or near infinity is tracked, only

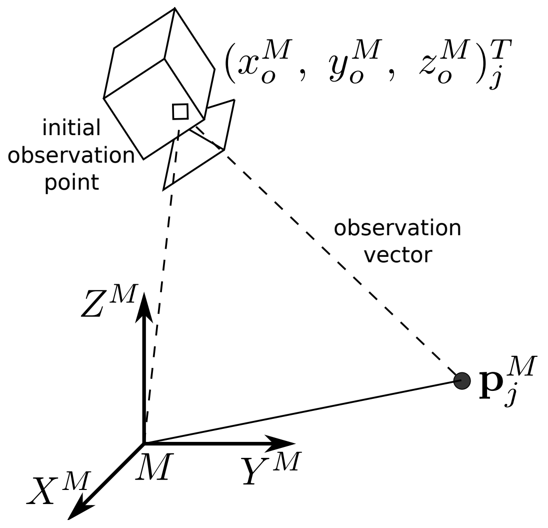


Figure 2.4: Each feature point is represented by the camera optical centre coordinates in the target model frame where the feature was first observed $[x_o^M,\ y_o^M,\ z_o^M]_j^\top$, the bearing to the point $(\alpha_j,\ \beta_j)$, and the inverse of the depth along that bearing, ρ_j.

relative orientation information is available from such measurements. IDP allows the EKF to integrate this information directly.

Finally, the initialization of new points into the EKF framework can be done with no prior knowledge of feature locations, and on the first observation of that new feature point. With previous monocular camera systems, initializing a new feature point had to be done in a separate batch optimization over several observations to try and reduce the uncertainty associated with the feature point Cartesian coordinates such that the EKF could correctly estimate the location [17] [24]. Without a roughly accurate initial estimate, the filter may diverge.

The initial inverse depth estimate has the most flexibility when assigning a value. Suppose that one is 95% confident that the features will have a depth on the interval $[d_{min}, d_{max}]$, then the associated inverse depth mean and variance can be set to,

$$\rho_j = \frac{d_{min} + d_{max}}{2 d_{max} d_{min}} \tag{2.31}$$

$$\sigma_\rho^2 = \left(\frac{d_{max} - d_{min}}{4 d_{max} d_{min}} \right)^2 . \tag{2.32}$$

The flexibility of this parameterization comes at the cost of using six parameters for each point feature instead of three as with the Cartesian coordinate case.

However, once the location of the feature is known with relative certainty and the distribution associated with the estimated position becomes a sharp peak about the estimated mean, the parameterization can be reduced to only the three Cartesian coordinates in the target model frame [12]. This transformation reduces the dimension of the state vector and therefore, the computational cost of the filter. The problem of how and when to convert the inverse depth parameters to the Cartesian representation is addressed in [11].

In [73], Sola *et al.* propose Anchored Homogeneous Points (AHP) as a replacement for IDP. In AHP, the two bearing angles from IDP are replaced with a 3-vector of Cartesian coordinates representing the bearing to the point feature. AHP maintains the initial observation point, as well as the inverse depth parameter, resulting in seven parameters to represent each feature point. Parameterizing the observation vector as a set of four homogeneous coordinates, effectively the vector representation in $\mathbb{P}^3$, creates an extra degree of freedom for every point feature, providing the estimator with more flexibility when pushing the over-parameterized representation to the correct solution. The resulting measurement model is more linear compared with IDP and Sola *et al.* demonstrate that using AHP results in better performance in terms of convergence and accuracy using an EKF.

Keyframes

In [45], Klein and Murray present a decoupled camera relative pose tracking and BA algorithm called Parallel Tracking and Mapping (PTAM). Their approach is to separate the tasks of pose tracking and structure recovery into two parallel processes. The mapping component is accomplished by a Bundle Adjustment process, which is run using a subset of camera poses collected through the motion sequence, called keyframes. Point features are extracted and triangulated using the images captured at each keyframe. Each keyframe has an associated estimate of its pose relative to the initial pose of the camera. An example is shown in Figure 2.5.

Once an initial map of feature points is generated using two keyframes, the camera pose is tracked using a simple nonlinear weighted least-squares localization algorithm using the triangulated feature map as fixed. This parallel approach allows the pose tracking estimator to only include the pose states and assumes the map parameters are known at any point in time. The mapping process continues to run in parallel on the keyframe set to increase the accuracy of the pose tracking, which will continue with the new map estimates as they become available. The set of keyframes is augmented at the discretion of the tracking thread, to ensure

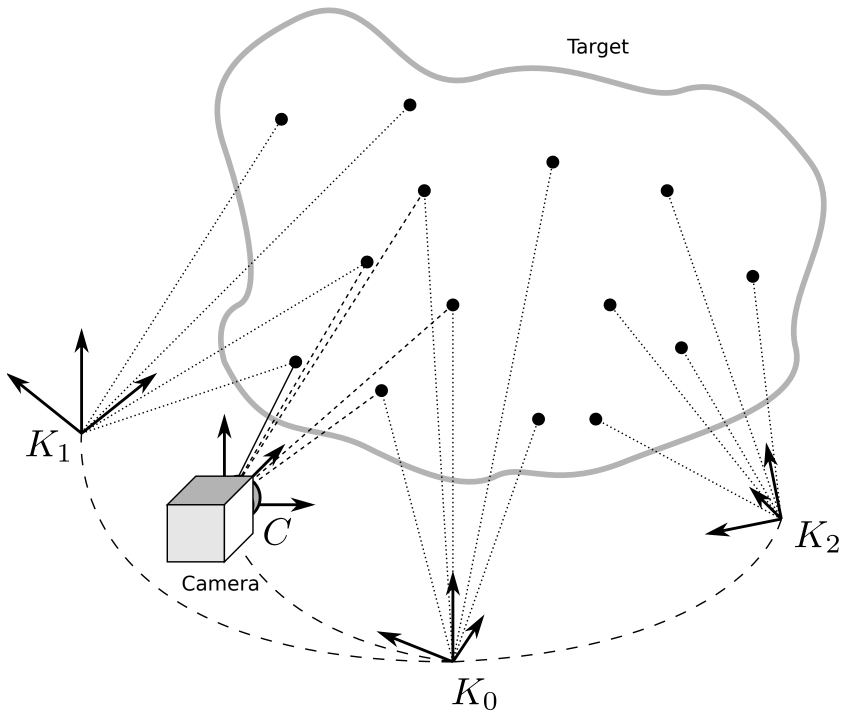


Figure 2.5: The mapping process performs a BA optimization at a set of sparsely-positioned keyframes, K_i, each with a pose and set of features observed as the camera moves through the environment. The current camera pose is tracked relative to these keyframes by making image measurements of the point features within them.

adequate keyframe coverage of the environment. When the camera is no longer able to see enough known feature points to maintain an accurate track, a new keyframe, and newly observed features are added to the map.

By limiting the number of keyframes in the map, the computational cost of the BA algorithm can be significantly reduced compared to a global optimization using all available camera images through the trajectory. Additionally, ensuring that the keyframes are well-separated spatially allows a small number of keyframes to cover the map and provide long baselines for accurate triangulation of point features.

Explicitly running the two processes in parallel allows the fast tracking algorithm to maintain a real-time estimate of camera pose. This alleviates the time constraints placed on the BA process, which is now updating and refining the map estimate when it is able. The separation of the tracking and mapping tasks is a powerful paradigm shift. Since the introduction of PTAM, researchers have begun to leverage the BA methods in real-time applications. The vision communities have moved away from using recursive filters and towards large-scale nonlinear optimization methods with parallel real-time tracking components localizing with respect to the most up-to-date map.

In [80], Strasdat *et al.* show that the BA solution is superior to the recursive filter solution, both in terms of accuracy and computational requirements in nearly every operating scenario, except in cases when the number of point features is very low and computational resources are limited. These results have subsequently mitigated the use of filters to solve the visual SLAM problem. Accordingly, the robotics vision community is interested in extremely large-scale mapping methods. These optimization methods aim to limit the number of keyframes during each mapping run, for example, by parameterizing the pose of keyframes in a chain with respect to one another, as in Relative BA [71], or performing a full-optimization on a local window of keyframe and point features, plus only the poses of a support set of outer-window keyframes, as in Double Window Optimization [79].

2.1.6 Ambiguities and Sensitivities

There are two main disadvantages associated with using a single camera to perform the estimation: ambiguities in the global scale; and ambiguities in motion from limited FOV. As demonstrated in Section 2.1.4, without prior knowledge of the target model or using supplementary sensors and measurements, it is impossible to recover the scale of the relative motion using a single monocular camera. This is due to the bearing-only nature of the sensor. For control purposes, scale information in the relative pose estimate is extremely important. Consider a simple proportional controller using the relative position estimates. The scale of the estimate affects the chosen gain and could lead to failure with the robot colliding with the target.

Another ambiguity arises when using a monocular camera with a limited FOV. The image of a target object in a perspective camera undergoing a small rotation about an axis parallel to the image plane creates a similar image to a small translation parallel to the camera image plane. As a result, any estimator will have problems distinguishing between these two motions using the image measurements.

These two effects combine to produce what is called the bas-relief ambiguity [31], which is the inability to distinguish the difference between a shallow object undergoing a large rotation and a deep object undergoing a small rotation. The problem is exaggerated as the camera FOV is limited. These ambiguities present difficulties for any monocular camera estimation scheme.

2.2 Multicamera Cluster Pose Estimation

In recent years, the trend has been to move away from single perspective cameras performing localization to using more complicated imaging systems, including clusters of perspective cameras fixed rigidly with respect to each other and facing in different directions such that their FOV have little or no overlap. In these calibrated camera clusters, the relative poses of all of the component cameras are assumed to be known precisely with respect to each other from a prior extrinsic calibration.

While there are motion estimation methods for non-overlapping multicamera clusters from more than 20 years ago which operate similarly to some of the current techniques [87], the more recent movement has largely been inspired by the observability analysis of Fermuller *et al.* [25]. Fermuller *et al.* show that a single perspective camera with a limited FOV will have fundamental problems estimating 3D motion due to confusion between some translations and rotations. For the five motion parameters (rotation and translation direction), the expected value of the errors are fundamentally mingled regardless of the estimation approach that is used. However, the ambiguities disappear when the FOV is increased to cover the entire viewing sphere, resulting in a well-defined global minimum for the cost function.

To take advantage of this, Baker *et al.* propose the Argus Eye system [5] composed of six perspective cameras placed in back-to-back pairs on each of the Cartesian axes. With this configuration, accuracy of the motion estimation is shown to be greatly improved. Further, since the cameras have their optical centres displaced from each other, the system allows for full metric reconstruction of the relative pose and target model, including scale, without using traditional stereo correspondence techniques between cameras.

In [63], Pless formulates the Fisher Information matrix for estimating camera motion using a variety of camera configurations, including the non-overlapping FOV camera cluster. This structure visibly highlights the ambiguities which exist with these different camera cluster configurations. Significantly, it is shown that a pair of cameras with overlapping FOV, as in traditional stereo, suffers from the same rotation-translation ambiguity as a single camera system. Further, the Argus Eye configuration eliminates this problem and each motion parameter can be estimated without confusion.

These analyses suggest that the ideal camera system would have as large a FOV as possible with the individual cameras arranged to complement each other.

An individual perspective camera is most sensitive to translational motion parallel to the image plane, and rotation around the optical axis. With this in mind, placing cameras at right angles to each other places the sensitive motions of each in alignment with the ambiguous motions of another, making the system collectively more sensitive. Additionally, using discrete cameras with their centres separated by some known non-zero distance allows for estimating the motion parameters completely, including the global scale. This is an improvement compared to a system like a omnidirectional catadioptric camera with one optical centre, which is like a single wide-angle perspective camera and therefore cannot recover global scale [43, 5].

Even in the case of a non-overlapping FOV camera cluster configuration, it is possible to recover the motion parameters when point features are tracked by each individual camera and not seen by another in the camera cluster. However, there are relative motions of the camera cluster and the target from which the motion and structure cannot be estimated using the image measurements. These are called critical motions [13], and can potentially cause an estimator to diverge or converge to an incorrect solution, if proper care is not taken.

The intuition behind how a calibrated cluster is able to track its relative motion without finding correspondence between point features across cameras is as follows. Each individual camera is able to estimate its own local motion increment up-to-scale in its own frame using its image sequence. Using the known cluster calibration, these local motions are combined to find the unique cluster translation and rotation which includes the proper scale value. Therefore, the world scale is embedded in the camera cluster extrinsic calibration.

2.2.1 Algorithm Classification Criteria

Previous motion estimation algorithms using camera clusters can be categorized using five criteria, each detailed in the subsections that follow.

A. Overlapping vs. Non-overlapping FOV

The first criterion is based on whether or not the algorithm requires (partial) overlap in the FOV of the component cameras in the cluster:

overlap FOV – There must be partial FOV overlap in at least two of the FOV of the cameras within the cluster for either initialization, subsequent operation, or both.

non-overlap FOV – The method is able to initialize and operate successfully using multicamera clusters with no overlap between any of the camera FOV.

The requirement that some of the component cameras have at least partial overlap in their FOV limits the collective FOV of the camera cluster. It is advantageous to have as much coverage of the entire viewing sphere as possible, and an algorithm which does not require FOV overlap for initialization and operation can make the most effective use of the available camera pixels.

B. Zero vs. Non-zero Baseline

The second criterion used to classify the previous algorithms relates to whether the camera clusters are approximated as a single spherical camera with a common optical centre:

zero baseline – The distances between the component camera optical centres is neglected and the camera is approximated as a spherical central camera.

non-zero baseline – The distances between the centres are used in the measurement model.

When the distance to the point features relative to the baseline between the cameras within the cluster are large, the image measurements are insensitive to errors on the system scale [43]. Accordingly, some authors propose to approximate the cluster as a single spherical camera with zero-length baselines and, effectively, a single optical centre. This allows the solution to be found using single-camera motion estimation techniques, but does not allow the global scale of the system to be recovered since it is embedded in the baseline length. As the cluster gets closer to the target point features, modelling error will deteriorate the solution as the spherical camera assumption is violated.

C. Decoupled vs. Coupled Operation

The third criterion relates to the point at which the camera images are combined to find the global cluster motion:

decoupled – Individual cameras in the cluster estimate their own local motion increment and global cluster motion is subsequently found by combining these motion estimates.

coupled – All camera image measurements are considered concurrently when generating the cluster global motion estimate.

Methods using the decoupled strategy, combining local motion estimates from individual cameras, suffer from the same issues of reduced accuracy and sensitivity to motion ambiguities as in monocular techniques, primarily stemming from limited FOV. Additionally, the combination step that resolves the global motion and scale does not properly account for camera image measurement noise. Finally, all cameras must each observe a sufficient number of feature points in order to estimate their own local motion. A preferable solution is the coupled strategy which considers all image measurements across all cameras to determine the global motion directly.

D. Visual Odometry vs. Localization

The fourth criterion relates to whether the global cluster motion is accumulated through increments or determined relative to a generated target environment model:

visual odometry (VO) – Considers only a small set of current and previous camera frames to estimate the cluster motion increment, which is then accumulated into the global motion trajectory.

localization – Builds a model of the unknown target object or environment through the image sequence, and the cluster position and orientation is localized with respect to the model.

Because the sensitivity of global solution scale is low due to the baseline-to-feature depth ratio requirement, and the prevalence of critical motions, the scale of each motion increment in the VO methods will accumulate error in the global estimate which will not be corrected through the sequence. For this reason, it is beneficial to maintain a model of the target as in the localization methods. If there is an error in the estimated scale of the solution, it will be reflected in the generated model and can be corrected over time when the relative motion is not critical.

E. Object-Space vs. Image-Space Error

The final criterion relates to the choice of distance error to be minimized in the cost function:

object-space error – Represented as distances between rays in 3D space for each measurement of a point feature at the different camera locations.

image-space error – The difference between the measured and reprojected pixel-coordinates on the camera image sensor for a particular point feature.

The object-space error methods solve systems of equations involving algebraic constraints transforming sets of rays in the images so that they intersect at the observed feature point locations in 3D space. The image-space error methods aim to minimize the error between the measured point feature image locations and those predicted by reprojection of a reconstructed model for the camera motion and target structure.

The image-space error methods are robust to image noise since they operate on the image measurement reprojection error directly. They are usually iterative or recursive optimization schemes which are accurate but can be computationally expensive and typically require a good initial estimate to converge to the global minimum. The object-space error methods are usually faster to execute and can converge from a larger set of initial conditions, but are more sensitive to image noise and have worse accuracy. Often, the object-space error methods are used to generate an initial condition for a more accurate image-space error method.

Ideal Method

The remainder of this chapter will outline the state-of-the-art for estimators using camera cluster configurations with respect to the criteria presented in this section. In terms of the goals of accurate real-time pose estimates for non-overlapping camera clusters, the ideal algorithm would have the following set of characteristics for the above criteria:

(A) non-overlap FOV

(B) non-zero baseline

(C) coupled

(D) localization

(E) image-space error

2.2.2 General Camera Model

Complex camera configurations, like the multicamera cluster, can be difficult to model by using traditional perspective camera models. This has lead to recent efforts to generalize the structure of camera models to allow complex imaging systems to be treated as one unified camera, resulting in the Generalized Camera Model (GCM) [28] with which Grossberg and Nayar propose to represent imaging systems more complex than regular perspective camera systems. The fundamental change is to replace the idea of pixels with a more general concept called a raxel. A raxel is a sample of light in a ray starting at a particular point, travelling in a particular direction. That is, each sensing element in the camera samples light travelling in a particular direction defined by the ray. With this concept, complex vision systems composed of multiple cameras or curved mirrors can be modelled by determining the rays of the incident light on the sensing elements. Calibrating a general camera means determining the mapping of the scene rays to the pixels on the image sensors.

The raxels can be parameterized by Plücker vectors [63]. The Plücker vectors for a line in $\mathbb{R}^3$ are two vectors $\mathbf{q},\ \mathbf{q}' \in \mathbb{R}^3$, the direction and moment respectively, such that,

$$\mathbf{q}' = \mathbf{q} \times \mathbf{p} \tag{2.33}$$

where $\mathbf{p} \in \mathbb{R}^3$ is a point on the line. Calibrating a general camera amounts to finding the set of $(\mathbf{q}, \mathbf{q}')_i$ for each raxel in the system.

This model is able to accommodate many different types of cameras, see [81], including catadioptric, wide-angle dioptric, and clusters, as shown in Figure 2.6.

An example of a raxel parameterization for a simple camera cluster is shown in Figure 2.7. A cluster coordinate frame C can be defined, in which the two optical centres are known, $\mathbf{o}_A$ and $\mathbf{o}_B$. Each of the four observed points, $\mathbf{p}_i^C$ has an associated direction vector, $\mathbf{q}_i$ and moment vectors corresponding to the centres of the cameras they are observed in, $\mathbf{q}_i' = \mathbf{q}_i \times \mathbf{o}_A$ for $i = 1, 2$ and $\mathbf{q}_j' = \mathbf{q}_j \times \mathbf{o}_B$ for $j = 3, 4$.

Solving for the relative motion of a GCM is more complicated than the estimation with individual perspective cameras. Stewenius *et al.* demonstrate that the

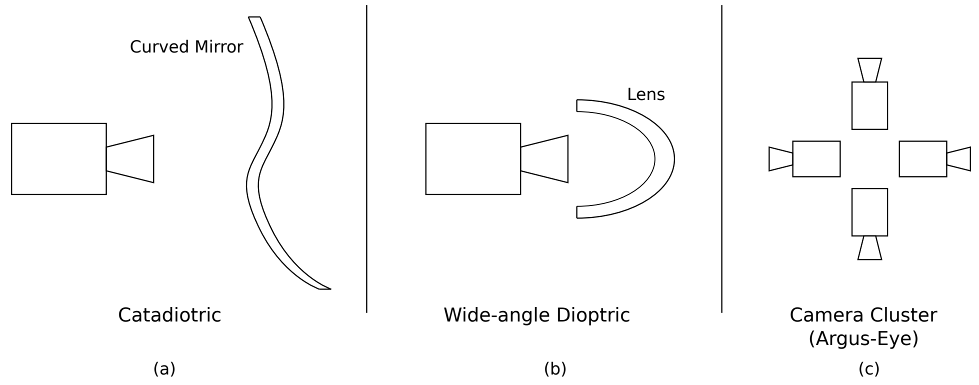


Figure 2.6: The GCM is able to model complex imaging systems, including (a) catadioptric, (b) wide-angle dioptric, and (c) camera clusters [28].

general camera framework can be solved minimally over two frames with six feature correspondences, leading to 64 solutions for the rigid motion [76]. Alternatively, there are 1320 solutions for three frames and four feature correspondences. As will be seen in this literature review, this complexity has lead some researchers to divide the problem of estimating camera cluster relative position into a decoupled estimation of motion parameters of the individual component cameras, then recombining these local motions to determine the global cluster motion.

Epipolar Geometry for GCM

Pless derives the GCM-equivalent to the single camera epipolar constraint for two general cameras separated by a rigid motion $(\boldsymbol{\mathcal{R}}, \mathbf{t})$, called the Generalized Epipolar Constraint (GEC) [63],

$$(\mathbf{q}_1)^\top \boldsymbol{\mathcal{R}} \mathbf{q}_2' + (\mathbf{q}_1)^\top [\mathbf{t}]_\times \boldsymbol{\mathcal{R}} \mathbf{q}_2 + (\mathbf{q}_1')^\top \boldsymbol{\mathcal{R}} \mathbf{q}_2 = 0, \tag{2.34}$$

where $(\mathbf{q}_1, \mathbf{q}_1')$ and $(\mathbf{q}_2, \mathbf{q}_2')$ are the raxels of the observed point feature in the first and second generalized camera, respectively. Note that the second term on the left hand side of (2.34) is equivalent to that of the single camera epipolar constraint (2.22). The expression can be written in matrix form as,

$$\begin{bmatrix} \mathbf{q}_1 \\ \mathbf{q}_1' \end{bmatrix}^\top \mathbf{E}_g \begin{bmatrix} \mathbf{q}_2 \\ \mathbf{q}_2' \end{bmatrix} = 0 \tag{2.35}$$

where $\mathbf{E}_g$ is the Generalized Essential Matrix (GEM) with the structure,

$$\mathbf{E}_g = \begin{bmatrix} \mathbf{E} & \boldsymbol{\mathcal{R}} \\ \boldsymbol{\mathcal{R}} & \mathbf{0}_{3\times3} \end{bmatrix} \tag{2.36}$$

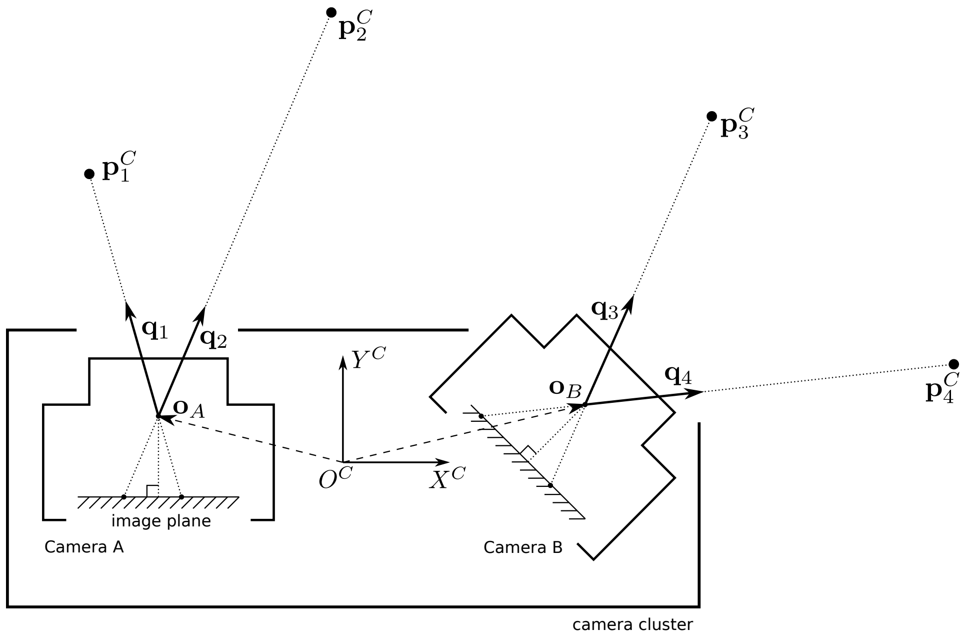


Figure 2.7: An example camera cluster GCM model for two central cameras at known locations in the cluster frame C.

with $\mathbf{E} = [\mathbf{t}]_\times \boldsymbol{\mathcal{R}}$.

Pless also proposes, without demonstration, that the GEC can be used to solve for $\mathbf{E}$ and $\boldsymbol{\mathcal{R}}$ linearly, and then to extract the motion between the cameras using 17 point correspondences in a manner similar to the linear eight-point algorithm for the single camera case. The feasibility of this simple algorithm has been studied by subsequent authors and will be addressed later in this chapter.

2.2.3 Literature Review

As noted in Section 2.1.5, the division between the techniques from the computer vision and robotics communities is becoming increasingly blurred as both areas are moving towards large-scale real-time BA methods. Therefore, the following literature review considers all of the calibrated multicamera cluster work collectively.

Known Target Model

Early methods in the literature suggest using the camera cluster along with a set of point features at known 3D positions within the environment. Similar to the single camera case, it is not surprising that scale is accurately recovered since it is embedded in the known model point feature positions. The large FOV of the cluster provides excellent localization results.

Chang and Chen propose a coupled localization method for a calibrated camera cluster modelled by GCM observing landmarks at known 3D locations [9]. Not specifically designed for non-overlapping clusters, it uses points observed at the same time in two or more cameras in a stereo-like technique to triangulate the relative pose of the camera. Also, for point features visible in only one camera, they are able to make use of relative pose algorithms for single perspective cameras observing points with known structure, to localize that particular camera and by extension, the camera cluster since the extrinsic calibration of the cluster is known and the scale is resolved by the known model. The relative pose is recalculated using the available point image measurements across all of the cameras, at each time step.

Sato *et al.* go one step further by adding the ability to estimate the locations of new, previously unknown feature points within the environment using at least six known feature points, called landmarks, as the base structure [67]. With the minimum of six landmarks, the scale is resolved and maintained for the added feature points. The landmarks are required to be visible for at least the first two frames to estimate the camera motion and initialize a set of previously unknown features. Once initialized, the solution is refined using a nonlinear optimization using gradient descent to estimate the relative pose parameters at each time step.

In [8], Carrera *et al.* propose a light-weight visual navigation algorithm which uses a non-overlapping camera cluster on a mobile robot to recognize previously visited areas in an environment for loop closure, and generate a 2D occupancy-grid map of the free-space in the world. The environment is assumed to consist of a planar floor which is visually distinct from all of the other components. Additionally, the localization is performed using odometric data from the robot and a high-level topological map is optimized when the cameras recognize a place that has been previously visited in the images.

Small FOV Overlap

The previous methods required some information about the environment model be known. As a result, the scale was automatically resolved by the imposed structure metrics. The methods that follow relax the requirement for known landmarks in the scene, but instead require small overlap in the FOV of at least some of the component cameras.

Work by Stewenius *et al.* involved mounting a calibrated camera cluster with only small overlap in the FOV, on a vehicle driving in planar motion [77]. The system made observations of the unknown feature points to build a model of the structure in 3D along with the planar pose parameters. Importantly, none of the point feature locations are known prior to the start of the estimation. Further, it is recognized that none of the camera views need to observe common points, even through the entire motion trajectory.

The method uses decoupled steps for localization and mapping, followed by BA refinement of the structure and motion combined, each using the result of the last, to proceed. As a result, initialization is a problem with this technique. Stewenius proposes to initially triangulate the 3D positions of a set of feature points assumed to be visible in more than one camera using stereo correspondence techniques. This seeds the localization step and the method can proceed. This requirement is restrictive in that at least two cameras must have some overlap in their FOV.

Stewenius further analyzes the problem and proves that for planar motion, the system can be solved minimally with either

- three cameras, each measuring one point over two frames, or
- two cameras, each measuring one point over three frames,

each case resulting in one or three real non-trivial solutions [75]. Additionally, they identify the important degenerate case when the motion of the camera cluster is a simple translation with no change in orientation. Under this motion, the scale cannot be recovered. More examples of critical motion will follow later in this section.

Kaess and Dellaert [36] mount eight perspective cameras to a robot in a circle pointing outwards with small FOV overlap in the adjacent cameras. Using a kinematic model, odometry measurements, and image measurements from all of the cameras, the system solves the SLAM problem using a nonlinear BA optimization on the pose parameters and model structure. In between the keyframes selected for the optimization, the system uses wheel odometry measurements to initialize each run of the optimizer and, as a result, there is an implicit assumption that the target environment is static. The system is further developed in [37] which applies a smoothing step to the generated map at each interval and proposes a probabilistic method for finding feature correspondences.

Harmat *et al.* rigidly-mount several wide-FOV cameras on an aerial vehicle in [30], and run a modified version of PTAM, called Multi-Camera PTAM (MCPTAM), which is able to track the pose of the cluster in real-time while simultaneously computing the BA solution in parallel. The system shows excellent tracking performance, but requires two of the component cameras to have FOV overlap during the initialization phase. Harmat *et al.* describe the additional robustness advantage of using clusters due to having cameras looking in certain directions being able to successfully track against stable point features while some of the component cameras are unable to see any good point features to track in the target environment. The SLAM framework described in this thesis is implemented as a modification to the MCPTAM algorithm to allow it to operate successfully with no FOV overlap in any of the component cameras.

Spherical Camera Approximation

The methods in this section do not require overlap in the FOV of the cluster cameras, but model the camera cluster as a single camera sensor with a common optical centre. By imposing a zero baseline, the global scale is no longer recoverable from the image measurements and modelling error is introduced, but the cluster motion can be solved for using the well-established single-camera techniques.

Kim *et al.* note that recovering the six pose parameters for a non-overlapping FOV multicamera cluster, such as the Argus Eye, is difficult when observing a scene where feature point depth is large relative to the distance between camera centres [43]. This is particularly true for the linear algebraic methods like the seventeen point algorithm when image noise is present [63]. In these cases, Kim *et al.* propose to approximate the camera cluster as a single omnidirectional camera with a common optical centre. The large FOV of this new spherical camera still eliminates the ambiguities between translation and rotation, but the scale can no longer be recovered since it is now effectively a single camera system. However, because the system is approximated as a single camera, the motion parameters can be estimated using monocular techniques.

Kim *et al.* suggest that the approximation is justified, compared to the seventeen point algorithm results, when the ratio of feature depth d, over distance between camera centres, T,

$$\frac{d}{T} > 15 \tag{2.37}$$

for cameras with 90 degree FOV with 300 pixels across the image. Notice that these are quite low resolution cameras. With any larger resolution, the ratio where this approximation is valid would only increase, requiring the points to be even further away. For executing precision control tasks, such as the ones that require close manoeuvring, this approximation may lead to poor performance.

Several other researchers have also considered approximating a non-overlapping camera cluster as a single spherical camera with a common optical centre. Tong *et al.* derive a way of approximately mapping the viewing rays from the cluster to a spherical camera model [83]. Unfortunately, the mapping is dependent on the depth of the points in the 3D scene and deviations from the assumed distance will lead to estimation errors using this model. Both Oskiper *et al.* [62] and Hui *et al.* [35] present VO algorithms with a spherical camera approximation. However, due to the single optical centre, the methods are unable to estimate the global scale of the motion and structure.

Meilland *et al.* arrange a set of cameras in a hexagonal ring facing outwards on top of a car, with adjacent cameras overlapping in half of their respective FOV [56]. The cluster is assumed to be a single spherical camera, but the depth to each point on the sphere is triangulated using stereo-correspondences on the set of pair-wise cameras. This results in a dense spherical point cloud which is then used to generate, and localize with respect to, a point-based map of the operating environment.

Non-overlapping FOV: Visual Odometry

Moving past the restrictions of planar motion, small FOV overlap, and common optical centre, the methods in this section propose to solve for the motion of the calibrated camera clusters using VO techniques, and are therefore vulnerable to drift in the solution.

A decoupled method by Gupta *et al.* calculates the optical flow for each new image in the component cameras individually to generate a local motion increment [29]. However, upon combining them to determine the global cluster motion, it is unable to determine the global scale metric despite the non-zero baselines between the cameras.

In [41], Kim *et al.* propose a decoupled strategy using the fact that the relative rotation of each component camera extracted from their individual essential matrices (using single camera techniques) are the same, a property originally observed by Baker *et al.* [5]. The resulting rotation matrices are averaged together to determine the rotation for the entire camera cluster. Each camera also has an estimate of its own translation direction. Using all of the estimates for the cameras leads to a triangulation problem to determine the scale of the cluster translation. Kim *et al.* propose to solve it using Second-Order Cone Programming (SOCP).

Kim *et al.* have refined their previous decoupled methods which found the global cluster rotation as an average of all of the rotations from individual cameras, by instead performing a branch-and-bound search over the space of all cluster rotations [42]. They show that the problem can be cast as a linear programming problem for finding the rotation, then determining the translation from this estimate. While the results are better than the previous local averaging method, it is expensive computationally and can take on the order of a few seconds to complete for each pair of frames.

An alternate solution, subsequently proposed by Clipp *et al.* works in a similar decoupled manner, extracting the rotation of the camera cluster using individual motion parameters from each camera using the five-point algorithm to estimate the essential matrix [13]. Then, writing out the constraints, only one point correspondence in any other camera is sufficient to solve for the system scale. As a result, only six feature correspondences are required to determine the full cluster motion. For robustness, more points can be added to accommodate for image noise and improve the recovered solution. Once the scale is recovered, the translation of the cluster can be recovered using the estimated translations of the individual cameras and the known transformations between them from calibration. A nearly identical algorithm was later proposed by So *et al.* [72] for a hopping rover.

In [38], Kazik *et al.* also use a decoupled VO strategy where the individual component cameras estimate their local motion up-to-scale, then combine the results to determine the true cluster translation scale.

A technique from Li *et al.* demonstrates how to use the GEC to linearly solve for the non-overlapping camera cluster motion using a coupled VO strategy [50], resulting in an algorithm similar to the eight-point algorithm for a single camera. With a more restrictive method, Lee *et al.* present a coupled GCM VO scheme which integrates the motion constraints of a car with an Ackerman motion model to simplify the estimation of the GEM and determine the cluster motion [48]. It is noted that the common scenario case of straight-forward driving with no rotation is a degenerate motion. The authors suggest to use an inter-camera feature correspondence to resolve the scale, implying that there is overlap in the component camera FOVs.

Hu *et al.* propose using pairs of laterally-placed outward facing cameras to solve the the cluster ego-motion using a Quasi-Parallax constraint [34]. Similar to the eye configuration for prey animals, they exploit this structure to create a coupled linear motion estimation method exploiting the optical flow vectors at diametrically-opposed ray directions in the two cameras. They are able to recover the global translation, then rotation in a two-step process.

Non-overlapping FOV: Recursive Filters

In this section, the presented methods operate without the requirement for FOV overlap and solve for the localization of the camera cluster relative to a target object using a recursive filter.

Both Sola *et al.* [74] and Kim *et al.* [40]. present coupled recursive SLAM systems which use a stereo camera setup as two monocular cameras with FOV overlap. The methods both use an EKF to estimate the structure of the target model, the relative motion, and in the case of Sola *et al.*, even the orientation parameters of the cluster extrinsic calibration. The estimators both make use of IDP for the point feature positions. Additionally, the systems both employ the overlap in the FOV of the cameras and explicitly match point features across the cameras in order to avoid the degenerate motions for the two-camera cluster. While this is good practice to take advantage of overlap when possible to constrain the localization, it is not necessary for the estimation to succeed and limiting the collective FOV negatively affects the pose accuracy since the estimation suffers from the same translation-rotation ambiguities as the single camera case.

Ragab and Wong [64] mount two back-to-back camera pairs on a robot. The pose of each camera is tracked individually, using two EKFs per camera. One EKF tracks the relative pose, while the other tracks the model structure, similar to the work of Deng *et al.* [21]. An optimization and correction phase is run using all of the estimates from the eight EKFs to determine the cluster's relative motion and resolve the global scale, resulting in a decoupled strategy.

In the related field of Visual Servoing, Comport *et al.* propose an Image-Based Visual Servoing (IBVS) control law for a non-overlapping stereo pair [14]. The system regulates the point feature image coordinates to a desired reference image given an approximate prior depth of each feature. Consequently, a relative pose of the cameras and a target can be maintained without the need for it to be explicitly calculated. Unfortunately, the desired reference image must either be captured, a priori, or calculated based on an accurate model of the target. This makes the IBVS methods inappropriate for exploration into novel views of the target environment. Additionally, the Cartesian-space trajectory of the camera motion cannot be controlled directly and operating on the image-space error may result in inefficient 3D motions of the cameras.

Non-overlapping FOV: Bundle Adjustment

This section presents a set of coupled localization algorithms for non-overlapping FOV camera clusters with non-zero baselines using BA optimization.

Some authors have demonstrated that the GCM BA optimization can be efficiently factorized when the cost function is changed to use the object-space error for point features, rather than the reprojection error in the image plane. Schweighofer *et al.* show how optimizing a BA formulation for a GCM is a linear operation for the keyframe translations and point feature positions for a given set of keyframe rotations [69, 70]. Therefore, the problem becomes one of iteratively determining the optimal keyframe rotations and the translation and feature positions then follow directly. The algorithm is shown to converge over a larger set of initial conditions when compared to regular BA, since there are fewer parameters to optimize and thus is less vulnerable to local minima, and it has a theoretical speed-up of a factor of eight. However, using the object-space error cost function results in a less optimal solution than when using image-space error.

In [16], Dai *et al.* propose a factorization of the GCM BA problem, similar to the idea of Schweighofer *et al.*, in which the parameters for the keyframe translations and point feature positions can be determined directly once the keyframe rotations are found. However, the cost function in this approach is again modified to use the error in object-space – how closely rays intersect in 3D space. As before, the rotations must be determined through iterative optimization and simulation results are presented showing good performance. The disadvantage of these object-space methods, as described previously, is that they are more sensitive to image noise and will produce less accurate results than the methods acting on reprojection error directly.

Valkenburg and Alwesh solve a problem, analogous to BA, localizing the positions of artificial point features within a room environment using sets of images from a mobile camera cluster taken at unknown locations [85]. The proposed algorithm uses conformal geometric algebra to formulate a closed-form initial estimate and then solves for the unknown cluster poses accurately using a nonlinear minimization of an object-space error cost function. Once the poses are determined, the locations of the points in the room are reconstructed through triangulation.

In a similar approach to PTAM and MCPTAM, Mouragnon *et al.* propose a coupled local BA method for a GCM based on the concept of keyframes [58]. Unlike the previous methods, the cost function is defined with image-space error. In order to initialize the algorithm, a linear solution to the GEC, from [63], is used to find an initial estimate which is then refined by the subsequent BA. However, solving the GEC using the seventeen point algorithm is not possible for non-overlapping multicamera clusters as this configuration leads to degeneracy of the solution [44]. As a result, this method cannot be directly applied to these non-overlapping FOV camera systems.

Summary

Along with the work of Harmat *et al.* [30], the method of Mouragnon *et al.* is the closest to that proposed in this thesis. However, all of the previous algorithms presented in this chapter fail to satisfy all of the selected criteria outlined in Section 2.2.1. The system presented in Chapter 3 addresses the specific problem posed by calibrated non-overlapping FOV multicamera clusters, and proposes several specializations to the BA scheme to address the difficulties related to parameter selection and system initialization, experienced by a camera configuration of this type. In doing so, the novel algorithm meets the criteria set forth in Section 2.2.1.

Chapter 3

Multicamera Cluster SLAM

In this chapter, a Simultaneous Localization and Mapping framework using a calibrated non-overlapping FOV multicamera cluster is presented. The algorithm is able to track the relative motion of the camera cluster and (moving) target object or environment through an image sequence, as well as construct a model representing the positions of point features on the target. Accordingly, the system considers the set of one or more rigidly-connected perspective cameras as a single vision sensor and forms the nonlinear optimization problem using state space manifolds. Subsequently, a novel parameterization for updating point features based on spherical coordinates is presented to allow the system to accurately track the motion and structure despite large global scale error in the current estimate caused by degenerate relative motions. Finally, an initialization procedure is proposed to generate an initial system estimate and allow the optimization to successfully converge to an accurate solution starting from the first set of camera images at the initial time step.

3.1 System Models

This section presents the models for the camera cluster image measurements and the target object structure.

3.1.1 Calibrated Multicamera Cluster

Collectively, the calibrated camera cluster is modelled as a set of n_c pin-hole cameras with known relative coordinate transformations between each camera coordinate frame. Accordingly, a point $\tilde{\mathbf{p}}^{C_h}$ in the camera frame C_h, can be transformed into any other camera frame C_i by,

$$\tilde{\mathbf{p}}^{C_i} = \mathbf{T}_{C_h}^{C_i} \tilde{\mathbf{p}}^{C_h} \tag{3.1}$$

where $\mathbf{T}_{C_h}^{C_i} \in SE(3)$, $\forall i, h = 1, 2, \ldots, n_c$. Without loss of generality, the coordinate frame for the camera cluster is chosen to coincide with the first camera frame, C_1. The transformation from camera h to the cluster frame can be written in shortened form as $\mathbf{T}_{C_h} \equiv \mathbf{T}_{C_h}^{C_1}$, where the cluster frame C_1 is implied when the superscript is neglected. The transformation is shown in Figure 3.1.

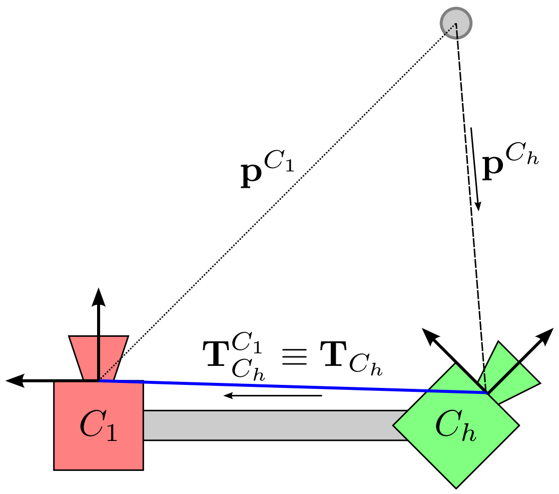


Figure 3.1: The relative position and orientation of each camera is known relative to the cluster frame, C_1 and therefore, the position of points in any camera frame can be found with respect to the cluster frame using the known transformation, $\mathbf{T}_{C_h}$.

3.1.2 Target Object Model

The target model consists of a set of point features organized into n_k keyframes, each a six degree of freedom (DOF) pose estimate with respect to the target model reference frame M, along with the n_c images from the cluster cameras captured at that location, as in [45] for a single camera. Each keyframe contains a set of point features that are said to be anchored within the respective camera coordinate frame at that keyframe. The coordinate frame of camera h at keyframe k is denoted C_hK_k.

The position of a point feature is expressed with respect to the camera coordinate frame at its anchor keyframe – the first keyframe in which it is observed. It is assumed in this work that the point features are a finite distance from the camera cluster at all time steps. This assumption excludes tracking point features such as stars or distant points on the horizon, which are effectively at infinite depth from a practical viewpoint.

Since the relative position and orientation of each component camera within the cluster is fixed at all times, the k^{th} keyframe pose is parameterized by the single homogeneous transformation for the cluster coordinate frame at the keyframe, C_1K_k, with respect to the target model reference frame, M, resulting in $\mathbf{T}^M_{C_1K_k} \in SE(3)$. The C_1 and M frames are implied in this keyframe pose definition, and therefore, the transformation will be written simply as $\mathbf{T}_{K_k} \equiv \mathbf{T}^M_{C_1K_k}$. The pose of camera h at keyframe k is easily found as,

$$\mathbf{T}^M_{C_hK_k} = \mathbf{T}_{K_k}\mathbf{T}_{C_h}. \tag{3.2}$$

An example system with a camera cluster composed of $n_c = 2$ cameras and $n_k = 2$ keyframes is shown in Figure 3.2. The cameras in this example are arranged back-to-back with the optical axes looking outwards along the green axes of the associated coordinate frames. The j^{th} point feature is anchored in the second camera at the second keyframe, C_2K_2, and its position with respect to this coordinate frame is represented as $\mathbf{p}_j^{C_2K_2}$.

Since the target model is initially unknown, the set of keyframes is accumulated as the estimation proceeds and new observations of the target object are made. At the beginning, the model consists of only one keyframe from the initial observation. As the algorithm continues through the sequence, it will determine when to add a new keyframe at the current cluster pose. This process grows the set of keyframes to cover the entire target object.

The parameters representing the set of keyframe poses together with the positions of the point features observed within them, compose the target model, as well as the full system state, $\mathbf{x} \in \mathcal{S}$, where $\mathcal{S}$ is the state space, which is defined in 3.2.2. These parameters are estimated using the point feature image measurements within the cluster cameras. The next section derives this relationship.

3.1.3 Camera Cluster Measurement Model

The position of a point in the coordinate frame of camera h and keyframe k, (C_hK_k), is denoted $\tilde{\mathbf{p}}^{h,k} \in \mathbb{P}^3$. Additionally, the transformation from the coordinate frame

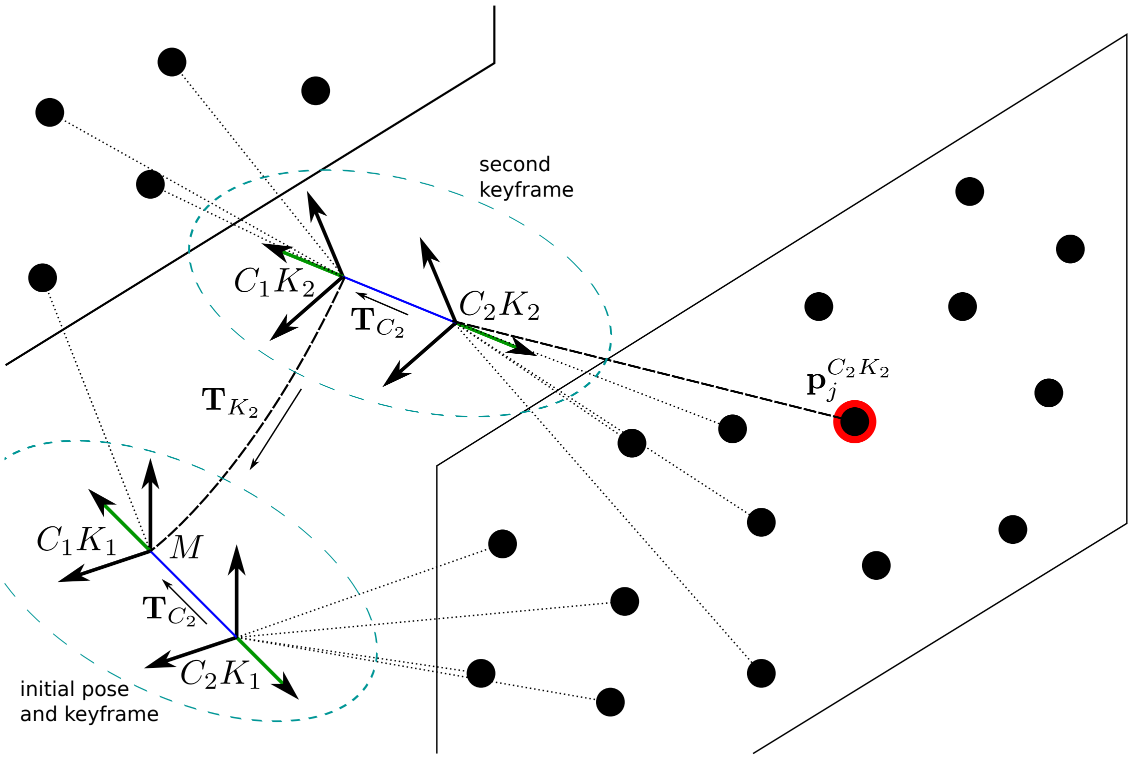


Figure 3.2: An example target object model with two keyframes for a two-camera back-to-back cluster. The cameras look outwards with the green arrows showing the optical axes. The point feature j is anchored, and therefore, positioned within the C_2K_2 coordinate frame. The relative pose of camera 2, $\mathbf{T}_{C_2}$, is known from calibration, but the relative pose of keyframe 2, $\mathbf{T}_{K_2}$, as well as the position of the point features must be estimated.

associated with camera h at keyframe k, (C_hK_k), to the coordinate frame of camera i at keyframe ℓ, (C_iK_ℓ), will be written,

$$\mathbf{T}_{h,k}^{i,\ell} = \begin{bmatrix} \boldsymbol{\mathcal{R}}_{h,k}^{i,\ell} & \mathbf{t}_{h,k}^{i,\ell} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}. \tag{3.3}$$

Therefore, the position of the point parameterized in C_hK_k, from the perspective of keyframe C_iK_ℓ is written,

$$\tilde{\mathbf{p}}^{i,\ell} = \mathbf{T}_{h,k}^{i,\ell}\tilde{\mathbf{p}}^{h,k}. \tag{3.4}$$

The measurement model, relating the observed feature point locations in the camera image planes, to the system states, can be written as a series of coordinate transformations. Suppose that the j^{th} point feature, anchored in the coordinate frame C_hK_k, is measured by camera i at C_iK_ℓ. An example of this chain of transformations is shown for the simple back-to-back two-camera cluster system with three keyframes in Figure 3.3. In this particular case, the point feature j is anchored in C_2K_2 and observed in C_2K_3.

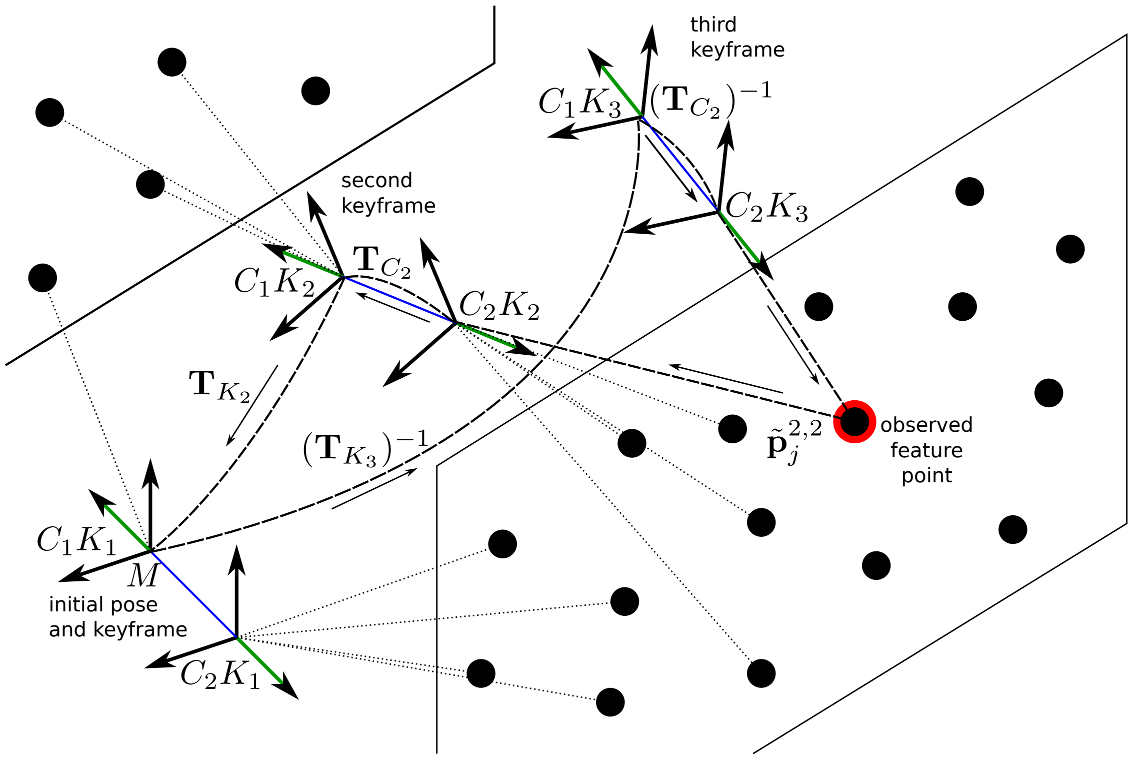


Figure 3.3: The coordinate transformations, for a two-camera back-to-back cluster, involved in finding the predicted image plane measurements of a point $\mathbf{p}^{2,2}$ anchored in C_2K_2 and observed in C_2K_3.

The feature point position parameters give the location of the j^{th} feature in its anchor keyframe and camera frame C_hK_k, resulting in $\mathbf{p}_j^{h,k}$. This point feature is first transformed into the target model coordinate frame by,

$$\tilde{\mathbf{p}}_j^M = \mathbf{T}_{h,k}^M \tilde{\mathbf{p}}_j^{h,k} \tag{3.5}$$

$$= \mathbf{T}_{K_k} \mathbf{T}_{C_h} \tilde{\mathbf{p}}_j^{h,k}, \tag{3.6}$$

which are transformations provided by the known cluster calibration and the estimated keyframe pose.

Next, the point is transformed into the coordinate frame of the observing keyframe and camera C_iK_ℓ using the observing keyframe pose states and the cluster calibration,

$$\begin{aligned}
\tilde{\mathbf{p}}_j^{i,\ell} &= \mathbf{T}_M^{i,\ell}\tilde{\mathbf{p}}_j^M && (3.7)\\
&= \left(\mathbf{T}_{i,\ell}^M\right)^{-1}\tilde{\mathbf{p}}_j^M && (3.8)\\
&= (\mathbf{T}_{C_i})^{-1}(\mathbf{T}_{K_\ell})^{-1}\tilde{\mathbf{p}}_j^M && (3.9)\\
&= (\mathbf{T}_{C_i})^{-1}(\mathbf{T}_{K_\ell})^{-1}\mathbf{T}_{K_k}\mathbf{T}_{C_h}\tilde{\mathbf{p}}_j^{h,k} && (3.10)
\end{aligned}$$

Finally, the point is projected into $\mathbb{P}^2$ and onto the image plane of camera C_i using the corresponding projection matrix, $\mathbf{K}_i$,

$$\begin{aligned}
\tilde{\mathbf{p}}_j^{D_i} &= \mathbf{K}_i\tilde{\mathbf{p}}_j^{i,\ell} && (3.11)\\
&= \begin{bmatrix} x_j^{D_i} \\ y_j^{D_i} \\ w_j^{D_i} \end{bmatrix}, && (3.12)
\end{aligned}$$

known from the intrinsic calibration of the individual cluster cameras.

Each of the four intermediate homogeneous transformation matrices in (3.10) are formed by either the system states, or the known cluster camera configurations from extrinsic calibration. Therefore, the measurement equation for feature j as seen in the observing keyframe and camera is,

$$\mathbf{z}_j^{i,\ell} = \mathbf{g}_j^{i,\ell}(\mathbf{x}) + \boldsymbol{\gamma}_j^{i,\ell} \tag{3.13}$$

where

$$\mathbf{g}_j^{i,\ell}(\mathbf{x}) = \boldsymbol{\pi}_2\left(\mathbf{K}_i(\mathbf{T}_{C_i})^{-1}(\mathbf{T}_{K_\ell})^{-1}\mathbf{T}_{K_k}\mathbf{T}_{C_h}\tilde{\mathbf{p}}_j^{h,k}\right) \tag{3.14}$$

and $\boldsymbol{\gamma}_j^{i,\ell} \sim \mathcal{N}\left(0, \mathbf{R}_j^{i,\ell}\right)$. In this system, the camera image noise is modelled as a zero-mean Gaussian random variable. It has been shown that this noise model is a good approximation of the actual image measurement noise from the feature extraction process [54].

The full system measurement vector $\mathbf{z} \in \mathcal{M}$ is made up of all of the individual point feature observations at all of the keyframes. It is modelled as a stacked column vector of measurements of the form (3.14). The measurement function,

$$\mathbf{g} : \mathcal{S} \to \mathcal{M}, \tag{3.15}$$

maps the current state of the system, $\mathbf{x} \in \mathcal{S}$, to the deterministic portion of the system measurement space. The complete system measurement model is,

$$\mathbf{z} = \mathbf{g}(\mathbf{x}) + \boldsymbol{\gamma}, \tag{3.16}$$

where $\boldsymbol{\gamma}$ is the measurement noise vector formed by stacking all of the individual noise vectors for feature observations into a column vector.

3.1.4 System Structure Graph

The calibrated multicamera cluster SLAM problem is formulated as a set of keyframes and point features connected by relative transformations and constrained by a set of image measurements of the point features at a subset of the system keyframes. Accordingly, it can be illustrated as a directed graph in which the nodes represent the position and orientation of body-fixed reference frames, and the edges represent coordinate and projection transformations corresponding to physical components, similar to that of mechanical systems in graph representation [55].

Each point feature in the target model is represented as a node P_j, each keyframe as a node K_i, and within each keyframe, the cluster cameras at that keyframe, are also nodes C_h. Each camera has its image plane represented as the nodes labelled D_h. The nodes are connected by edges representing the relative positions, transformations, and measurements in the system. Each keyframe node K_i is connected to the target model reference frame by the estimated keyframe pose transformation, $\mathbf{T}_{K_i}$. The camera nodes, C_h, at a specific keyframe, are connected to the keyframe node with the known relative pose from extrinsic calibration, $\mathbf{T}_{C_h}$. Each camera node is also connected to the node representing the image-plane, D_h, by the respective projection matrix $\mathbf{K}_h$. The point feature node, P_j, is connected to the camera node in which it is anchored by an edge representing its position in that coordinate frame, $\tilde{\mathbf{p}}_j$, and also connected to the image-plane of all of the cameras which observe that feature point through the motion sequence by the image measurement coordinates, $\mathbf{z}_j$.

A structural graph of an example three-camera cluster system (C_1, C_2, C_3), containing three keyframes (K_1, K_2, K_3), and three point features (P_1, P_2, P_3), is shown in Figure 3.4. The shaded nodes represent those with relative positions and orientations which are fixed. In this system, the point features are organized as follows:

- Point feature P_1 is anchored in C_1K_1, and measured in the image plane at C_1K_1 and C_1K_2.
- Point feature P_2 is anchored in C_2K_1, and measured in C_2K_1, C_3K_2, and C_1K_3.
- Point feature P_3 is anchored in C_3K_3, and measured in C_3K_3 and C_2K_2.

The graph contains a circuit for each measurement edge that includes no other measurement edges. The measurement equations for each feature point observation

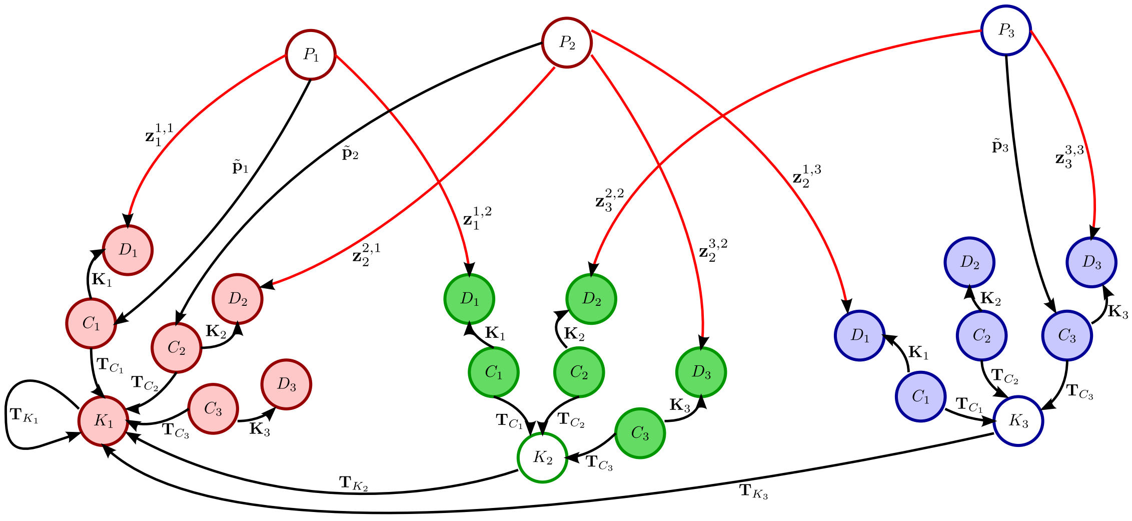


Figure 3.4: An example structural graph for a three-camera cluster with three keyframes, three point features, and seven observations. The nodes are connected by the black edges representing the relative positions and transformations, as well as the red edges representing measurements of the point features in the camera image planes at the various keyframes.

are found by traversing this loop for the measurement edge from tail to tip. The direction of each edge in the loop indicates if the respective edge transformation is applied directly (the circuit is traversed in the direction of the edge), or inverted first (against the direction of the edge). Note that in some cases, the point feature will be anchored and observed in the same camera, and/or the same keyframe, which will cause some of the transformations to resolve to identity. Suppose that the feature point j, anchored in C_hK_k is observed in C_iK_ℓ, the constraint equation is that of (3.14).

In the case of the example system in Figure 3.4, the constraint equations are,

$$\begin{bmatrix} \mathbf{z}_1^{1,1} \\ \mathbf{z}_1^{1,2} \\ \mathbf{z}_2^{2,1} \\ \mathbf{z}_2^{3,2} \\ \mathbf{z}_2^{1,3} \\ \mathbf{z}_3^{2,2} \\ \mathbf{z}_3^{3,3} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\pi}_2\left(\mathbf{K}_1\tilde{\mathbf{p}}_1^{1,1}\right) \\ \boldsymbol{\pi}_2\left(\mathbf{K}_1(\mathbf{T}_{K_2})^{-1}\tilde{\mathbf{p}}_1^{1,1}\right) \\ \boldsymbol{\pi}_2\left(\mathbf{K}_2\tilde{\mathbf{p}}_2^{2,1}\right) \\ \boldsymbol{\pi}_2\left(\mathbf{K}_3(\mathbf{T}_{C_3})^{-1}(\mathbf{T}_{K_2})^{-1}\mathbf{T}_{C_2}\tilde{\mathbf{p}}_2^{2,1}\right) \\ \boldsymbol{\pi}_2\left(\mathbf{K}_1(\mathbf{T}_{K_3})^{-1}\mathbf{T}_{C_2}\tilde{\mathbf{p}}_2^{2,1}\right) \\ \boldsymbol{\pi}_2\left(\mathbf{K}_2(\mathbf{T}_{C_2})^{-1}(\mathbf{T}_{K_2})^{-1}\mathbf{T}_{K_3}\mathbf{T}_{C_3}\tilde{\mathbf{p}}_3^{3,3}\right) \\ \boldsymbol{\pi}_2\left(\mathbf{K}_3\tilde{\mathbf{p}}_3^{3,3}\right) \end{bmatrix} + \begin{bmatrix} \boldsymbol{\gamma}_1^{1,1} \\ \boldsymbol{\gamma}_1^{1,2} \\ \boldsymbol{\gamma}_2^{2,1} \\ \boldsymbol{\gamma}_2^{3,2} \\ \boldsymbol{\gamma}_2^{1,3} \\ \boldsymbol{\gamma}_3^{2,2} \\ \boldsymbol{\gamma}_3^{3,3} \end{bmatrix} \tag{3.17}$$

Representing the system in this graphical manner produces a structured and systematic way of generating the measurement equations given the set of states and point feature observations. The set of measurements are used in the next section to optimize the estimated keyframe poses and point feature positions in the state space.

3.2 Optimization

Like many problems in engineering, the feature-based SLAM problem using cameras is, at its core, simply a large nonlinear optimization. The parameter vector contains information on the 3D poses of the cameras at the time of capturing images, as well as 3D positions of the observed point features measured within those images. A measurement function maps the tracked parameters into the measurement space where the predicted image locations are compared against where the points are actually observed in the images. The goal of the optimization, then, is to find the set of parameters such that the predicted feature measurement coordinates best match those extracted from the collected images.

It is possible to solve a system of moderate size simply by naively applying a nonlinear least-squares solver to the system with a reasonable initial condition. However, recent efforts have focused on identifying intelligent approximations to the full optimization that tradeoff complexity for solution accuracy. New parameterizations (e.g. [12, 73]), noise process models (e.g. [82]), or sparseness exploits (e.g. [86, 46]) narrow the distance between the speed and accuracy extremes. Ultimately, the proper choice of the algorithm to use depends critically on the application.

This section begins with an overview of optimization using the $\boxplus$-manifold [33], which encapsulates the global topology of the multicamera cluster system state consisting of the target model keyframe poses and point feature positions. The novel application of this concept to the particular case of non-overlapping FOV multicamera clusters is claimed. Furthermore, a new parameterization and state update method for the point feature positions is proposed.

3.2.1 State Representation Using $\boxplus$-Manifolds

The state space for the BA system is composed of a set of keyframe poses and point feature positions. The parameters in the target model state are assumed to be constant and therefore, do not have any associated dynamic model, besides the stationary process. There are many ways to parameterize these states as real-valued vectors and estimation methods have been proposed using a wide variety of parameterizations, which for rotation include the minimal representations such as, Euler angles [89], unit quaternions [17], or modified Rodrigues parameters [15]. However, all of the minimal representations for the $SO(3)$ group as a flat vector in $\mathbb{R}^3$ or $\mathbb{R}^4$ have singularities or extra constraints on the parameters which present unnecessary challenges to the optimization process. For example, Euler angles suffer from the gimbal lock singularity when the second angle goes to $\pm\frac{\pi}{2}$ rad, and unit quaternions must maintain the unit-length constraint. The least-squares optimization methods cannot enforce this constraint without requiring the addition of extra equations or an explicit normalization step [33].

In this work, the state space $\mathcal{S}$ is represented as a $\boxplus$-manifold, as proposed by Hertzberg *et al.* [33]. The formal definition and properties of these objects are provided in Appendix A. These manifolds act as a real vector space locally, but can encode a more complex global topology, such as that of the space of 3D orientations in the group $SO(3)$, as well as 3D rigid-body motions in $SE(3)$.

The classical least-squares optimization methods seek to update an estimate of the state vector, $\mathbf{x} \in \mathbb{R}^n$, at each iteration with,

$$\mathbf{x} \mapsto \mathbf{x} + \boldsymbol{\delta} \in \mathbb{R}^n, \tag{3.18}$$

using the calculated update vector, $\boldsymbol{\delta} \in \mathbb{R}^n$, which is a function of the measurement error vector,

$$\bar{\mathbf{z}} = \mathbf{z} - \mathbf{g}(\mathbf{x}) \in \mathbb{R}^m, \tag{3.19}$$

where $\mathbf{z}$ is the system measurement and $\mathbf{g}(\mathbf{x})$ is the system measurement model.

The general idea behind the $\boxplus$-manifold is to change the iterative update and error calculations to replace the $+$ and $-$ operators with the more general operators $\boxplus$ and $\boxminus$, which respect the underlying topology of the state space, but interface with optimization algorithms using the real vector space [33]. The operator $\boxplus$ is used to apply updates to the state manifold, $\mathbf{x} \in \mathcal{S}$, given a perturbation vector using a selected minimal representation for the update vector, $\boldsymbol{\delta} \in \mathbb{R}^n$,

$$\mathbf{x} \mapsto \mathbf{x} \boxplus \boldsymbol{\delta} \in \mathcal{S}, \tag{3.20}$$

producing an updated $\mathbf{x} \in \mathcal{S}$ that maintains the global topology of the manifold $\mathcal{S}$. This allows the state $\mathbf{x}$ to be a (possibly over-parameterized) representation free of singularities, manipulated by small-magnitude perturbations $\boldsymbol{\delta} \in \mathbb{R}^n$. Since the iterative optimization methods apply small refinements to the state, the update vector $\boldsymbol{\delta}$ is a minimal representation kept sufficiently far from the respective singularities.

Furthermore, the $\boxminus$ operator generalizes the idea of comparisons on the state manifold, producing a minimal representation vector difference between two elements in $\mathcal{S}$. Accordingly, the measurement error, calculated for two elements in the measurement manifold, $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{M}$ is found,

$$\bar{\mathbf{z}} = \mathbf{z}_1 \boxminus \mathbf{z}_2 \in \mathbb{R}^m. \tag{3.21}$$

By using these manifolds, the inner-workings of the state manifold topologies are isolated from the least-squares methods, which are able to treat them as a black-box by simply using the $\boxplus$ and $\boxminus$ operators for the respective manifolds, in place of the $+$ and $-$ operators. The adaptation of the Levenberg-Marquardt optimization method is now straight-forward and is shown in Section 3.2.3.

As a result, the state and measurement spaces for non-overlapping calibrated camera cluster BA can be represented as $\boxplus$-manifolds $\mathcal{S}$ and $\mathcal{M}$, respectively. The measurement space is simply the real vector space $\mathbb{R}^m$ since the measurements of the point features are on the image planes of the cluster cameras, individually represented in $\mathbb{R}^2$. Therefore, the conventional $+$ and $-$ operators are used as normally defined. The state manifold, however, contains the keyframe poses, in $SE(3)$, and the point feature positions. The $\boxplus$-manifold representation is especially useful for the keyframe relative orientation states to avoid the singularities in the minimal representations of a rotation matrix.

Another strength of the $\boxplus$-manifold approach is that the update equation (3.20) can be selected using prior knowledge of the problem. For the point feature states, a non-overlapping cluster camera has difficulties recovering the properly-scaled depth to the feature, particularly in the initial time steps when there is little information due to small relative motions. Therefore, even though the point position parameters are represented as the Cartesian coordinates in $\mathbb{R}^3$, the proposed update step treats the point as if it were on the surface of a sphere, centred at the camera coordinate frame, and updates the bearing along the surface separate from the update to the radial distance. This allows the bearing to converge quickly while the depth remains uncertain. This new parameter update is detailed in Section 3.2.2.

3.2.2 $\boxplus$-Manifolds for 3D Poses and Points

The cluster system state consists of keyframe poses and point feature positions to describe the target object model. To successfully track the relative motion and structure, suitable $\boxplus$-manifolds are selected in the following sections to effectively address the specific challenges of the non-overlapping cluster SLAM problem.

Keyframe Pose Manifold

The $\boxplus$-manifold $\mathcal{K}$ for the keyframe pose states is the $\boxplus$-manifold on the group $SE(3)$. The pose of each keyframe in the system is represented directly as a homogeneous transformation, there is no need to use a minimal vector representation in the real vector space, except for the update operation.

The group $SO(3)$ is a compact connected Lie group [33], and as a result, the mapping from $\mathbb{R}^3$ to $SO(3)$ can be performed using the exponential map defined on that group [49],

$$\exp_{SO(3)} : \mathbb{R}^3 \rightarrow SO(3) \tag{3.22}$$

which is found using the Rodrigues rotation formula [31], for $\boldsymbol{\omega} = \theta\hat{\boldsymbol{\omega}} \in \mathbb{R}^3$ where $\theta = \|\boldsymbol{\omega}\| \in \mathbb{R}$ is the rotation angle and $\hat{\boldsymbol{\omega}}$ is the unit-length rotation axis,

$$\exp_{SO(3)}(\boldsymbol{\omega}) = \mathbf{I}_{3\times3} + (\sin\theta)\,[\hat{\boldsymbol{\omega}}]_\times + (1-\cos\theta)\,[\hat{\boldsymbol{\omega}}]_\times^2\,. \tag{3.23}$$

This operation forms the 3×3 orthonormal rotation matrix associated with rotating a point around the axis $\hat{\boldsymbol{\omega}}$ through the angle θ.

The inverse function to map a rotation matrix representation to a 3-vector angle-axis representation is the corresponding logarithm on $SO(3)$ [3],

$$\log_{SO(3)} : SO(3) \rightarrow \mathbb{R}^3, \tag{3.24}$$

where the rotation matrix $\boldsymbol{\mathcal{R}} \in SO(3)$ goes to,

$$\log_{SO(3)}(\boldsymbol{\mathcal{R}}) = \theta\hat{\boldsymbol{\omega}}, \tag{3.25}$$

such that

$$\theta = \arccos\left(\frac{\operatorname{trace}(\boldsymbol{\mathcal{R}}) - 1}{2}\right), \tag{3.26}$$

and

$$[\boldsymbol{\omega}]_\times = \begin{cases} 0 & \text{if } \theta = 0 \\ \dfrac{\theta}{2\sin\theta}(\boldsymbol{\mathcal{R}} - \boldsymbol{\mathcal{R}}^\top) & \text{if } \theta \neq 0 \text{ and } \theta \in (-\pi, \pi). \end{cases} \tag{3.27}$$

Similarly, the exponential map for $SE(3)$,

$$\exp_{SE(3)} : \mathbb{R}^6 \rightarrow SE(3) \tag{3.28}$$

takes the vector $\mathbf{x} = [\mathbf{v}^\top \boldsymbol{\omega}^\top]^\top \in \mathbb{R}^6$ with $\boldsymbol{\omega} = \theta\hat{\boldsymbol{\omega}} \in \mathbb{R}^3$ and $\theta = \|\boldsymbol{\omega}\|$, into $SE(3)$ using the exponential on $SO(3)$ for the orientation and calculates the transformation matrix as [3],

$$\exp_{SE(3)}(\mathbf{x}) = \begin{bmatrix} \exp_{SO(3)}(\boldsymbol{\omega}) & \left(\mathbf{I}_{3\times3} + \left(\frac{1-\cos\theta}{\theta}\right)[\hat{\boldsymbol{\omega}}]_\times + \left(1 - \frac{\sin\theta}{\theta}\right)[\hat{\boldsymbol{\omega}}]_\times^2\right)\mathbf{v} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}. \tag{3.29}$$

A transformation matrix is mapped back to the real vector space $\mathbb{R}^6$ using the logarithm on the group $SE(3)$,

$$\log_{SE(3)} : SE(3) \to \mathbb{R}^6 \tag{3.30}$$

such that a homogeneous transformation,

$$\mathbf{T} = \begin{bmatrix} \boldsymbol{\mathcal{R}} & \mathbf{t} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \in SE(3), \tag{3.31}$$

becomes the vector,

$$\log_{SE(3)}(\mathbf{T}) = \begin{bmatrix} \left(\mathbf{I}_{3\times3} - \frac{\theta}{2}[\hat{\boldsymbol{\omega}}]_\times + \left(1 - \frac{\theta(1+\cos\theta)}{2\sin\theta}\right)[\hat{\boldsymbol{\omega}}]_\times^2\right)\mathbf{t} \\ \boldsymbol{\omega} \end{bmatrix}. \tag{3.32}$$

where $\boldsymbol{\omega} = \log_{SO(3)}(\boldsymbol{\mathcal{R}}) = \theta\hat{\boldsymbol{\omega}}$ and $\theta = \|\boldsymbol{\omega}\|$.

For notational convenience, the exponential map on $SE(3)$ as applied to the manifold for the keyframe poses will be given the alias,

$$\mathbf{T}_{\mathcal{K}} : \mathbb{R}^6 \to SE(3), \tag{3.33}$$

such that for $\boldsymbol{\delta}_K \in \mathbb{R}^6$,

$$\mathbf{T}_{\mathcal{K}}(\boldsymbol{\delta}_K) \equiv \exp_{SE(3)}(\boldsymbol{\delta}_K). \tag{3.34}$$

Similarly, for the exponential map on $SO(3)$, the alias,

$$\boldsymbol{\mathcal{R}}_{\mathcal{K}} : \mathbb{R}^3 \to SO(3), \tag{3.35}$$

will be used such that for $\boldsymbol{\omega} \in \mathbb{R}^3$,

$$\boldsymbol{\mathcal{R}}_{\mathcal{K}}(\boldsymbol{\omega}) \equiv \exp_{SO(3)}(\boldsymbol{\omega}). \tag{3.36}$$

The operators $\boxplus_{\mathcal{K}}$ and $\boxminus_{\mathcal{K}}$ to modify the state estimates of the keyframe poses are then defined:

$$\boxplus_{\mathcal{K}} : SE(3) \times \mathbb{R}^6 \to SE(3) \tag{3.37}$$

$$\boxminus_{\mathcal{K}} : SE(3) \times SE(3) \to \mathbb{R}^6 \tag{3.38}$$

such that for two keyframes, $\mathbf{T}_{K_1}, \mathbf{T}_{K_2} \in SE(3)$ and update vector $\boldsymbol{\delta}_K \in \mathbb{R}^6$,

$$\mathbf{T}_{K_1} \boxplus_{\mathcal{K}} \boldsymbol{\delta}_K = \mathbf{T}_{\mathcal{K}}(\boldsymbol{\delta}_K)\mathbf{T}_{K_1} \tag{3.39}$$

$$\mathbf{T}_{K_1} \boxminus_{\mathcal{K}} \mathbf{T}_{K_2} = \log_{SE(3)}\big((\mathbf{T}_{K_2})^{-1}\mathbf{T}_{K_1}\big). \tag{3.40}$$

Note that in the optimization algorithm, from Section 3.2.3, the $\boxminus_{\mathcal{S}}$ operator is not used since the elements of the state space are never measured directly. It is only through the measurements that information is gained regarding the state. This means that the keyframe poses do not ever need to be reduced to the flattened vector in the real vector space. Instead, the system state consists of a set of 4×4 coordinate transformation matrices.

Point Position Manifold

It is possible to define the manifold for the point features as either $\mathbb{P}^3$ or $\mathbb{R}^3$ depending on the requirements of the application. The practical difference between the two spaces is that $\mathbb{P}^3$ can represent point features which are at, or near, infinite depth from the camera, while $\mathbb{R}^3$ cannot. This is the case when points are, for practical purposes, infinitely far from the camera such as tracking stars, or points on the horizon. Since the fourth coordinate of the projective vector is zero, the finite translation vectors in the transformation matrices have no effect on the position of the point feature in the camera frame. As a result, these point features can only provide information about the relative orientation of the camera and the target. When the points are not represented in the projective space, the values of the coordinates in $\mathbb{R}^3$ can be large and require the translations to be sufficiently large for an accurate position estimate in the presence of measurement noise.

In this work, the manifold for the point features will be based in the real vector space $\mathbb{R}^3$. This implicitly assumes, as was stated previously, that points are a finite distance from the camera at all time in the relative motion profile. Therefore, the $\boxplus$-manifold $\mathcal{P}$ for the point feature position states is defined as the vector space $\mathbb{R}^3$ along with the operators:

$$\boxplus_{\mathcal{P}} : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3, \tag{3.41}$$

$$\boxminus_{\mathcal{P}} : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3 \tag{3.42}$$

As with the keyframe state manifold, the point positions are not measured directly and the $\boxminus_{\mathcal{P}}$ operator is not used in the optimization algorithm, but it is included here for completeness. Since the underlying space is the real vector space, the $\boxminus_{\mathcal{P}}$ can be simply defined as the direct subtraction operator,

$$\mathbf{p}_1 \boxminus_{\mathcal{P}} \mathbf{p}_2 = \mathbf{p}_1 - \mathbf{p}_2, \quad \mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^3. \tag{3.43}$$

In defining the $\boxplus_{\mathcal{P}}$ operator, this section will present two alternatives. For a point feature, $\mathbf{p}_1 \in \mathbb{R}^3$ and an increment $\boldsymbol{\delta}_P \in \mathbb{R}^3$. The first, and most obvious option is to define it as the usual addition operator in $\mathbb{R}^3$,

$$\mathbf{p}_1 \boxplus_{\mathcal{P}} \boldsymbol{\delta}_P = \mathbf{p}_1 + \boldsymbol{\delta}_P. \tag{3.44}$$

This is the simplest operator to implement and there is no difference in this representation and the classical flat vector representation of the model point feature positions.

The second option is a novel state update based on spherical coordinates and inspired by the Inverse Depth Parameterization (IDP) from Civera *et al.* [12]. In the calibrated multicamera cluster tracking system with non-overlapping camera FOV, the image measurements are quite insensitive to the global scale of the solution, particularly when the relative motion of the cluster and the target object is (near) degenerate (refer to Chapter 4). As a result, the directions of all of the position vectors in the system (keyframe translations and point positions), as well as the keyframe orientations can be accurately determined, while the global scale may be ambiguous or inaccurate.

With the point features anchored in a keyframe, scaling the solution is simply a matter of scaling the feature point position parameters directly. Note that if the feature points are all anchored in a common world frame, this is no longer true since the known scale of the camera baselines within the cluster are embedded in the feature position and will be affected by a simple scale factor multiplication.

This new update treats the feature position as a point on a sphere centred at the anchor camera coordinate frame. The point is moved on the surface of the sphere by the angle increments, δ_α and δ_β, and then moved in the radial direction by a scaling factor increment δ_r, as shown in Figure 3.5. The angle increments move the point in the local (X', Y', Z') coordinate system along the surface of the sphere, while the scale increment changes the radial distance to the point.

Using this update parameterization isolates the parts of the point position that can be estimated precisely using a camera, the bearing to the point on the sphere,

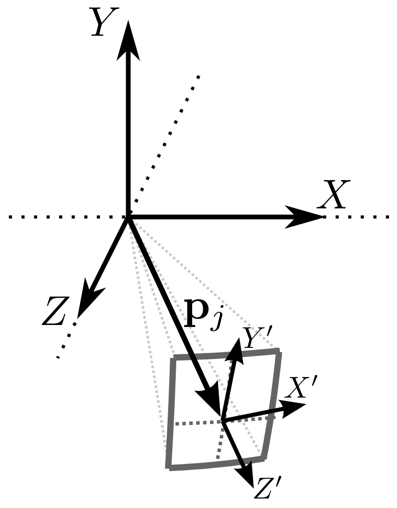


Figure 3.5: The point feature is incrementally rotated in the local (X', Y', Z') for the state update, then scaled.

from the part which is difficult to determine without sufficient motion, the radial depth to the point. This separation allows for a point feature position estimate, which is accurate in bearing but may not be in depth, to quickly converge to the proper scale when there is sufficient information from non-degenerate relative motion.

The $\boxplus_{\mathcal{P}}$ operator is thus defined, for $\mathbf{p}_1, \boldsymbol{\delta}_P \in \mathbb{R}^3$,

$$\mathbf{p}_1 \boxplus_{\mathcal{P}} \boldsymbol{\delta}_P = \boldsymbol{\pi}_3 \left(\mathbf{T}_{\mathcal{P}}(\boldsymbol{\delta}_P, \mathbf{p}_1) \tilde{\mathbf{p}}_1 \right) \tag{3.45}$$

where the operator $\mathbf{T}_{\mathcal{P}}$ forms the transformation matrix,

$$\mathbf{T}_{\mathcal{P}} : \mathbb{R}^3 \times \mathbb{R}^3 \to SE(3) \tag{3.46}$$

such that if $\boldsymbol{\delta}_P = [\delta_\alpha, \delta_\beta, \delta_r]^\top$,

$$\mathbf{T}_{\mathcal{P}}(\boldsymbol{\delta}_P, \mathbf{p}_1) = \begin{bmatrix} (1+\delta_r)\boldsymbol{\mathcal{R}}_{\mathcal{P}}(\mathbf{p}_1)\boldsymbol{\mathcal{R}}_{\mathcal{K}}(\delta_\alpha \hat{\mathbf{i}} + \delta_\beta \hat{\mathbf{j}})\boldsymbol{\mathcal{R}}_{\mathcal{P}}(\mathbf{p}_1)^\top & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \tag{3.47}$$

where a prerotation $\boldsymbol{\mathcal{R}}_{\mathcal{P}}(\mathbf{p}_1)^\top$ aligns the vector $\mathbf{p}_1$ with the camera frame z-axis, and is calculated by the operator,

$$\boldsymbol{\mathcal{R}}_{\mathcal{P}} : \mathbb{R}^3 \to SO(3) \tag{3.48}$$

according to,

$$\boldsymbol{\mathcal{R}}_{\mathcal{P}}(\mathbf{p}_1) = \exp_{SO(3)}(\theta \hat{\boldsymbol{\omega}}) \tag{3.49}$$

with

$$\theta = -\arcsin \|\boldsymbol{\omega}\| \tag{3.50}$$

and

$$\boldsymbol{\omega} = \hat{\mathbf{p}}_1 \times \hat{\mathbf{k}}. \tag{3.51}$$

Next, the vector is perturbed by the two incremental angles δ_α and δ_β, which preserves the length of the vector. This changes the direction of the vector using the local coordinate system, (X', Y', Z'), on the sphere. Finally, the vector is rotated back to the original neighbourhood using $\boldsymbol{\mathcal{R}}_{\mathcal{P}}(\mathbf{p}_1)$ and then scaled by the increment $(1 + \delta_r)$.

Updates to the bearing to the point feature are exclusively performed using the incremental angles, δ_α and δ_β, while updates to the the depth and global scale are performed through δ_r. Isolating these two effects means that the system can refine the bearing states quickly during small or degenerate motions, and when the motion is appropriate to recover the global scale, the radial increment can be used to push the depth to the accurate value.

Compared with IDP, this parameterization has the advantage of maintaining the state representation of the point feature position in the Cartesian coordinates, while updating the position in a similar spherical manner. Additionally, the parameterization is not vulnerable to the singularity present in IDP associated with the altitude angle going to $\pm\frac{\pi}{2}$ rad. As a result, it can accommodate component cameras with greater than 180 degree FOV.

3.2.3 Optimization on $\boxplus$-Manifolds

It is possible to modify the common nonlinear least-squares optimization algorithms to work with $\boxplus$-manifolds by replacing the $+$ and $-$ operators with the $\boxplus$ and $\boxminus$ operators for the respective space [33]. An optimization algorithm extensively used for bundle adjustment and SLAM problems is the Levenberg-Marquardt (LM) method [31], which can be adapted to work with the $\boxplus$-manifold framework following the LM derivation in [31] for systems in the real vector space. The optimization attempts to minimize a nonlinear cost function, $c : \mathcal{S} \to \mathbb{R}$, and determine the optimal state vector estimate, $\breve{\mathbf{x}}^* \in \mathcal{S}$, such that,

$$\breve{\mathbf{x}}^* = \arg\min_{\mathbf{x}} c(\mathbf{x}). \tag{3.52}$$

The cost function evaluates a weighted sum of squared measurement error,

$$\begin{aligned} c(\breve{\mathbf{x}}) &= \frac{1}{2}\bar{\mathbf{z}}^\top \mathbf{R}^{-1}\bar{\mathbf{z}} && (3.53)\\ &= \frac{1}{2}(\mathbf{z} \boxminus_{\mathcal{M}} \breve{\mathbf{z}})^\top \mathbf{R}^{-1}(\mathbf{z} \boxminus_{\mathcal{M}} \breve{\mathbf{z}}) && (3.54)\\ &= \frac{1}{2}(\mathbf{z} \boxminus_{\mathcal{M}} \mathbf{g}(\breve{\mathbf{x}}))^\top \mathbf{R}^{-1}(\mathbf{z} \boxminus_{\mathcal{M}} \mathbf{g}(\breve{\mathbf{x}})) && (3.55) \end{aligned}$$

where $\mathbf{R} \in \mathbb{R}^{m\times m}$ is the covariance matrix of the measurement noise, $\bar{\mathbf{z}} = \mathbf{z} \boxminus_{\mathcal{M}} \breve{\mathbf{z}} \in \mathbb{R}^m$ is the measurement error vector – the difference between the actual point feature measurements, $\mathbf{z} \in \mathcal{M}$, and the predicted feature measurements mapped from the estimated state vector, $\breve{\mathbf{z}} = \mathbf{g}(\breve{\mathbf{x}}) \in \mathcal{M}$.

The optimization proceeds iteratively, starting with an initial estimate, $\breve{\mathbf{x}}_0 \in \mathcal{S}$. Each iteration seeks to update the current state estimate, $\breve{\mathbf{x}}_k$, with a vector $\boldsymbol{\delta}_k \in \mathbb{R}^n$,

$$\breve{\mathbf{x}}_{k+1} = \breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k, \tag{3.56}$$

such that the sequence $\{\breve{\mathbf{x}}_0, \breve{\mathbf{x}}_1, \breve{\mathbf{x}}_2, \dots\} \to \breve{\mathbf{x}}^*$. To find the value of the update vector, the cost function is approximated about the current state estimate $\breve{\mathbf{x}}_k$ using the second-order Taylor-series expansion,

$$c(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k) \approx c(\breve{\mathbf{x}}_k) + \left(\left.\frac{\partial c(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k)}{\partial \boldsymbol{\delta}_k}\right|_{\boldsymbol{\delta}_k=\mathbf{0}} \right)^\top \boldsymbol{\delta}_k + \frac{1}{2}\boldsymbol{\delta}_k{}^\top \left(\left.\frac{\partial^2 c(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k)}{\partial \boldsymbol{\delta}_k{}^2}\right|_{\boldsymbol{\delta}_k=\mathbf{0}} \right) \boldsymbol{\delta}_k. \tag{3.57}$$

In order to minimize the cost approximation with respect to the update $\boldsymbol{\delta}_k$, the derivative of (3.57) is set to zero, resulting in,

$$\left(\left.\frac{\partial^2 c(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k)}{\partial \boldsymbol{\delta}_k{}^2}\right|_{\boldsymbol{\delta}_k=\mathbf{0}} \right) \boldsymbol{\delta}_k = -\left(\left.\frac{\partial c(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k)}{\partial \boldsymbol{\delta}_k}\right|_{\boldsymbol{\delta}_k=\mathbf{0}} \right). \tag{3.58}$$

For the chosen cost function, the first partial derivative evaluates to,

$$\left.\frac{\partial c(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k)}{\partial \boldsymbol{\delta}_k}\right|_{\boldsymbol{\delta}_k=\mathbf{0}} = -\mathbf{J}^\top \mathbf{R}^{-1}\bar{\mathbf{z}}_k \tag{3.59}$$

where

$$\mathbf{J} = \left.\frac{\partial \mathbf{g}(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k)}{\partial \boldsymbol{\delta}_k}\right|_{\boldsymbol{\delta}_k=\mathbf{0}} \tag{3.60}$$

is the Jacobian of the estimated measurement vector with respect to the update vector $\boldsymbol{\delta}_k$. For the LM method, the Hessian of the cost function is approximated by,

$$\left.\frac{\partial^2 c(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k)}{\partial \boldsymbol{\delta}_k{}^2}\right|_{\boldsymbol{\delta}_k=\mathbf{0}} \approx \mathbf{J}^\top \mathbf{R}^{-1}\mathbf{J} + \lambda_k \mathbf{I}_{n\times n} \tag{3.61}$$

where $\lambda_k > 0 \in \mathbb{R}$ is the convergence parameter which trades off against a Gauss-Newton and gradient descent step.

Substituting these values back into Equation (3.58),

$$\left(\mathbf{J}^\top \mathbf{R}^{-1} \mathbf{J} + \lambda_k \mathbf{I}_{n\times n}\right) \boldsymbol{\delta}_k = \mathbf{J}^\top \mathbf{R}^{-1} \bar{\mathbf{z}}_k \tag{3.62}$$

which can then be solved for $\boldsymbol{\delta}_k$,

$$\boldsymbol{\delta}_k = \left(\mathbf{J}^\top \mathbf{R}^{-1} \mathbf{J} + \lambda_k \mathbf{I}_{n\times n}\right)^{-1} \mathbf{J}^\top \mathbf{R}^{-1} \bar{\mathbf{z}}_k \tag{3.63}$$

as long as the matrix $\left(\mathbf{J}^\top \mathbf{R}^{-1} \mathbf{J} + \lambda_k \mathbf{I}_{n\times n}\right)$ is invertible. If it is singular, the system is degenerate and under-constrained. The cases where the system becomes degenerate are investigated in Chapter 4.

The new estimated state is $\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k$. If the value of the cost function with the new state estimate is less than the previous cost, $c(\breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k) < c(\breve{\mathbf{x}}_k)$, the state estimate update is applied to the current state, and the convergence parameter is decreased,

$$\breve{\mathbf{x}}_{k+1} = \breve{\mathbf{x}}_k \boxplus_{\mathcal{S}} \boldsymbol{\delta}_k \tag{3.64}$$

$$\lambda_{k+1} = \lambda_k / 10. \tag{3.65}$$

Otherwise, the update is not applied to the state estimate and the convergence parameter is increased,

$$\breve{\mathbf{x}}_{k+1} = \breve{\mathbf{x}}_k \tag{3.66}$$

$$\lambda_{k+1} = 10\lambda_k. \tag{3.67}$$

The optimization proceeds iteratively using the new state estimate, $\breve{\mathbf{x}}_{k+1}$, until the termination criteria are fulfilled. These include a maximum iteration limit being reached, the magnitude of the state update falling below a threshold, or the magnitude of the cost reduction becoming less than a selected threshold. Ideally, the solution will converge to the global minimum, however, the system can settle to local minima depending on the shape of the cost function and the initial estimate of the solution. Therefore, it is vital to supply the optimization algorithm with a reasonably accurate initial state estimate to help with convergence to the correct solution. A process for identifying a suitable initial estimate for the non-overlapping calibrated multicamera cluster system is presented in Section 3.3.3.

3.2.4 Jacobian Calculation

At each iteration of the optimization algorithm, the update vector $\boldsymbol{\delta}_k$ is calculated using (3.63) containing the measurement Jacobian matrix, $\mathbf{J}$. This matrix represents the change in the predicted point feature measurements for a change in the update vector,

$$\mathbf{J} = \left. \frac{\partial \mathbf{g}(\breve{\mathbf{x}} \boxplus_{\mathcal{S}} \boldsymbol{\delta})}{\partial \boldsymbol{\delta}} \right|_{\boldsymbol{\delta}=\mathbf{0}} . \tag{3.68}$$

The Jacobian is formed by vertically stacking the $2 \times n$ Jacobians for the individual point feature observations,

$$\mathbf{J}_j^{i,\ell} = \left. \frac{\partial \mathbf{g}_j^{i,\ell}(\breve{\mathbf{x}} \boxplus_{\mathcal{S}} \boldsymbol{\delta})}{\partial \boldsymbol{\delta}} \right|_{\boldsymbol{\delta}=\mathbf{0}} = \mathbf{H}_j^{i,\ell} \mathbf{G}_j^{i,\ell}, \tag{3.69}$$

where

$$\mathbf{H}_j^{i,\ell} = \left(\left. \frac{\partial \boldsymbol{\pi}_2 \left(\tilde{\mathbf{p}}_j^{D_i} \right)}{\partial \tilde{\mathbf{p}}_j^{D_i}} \right|_{\boldsymbol{\delta}=\mathbf{0}} \right) \left(\left. \frac{\partial \tilde{\mathbf{p}}_j^{D_i}}{\partial \tilde{\mathbf{p}}_j^{i,\ell}} \right|_{\boldsymbol{\delta}=\mathbf{0}} \right), \tag{3.70}$$

and

$$\mathbf{G}_j^{i,\ell} = \left. \frac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}} \right|_{\boldsymbol{\delta}=\mathbf{0}} . \tag{3.71}$$

The term $\mathbf{H}_j^{i,\ell}$ is a 2×4 matrix dependent on the camera structure and the point position in the observing camera and keyframe coordinate frame. For the pin-hole camera model the terms are,

$$\left. \frac{\partial \boldsymbol{\pi}_2 \left(\tilde{\mathbf{p}}_j^{D_i} \right)}{\partial \tilde{\mathbf{p}}_j^{D_i}} \right|_{\boldsymbol{\delta}=\mathbf{0}} = \frac{1}{\left(w_j^{D_i} \right)^2} \begin{bmatrix} w_j^{D_i} & 0 & -x_j^{D_i} \\ 0 & w_j^{D_i} & -y_j^{D_i} \end{bmatrix} \tag{3.72}$$

and

$$\frac{\partial \tilde{\mathbf{p}}_j^{D_i}}{\partial \tilde{\mathbf{p}}_j^{i,\ell}} = \mathbf{K}_i \tag{3.73}$$

such that,

$$\mathbf{H}_j^{i,\ell} = \frac{1}{\left(w_j^{D_i} \right)^2} \begin{bmatrix} w_j^{D_i} & 0 & -x_j^{D_i} \\ 0 & w_j^{D_i} & -y_j^{D_i} \end{bmatrix} \mathbf{K}_i . \tag{3.74}$$

The term $\mathbf{G}_j^{i,\ell}$ is a $4 \times (6n_k + 3n_f)$ matrix, where n_k and n_f are the number of keyframes and point features in the target model, respectively. This matrix represents the change of the feature position within the observing camera and keyframe coordinate frame for a change in the update vector. The fourth row of this matrix is all zeros since the points are restricted to lie in $\mathbb{R}^3$.

Assume the point feature $\mathbf{p}_j^{h,k}$, anchored in camera C_h of keyframe K_k, is observed at keyframe K_ℓ by camera C_i. The coordinates of the point feature position in the coordinate frame C_iK_ℓ are given in (3.10). The update vector consists of components for all of the keyframes and points in the system.

$$\boldsymbol{\delta} = \begin{bmatrix} {\boldsymbol{\delta}_{K_1}}^\top & \dots & {\boldsymbol{\delta}_{K_{n_k}}}^\top & {\boldsymbol{\delta}_{P_1}}^\top & \dots & {\boldsymbol{\delta}_{P_{n_f}}}^\top \end{bmatrix}^\top \in \mathbb{R}^{(6n_k+3n_f)} \tag{3.75}$$

The position of the point feature in the observing coordinate frame subject to the state perturbations is written,

$$\tilde{\mathbf{p}}_j^{i,\ell} = {\mathbf{T}_{C_i}}^{-1}(\mathbf{T}_{K_\ell} \boxplus_{\mathcal{K}} \boldsymbol{\delta}_{K_\ell})^{-1} (\mathbf{T}_{K_k} \boxplus_{\mathcal{K}} \boldsymbol{\delta}_{K_k})\, \mathbf{T}_{C_h} \tilde{\boldsymbol{\rho}} \left(\mathbf{p}_j \boxplus_{\mathcal{P}} \boldsymbol{\delta}_{P_j}\right) \tag{3.76}$$

Applying the operators for the various $\boxplus$-manifolds in the state space,

$$\begin{aligned} \tilde{\mathbf{p}}_j^{i,\ell} &= {\mathbf{T}_{C_i}}^{-1}(\mathbf{T}_{\mathcal{K}}(\boldsymbol{\delta}_{K_\ell})\mathbf{T}_{K_\ell})^{-1} (\mathbf{T}_{\mathcal{K}}(\boldsymbol{\delta}_{K_k})\mathbf{T}_{K_k})\, \mathbf{T}_{C_h} \left(\mathbf{T}_{\mathcal{P}}(\boldsymbol{\delta}_{P_j})\tilde{\mathbf{p}}_j\right) &(3.77)\\ &= {\mathbf{T}_{C_i}}^{-1}{\mathbf{T}_{K_\ell}}^{-1}(\mathbf{T}_{\mathcal{K}}(\boldsymbol{\delta}_{K_\ell}))^{-1}\mathbf{T}_{\mathcal{K}}(\boldsymbol{\delta}_{K_k})\mathbf{T}_{K_k}\mathbf{T}_{C_h}\mathbf{T}_{\mathcal{P}}(\boldsymbol{\delta}_{P_j})\tilde{\mathbf{p}}_j &(3.78)\end{aligned}$$

where $\tilde{\mathbf{p}}_j = \tilde{\boldsymbol{\rho}}(\mathbf{p}_j)$.

The only non-zero blocks in $\mathbf{G}_j^{i,\ell}$ are those corresponding to the anchor and observing keyframes, as well as the feature position,

$$\mathbf{G}_j^{i,\ell} = \begin{bmatrix} \mathbf{0} & \dfrac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}_{K_k}} & \mathbf{0} & \dots & \mathbf{0} & \dfrac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}_{K_\ell}} & \mathbf{0} & \dots & \mathbf{0} & \dfrac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}_{P_j}} & \mathbf{0} \end{bmatrix} \tag{3.79}$$

Accordingly, these blocks will be investigated separately in the following.

The Jacobian block associated with the update vector for the anchor keyframe, K_k, are [71],

$$\begin{aligned} \frac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}_{K_k}} &= \left((\mathbf{T}_{C_i})^{-1}(\mathbf{T}_{K_\ell})^{-1}\right) \frac{\partial \mathbf{T}_{\mathcal{K}}(\boldsymbol{\delta}_{K_k})}{\partial \boldsymbol{\delta}_{K_k}} \left(\mathbf{T}_{K_k}\mathbf{T}_{C_h}\tilde{\mathbf{p}}_j\right) &(3.80)\\ &= \mathbf{T}_M^{i,\ell} \begin{bmatrix} \mathbf{I}_{3\times 3} & -\left[\mathbf{p}_j^M\right]_\times \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} \end{bmatrix} &(3.81)\\ &= \begin{bmatrix} \boldsymbol{\mathcal{R}}_M^{i,\ell} & -\boldsymbol{\mathcal{R}}_M^{i,\ell}\left[\mathbf{p}_j^M\right]_\times \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} \end{bmatrix}, &(3.82)\end{aligned}$$

where $\boldsymbol{\mathcal{R}}_M^{i,\ell}$ is the rotation matrix within $\mathbf{T}_M^{i,\ell}$ as in (3.3).

Similarly, for the Jacobian block associated with the observing keyframe update vector,

$$\frac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}_{K_\ell}} = \left((\mathbf{T}_{C_i})^{-1}(\mathbf{T}_{K_\ell})^{-1}\right) \frac{\partial\left(\mathbf{T}_{\mathcal{K}}(\boldsymbol{\delta}_{K_\ell})^{-1}\right)}{\partial \boldsymbol{\delta}_{K_\ell}} \left(\mathbf{T}_{K_k}\mathbf{T}_{C_h}\tilde{\mathbf{p}}_j\right) \tag{3.83}$$

$$= \left((\mathbf{T}_{C_i})^{-1}(\mathbf{T}_{K_\ell})^{-1}\right) \frac{\partial \mathbf{T}_{\mathcal{K}}(-\boldsymbol{\delta}_{K_\ell})}{\partial \boldsymbol{\delta}_{K_\ell}} \left(\mathbf{T}_{K_k}\mathbf{T}_{C_h}\tilde{\mathbf{p}}_j\right) \tag{3.84}$$

$$= \begin{bmatrix} -\boldsymbol{\mathcal{R}}_M^{i,\ell} & \boldsymbol{\mathcal{R}}_M^{i,\ell}\left[\mathbf{p}_j^M\right]_\times \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} \end{bmatrix}. \tag{3.85}$$

In the event that the anchor and observing keyframes are the same, $k = \ell$, updating the pose will have no effect on the feature measurement since the transformation matrices will always combine to identity in (3.14). As a result, the block in $\mathbf{G}_j^{i,\ell}$ for that keyframe will be,

$$\frac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}_{K_\ell}} = \frac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}_{K_k}} = \mathbf{0}_{4\times 6}. \tag{3.86}$$

Finally, for the novel point feature position update, the Jacobian block has the form,

$$\frac{\partial \tilde{\mathbf{p}}_j^{i,\ell}}{\partial \boldsymbol{\delta}_{P_j}} = \|\mathbf{p}_j\| \begin{bmatrix} \boldsymbol{\mathcal{R}}_{i,\ell}^{h,k}\boldsymbol{\mathcal{R}}_{P_j} \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ \mathbf{0}_{1\times 3} \end{bmatrix}. \tag{3.87}$$

For the example system introduced in Section 3.1.4, the block structure of the resulting measurement Jacobian is shown in Figure 3.6, where the shaded blocks represent the non-zero blocks in the matrix.

The full Jacobian $\mathbf{J}$ for the general BA optimization can be calculated given the current estimate of the state and used to find the new update vector $\boldsymbol{\delta}$ for the next iteration, from (3.63). Provided that the initial estimate is in the neighbourhood of the true solution the optimization will converge to the global minimum and produce an accurate target model. The proposed algorithm along with the process to form an initial estimate for an unknown target is described in the next section.

3.3 Algorithm

The modelling and optimization mechanisms described in the previous sections are now combined into a real-time relative motion and target model estimation

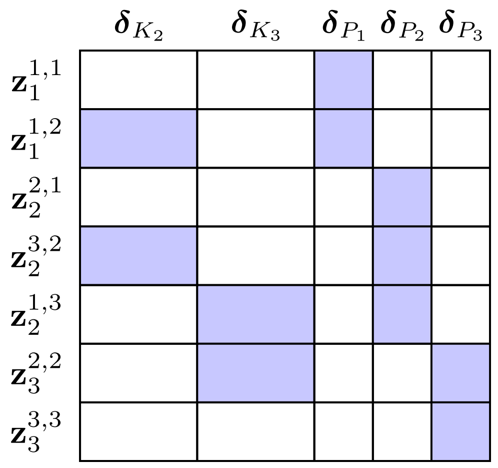


Figure 3.6: The Jacobian block structure for the example multicamera cluster system. The non-zero blocks for each set of observation rows are shown as the shaded blocks.

algorithm using the non-overlapping calibrated camera cluster. The proposed parameterization and optimization method, along with the novel initialization scheme, presented in Section 3.3.3, allow the solution to accurately converge despite no prior knowledge of the target object model or relative motion.

Similar to PTAM [45] and MCPTAM [30], the tasks of real-time motion tracking and accurate reconstruction of the target model structure are divided into parallel tasks running as distinct processes. The full optimization, as described in Section 3.2, is implemented in the BA module to accurately determine the poses of the keyframes and positions of the point features within them. The included keyframes are carefully selected to sufficiently constrain the target model solution, while limiting the total number of keyframes in the target model to keep the computational requirements low. Concurrently, a pose tracking process localizes the current camera cluster coordinate frame within the most recent target model generated by the BA module.

The separation of the motion tracking and BA optimization tasks alleviates the real-time requirement on the full nonlinear optimization. The challenging part is to boot-strap the process successfully despite no overlap in the cluster camera FOV and no prior knowledge of the target object.

3.3.1 Bundle Adjustment

A dedicated process generates and continually refines the model of the target object or environment using the full nonlinear BA framework detailed in Section 3.2. The target model consists of a set of permanent keyframes, in which all of the corresponding point feature positions are parameterized, and a single temporary keyframe representing the most recently acquired cluster pose from the tracking thread. The distinction between permanent and temporary keyframes is only their persistence in the target model past the conclusion of the optimization.

Permanent keyframes are, as the name suggests, permanently part of the target model and can be used to anchor point feature positions. A temporary keyframe is added to the target model for the purposes of triangulating point features within the permanent keyframes and constraining the poses of the permanent keyframes, and is discarded when a BA optimization completes. However, if the algorithm determines that the temporary keyframe significantly improves the target model by observing a set of new point features or has a long baseline from the neighbouring permanent keyframes, the temporary keyframe can be promoted to a permanent keyframe in the target model. As a result, new point features are anchored in this frame and added to the target model. The mechanics for optimizing the permanent and temporary keyframes are identical in the BA process.

When the BA process completes, the updated target model is sent to the tracking thread for use in localizing the current relative pose of the camera cluster. The BA process will then retrieve the latest cluster frame, including the images from the individual cameras, and begin the optimization with this pose as a new temporary keyframe.

3.3.2 Pose Tracking

In the pose tracking process, the target model parameters are held fixed. Only the current pose of the cluster coordinate frame, denoted U, with respect to the target model frame, M, is optimized given the measurements of the existing target model point features in the current set of component camera images. As a result, the tracking system state is simply,

$$\mathbf{x} = \mathbf{T}_U^M \equiv \mathbf{T}_U = \begin{bmatrix} \boldsymbol{\mathcal{R}}_U & \mathbf{t}_U \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in SE(3), \tag{3.88}$$

while the system measurement vector consists of all of the image-plane measurements of all of the currently visible point features from within the target model. For the point feature j, anchored in camera h of keyframe k, the measurements in the i^{th} camera at the current cluster pose are as follows,

$$\mathbf{z}_j^{i,u} = \boldsymbol{\pi}_2 \left(\mathbf{K}_i (\mathbf{T}_{C_i})^{-1} (\mathbf{T}_U)^{-1} \mathbf{T}_{K_k} \mathbf{T}_{C_h} \tilde{\mathbf{p}}_j^{h,k} \right). \tag{3.89}$$

By stacking all of the current point feature measurements into the system measurement vector, the relative pose is then estimated using an iterative nonlinear least squares optimization method, as in Section 3.2.3.

Since both the camera cluster and the target object or environment are able to move, the relative motion dynamics are approximated as a simple constant-velocity model. Between time steps, the two previous relative pose estimates are combined to predict the current pose of the cluster with respect to the target. However, this only serves as the initial condition for the pose estimate and the optimization proceeds using just the current image measurements to constrain the current solution.

With this system, the individual Jacobians are calculated in the same manner as for the observing keyframes in (3.83),

$$\mathbf{J}_j^{i,u} = \frac{1}{\left(w_j^{D_i}\right)^2} \begin{bmatrix} w_j^{D_i} & 0 & -x_j^{D_i} \\ 0 & w_j^{D_i} & -y_j^{D_i} \end{bmatrix} \mathbf{K}_i \begin{bmatrix} -\boldsymbol{\mathcal{R}}_M^{i,u} & \boldsymbol{\mathcal{R}}_M^{i,u} \left[\mathbf{p}_j^M\right]_\times \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} \end{bmatrix} \tag{3.90}$$

where $\boldsymbol{\mathcal{R}}_M^{i,u} = {\boldsymbol{\mathcal{R}}_{C_i}}^\top {\boldsymbol{\mathcal{R}}_U}^\top$, and

$$\begin{bmatrix} x_j^{D_i} \\ y_j^{D_i} \\ w_j^{D_i} \end{bmatrix} = \mathbf{K}_i (\mathbf{T}_{C_i})^{-1} (\mathbf{T}_U)^{-1} \mathbf{T}_{K_k} \mathbf{T}_{C_h} \tilde{\mathbf{p}}_j^{h,k}. \tag{3.91}$$

The full system measurement Jacobian is formed by stacking the individual Jacobians vertically.

The small dimension of this state space allows this process to run at the frame rate of the cluster cameras to maintain a real-time pose estimate for the camera cluster with respect to the target model. The accuracy of the resulting estimates is dependent on the accuracy of the most recent target model from the BA process. In practice, this methodology has proven to work effectively once the target model is refined over a number of BA optimization runs. The most critical portion is the initialization phase when the generated target model is uncertain due to the limited information available.

It is possible for the target model to change significantly between time steps of the pose tracking process, particularly if recently-added keyframes lead to significant corrections to the global scale metric in the model generated by the BA process. As a result, the tracking process could potentially fail if the previous estimate becomes a poor initial condition for the optimization with respect to the new model. However, the large scale corrections occur towards the beginning of the motion trajectory as the relative motion becomes sufficiently large to constrain the global scale metric. Assuming that the update rate of the BA process is fast enough given the magnitude of the relative motion, the point feature measurements will still be relatively insensitive to an incorrect scale metric (refer to Chapter 4). As a result, the pose tracking optimization will converge to the new current estimate despite the poorly-scaled initial condition.

For a more proactive solution, the system could calculate the change in the global scale metric before and after the BA process completes, using the translational distance between keyframes, and scale the previous pose tracking estimate accordingly. Results have demonstrated that even without the proactive strategy, the pose tracking process is robust to the changes in the target model produced during normal operation (refer to Chapter 5).

3.3.3 Initialization

Different from the PTAM approach, the proposed tracking and BA processes operate in parallel right from the start of the motion sequence. Further, compared to MCPTAM, it adds the ability to successfully operate even when there is no overlap in the FOV of the cluster cameras. The system is parameterized to allow the solution to converge using only an initial estimate obtained immediately at start-up as the base solution. More detailed descriptions of the PTAM and MCPTAM algorithms are provided in Appendix B.

Upon capturing an initial set of images from all of the component cameras in the cluster, a permanent keyframe is added to the target model and fixed collocated at the target model reference frame, M. The observed feature points are initialized using the image space measurements to determine the bearing to the points in space at this keyframe. These features are anchored at this keyframe within the cameras in which they are observed. Since there is no overlap in the FOV of the cameras, there is no information about the depth of the point features in the scene, except that the points must sit in front of the cameras.

For a measurement of the j^{th} point feature $[u_j, v_j] \in \mathbb{R}^2$ first observed, and subsequently anchored in camera h at the new keyframe being added to the model, the ray along which it lies can be found by multiplying by the left-inverse of the camera projection matrix, ${\mathbf{K}_h}^{-1}$ from (2.15),

$$\mathbf{q}_j = \boldsymbol{\pi}_3 \left({\mathbf{K}_h}^{-1} \begin{bmatrix} u_j \\ v_j \\ 1 \end{bmatrix} \right) \in \mathbb{R}^3. \tag{3.92}$$

Subsequently, the initial estimate of the point feature position can be calculated using this direction ray,

$$\mathbf{p}_j = d_0 \frac{\mathbf{q}_j}{\|\mathbf{q}_j\|} \tag{3.93}$$

where the initial depth along the feature ray d_0 can be set using even poor a priori information regarding the target model, such as expected average feature point depth. If none is available, a reasonable nominal value can be selected, such as $d_0 = 1$. This forms a hemisphere of point features around each camera coordinate frame. An example initial keyframe for a four-camera cluster is shown in Figure 3.7. This rough target model is immediately available to the tracking thread to localize the camera cluster frame against.

At the second time-step, a temporary keyframe is added to the model and the BA thread begins the optimization. Simultaneously, the tracker localizes the camera cluster with respect to the most recent target model that it has received from the BA thread. Initially, the model is inaccurate, particularly in the global scale, due to the incorrect point feature depths, the small triangulation baselines, and the motion being close to degenerate (Chapter 4). However, results have shown that the tracking thread is able to consistently localize with respect to the poor target model with a small amount of steady-state error and incorrect global scale. In fact, the small translational baselines between cluster poses are beneficial, in this case, since the incorrect point feature depth estimates do not produce large errors in the reprojection of those features onto the camera image plane when the motion is small. Therefore the initial tracking solution is not very sensitive to errors in the depth of the point features, in the local neighbourhood around the initial cluster pose.

When the BA process finishes an optimization, it will send the corrected permanent portion of the target model to the tracking thread and retrieve the current pose and measurement set from the tracker to act as the new temporary keyframe,

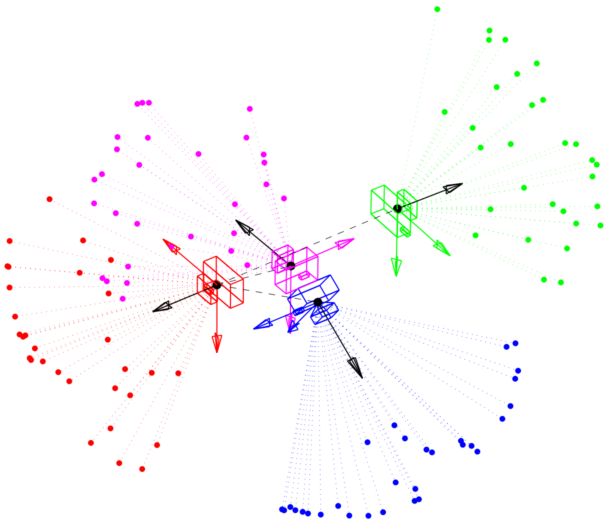


Figure 3.7: The initial permanent keyframe generated using the first set of camera images for an example four-camera cluster. The point estimates are generated using the bearing from the image measurement at an uncertain depth estimate, resulting in a set spheres of point feature estimates centred about the camera coordinate frame origins.

and start the optimization again. The second permanent keyframe is added at the cluster pose when the maximum trace of the point feature covariance matrices falls below a selected threshold. This metric is selected to keep the number of permanent keyframes low during initialization and avoid adding keyframes before the solution is constrained. This allows the BA to converge quickly to a solution even when there are many point features in the model. Because of the small baselines and rotation angles, the global scale of the solution may be inaccurate, but it will be consistent, and as a result, the BA thread is able to supply the tracking thread with an updated version of the target model at or near the frame rate of cluster cameras. In the event that the camera cluster undergoes an extremely fast motion that makes the measurements more sensitive to scale, such as large rotations, the tracking thread may become lost if the model is not updated in time. A diagram of the program flow for the algorithm from initialization is shown in Figure 3.8 for an example system.

The time steps in the graph correspond to the instant when the camera devices capture an image. There is a small delay involved in acquiring the images due to the data transfer processes. At the first time step, the images are used to

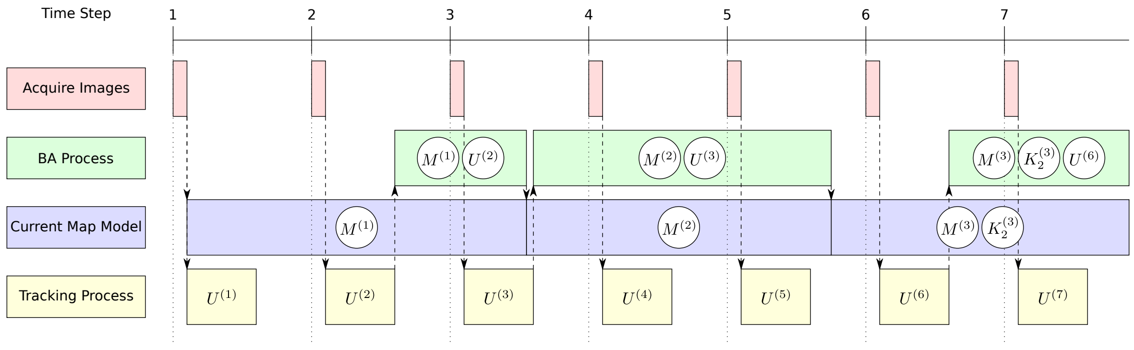


Figure 3.8: A timing graph showing interaction between the system components for an example execution of the proposed algorithm at start-up.

construct the initial keyframe, $M^{(1)}$, where the superscript indicates the most recent camera image set used to generate the keyframe estimate. The target model now consists of only this keyframe and the anchored point features within it, as described previously.

When the second set of frames arrives, the tracking thread localizes the cluster frame pose, $U^{(2)}$, with respect to the current model. Subsequently, it passes this pose estimate to the BA process, which begins optimizing the most recent set of permanent keyframes, $M^{(1)}$, and the temporary keyframe at the current cluster pose. Before the BA process completes, the third set of camera images arrives and the tracking thread again localizes against the available target model. When the BA completes, the temporary keyframe $U^{(2)}$ is discarded and the updated permanent keyframe, $M^{(2)}$, is set as the current model. The BA process then begins again using the most recent pose estimate, $U^{(3)}$, as the temporary keyframe.

At the next time step, the tracking thread now localizes the current cluster frame with respect to the updated target model. At time step five, the BA process completes and determines from analyzing the point feature covariances that it should promote the temporary keyframe to the permanent portion of the model. The updated model now consists of the initial pose, $M^{(3)}$, and a second keyframe, $K_2^{(3)}$. The tracking thread is now able to localize relative to the more accurate target model, and the BA process proceeds with two permanent keyframes and the next cluster pose as the temporary third keyframe.

The proposed algorithm continues in this manner as more keyframes are added and refined to improve the pose tracking process. As more permanent keyframes are accumulated, the optimization time required for the BA thread will begin to grow cubically. This constrains the application of this method to tracking target objects and environments of a certain size, such that they are sufficiently modelled using a moderate number of keyframes. For general motion trajectories, the global scale of the target model will converge and the tracking thread is able to successfully and accurately localize the camera cluster despite only receiving the low frequency refinements from the BA process.

For the BA thread to produce an accurate target model through iterative updates $\boldsymbol{\delta}_k$ determined in (3.63), the matrix $(\mathbf{J}^\top R^{-1}\mathbf{J} + \lambda_k \mathbf{I}_{n\times n})$ must be non-singular as $\lambda_k \to 0$. If it is singular, $\boldsymbol{\delta}_k$ is not unique and the system is degenerate. The matrix Jacobian $\mathbf{J}$ is composed of the relative motion of the cluster and target, as well as the cluster configuration and the point feature locations. The combinations of these parameters that lead to the system becoming degenerate are investigated in Chapter 4 to identify situations where the proposed algorithm may be unable to recover an accurate pose estimate for the case of a camera cluster with non-overlapping FOV.

Chapter 4

Degeneracies of Multicamera Cluster SLAM

In order for any estimation system to operate successfully, the current state of the system must be uniquely recoverable given the measurable outputs up to, and including the current time step. In the context of the multicamera cluster relative pose system presented in this thesis, this means that the image measurements must contain sufficient information to recover the cluster motion and the target model parameters, including the proper global scale metric. Furthermore, the solution should be unique since convergence to a different configuration, which may also agree with the measurements, would likely result in failure of perception and control operations.

This chapter investigates the degenerate configurations when estimating the system states of a two-camera cluster over two keyframes while observing a set of point features in each camera and using an iterative optimization or recursive filter-based approach. This includes Bundle Adjustment schemes, including the one presented in Chapter 3, as well as recursive filters such as an extended Kalman filter. In Section 4.2, the degeneracy of only the scale parameter will be analyzed to find when the global scale metric for the system cannot be recovered, despite each individual camera being able to determine its own local motion increment. Next in Section 4.3, the analysis will be extended to look at the larger problem of estimating the full state and parameter set. A non-zero determinant condition is presented to determine when there is sufficient information in the measurements to uniquely estimate the system states.

The main contribution of this chapter is the identification of configurations of relative motion and target structure model leading to failure of an iterative nonlinear optimization-based solution. Degenerate configurations for both the scale-only and full state scenarios are analyzed. While other researchers have provided analyses of other solution methods [40, 44, 13], this is the first work to consider degeneracies for multicamera cluster motion estimation schemes employing Bundle Adjustment or recursive filters.

Determining the system configurations leading to solution degeneracy is closely related to the concept of observability in control systems. In the study of observability for nonlinear systems, the local weak observability of the system can be determined by checking the observability rank condition about any point in the state space [32]. This involves checking the column rank of a matrix containing the partial derivatives with respect to the system states, for increasing orders of Lie derivatives of the measurement model with respect the the system dynamics. When the matrix has full column rank, the system is locally weakly observable about that point. For the BA system presented in Chapter 3, the system does not have any dynamics and therefore, only the zeroth-order Lie derivatives are non-zero. In this case, evaluating the observability rank condition is equivalent to checking the rank of the measurement Jacobian matrix, as will be done in the degeneracy analysis in this chapter. If the system were to contain a model of the relative motion dynamics, and the extra information that comes with it, the higher-order Lie derivatives of the measurement model would contain non-zero terms and the added matrix rows would only increase the likelihood that the matrix has full column rank at any point in the state space. However, in this analysis, no such assumptions about the relative motion dynamics are made and the degenerate configurations arising from only using image measurements for a set of point features over two keyframes, are identified.

4.1 Problem Formulation

Recall the notation used to represent a point position in a coordinate frame for a certain camera of a particular keyframe, is as follows. The symbol $\mathbf{p}_j^{i,k}$ represents the j^{th} point in the coordinate frame corresponding to the i^{th} camera frame of keyframe k. The associated unit vector is denoted $\hat{\mathbf{p}}_j^{i,k}$ and is related to the point quantity by,

$$\hat{\mathbf{p}}_j^{i,k} = \frac{\mathbf{p}_j^{i,k}}{\left\|\mathbf{p}_j^{i,k}\right\|}, \quad \left\|\mathbf{p}_j^{i,k}\right\| > 0 \tag{4.1}$$

where $\|\cdot\|$ is the vector 2-norm.

Consider the combined state and parameter estimation for a two-camera cluster observing a set of point features over two keyframes, as shown in Figure 4.1. Each camera has its own mutually exclusive set of point features which it observes at both keyframes. That is, if a point feature is observed by camera i at the first keyframe, it is observed by only camera i at the second keyframe and no other cameras. In the event that a point feature falls within the intersection of the FOVs of two cameras at both keyframes, each camera will consider the point feature as a separate feature and it will be tracked twice. The correspondence of point features is only performed across keyframes within the same camera. In a practical implementation, it would be advantageous to track the point features across the different cameras. This possibility is not included here to find the theoretical limits of the problem.

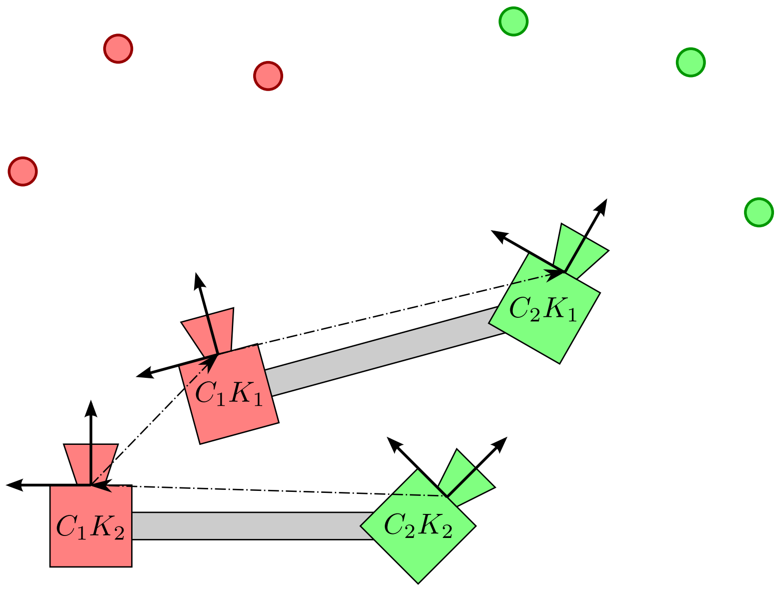


Figure 4.1: The two-camera cluster observes point features over two keyframes.

4.1.1 System Parameterization

For the purposes of the two-camera cluster system studied in this chapter, the system parameters and measurement equations from Chapter 3 are briefly reintroduced below. The state space is not represented using the $\boxplus$-manifolds from Section 3.2.2, but as a vector in $\mathbb{R}^n$. However, for the purposes of this analysis, the results are unaffected since the resulting Jacobian has the same structure as that in Section 3.2.4. Subsequently, the generic formulation is specialized and simplified for the case studied, and intermediate variables are introduced to help in interpreting the results.

The system measurements (3.10) are parameterized as a set of four transformations, followed by a projection transformation. Before the projection, the position of the point feature in the coordinate frame of the observing camera and keyframe is written,

$$\tilde{\mathbf{p}}_j^{i,\ell} = (\mathbf{T}_{C_i})^{-1}(\mathbf{T}_{K_\ell})^{-1}\mathbf{T}_{K_k}\mathbf{T}_{C_h}\tilde{\mathbf{p}}_j^{h,k}. \tag{4.2}$$

This equation describes the position of the j^{th} point feature, anchored in the k^{th} keyframe of camera h, in the coordinate frame of camera i at keyframe ℓ. Each of these intermediate transformations can be represented by a rotation matrix and translation vector,

$$\mathbf{T}_{C_h} = \begin{bmatrix} \boldsymbol{\mathcal{R}}_{C_h} & \mathbf{t}_{C_h} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \tag{4.3}$$

$$\mathbf{T}_{K_k} = \begin{bmatrix} \boldsymbol{\mathcal{R}}_{K_k} & \mathbf{t}_{K_k} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \tag{4.4}$$

$$\mathbf{T}_{K_\ell} = \begin{bmatrix} \boldsymbol{\mathcal{R}}_{K_\ell} & \mathbf{t}_{K_\ell} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \tag{4.5}$$

$$\mathbf{T}_{C_i} = \begin{bmatrix} \boldsymbol{\mathcal{R}}_{C_i} & \mathbf{t}_{C_i} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}. \tag{4.6}$$

When these values are substituted into Equation (4.2), it becomes,

$$\mathbf{p}_j^{i,\ell} = \boldsymbol{\mathcal{R}}_{C_i}^\top\boldsymbol{\mathcal{R}}_{K_\ell}^\top\boldsymbol{\mathcal{R}}_{K_k}\boldsymbol{\mathcal{R}}_{C_i}\mathbf{p}_j^{h,k} + \boldsymbol{\mathcal{R}}_{C_i}^\top\boldsymbol{\mathcal{R}}_{K_\ell}^\top\boldsymbol{\mathcal{R}}_{K_k}\mathbf{t}_{C_h} + \boldsymbol{\mathcal{R}}_{C_i}^\top\boldsymbol{\mathcal{R}}_{K_\ell}^\top\mathbf{t}_{K_k} - \boldsymbol{\mathcal{R}}_{C_i}^\top\boldsymbol{\mathcal{R}}_{K_\ell}^\top\mathbf{t}_{K_\ell} - \boldsymbol{\mathcal{R}}_{C_i}^\top\mathbf{t}_{C_i}. \tag{4.7}$$

This is the most general form of the transformation chain. In this section, the camera cluster is assumed to be rigid and a particular point feature is only observed by one camera at all of the keyframes,

$$\boldsymbol{\mathcal{R}}_{C_i} = \boldsymbol{\mathcal{R}}_{C_h} \tag{4.8}$$

$$\mathbf{t}_{C_i} = \mathbf{t}_{C_h}. \tag{4.9}$$

Furthermore, the point feature is assumed, without loss of generality, to be parameterized in the first keyframe coordinate frame,

$$\boldsymbol{\mathcal{R}}_{K_k} = \mathbf{I}_{3\times3} \tag{4.10}$$

$$\mathbf{t}_{K_k} = \mathbf{0}_{3\times1}. \tag{4.11}$$

When these constraints are applied to (4.7), it simplifies to,

$$\mathbf{p}_j^{i,\ell} = \boldsymbol{\mathcal{R}}_{C_h}^\top \boldsymbol{\mathcal{R}}_{K_\ell}^\top \boldsymbol{\mathcal{R}}_{C_h} \mathbf{p}_j^{h,k} + \boldsymbol{\mathcal{R}}_{C_h}^\top \boldsymbol{\mathcal{R}}_{K_\ell}^\top \mathbf{t}_{C_h} - \boldsymbol{\mathcal{R}}_{C_h}^\top \boldsymbol{\mathcal{R}}_{K_\ell}^\top \mathbf{t}_{K_\ell} - \boldsymbol{\mathcal{R}}_{C_h}^\top \mathbf{t}_{C_h} \tag{4.12}$$

$$= \boldsymbol{\mathcal{R}}_{C_h}^\top \left(\boldsymbol{\mathcal{R}}_{K_\ell}^\top \boldsymbol{\mathcal{R}}_{C_h} \mathbf{p}_j^{h,k} + \left(\boldsymbol{\mathcal{R}}_{K_\ell}^\top - \mathbf{I}_{3\times3} \right) \mathbf{t}_{C_h} - \boldsymbol{\mathcal{R}}_{K_\ell}^\top \mathbf{t}_{K_\ell} \right). \tag{4.13}$$

For simplicity of notation, the remaining camera and keyframe transformation will be redefined,

$$\mathbf{T}_K \equiv \left(\mathbf{T}_{K_\ell}\right)^{-1} \tag{4.14}$$

such that,

$$\boldsymbol{\mathcal{R}}_K \equiv \boldsymbol{\mathcal{R}}_{K_\ell}^\top \tag{4.15}$$

$$\mathbf{t}_K \equiv -\boldsymbol{\mathcal{R}}_{K_\ell}^\top \mathbf{t}_{K_\ell}, \tag{4.16}$$

and

$$\mathbf{T}_C \equiv \mathbf{T}_{C_h} \tag{4.17}$$

such that,

$$\boldsymbol{\mathcal{R}}_C \equiv \boldsymbol{\mathcal{R}}_{C_h} \tag{4.18}$$

$$\mathbf{t}_C \equiv \mathbf{t}_{C_h}. \tag{4.19}$$

Accordingly, Equation (4.20) becomes,

$$\mathbf{p}_j^{i,\ell} = \boldsymbol{\mathcal{R}}_C^\top \left(\boldsymbol{\mathcal{R}}_K \boldsymbol{\mathcal{R}}_C \mathbf{p}_j^{h,k} + \left(\boldsymbol{\mathcal{R}}_K - \mathbf{I}_{3\times3} \right) \mathbf{t}_C + \mathbf{t}_K \right). \tag{4.20}$$

The relative translation and orientation of the second camera frame with respect to the first camera frame is assumed known from extrinsic calibration, fixed over both keyframes, and represented by the translation vector, $\mathbf{t}_C$, and the rotation matrix,

$$\boldsymbol{\mathcal{R}}_C = \begin{bmatrix} \hat{\mathbf{n}}_x & \hat{\mathbf{n}}_y & \hat{\mathbf{n}}_z \end{bmatrix} \in SO(3), \tag{4.21}$$

where $\hat{\mathbf{n}}_x$, $\hat{\mathbf{n}}_y$, $\hat{\mathbf{n}}_z$ are the orthonormal basis vectors for the second camera frame. Accordingly, the position of a point in the coordinate frame of camera 2 at keyframe k, $\mathbf{p}^{2,k}$, can be transformed into the coordinate frame of camera 1 at keyframe k by,

$$\mathbf{p}^{1,k} = \boldsymbol{\mathcal{R}}_C \mathbf{p}^{2,k} + \mathbf{t}_C, \tag{4.22}$$

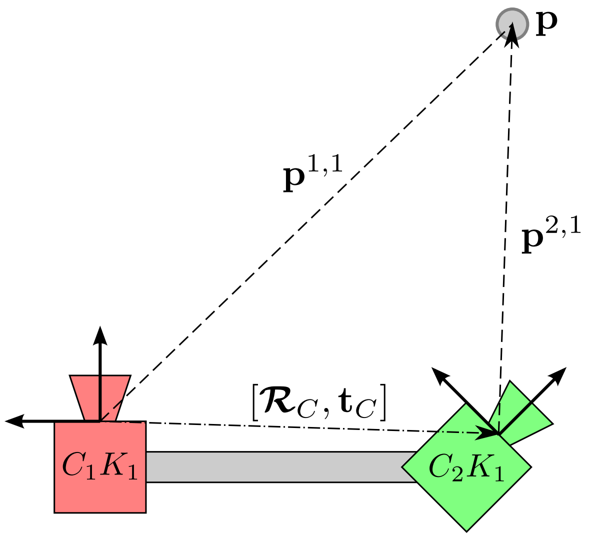


Figure 4.2: Transforming point features between camera frames using the known extrinsic camera calibration.

as shown in Figure 4.2.

Let $p, q \in \mathbb{Z}^+$ be the number of point features observed over both keyframes by the first and second cameras, respectively. The position of the j^{th} point feature, first observed in the i^{th} camera at keyframe one, is parameterized by the azimuth and altitude angles of the vector through the feature, $\phi_j \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, $\theta_j \in (-\pi, \pi]$. The depth along this bearing to the point feature, is the value $r_j \in \mathbb{R}^+$ or $s_j \in \mathbb{R}^+$, for the first or second cameras, respectively. The bearing angles are used to form the unit vector in the camera frames,

$$\hat{\mathbf{p}}_j^{1,1} = \begin{bmatrix} \sin\phi_j \cos\theta_j \\ -\sin\theta_j \\ \cos\phi_j \cos\theta_j \end{bmatrix} \tag{4.23}$$

$$\hat{\mathbf{p}}_j^{2,1} = \begin{bmatrix} \sin\phi_{p+j} \cos\theta_{p+j} \\ -\sin\theta_{p+j} \\ \cos\phi_{p+j} \cos\theta_{p+j} \end{bmatrix},$$

and the point feature positions are along this bearing at the distance r_j or s_j,

$$\mathbf{p}_j^{1,1} = r_j \hat{\mathbf{p}}_j^{1,1} \tag{4.24}$$

$$\mathbf{p}_j^{2,1} = s_j \hat{\mathbf{p}}_j^{2,1} \tag{4.25}$$

In the analyses that follow, the point features are assumed to lie in front of the observing cameras at both keyframes. The central perspective cameras considered in this thesis, are only able to measure point features which appear strictly in front of the camera, with the assumed properties,

$$-\pi < \phi_j < \pi \tag{4.26}$$

$$-\frac{\pi}{2} < \theta_j < \frac{\pi}{2} \tag{4.27}$$

$$r_j > 0 \text{ or } s_j > 0 \tag{4.28}$$

which guarantee that the z-axis coordinate of the point position in the camera coordinate frame is positive,

$$z_j^{i,1} > 0. \tag{4.29}$$

It is further assumed that the motion of the camera cluster is such that the point features are in the positive z-direction of their respective cameras at the second keyframe as well, and

$$z_j^{i,2} > 0. \tag{4.30}$$

The motion of the camera cluster is parameterized by six values describing the relative translation and orientation of the first keyframe with respect to the second keyframe, for the first camera. The translation parameters, t_x, t_y, t_z, form the relative translation vector, $\mathbf{t}_K = [t_x, t_y, t_z]^\top$, and the rotation parameters, $\boldsymbol{\omega}_K = [\omega_x, \omega_y, \omega_z]^\top$, form the relative rotation matrix, $\boldsymbol{\mathcal{R}}_K \in SO(3)$.

To find the minimal number of observed point features necessary to solve for the system states, the number of equations must be made larger than or equal to the number of system states. For the system with two keyframes, each feature point results in two pairs of image coordinates, or four equations.

These measurement equations for the j^{th} point feature in the first camera frame at both keyframes are represented in Figure 4.3, and written using the standard pin-hole camera model,

$$\begin{bmatrix} u_j^{1,1} & v_j^{1,1} & u_j^{1,2} & v_j^{1,2} \end{bmatrix}^\top = \begin{bmatrix} \frac{x_j^{1,1}}{z_j^{1,1}} & \frac{y_j^{1,1}}{z_j^{1,1}} & \frac{x_j^{1,2}}{z_j^{1,2}} & \frac{y_j^{1,2}}{z_j^{1,2}} \end{bmatrix}^\top, \tag{4.31}$$

where

$$\begin{bmatrix} x_j^{1,1} \\ y_j^{1,1} \\ z_j^{1,1} \end{bmatrix} = r_j \hat{\mathbf{p}}_j^{1,1} \tag{4.32}$$

and

$$\begin{bmatrix} x_j^{1,2} \\ y_j^{1,2} \\ z_j^{1,2} \end{bmatrix} = r_j \boldsymbol{\mathcal{R}}_K \hat{\mathbf{p}}_j^{1,1} + \mathbf{t}_K \tag{4.33}$$

$$= r_j \hat{\mathbf{d}}_j + \mathbf{f}, \tag{4.34}$$

with $\hat{\mathbf{d}}_j = \boldsymbol{\mathcal{R}}_K \hat{\mathbf{p}}_j^{1,1}$, and $\mathbf{f} = \mathbf{t}_K$. For this analysis, the projection matrices for both cameras are assumed to be $\mathbf{K}_i = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times1} \end{bmatrix}$ for $i = 1, 2$, without loss of generality.

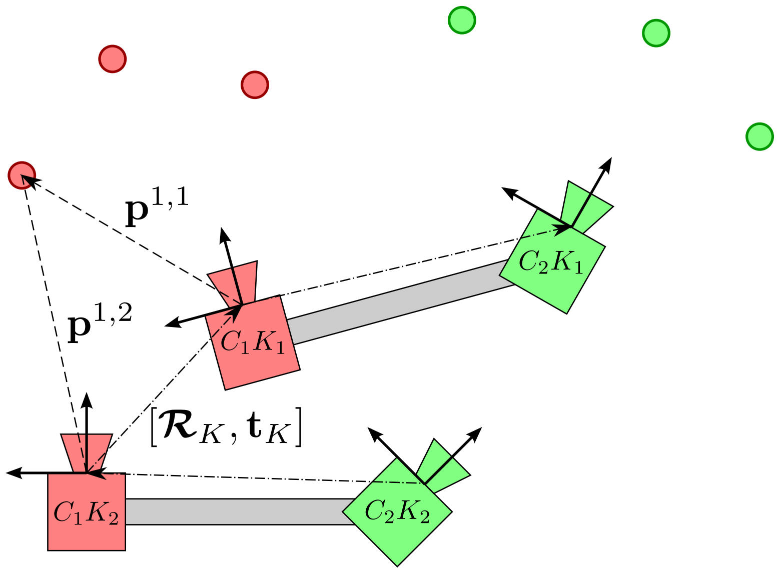


Figure 4.3: Measurements of a feature in the first camera at the two keyframes.

The measurement equations of the j^{th} point feature in the second camera at both keyframes, shown in Figure 4.4, are written in a similar manner, but also include the cluster calibration parameters,

$$\begin{bmatrix} u_j^{2,1} & v_j^{2,1} & u_j^{2,2} & v_j^{2,2} \end{bmatrix}^\top = \begin{bmatrix} \frac{x_j^{2,1}}{z_j^{2,1}} & \frac{y_j^{2,1}}{z_j^{2,1}} & \frac{x_j^{2,2}}{z_j^{2,2}} & \frac{y_j^{2,2}}{z_j^{2,2}} \end{bmatrix}^\top, \tag{4.35}$$

where

$$\begin{bmatrix} x_j^{2,1} \\ y_j^{2,1} \\ z_j^{2,1} \end{bmatrix} = s_j \hat{\mathbf{p}}_j^{2,1} \tag{4.36}$$

and

$$\begin{bmatrix} x_j^{2,2} \\ y_j^{2,2} \\ z_j^{2,2} \end{bmatrix} = \boldsymbol{\mathcal{R}}_C^\top (s_j \boldsymbol{\mathcal{R}}_K \boldsymbol{\mathcal{R}}_C \hat{\mathbf{p}}_j^{2,1} + \mathbf{t}_K + (\boldsymbol{\mathcal{R}}_K - \mathbf{I}_{3\times 3})\mathbf{t}_C) \tag{4.37}$$

$$= \boldsymbol{\mathcal{R}}_C^\top (s_j \hat{\mathbf{e}}_j + \mathbf{f} + \mathbf{g} + \mathbf{h}) \tag{4.38}$$

with $\hat{\mathbf{e}}_j = \boldsymbol{\mathcal{R}}_K \boldsymbol{\mathcal{R}}_C \hat{\mathbf{p}}_j^{2,1}$, $\mathbf{g} = \boldsymbol{\mathcal{R}}_K \mathbf{t}_C$, and $\mathbf{h} = -\mathbf{t}_C$.

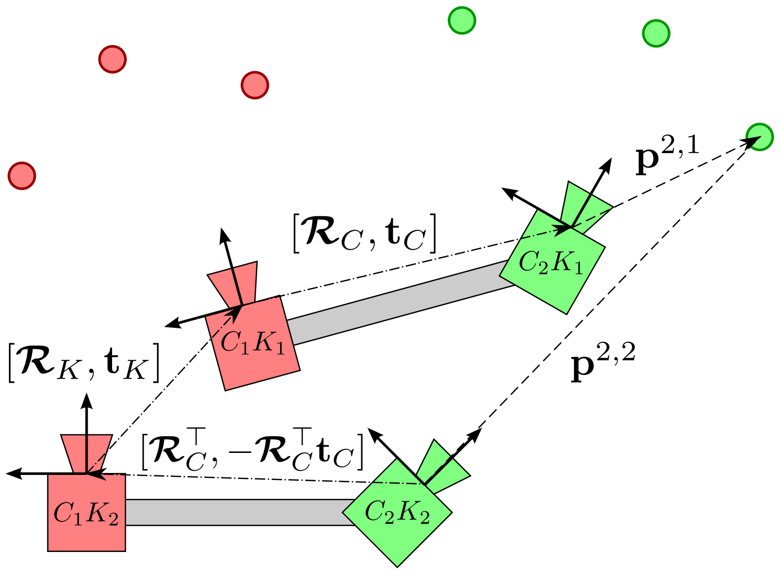

$\mathbf{p}^{2,1}$
$[\boldsymbol{\mathcal{R}}_C, \mathbf{t}_C]$
C_2K_1
C_1K_1
$[\boldsymbol{\mathcal{R}}_K, \mathbf{t}_K]$
$\mathbf{p}^{2,2}$
$[\boldsymbol{\mathcal{R}}_C^\top, -\boldsymbol{\mathcal{R}}_C^\top \mathbf{t}_C]$
C_1K_2
C_2K_2

Figure 4.4: Measurements of the features in the second camera.

The system measurement vector, $\mathbf{z} \in \mathbb{R}^{4(p+q)}$, is composed of the observations of the $p + q$ point features at both keyframes in their respective cameras,

$$\mathbf{z} = \begin{bmatrix} \mathbf{z}_2 \\ \mathbf{z}_1 \end{bmatrix} = \boldsymbol{\varpi}(\mathbf{x}) = \begin{bmatrix} \boldsymbol{\varpi}_2(\mathbf{x}) \\ \boldsymbol{\varpi}_1(\mathbf{x}) \end{bmatrix}, \tag{4.39}$$

where

$$\mathbf{z}_2 = \boldsymbol{\varpi}_2(\mathbf{x}) = \begin{bmatrix} [u_1^{1,2}, v_1^{1,2}]^\top \\ \vdots \\ [u_p^{1,2}, v_p^{1,2}]^\top \\ [u_1^{2,2}, v_1^{2,2}]^\top \\ \vdots \\ [u_q^{2,2}, v_q^{2,2}]^\top \end{bmatrix} \in \mathbb{R}^{2(p+q)}, \tag{4.40}$$

are the measurements at the second keyframe, and

$$\mathbf{z}_1 = \boldsymbol{\varpi}_1(\mathbf{x}) = \begin{bmatrix} [u_1^{1,1}, v_1^{1,1}]^\top \\ \vdots \\ [u_p^{1,1}, v_p^{1,1}]^\top \\ [u_1^{2,1}, v_1^{2,1}]^\top \\ \vdots \\ [u_q^{2,1}, v_q^{2,1}]^\top \end{bmatrix} \in \mathbb{R}^{2(p+q)}, \tag{4.41}$$

are the measurements at the first keyframe. The vector $\mathbf{x}$ is the state vector for the system and the measurement function, $\boldsymbol{\varpi} : \mathbb{R}^n \to \mathbb{R}^{4(p+q)}$ maps the system states to the measurements using $\boldsymbol{\varpi}_1 : \mathbb{R}^n \to \mathbb{R}^{2(p+q)}$ and $\boldsymbol{\varpi}_2 : \mathbb{R}^n \to \mathbb{R}^{2(p+q)}$ for the corresponding keyframes. The specific ordering of the measurements, with the measurements from the second keyframe first, was chosen so that the Jacobian in Section 4.3, to follow, has an upper-block triangular structure.

4.1.2 System Degeneracies

A typical iterative least-squares optimization method, including the one presented in Section 3.2.3, will seek to refine a parameter vector estimate, $\breve{\mathbf{x}} \in \mathbb{R}^n$, using a first-order Taylor-series expansion about the current estimated operating point,

$$\breve{\mathbf{x}}_{i+1} = \breve{\mathbf{x}}_i + \boldsymbol{\delta}_i, \tag{4.42}$$

where the parameter update, $\boldsymbol{\delta}_i \in \mathbb{R}^n$, is found by solving the system,

$$\mathbf{J}\boldsymbol{\delta}_i = \bar{\mathbf{z}}_i, \tag{4.43}$$

with the measurement error,

$$\bar{\mathbf{z}}_i = \mathbf{z} - \boldsymbol{\varpi}(\breve{\mathbf{x}}_i), \tag{4.44}$$

and the associated measurement Jacobian matrix,

$$\mathbf{J} = \left. \frac{\partial \boldsymbol{\varpi}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\mathbf{x}=\breve{\mathbf{x}}_i}. \tag{4.45}$$

Solving for $\boldsymbol{\delta}_i$ depends critically on the column rank of matrix $\mathbf{J}$. Therefore, when $\operatorname{rank}(\mathbf{J}) < n$, one cannot find a unique solution for $\boldsymbol{\delta}_i$ given the measurement error and the system is under-constrained and degenerate. Accordingly, the rank of the Jacobian $\mathbf{J}$ under various motion and structure configurations is studied in the sequel to identify those situations where the system is degenerate when using an estimator based on this approach.

For the particular system studied in this chapter, the Jacobian defined in Chapter 3 takes the form,

$$\mathbf{J} = \begin{bmatrix} \begin{bmatrix} \mathbf{J}_1^{1,2} \\ \vdots \\ \mathbf{J}_p^{1,2} \\ \mathbf{J}_1^{2,2} \\ \vdots \\ \mathbf{J}_q^{2,2} \end{bmatrix}^\top & \begin{bmatrix} \mathbf{J}_1^{1,1} \\ \vdots \\ \mathbf{J}_p^{1,1} \\ \mathbf{J}_1^{2,1} \\ \vdots \\ \mathbf{J}_q^{2,1} \end{bmatrix}^\top \end{bmatrix}^\top . \tag{4.46}$$

Degenerate configurations of the relative motion and target model parameters are then identified as those at which the point feature observations lead to the rank of the Jacobian matrix falling below full column-rank,

$$\operatorname{rank}(\mathbf{J}) < n. \tag{4.47}$$

In the subsequent sections, two estimation scenarios will be investigated for the conditions when the system states, $\mathbf{x}$, cannot be uniquely determined given the configuration of the system during the point feature observations. These configurations are identified by observing when the measurement Jacobian becomes rank-deficient for each set of parameters being estimated.

The first scenario in Section 4.2 assumes that each camera's relative orientation and translation directions are known for the two keyframes by using single camera techniques on the images from the cameras in a decoupled methodology. These individual motion increments are then combined to find the rotation and translation for the camera cluster. The only parameters left to be estimated are the point positions and the magnitude of the translation vector. The configurations in which it is impossible to uniquely determine these parameters are identified and compared to previous results in the literature for similar systems.

The second scenario, found in Section 4.3, extends the analysis to include the estimation of the full cluster motion parameters, which includes the six parameters representing the relative translation and orientation. The resulting degenerate motions are far more complex when including the keyframe parameters in the state vector.

4.2 Observability of Scale

In this section, it is assumed that the five degrees of freedom describing relative orientation and translation direction of the cluster are known from the well-studied single camera ego-motion estimation techniques [31]. Of interest are the conditions when the measurements from the camera cluster are able to allow for estimation of the final degree of freedom, corresponding to the translation magnitude and therefore, global system scale. Similar studies to identify the degenerate motions, though by different methods, were carried out by Kim *et al.* [40], and then subsequently, Clipp *et al.* [13]. The main contribution in this section is the analysis using an alternative procedure which results in confirmation of the previous results plus identification of a further degenerate case.

For this particular estimation scenario, the keyframe translation, $\mathbf{t}_K$, will now be written as,

$$\mathbf{t}_K = s_t \hat{\mathbf{t}}_K, \tag{4.48}$$

where the translation unit vector $\hat{\mathbf{t}}_K$ is assumed known, but the magnitude, s_t, is estimated and included in the state vector. Furthermore, the three parameters for bearing and depth of each of the system feature points are also included as part of the state vector.

In order to estimate the unknown states, there must be a necessary number of measurements containing these values. Each point feature in the system adds four measurements – two image coordinates at the two keyframes – and adds three parameters to the system state. The system state consists of the $3(p+q)$ point feature parameters, along with the one unknown scale factor for the cluster translation. Therefore, the number of observed point features required so that the number of equations are greater than or equal to the number of unknown parameters must satisfy,

$$4(p+q) \geq 1 + 3(p+q) \tag{4.49}$$

$$p+q \geq 1. \tag{4.50}$$

In this system, the scale of the world is embedded in the translation between the camera centres, known from extrinsic calibration. The extrinsic calibration parameters only appear in the measurement equations for point features observed in the second camera. Therefore, at least one feature point must be observed in the second camera to solve for all of the system states, $q \geq 1$.

With one point feature observed in both keyframes of the second camera, the state vector $\mathbf{x} \in \mathbb{R}^4$, is written,

$$\mathbf{x} = \begin{bmatrix} s_t & s_1 & \phi_1 & \theta_1 \end{bmatrix}^\top, \tag{4.51}$$

where s_1, ϕ_1, and θ_1 are the depth and bearings to the single point feature. The bearings form the unit-vector to the point feature according to (4.23).

The measurement equations for this point feature have the same structure as (4.36) and (4.37), but with the scale factor on the keyframe translation,

$$\begin{bmatrix} x^{2,2} \\ y^{2,2} \\ z^{2,2} \end{bmatrix} = \boldsymbol{\mathcal{R}}_C^\top \left(s_1 \boldsymbol{\mathcal{R}}_K \boldsymbol{\mathcal{R}}_C \hat{\mathbf{p}}_1^{2,1} + s_t \hat{\mathbf{t}}_K + (\boldsymbol{\mathcal{R}}_K - \mathbf{I})\mathbf{t}_C \right) \tag{4.52}$$

$$= \boldsymbol{\mathcal{R}}_C^\top \left(s_1 \hat{\mathbf{e}}_1 + s_t \hat{\mathbf{f}} + \mathbf{g} + \mathbf{h} \right), \tag{4.53}$$

such that $\hat{\mathbf{e}}_1 = \boldsymbol{\mathcal{R}}_K \boldsymbol{\mathcal{R}}_C \hat{\mathbf{p}}_1^{2,1}$, and $\hat{\mathbf{f}} = \hat{\mathbf{t}}_K$. The system measurement vector, $\mathbf{z} \in \mathbb{R}^4$, consists of the four image coordinates for the point feature at the two keyframes,

$$\mathbf{z} = \begin{bmatrix} u_j^{2,2} \\ v_j^{2,2} \\ u_j^{2,1} \\ v_j^{2,1} \end{bmatrix}. \tag{4.54}$$

The measurement Jacobian, $\mathbf{J} \in \mathbb{R}^{4\times 4}$, is composed of the partial derivatives of the measurement equations with respect to the system states. These partials are formed using the partial derivatives of the point feature position in each keyframe, with respect to the states. The non-zero elements are as follows,

$$\frac{\partial \mathbf{p}_1^{2,2}}{\partial s_t} = \boldsymbol{\mathcal{R}}_C^\top \hat{\mathbf{f}} \tag{4.55}$$

$$\frac{\partial \mathbf{p}_1^{2,2}}{\partial s_1} = \boldsymbol{\mathcal{R}}_C^\top \hat{\mathbf{e}}_1 \tag{4.56}$$

$$\frac{\partial \mathbf{p}_1^{2,1}}{\partial \phi_1} = s_1 \begin{bmatrix} \cos\phi_1 \cos\theta_1 \\ 0 \\ -\sin\phi_1 \cos\theta_1 \end{bmatrix} \tag{4.57}$$

$$\frac{\partial \mathbf{p}_1^{2,1}}{\partial \theta_1} = s_1 \begin{bmatrix} -\sin\phi_1 \sin\theta_1 \\ -\cos\theta_1 \\ -\cos\phi_1 \sin\theta_1 \end{bmatrix}. \tag{4.58}$$

The terms $\frac{\partial \mathbf{p}_1^{2,2}}{\partial \phi_1}$ and $\frac{\partial \mathbf{p}_1^{2,2}}{\partial \theta_1}$ are also non-zero, but do not affect the rank condition since they appear in the upper-right block of the upper-block triangular matrix, and so are not shown here.

The resulting measurement Jacobian can be written as,

$$\mathbf{J} = \begin{bmatrix} (z_1^{2,2})^{-2}\mathbf{J}_2 & (z_1^{2,2})^{-2}\mathbf{J}_3 \\ \mathbf{0} & (z_1^{2,1})^{-2}\mathbf{J}_1 \end{bmatrix}, \tag{4.59}$$

where

$$\mathbf{J}_1 = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \left(\mathbf{p}_1^{2,1} \times \frac{\partial \mathbf{p}_1^{2,1}}{\partial [\phi_1 \ \theta_1]^\top} \right), \tag{4.60}$$

$$\mathbf{J}_2 = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \left(\mathbf{p}_1^{2,2} \times \frac{\partial \mathbf{p}_1^{2,2}}{\partial [s_t \ s_1]^\top} \right), \tag{4.61}$$

$$\mathbf{J}_3 = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \left(\mathbf{p}_1^{2,2} \times \frac{\partial \mathbf{p}_1^{2,2}}{\partial [\phi_1 \ \theta_1]^\top} \right). \tag{4.62}$$

The determinant of this upper-block triangular matrix is the product of the determinants of only the diagonal blocks,

$$\det(\mathbf{J}) = \frac{\det(\mathbf{J}_2)\det(\mathbf{J}_1)}{(z_1^{2,2})^2 (z_1^{2,1})^2}. \tag{4.63}$$

As a result, it is not necessary to use the submatrix $\mathbf{J}_3$ when finding the determinant.

The configurations when each of the terms goes to zero can be analyzed separately. The term involving the matrix $\mathbf{J_1}$ corresponds to the position of the point feature,

$$\det(\mathbf{J}_1) = -s_1^4 \cos(\phi_1)\cos(\theta_1)^2 \tag{4.64}$$

$$= -s_1^3 \cos(\theta_1) z_1^{2,1}. \tag{4.65}$$

Therefore, the determinant of the $\mathbf{J}_1$ submatrix is zero only if the z-coordinate of the point feature position in the anchor keyframe is zero. That is, the point feature lies on the camera x-y plane in the first keyframe. This configuration is not possible due to the assumption from (4.29). Therefore,

$$\det(\mathbf{J}_1) \neq 0. \tag{4.66}$$

The term involving the matrix $\mathbf{J_2}$ relates the scale parameters to the point feature measurements,

$$\mathbf{J}_2 = \begin{bmatrix} \hat{\mathbf{n}}_y \cdot \Big((s_1\hat{\mathbf{e}}_1 + \mathbf{g} + \mathbf{h}) \times \hat{\mathbf{f}} \Big) & \hat{\mathbf{n}}_y \cdot \Big(\Big(s_t\hat{\mathbf{f}} + \mathbf{g} + \mathbf{h} \Big) \times \hat{\mathbf{e}}_1 \Big) \\ -\hat{\mathbf{n}}_x \cdot \Big((s_1\hat{\mathbf{e}}_1 + \mathbf{g} + \mathbf{h}) \times \hat{\mathbf{f}} \Big) & -\hat{\mathbf{n}}_x \cdot \Big(\Big(s_t\hat{\mathbf{f}} + \mathbf{g} + \mathbf{h} \Big) \times \hat{\mathbf{e}}_1 \Big) \end{bmatrix}, \tag{4.67}$$

leading to the determinant expression,

$$\det(\mathbf{J}_2) = \left(\left(\hat{\mathbf{f}} \times (\mathbf{g}+\mathbf{h})\right) \cdot \hat{\mathbf{e}}_1\right) \left(\hat{\mathbf{n}}_z \cdot \left(s_1\hat{\mathbf{e}}_1 + s_t\hat{\mathbf{f}} + \mathbf{g} + \mathbf{h}\right)\right) \tag{4.68}$$

$$= \left(\left(\hat{\mathbf{f}} \times (\mathbf{g}+\mathbf{h})\right) \cdot \hat{\mathbf{e}}_1\right) \left(z_1^{2,2}\right). \tag{4.69}$$

The second term specifies that the determinant is zero when the feature point lies on the camera x-y plane in the second keyframe, which is impossible by the assumptions in the problem formulation. The first term dictates three more cases when the determinant is zero:

a) $\mathbf{g}+\mathbf{h} = \mathbf{0}_{3\times1}$ **:** implies that $\boldsymbol{\mathcal{R}}_K = \mathbf{I}_{3\times3}$, meaning that there is no relative rotation between the keyframes, or there is no baseline between the camera centres.

b) $\hat{\mathbf{f}} \times (\mathbf{g}+\mathbf{h}) = \mathbf{0}_{3\times1}$ **:** the relative translation, $\hat{\mathbf{f}}$, and the baseline change, $\mathbf{g}+\mathbf{h}$, are collinear. This occurs when the camera centres move in concentric circles with a common centre on the line through the optical centres of the two cameras, as observed in Clipp *et al.* [13]. An example of such a motion is shown in Figure 4.5.

c) $\left(\hat{\mathbf{f}} \times (\mathbf{g}+\mathbf{h})\right) \cdot \hat{\mathbf{e}}_1 = 0$ **:** when the motion ($\hat{\mathbf{f}}$), baseline change ($\mathbf{g}+\mathbf{h}$), and point feature ($\hat{\mathbf{e}}_1$), are coplanar. This shows that, among other configurations, a completely planar system can never be solved to include the proper scale. Additionally, if the motion of the second camera, as viewed from its own coordinate frame, is collinear with the point feature position in the same frame, the system is degenerate. While the (a) and (b) degeneracies are known in the literature, this final configuration is novel and an example of a planar system that falls into this degeneracy is shown in Figure 4.6.

The determinant of $\mathbf{J}$ is now written,

$$\det(\mathbf{J}) = \frac{-s_1^3\cos(\theta_1)\left(\left(\hat{\mathbf{f}} \times (\mathbf{g}+\mathbf{h})\right) \cdot \hat{\mathbf{e}}_1\right)}{(z_1^{2,2})(z_1^{2,1})}, \tag{4.70}$$

and goes to zero if and only if,

$$\frac{\left(\hat{\mathbf{f}} \times (\mathbf{g}+\mathbf{h})\right) \cdot \hat{\mathbf{e}}_1}{(z_1^{2,2})(z_1^{2,1})} = 0. \tag{4.71}$$

Accordingly, the system configuration is degenerate and the global scale is ambiguous in the following cases:

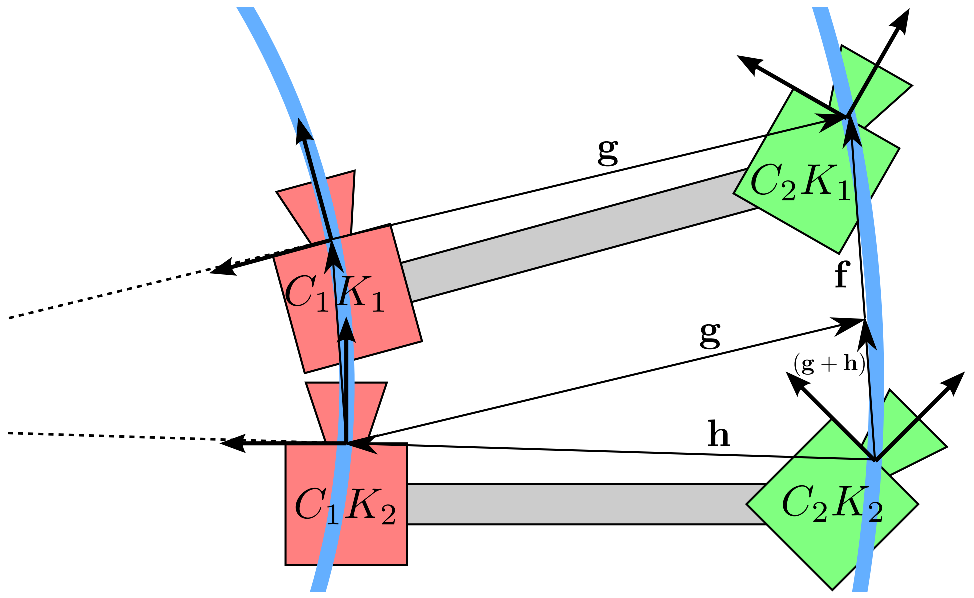


Figure 4.5: When the camera centres move in concentric circles with the common centre on the line formed by the camera centres, $\hat{\mathbf{f}} \times (\mathbf{g}+\mathbf{h}) = \mathbf{0}_{1\times 3}$, and as a result, the system is degenerate.

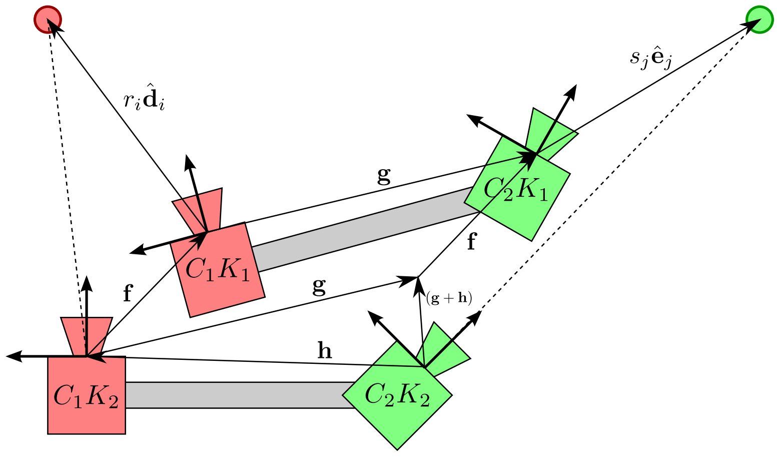


Figure 4.6: When the cluster motion ($\mathbf{f}$), baseline change ($\mathbf{g}+\mathbf{h}$), and point feature ($\hat{\mathbf{e}}_1$), are coplanar, the system is degenerate. These configurations include a completely planar system.

1.) $\left|z_1^{2,1}\right| \to \infty$ **or** $\left|z_1^{2,2}\right| \to \infty$ **:** The point feature is infinitely far from the second camera at either keyframe.

2.) $z_1^{2,1} = 0$ **or** $z_1^{2,2} = 0$ **:** The point feature is on the x-y plane of the second camera at either keyframe. However, this is not possible by the assumptions in the formulation.

3.) $\mathbf{t}_C = \mathbf{0}_{3\times 1}$ **:** The two cameras have the same optical centre.

4.) $\boldsymbol{\mathcal{R}}_K = \mathbf{I}_{3\times 3}$ **:** There is no rotation between the keyframes.

5.) $\mathbf{t}_K = \mathbf{0}_{3\times 1}$ **:** There is no translation between the keyframes.

6.) $\mathbf{t}_K \times (\boldsymbol{\mathcal{R}}_K - \mathbf{I}_{3\times 3})\, \mathbf{t}_C = \mathbf{0}_{3\times 1}$ **:** The cameras move in concentric circles with a common centre along the line through the two optical centres.

7.) $(\mathbf{t}_K \times (\boldsymbol{\mathcal{R}}_K - \mathbf{I}_{3\times 3})\, \mathbf{t}_C) \cdot \boldsymbol{\mathcal{R}}_K \boldsymbol{\mathcal{R}}_C \mathbf{p}_1^{2,1} = 0$ **:** The translations of the cameras and the point feature positions are all coplanar.

4.3 Observability of Motion and Structure

In this section, the cluster relative motion parameters are included in the system state vector. As a result, it is not necessary to assume that the motion increments of the individual cameras are known a priori. This is a more realistic scenario where all of the parameters are to be estimated simultaneously. This matches the framework detailed in Chapter 3. The main contribution of this section is to present the first analysis of the configurations leading to degeneracies in the iterative least-squares type optimization methods. It also provides a structured systematic method for identifying degenerate configurations for other sets of system parameters and measurements.

4.3.1 Previous Work

In the literature, analysis of degeneracies in the estimation of the full motion and structure for multicamera clusters has thus far focused on those associated with linearly solving the GEM (2.36) using the seventeen point algorithm [63]. Sturm [81] and Mouragnon *et al.* [58] discuss some degenerate cases when describing methods for solving using the GEM, but Kim and Kanade [44] provide the most complete analysis. The latter work will be summarized here.

Within the GEM, there are 18 unknown elements, up to a scale factor, within the two 3×3 matrices to be solved for, $\boldsymbol{\mathcal{R}}$ and $\mathbf{E}$. Therefore a linear homogeneous system can be created by vectorizing the elements of the matrices and stacking the constraints for 17 point feature correspondences into the measurement matrix $\mathbf{A}$, such that,

$$\mathbf{A}\begin{bmatrix}\mathbf{e}\\ \mathbf{r}\end{bmatrix} = \mathbf{0}_{17\times 1}, \tag{4.72}$$

where $\mathbf{e}$ and $\mathbf{r}$ are vectorized versions of the $\mathbf{E}$ and $\boldsymbol{\mathcal{R}}$ matrix elements, respectively.

Kim and Kanade show that the matrix $\mathbf{A}$ can be decomposed as,

$$\mathbf{A} = \mathbf{A}_d\mathbf{A}_c, \tag{4.73}$$

where $\mathbf{A}_d$ depends on the directions of the observation rays, while $\mathbf{A}_c$ depends only on the camera centres. It is then proven that the system is degenerate when the matrix $\mathbf{A}_c$ has a nontrivial null space. Subsequently, the following degenerate configurations for the generalized cameras are identified:

1. All of the observation rays pass through one common point before and after the motion.
2. The camera centres are on a line before and after the motion.
3. Each corresponding ray pair passes through the same local point before and after the motion.

The two-camera cluster at two keyframes with non-overlapping FOV studied in this chapter meets the last two criteria for degeneracy. Consequently, the seventeen point algorithm is not applicable to this system. This problem was noticed by Li *et al.* [50] and they modified the algorithm for use with non-overlapping clusters, but the subsequent degeneracy analysis has not been carried out.

More importantly, the degenerate configurations are specific to the linear method of estimation. In following section, the iterative least-squares solver from Chapter 3 is considered which operates on the nonlinear system model directly. The configurations for which the particular solver will fail are identified through analysis and differ from those of the linear algorithm discussed above.

4.3.2 Identification of Degenerate Configurations

To determine the necessary number of point features to observe, the number of equations must again be larger than or equal to the number of unknowns. Over the two keyframes, four measurements are added for each point feature, resulting in

$4(p+q)$ system equations. There are six parameters representing the relative pose of the two keyframes, and three position parameters for tracked each point feature. This constraint can be written algebraically as,

$$4(p+q) \geq 6 + 3(p+q) \tag{4.74}$$

$$p+q \geq 6. \tag{4.75}$$

This indicates that at least six point features are required to be able to solve for the system states. The minimal case of $p+q=6$ will be used in this analysis.

The resulting state vector, $\mathbf{x} \in \mathbb{R}^{24}$, is composed of the parameters for the six point features, along with the relative translation and orientation states for the cluster motion,

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix}, \tag{4.76}$$

where

$$\mathbf{x}_1 = [r_1, \dots, r_p, s_1, \dots, s_q]^\top \in \mathbb{R}^{p+q}, \tag{4.77}$$

are the depths to the point features,

$$\mathbf{x}_2 = [t_x, t_y, t_z, \omega_x, \omega_y, \omega_z]^\top \in \mathbb{R}^6, \tag{4.78}$$

are the position and orientation of the first keyframe with respect to the second keyframe and,

$$\mathbf{x}_3 = [\phi_1, \theta_1, \phi_2, \theta_2, \phi_3, \theta_3, \phi_4, \theta_4, \phi_5, \theta_5, \phi_6, \theta_6]^\top \in \mathbb{R}^{2(p+q)}, \tag{4.79}$$

are the bearings to the point features in their respective anchor keyframes. This state order has been specifically chosen in order to simplify the analysis of the degeneracies of the solution.

The relevant non-zero partial derivatives for the point feature location with respect to the states are as follows. The derivative of the point feature position in the observing camera frame with respect to the depth to the point feature are,

$$\frac{\partial \mathbf{p}_j^{1,2}}{\partial r_j} = \hat{\mathbf{d}}_j \tag{4.80}$$

$$\frac{\partial \mathbf{p}_j^{2,2}}{\partial s_j} = \boldsymbol{\mathcal{R}}_C^\top \hat{\mathbf{e}}_j. \tag{4.81}$$

The derivatives of the point position with respect to the keyframe translations in the two cameras are,

$$\frac{\partial \mathbf{p}_j^{1,2}}{\partial \mathbf{t}_K} = \mathbf{I}_{3\times 3} \tag{4.82}$$

$$\frac{\partial \mathbf{p}_j^{2,2}}{\partial \mathbf{t}_K} = \boldsymbol{\mathcal{R}}_C^\top, \tag{4.83}$$

while those with respect to the keyframe rotation in each camera are,

$$\frac{\partial \mathbf{p}_j^{1,2}}{\partial \boldsymbol{\omega}_K} = -\left[r_j \hat{\mathbf{d}}_j\right]_\times \tag{4.84}$$

$$\frac{\partial \mathbf{p}_j^{2,2}}{\partial \boldsymbol{\omega}_K} = -\boldsymbol{\mathcal{R}}_C^\top \left[s_j \hat{\mathbf{e}}_j + \mathbf{g}\right]_\times . \tag{4.85}$$

Finally, the derivatives of the point feature positions with respect to the bearing to the features in the anchor keyframe are,

$$\frac{\partial \mathbf{p}_j^{1,1}}{\partial \phi_j} = r_j \begin{bmatrix} \cos\phi_j \cos\theta_j \\ 0 \\ -\sin\phi_j \cos\theta_j \end{bmatrix} \tag{4.86}$$

$$\frac{\partial \mathbf{p}_j^{1,1}}{\partial \theta_j} = r_j \begin{bmatrix} -\sin\phi_j \sin\theta_j \\ -\cos\theta_j \\ -\cos\phi_j \sin\theta_j \end{bmatrix} \tag{4.87}$$

$$\frac{\partial \mathbf{p}_j^{2,1}}{\partial \phi_{p+j}} = s_j \begin{bmatrix} \cos\phi_{p+j} \cos\theta_{p+j} \\ 0 \\ -\sin\phi_{p+j} \cos\theta_{p+j} \end{bmatrix} \tag{4.88}$$

$$\frac{\partial \mathbf{p}_j^{2,1}}{\partial \theta_{p+j}} = s_j \begin{bmatrix} -\sin\phi_{p+j} \sin\theta_{p+j} \\ -\cos\theta_{p+j} \\ -\cos\phi_{p+j} \sin\theta_{p+j} \end{bmatrix} . \tag{4.89}$$

The measurement Jacobian that results from finding the partial derivatives of the measurements with respect to the states, has the following structure,

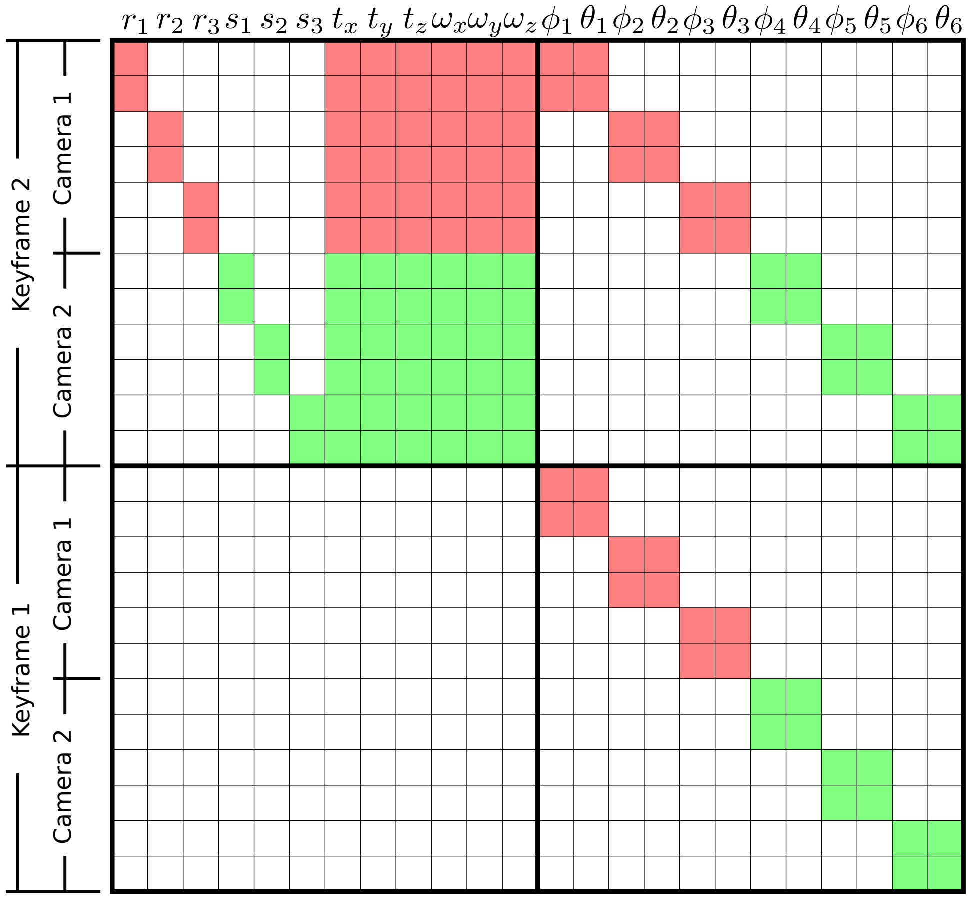
$$\mathbf{J} = \begin{bmatrix} \frac{\partial \boldsymbol{\varpi}_2(\mathbf{x})}{\partial [\mathbf{x}_1{}^\top \ \mathbf{x}_2{}^\top]^\top} & \frac{\partial \boldsymbol{\varpi}_2(\mathbf{x})}{\partial \mathbf{x}_3} \\ \frac{\partial \boldsymbol{\varpi}_1(\mathbf{x})}{\partial [\mathbf{x}_1{}^\top \ \mathbf{x}_2{}^\top]^\top} & \frac{\partial \boldsymbol{\varpi}_1(\mathbf{x})}{\partial \mathbf{x}_3} \end{bmatrix} \in \mathbb{R}^{24\times 24} \tag{4.90}$$

$$= \begin{bmatrix} \mathbf{Z}_2^{-1}\mathbf{J}_2 & \mathbf{Z}_2^{-1}\mathbf{J}_3 \\ \mathbf{0}_{12\times 12} & \mathbf{Z}_1^{-1}\mathbf{J}_1 \end{bmatrix} \tag{4.91}$$

where

$$\mathbf{Z}_k = \mathbf{\Phi}\left(\begin{bmatrix} \left[(z_1^{1,k})^2 \quad (z_1^{1,k})^2\right]^\top \\ \vdots \\ \left[(z_p^{1,k})^2 \quad (z_p^{1,k})^2\right]^\top \\ \left[(z_1^{2,k})^2 \quad (z_1^{2,k})^2\right]^\top \\ \vdots \\ \left[(z_q^{2,k})^2 \quad (z_q^{2,k})^2\right]^\top \end{bmatrix}\right) \in \mathbb{R}^{12\times 12}, \quad \text{for } k = 1, 2 \tag{4.92}$$

and $\mathbf{\Phi} : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ creates a diagonal matrix with the diagonal entries given by the elements of the vector argument. The sub-blocks of the Jacobian, $\mathbf{J}_1$, $\mathbf{J}_2$, $\mathbf{J}_3 \in \mathbb{R}^{12\times 12}$, relate the subsets of measurements to a subset of the system parameters.



r_1 r_2 r_3 s_1 s_2 s_3 t_x t_y t_z ω_x ω_y ω_z ϕ_1 θ_1 ϕ_2 θ_2 ϕ_3 θ_3 ϕ_4 θ_4 ϕ_5 θ_5 ϕ_6 θ_6
Keyframe 2
Camera 1
Camera 2
Keyframe 1
Camera 1
Camera 2

Figure 4.7: Non-zero element structure of Jacobian $\mathbf{J}$ for $p = 3$ (red) and $q = 3$ (green).

The structure of $\mathbf{J}$ for an example system with $p = q = 3$ is shown in Figure 4.7. For different combinations of $p + q = 6$, the elements which contain non-zero entries are the same as for this chosen system, only the proportion of the features in each camera would change. Since **J** is a square matrix when $p+q=6$, a non-zero determinant indicates that it is full rank. As a result, the degenerate configurations are those for which the system measurement Jacobian **J** has a zero determinant,

$$\det(\mathbf{J}) = 0 \iff \operatorname{rank}(\mathbf{J}) < 6 + 3(p+q). \tag{4.93}$$

Analytically solving for the determinant of the full system matrix **J** directly is difficult, however, the underlying block-triangular structure of **J** allows the determinant to be simplified,

$$\begin{aligned}\det(\mathbf{J}) &= \det\left(\mathbf{Z}_2^{-1}\mathbf{J}_2\right)\det\left(\mathbf{Z}_1^{-1}\mathbf{J}_1\right) &(4.94)\\ &= \det\left(\mathbf{Z}_2^{-1}\right)\det\left(\mathbf{Z}_1^{-1}\right)\det(\mathbf{J}_2)\det(\mathbf{J}_1) &(4.95)\\ &= \frac{\det(\mathbf{J}_2)\det(\mathbf{J}_1)}{\det(\mathbf{Z}_2)\det(\mathbf{Z}_1)}. &(4.96)\end{aligned}$$

Now that the determinant consists of a series of terms, the roots of the full system can be found by investigating the roots of each term individually. The total set of degenerate configurations is the union of all of the degenerate configurations for the determinant terms. Each term will be investigated in its own subsection below.

Determinant of $\mathbf{Z}_k$

The denominator of (4.94) consists of products of the z-component of the point feature positions in the respective camera frames at each keyframe,

$$\det(\mathbf{Z}_k) = \left(\prod_{j=1}^{p} z_j^{1,k}\right)^4 \left(\prod_{j=1}^{q} z_j^{2,k}\right)^4 \quad k = 1, 2. \tag{4.97}$$

Accordingly, the determinant goes to zero when any of the feature points are infinitely far from the respective camera, in the z-axis direction, at either keyframe.

Determinant of $\mathbf{J}_1$

The Jacobian block, $\mathbf{J}_1$, relates the set of point feature measurements in the first keyframe for both cameras, to the feature bearing states. It is a 12×12 block-diagonal matrix composed of 2×2 sub-blocks for each point feature,

$$\mathbf{J}_1 = \boldsymbol{\Psi}\left(\mathbf{J}_{1,1}, \mathbf{J}_{1,2}, \mathbf{J}_{1,3}, \mathbf{J}_{1,4}, \mathbf{J}_{1,5}, \mathbf{J}_{1,6}\right), \tag{4.98}$$

where $\mathbf{\Psi}\left(\right)$ creates a block-diagonal matrix with the sub-blocks given by the arguments, and

$$\mathbf{J}_{1,j} = \begin{cases} r_j^2 \begin{bmatrix} \cos(\theta_j)^2 & 0 \\ -\sin(\phi_j)\cos(\theta_j)\sin(\theta_j) & -\cos(\phi_j) \end{bmatrix} & \forall j = 1, \ldots, p \\ s_{(j-p)}^2 \begin{bmatrix} \cos(\theta_j)^2 & 0 \\ -\sin(\phi_j)\cos(\theta_j)\sin(\theta_j) & -\cos(\phi_j) \end{bmatrix} & \forall j = p+1, \ldots, p+q \end{cases} \tag{4.99}$$

such that

$$\det\left(\mathbf{J}_{1,j}\right) = \begin{cases} -r_j^4 \cos(\phi_j)\cos(\theta_j)^2 & \forall j = 1, \ldots, p \\ -s_{(j-p)}^4 \cos(\phi_j)\cos(\theta_j)^2 & \forall j = p+1, \ldots, p+q \end{cases} \tag{4.100}$$

$$= \begin{cases} -r_j^3 \cos(\theta_j) z_j^{1,1} & \forall j = 1, \ldots, p \\ -s_{(j-p)}^3 \cos(\theta_j) z_{(j-p)}^{2,1} & \forall j = p+1, \ldots, p+q \end{cases}. \tag{4.101}$$

The $\mathbf{J}_1$ block from the example system is shown in Figure 4.8. The determinant of the block $\mathbf{J}_1$ is the product of the determinants of the diagonal blocks,

$$\det\left(\mathbf{J}_1\right) = \prod_{j=1}^{p+q} \det\left(\mathbf{J}_{1,j}\right) \tag{4.102}$$

$$= (-1)^{(p+q)} \left(\prod_{j=1}^{p} r_j^3 \cos(\theta_j) z_j^{1,1}\right) \left(\prod_{j=1}^{q} s_j^3 \cos(\theta_{p+j}) z_j^{2,1}\right) \tag{4.103}$$

$$= (-1)^{(p+q)} \left(\prod_{j=1}^{p} z_j^{1,1}\right) \left(\prod_{j=1}^{q} z_j^{2,1}\right) \left(\prod_{j=1}^{p} r_j^3 \cos(\theta_j)\right) \left(\prod_{j=1}^{q} s_j^3 \cos(\theta_{p+j})\right). \tag{4.104}$$

As with the previous scale-only system in Section 4.2, this determinant is zero when any of the features lie on its respective camera x-y plane at the first keyframe. However, this case is not possible due to the assumption that the point features are always in front of the respective cameras. Furthermore, with the constraints on the point feature parameters,

$$\det\left(\mathbf{J}_1\right) \neq 0. \tag{4.105}$$

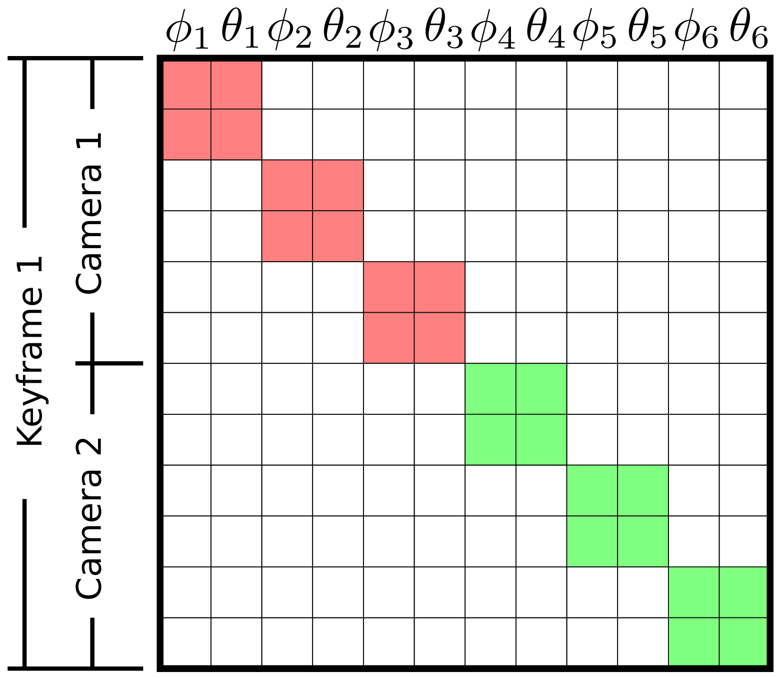

ϕ_1 θ_1 ϕ_2 θ_2 ϕ_3 θ_3 ϕ_4 θ_4 ϕ_5 θ_5 ϕ_6 θ_6
Keyframe 1
Camera 1
Camera 2

Figure 4.8: Structure of the Jacobian block $\mathbf{J}_1$ for $p = 3$ and $q = 3$.

Determinant of $\mathbf{J}_2$

The final determinant term involving $\mathbf{J}_2$ is not as straight-forward to decompose. The matrix $\mathbf{J}_2$ is computed as,

$$\mathbf{J}_2 = \begin{bmatrix} \begin{bmatrix} \hat{\mathbf{j}}^\top \\ -\hat{\mathbf{i}}^\top \end{bmatrix} \left[\mathbf{p}_1^{1,2}\right]_\times \mathbf{G}_1 \\ \vdots \\ \begin{bmatrix} \hat{\mathbf{j}}^\top \\ -\hat{\mathbf{i}}^\top \end{bmatrix} \left[\mathbf{p}_p^{1,2}\right]_\times \mathbf{G}_p \\ \begin{bmatrix} \hat{\mathbf{j}}^\top \\ -\hat{\mathbf{i}}^\top \end{bmatrix} \left[\mathbf{p}_1^{2,2}\right]_\times \mathbf{G}_{(p+1)} \\ \vdots \\ \begin{bmatrix} \hat{\mathbf{j}}^\top \\ -\hat{\mathbf{i}}^\top \end{bmatrix} \left[\mathbf{p}_q^{2,2}\right]_\times \mathbf{G}_{(p+q)} \end{bmatrix} \tag{4.106}$$

where

$$\mathbf{G}_j = \begin{cases} \begin{bmatrix} \mathbf{0}_{3\times(j-1)} & \hat{\mathbf{d}}_j & \mathbf{0}_{3\times(p+q-j)} & \mathbf{I}_{3\times3} & -\left[r_j\hat{\mathbf{d}}_j\right]_\times \end{bmatrix} & \forall j = 1,\ldots,p \\ \boldsymbol{\mathcal{R}}_C^\top \begin{bmatrix} \mathbf{0}_{3\times(j-1)} & \hat{\mathbf{e}}_{(j-p)} & \mathbf{0}_{3\times(p+q-j)} & \mathbf{I}_{3\times3} & -\left[s_{(j-p)}\hat{\mathbf{e}}_{(j-p)} + \mathbf{g}\right]_\times \end{bmatrix} & \forall j = p+1,\ldots,p+q. \end{cases} \tag{4.107}$$

It relates the set of observations of the point features in the second keyframe to the point feature depth and keyframe pose states. The example system $\mathbf{J}_2$ block is shown in Figure 4.9.

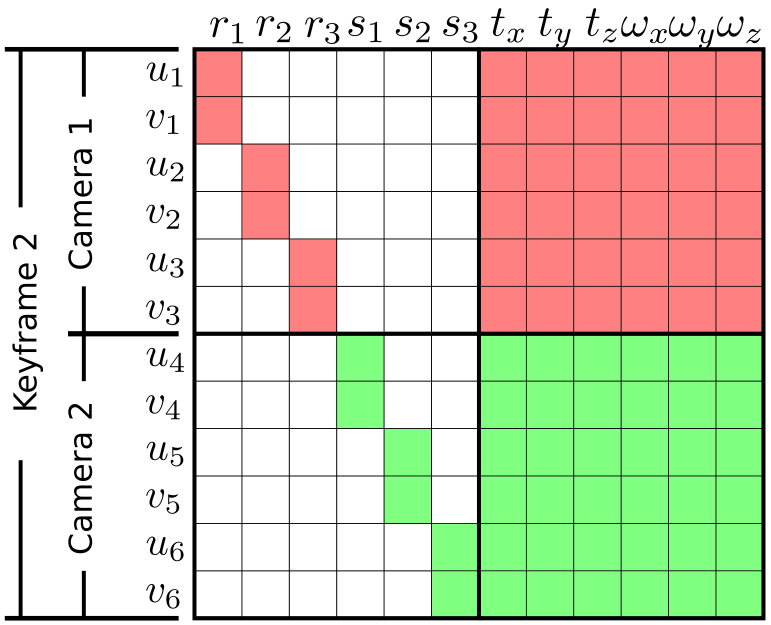

r_1 r_2 r_3 s_1 s_2 s_3 t_x t_y t_z ω_x ω_y ω_z
u_1 v_1 u_2 v_2 u_3 v_3 u_4 v_4 u_5 v_5 u_6 v_6
Camera 1
Camera 2
Keyframe 2

Figure 4.9: Structure of the Jacobian block $\mathbf{J}_2$ for $p = 3$ and $q = 3$.

The first $p + q$ columns of $\mathbf{J}_2$ describe the change in the point feature image location with respect to a change in the respective feature depth. A column will be zero when the motion of the camera is collinear with the initial bearing to the point feature in the respective anchor camera frame,

$$\hat{\mathbf{d}}_j \times \mathbf{f} = \mathbf{0}_{3\times1}, \quad \forall j = 1, \ldots, p \tag{4.108}$$

and

$$\boldsymbol{\mathcal{R}}_C^{\top} \left(\hat{\mathbf{e}}_j \times (\mathbf{f} + \mathbf{g} + \mathbf{h})\right) = \mathbf{0}_{3\times1}, \quad \forall j = 1, \ldots, q. \tag{4.109}$$

This results in the depth parameters having no effect on the measurement since they move the feature along the measured bearing at each keyframe, and therefore, they cannot be determined uniquely. With a column of all zeros, the matrix $\mathbf{J}_2$ is rank deficient and the solution is degenerate for this set of point features.

When at least one of the elements in each of the columns is non-zero, the matrix $\mathbf{J}_2$ can be manipulated to determine the rest of the degenerate configurations. Through row operations, a rank-equivalent upper-block triangular matrix $\mathbf{K}$ is derived from $\mathbf{J}_2$ as,

$$\mathbf{K} = \mathbf{LMJ}_2, \tag{4.110}$$

where $\mathbf{L}$ is a matrix representing elementary row operations to stack the odd-numbered rows on top of the even-numbered rows, then subtract the odd-numbered rows from the even-numbered rows. As a result, $\det(\mathbf{L}) = -1$. The matrix $\mathbf{M}$ scales the corresponding rows so that the subtraction in $\mathbf{L}$ results in the lower-left block being all zeros,

$$\mathbf{M} = \mathbf{\Phi}\left(\begin{bmatrix} \hat{\mathbf{i}} \cdot (\hat{\mathbf{d}}_1 \times \mathbf{f}) \\ -\hat{\mathbf{j}} \cdot (\hat{\mathbf{d}}_1 \times \mathbf{f}) \\ \vdots \\ \hat{\mathbf{i}} \cdot (\hat{\mathbf{d}}_p \times \mathbf{f}) \\ -\hat{\mathbf{j}} \cdot (\hat{\mathbf{d}}_p \times \mathbf{f}) \\ \hat{\mathbf{n}}_x \cdot (\hat{\mathbf{e}}_1 \times (\mathbf{f}+\mathbf{g}+\mathbf{h})) \\ -\hat{\mathbf{n}}_y \cdot (\hat{\mathbf{e}}_1 \times (\mathbf{f}+\mathbf{g}+\mathbf{h})) \\ \vdots \\ \hat{\mathbf{n}}_x \cdot (\hat{\mathbf{e}}_q \times (\mathbf{f}+\mathbf{g}+\mathbf{h})) \\ -\hat{\mathbf{n}}_y \cdot (\hat{\mathbf{e}}_q \times (\mathbf{f}+\mathbf{g}+\mathbf{h})) \end{bmatrix}\right). \tag{4.111}$$

This matrix has a determinant of the form,

$$\det(\mathbf{M}) = (-1)^{p+q}\left(\prod_{j=1}^{p}\left(\hat{\mathbf{i}} \cdot (\hat{\mathbf{d}}_j \times \mathbf{f})\right)\left(\hat{\mathbf{j}} \cdot (\hat{\mathbf{d}}_j \times \mathbf{f})\right)\right) \left(\prod_{j=1}^{q}(\hat{\mathbf{n}}_x \cdot (\hat{\mathbf{e}}_j \times (\mathbf{f}+\mathbf{g}+\mathbf{h})))(\hat{\mathbf{n}}_y \cdot (\hat{\mathbf{e}}_j \times (\mathbf{f}+\mathbf{g}+\mathbf{h})))\right). \tag{4.112}$$

The matrix $\mathbf{M}$ contains the non-zero elements of the first $p+q$ columns of $\mathbf{J}_2$. As long as none of the first $p+q$ columns of $\mathbf{J}_2$ are all zeros, but the i^{th} diagonal element of $\mathbf{M}$ is zero, the following procedure can be used to create a non-singular $\mathbf{M}$ and $\mathbf{L}$, resulting in the same structure for $\mathbf{K}$.

- Set $j := \lfloor \frac{i-1}{2} \rfloor$, $k := i \bmod 2$.
- Replace the i^{th} diagonal element in $\mathbf{M}$ with 1.
- Set the $(p+q+j+1, 2j+1)$ element of $\mathbf{L}$ to 0.
- If $k = 1$, swap columns $2j+1$ and $2j+2$ of $\mathbf{L}$.

As a result of the swapping of some columns, the determinant $\det(\mathbf{L})$ will be ± 1.

After the matrix manipulations, the resulting matrix $\mathbf{K}$ has the upper-block triangular form,

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_1 & \mathbf{K}_3 \\ \mathbf{0} & \mathbf{K}_2 \end{bmatrix}, \tag{4.113}$$

which is shown for the example system in Figure 4.10.

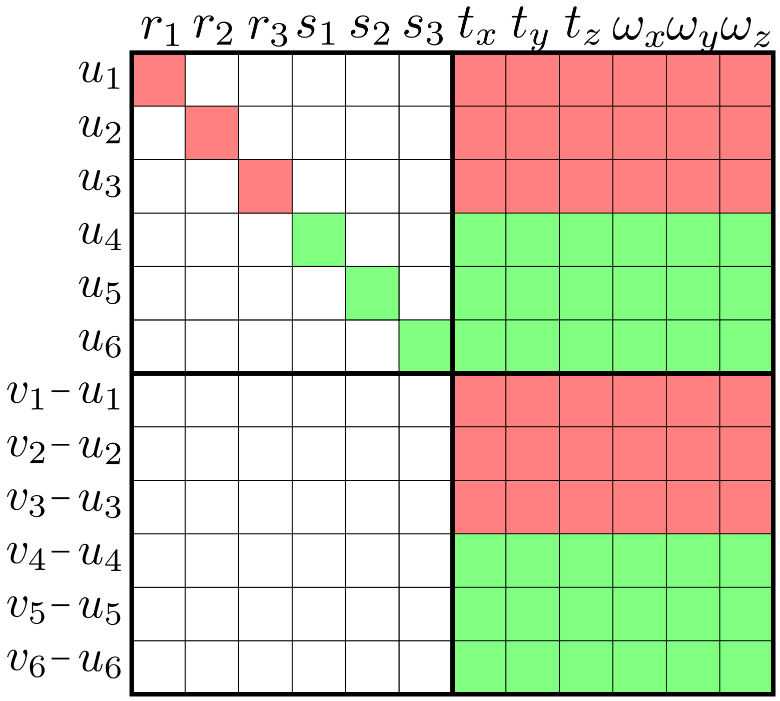


Figure 4.10: Structure of the Jacobian block $\mathbf{K}$ for $p = 3$ and $q = 3$.

Taking the determinant of (4.110) results in,

$$\det(\mathbf{K}) = \det(\mathbf{L}\mathbf{M}\mathbf{J}_2) \tag{4.114}$$

$$\det(\mathbf{K}) = \det(\mathbf{L})\det(\mathbf{M})\det(\mathbf{J}_2) \tag{4.115}$$

$$\det(\mathbf{K}_1)\det(\mathbf{K}_2) = \det(\mathbf{L})\det(\mathbf{M})\det(\mathbf{J}_2)\,. \tag{4.116}$$

The matrix $\mathbf{K}_1$ has the following structure,

$$\mathbf{K}_1 = \mathbf{\Phi}\left(\begin{bmatrix} -\left(\hat{\mathbf{i}}\cdot(\hat{\mathbf{d}}_1\times\mathbf{f})\right)\left(\hat{\mathbf{j}}\cdot(\hat{\mathbf{d}}_1\times\mathbf{f})\right) \\ \vdots \\ -\left(\hat{\mathbf{i}}\cdot(\hat{\mathbf{d}}_p\times\mathbf{f})\right)\left(\hat{\mathbf{j}}\cdot(\hat{\mathbf{d}}_p\times\mathbf{f})\right) \\ -\left(\hat{\mathbf{n}}_x\cdot(\hat{\mathbf{e}}_1\times(\mathbf{f}+\mathbf{g}+\mathbf{h}))\right)\left(\hat{\mathbf{n}}_y\cdot(\hat{\mathbf{e}}_1\times(\mathbf{f}+\mathbf{g}+\mathbf{h}))\right) \\ \vdots \\ -\left(\hat{\mathbf{n}}_x\cdot(\hat{\mathbf{e}}_q\times(\mathbf{f}+\mathbf{g}+\mathbf{h}))\right)\left(\hat{\mathbf{n}}_y\cdot(\hat{\mathbf{e}}_q\times(\mathbf{f}+\mathbf{g}+\mathbf{h}))\right) \end{bmatrix}\right), \tag{4.117}$$

such that the determinant is,

$$\det(\mathbf{K}_1) = (-1)^{p+q}\left(\prod_{j=1}^{p}\left(\hat{\mathbf{i}}\cdot(\hat{\mathbf{d}}_j\times\mathbf{f})\right)\left(\hat{\mathbf{j}}\cdot(\hat{\mathbf{d}}_j\times\mathbf{f})\right)\right) \tag{4.118}$$
$$\left(\prod_{j=1}^{q}\left(\hat{\mathbf{n}}_x\cdot(\hat{\mathbf{e}}_j\times(\mathbf{f}+\mathbf{g}+\mathbf{h}))\right)\left(\hat{\mathbf{n}}_y\cdot(\hat{\mathbf{e}}_j\times(\mathbf{f}+\mathbf{g}+\mathbf{h}))\right)\right)$$

which, comparing to (4.112), means $\det(\mathbf{K}_1) = \det(\mathbf{M})$. Substituting back into (4.116),

$$\det(\mathbf{M})\det(\mathbf{K}_2) = \det(\mathbf{L})\det(\mathbf{M})\det(\mathbf{J}_2)\,. \tag{4.119}$$

Now, since $\det(\mathbf{M}) \neq 0$, the determinant of $\mathbf{J}_2$ can be written,

$$\det(\mathbf{J}_2) = \frac{\det(\mathbf{M})\det(\mathbf{K}_2)}{\det(\mathbf{M})\det(\mathbf{L})} \tag{4.120}$$
$$= \frac{\det(\mathbf{K}_2)}{\det(\mathbf{L})}. \tag{4.121}$$

Therefore, the determinant of the 12×12 matrix $\mathbf{J}_2$ is the same as that of the 6×6 matrix $\mathbf{K}_2$, up to the sign of $\det(\mathbf{L})$.

The matrix $\mathbf{K}_2$ can be further decomposed by factoring the z-component of each point feature observation at the second keyframe, out of each row,

$$\mathbf{K}_2 = \mathbf{\Phi}\left([z_1^{1,2},\ldots,z_p^{1,2},z_1^{2,2},\ldots,z_q^{2,2}]\right)\mathbf{K}_4 \tag{4.122}$$

where

$$\mathbf{K}_4 = \begin{bmatrix} \left(\mathbf{f}\times\hat{\mathbf{d}}_1\right)^\top & \left(r_1\hat{\mathbf{d}}_1\times\left(\mathbf{f}\times\hat{\mathbf{d}}_1\right)\right)^\top \\ \vdots & \vdots \\ \left(\mathbf{f}\times\hat{\mathbf{d}}_p\right)^\top & \left(r_p\hat{\mathbf{d}}_p\times\left(\mathbf{f}\times\hat{\mathbf{d}}_p\right)\right)^\top \\ \left((\mathbf{f}+\mathbf{g}+\mathbf{h})\times\hat{\mathbf{e}}_1\right)^\top & \left((\mathbf{g}+s_1\hat{\mathbf{e}}_1)\times((\mathbf{f}+\mathbf{g}+\mathbf{h})\times\hat{\mathbf{e}}_1)\right)^\top \\ \vdots & \vdots \\ \left((\mathbf{f}+\mathbf{g}+\mathbf{h})\times\hat{\mathbf{e}}_q\right)^\top & \left((\mathbf{g}+s_q\hat{\mathbf{e}}_q)\times((\mathbf{f}+\mathbf{g}+\mathbf{h})\times\hat{\mathbf{e}}_q)\right)^\top \end{bmatrix} \tag{4.123}$$

and the determinant of $\mathbf{K}_2$ is,

$$\det(\mathbf{K}_2) = \left(\prod_{j=1}^{p} z_j^{1,2}\right)\left(\prod_{j=1}^{q} z_j^{2,2}\right)\det(\mathbf{K}_4)\,. \tag{4.124}$$

The determinant of $\mathbf{K}_2$ is zero if and only if $\det(\mathbf{K}_4) = 0$ since none of the point features lie on their respective camera x-y planes at the second keyframe, by assumption.

The structure of matrix $\mathbf{K}_4$ for the example system is shown in Figure 4.11. On closer inspection of the matrix $\mathbf{K}_4{}^\top$, the columns represent the Plücker coordinates of six lines in $\mathbb{R}^3$,

$$\mathbf{K}_4{}^\top = \begin{bmatrix} \boldsymbol{\ell}_1 & \boldsymbol{\ell}_2 & \boldsymbol{\ell}_3 & \boldsymbol{\ell}_4 & \boldsymbol{\ell}_5 & \boldsymbol{\ell}_6 \end{bmatrix}, \tag{4.125}$$

where the line vectors

$$\boldsymbol{\ell}_j = \begin{bmatrix} \mathbf{q}_j \\ \mathbf{q}'_j \end{bmatrix} \in \mathbb{R}^6 \tag{4.126}$$

$$= \begin{cases} \begin{bmatrix} \mathbf{f} \times \hat{\mathbf{d}}_j \\ r_j \hat{\mathbf{d}}_j \times \left(\mathbf{f} \times \hat{\mathbf{d}}_j\right) \end{bmatrix} & \forall j = 1, \ldots, p \\ \begin{bmatrix} (\mathbf{f} + \mathbf{g} + \mathbf{h}) \times \hat{\mathbf{e}}_{(j-p)} \\ (\mathbf{g} + s_{(j-p)} \hat{\mathbf{e}}_{(j-p)}) \times \left((\mathbf{f} + \mathbf{g} + \mathbf{h}) \times \hat{\mathbf{e}}_{(j-p)}\right) \end{bmatrix} & \forall j = p+1, \ldots, p+q, \end{cases} \tag{4.127}$$

each satisfy the Grassmann-Plücker relation [88] for the coordinates of a line,

$$\mathbf{q}_j \cdot \mathbf{q}'_j = 0. \tag{4.128}$$

The coordinates for the lines in the columns of $\mathbf{K}_4{}^\top$ satisfy this constraint trivially for the first camera,

$$\left(\mathbf{f} \times \hat{\mathbf{d}}_i\right) \cdot \left(r_i \hat{\mathbf{d}}_i \times \left(\mathbf{f} \times \hat{\mathbf{d}}_i\right)\right) \tag{4.129}$$

$$= r_i \hat{\mathbf{d}}_i \cdot \underbrace{\left(\left(\mathbf{f} \times \hat{\mathbf{d}}_i\right) \times \left(\mathbf{f} \times \hat{\mathbf{d}}_i\right)\right)}_{=\mathbf{0}_{3\times 1}} \tag{4.130}$$

$$= 0, \tag{4.131}$$

and for the second camera,

$$((\mathbf{f} + \mathbf{g} + \mathbf{h}) \times \hat{\mathbf{e}}_j) \cdot ((\mathbf{g} + s_j \hat{\mathbf{e}}_j) \times ((\mathbf{f} + \mathbf{g} + \mathbf{h}) \times \hat{\mathbf{e}}_j)) \tag{4.132}$$

$$= (\mathbf{g} + s_j \hat{\mathbf{e}}_j) \cdot \underbrace{(((\mathbf{f} + \mathbf{g} + \mathbf{h}) \times \hat{\mathbf{e}}_j) \times ((\mathbf{f} + \mathbf{g} + \mathbf{h}) \times \hat{\mathbf{e}}_j))}_{=\mathbf{0}_{3\times 1}} \tag{4.133}$$

$$= 0. \tag{4.134}$$

The first three columns in $\mathbf{K}_4$ are a stack of cross products. When they all have a common collinear vector operand, the resulting row vectors are all coplanar, in a plane with the normal defined by the collinear vector operand. In this case,

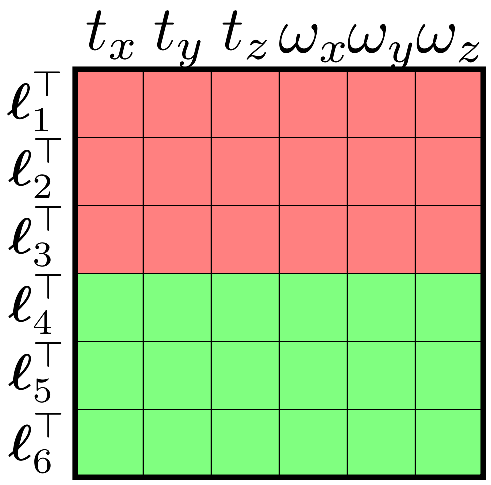

t_x t_y t_z ω_x ω_y ω_z
$\boldsymbol{\ell}_1^\top$
$\boldsymbol{\ell}_2^\top$
$\boldsymbol{\ell}_3^\top$
$\boldsymbol{\ell}_4^\top$
$\boldsymbol{\ell}_5^\top$
$\boldsymbol{\ell}_6^\top$

Figure 4.11: Structure of the Jacobian block $\mathbf{K}_4$ for $p = 3$ and $q = 3$.

the matrix $\mathbf{K}_4$ will have less than full rank. In the two-camera cluster system, the operands $\mathbf{f}$ and $\mathbf{f} + \mathbf{g} + \mathbf{h}$ are collinear when,

$$\mathbf{f} \times (\mathbf{f} + \mathbf{g} + \mathbf{h}) = \mathbf{f} \times (\mathbf{g} + \mathbf{h}) = \mathbf{0}_{3\times 1}, \tag{4.135}$$

and the operands can be written as a scalar multiple of one another,

$$\mathbf{f} + \mathbf{g} + \mathbf{h} = \alpha \mathbf{f}, \quad \alpha \in \mathbb{R}, \tag{4.136}$$

so that the first three columns of the matrix $\mathbf{K}_4$ have degenerate rank,

$$\operatorname{rank}\left(\begin{bmatrix} \left(\mathbf{f} \times \hat{\mathbf{d}}_1\right)^\top \\ \vdots \\ \left(\mathbf{f} \times \hat{\mathbf{d}}_p\right)^\top \\ (\alpha\mathbf{f} \times \hat{\mathbf{e}}_1)^\top \\ \vdots \\ (\alpha\mathbf{f} \times \hat{\mathbf{e}}_q)^\top \end{bmatrix}\right) \leq 2 < 3. \tag{4.137}$$

This shows that the concentric circle degeneracy (4.135), determined in the scale-only system studied in Section 4.2, remains in the full system estimation.

For the six point features, p and q must be selected such that $p + q = 6$. It is apparent from (4.123) by the same logic as above that when all of the features are observed by one of the cameras, that is, $p = 6$ and $q = 0$, or $p = 0$ and $q = 6$, the first three columns are linearly dependent since all of the rows are again coplanar. Therefore, at least one point feature must be observed in each camera.

When the determinant of the 6×6 matrix $\mathbf{K}_4$ is zero, the overall system is degenerate. The zero determinant expression,

$$\det(\mathbf{K}_4) = 0, \tag{4.138}$$

defines an implicit equation of a quintic surface in terms of the cluster translation $\mathbf{t}_K$, defined by the motion and structure values. An example of the fifth-order surface for a system with $p = 3$, $q = 3$, and a set of manually chosen system parameters is shown in 4.12. The magenta line along the surface represents the concentric circles condition, $\mathbf{f} \times (\mathbf{g} + \mathbf{h}) = \mathbf{0}_{3\times1}$.

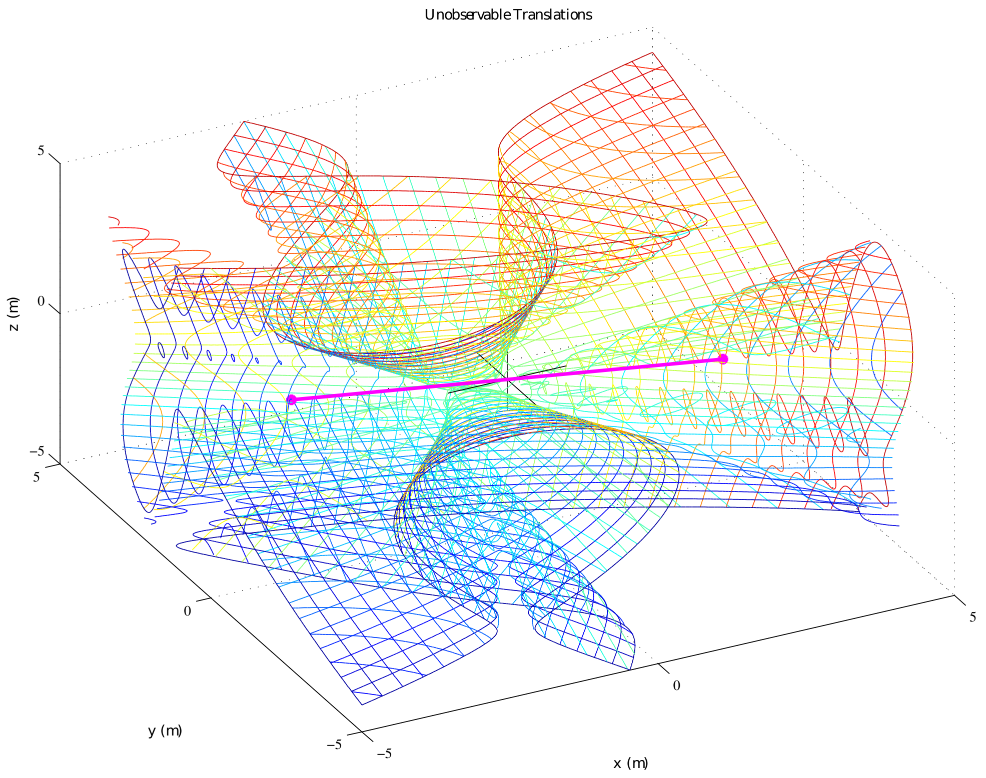


Figure 4.12: An example of the fifth-order surface in 3-space defining the cluster keyframe translations, $\mathbf{t}_K$, leading to system degeneracy.

The specific shape of the surface is a function of the positions of the set of six point features, as well as the cluster geometry and the relative keyframe rotation. The zero translation lies on the surface, as well as the transformations which lead to collinear motion of the camera centres with the anchor keyframe bearings of the point features.

Overall Determinant of J

Substituting back into (4.94), the overall determinant for the measurement Jacobian is,

$$\det(\mathbf{J}) = \left(\frac{\left(\prod_{j=1}^{p} r_j^3 \cos(\theta_j)\right)\left(\prod_{j=1}^{q} s_j^3 \cos(\theta_{p+j})\right)}{\left(\prod_{j=1}^{p} z_j^{1,1}\right)^3 \left(\prod_{j=1}^{q} z_j^{2,1}\right)^3 \left(\prod_{j=1}^{p} z_j^{1,2}\right)^3 \left(\prod_{j=1}^{q} z_j^{2,2}\right)^3} \right) \frac{\det(\mathbf{K}_4)}{\det(\mathbf{L})}. \tag{4.139}$$

The analysis in this section, the determinant of the 24×24 matrix, $\mathbf{J}$, has been reduced to a determinant of a 6×6 matrix in (4.139). This expression goes to zero, if and only if,

$$\frac{\det(\mathbf{K}_4)}{\left(\prod_{j=1}^{p} z_j^{1,1}\right)\left(\prod_{j=1}^{q} z_j^{2,1}\right)\left(\prod_{j=1}^{p} z_j^{1,2}\right)\left(\prod_{j=1}^{q} z_j^{2,2}\right)} = 0. \tag{4.140}$$

That is, the system is degenerate in all of the previously identified configurations in Section 4.2, plus the additional cases:

8.) All of the point features are observed by only the first camera at both keyframes, or all are observed by only the second camera at both keyframes.

9.) The translation of the cluster lies on the quintic surface defined by $\det(\mathbf{K}_4)$. This includes the cases of no translation, no rotation, motion resulting in the camera centres moving in concentric circles with the circle centre lying on the line through both camera centres, and motion of either camera in the direction of one of its point features as observed at the first keyframe. It also includes a family of other, more complex motions that prevent the estimator from determining a unique solution.

Detecting Degenerate Configurations

The analysis presented here identifies degenerate configurations of the two-camera cluster system observing point features over two keyframes, such that the system is degenerate using an Bundle Adjustment method or recursive filter. It is possible to detect these degenerate configurations during the operation of the estimator by quickly evaluating each of the individual determinant terms and looking for the expressions to be at, or approaching, zero.

In a practical system, it is reasonable to assume that more than six point features will be tracked within each keyframe. It is therefore unlikely that when the relative motion has the property,

$$\mathbf{f} \times (\mathbf{g} + \mathbf{h}) \neq \mathbf{0}_{1\times 3}, \tag{4.141}$$

the first three columns of $\mathbf{K}_4$ will be rank-deficient. When the motion is near degenerate, however, the matrix $\mathbf{K}_4$ will be ill-conditioned. This could lead to poor estimator performance if the motion is not sufficiently far from degenerate.

For a system with multiple keyframes which observe the same set of point features, it is sufficient to check each of the keyframes against only one in the set since the property is transitive. If there is at least one pair of keyframes in the target model for which the relative configuration is not degenerate, then the collective system can be solved, including the global scale.

The test for degenerate configurations is vital in the keyframe selection process to ensure that each added keyframe helps to accurately constrain the system solution, including the global scale. If a keyframe with a degenerate configuration is added to the target model, the optimization process in Chapter 3 will fail to determine a unique solution to the SLAM problem. The use of the relative degeneracy as a metric for the decision of when to add a keyframe is left as a possible future direction for research.

4.4 Summary

This chapter presented an analysis of the degenerate configurations of the relative cluster motion and point feature constellations for an optimization process based on minimizing a least-squares cost function with respect to the image error. The configurations in which the solution is ambiguous and non-unique are identified for a two-camera cluster for two cases: each camera is able to determine its own local motion increment without scale and the estimator must estimate the scale by combining them; and the full SLAM optimization where all of the relative pose parameters and the target model point feature positions are estimated. Both scenarios share the common degeneracy when the centres of the cameras move in concentric circles with the common centre on the line formed by the camera centres. This includes motions with no relative rotation, which are motions along concentric circles with infinite radius.

Future work will focus on extending the analysis to the case of more than six feature points, more than two cameras, the case when a point feature is observed by more than one camera over time, and the proper use of the analysis results as a metric for keyframe selection. More details will be presented in Chapter 6.

Chapter 5

Experiments

The algorithms and analysis proposed in this thesis were implemented and verified in experiment. This chapter presents a description of the software implementation of the algorithms, the design and construction of an actual four-camera cluster in hardware mounted on a quadrotor aerial robot, and the estimator performance under a set of relative motion and target structure profiles, including degenerate motions, and compared against high-accuracy ground truth measurements collected from an optical Indoor Positioning System (IPS).

5.1 Software Implementation

The software implementation of the algorithm described in Chapter 3 was accomplished, in collaboration with Adam Harmat at McGill University, by modifying the Multi-Camera Parallel Tracking and Mapping (MCPTAM) software by Harmat *et al.* [30]. MCPTAM itself, is an enhanced version of the Parallel Tracking and Mapping (PTAM) monocular SLAM system from Klein and Murray [45], adapted to work with multiple cameras consisting of overlap within their FOV. A summary of the capabilities of the PTAM and MCPTAM algorithms is provided in Appendix B to serve as a baseline for the changes made for the proposed implementation in this thesis.

5.1.1 Non-overlapping MCPTAM

The previous MCPTAM algorithm requires at least a small amount of FOV overlap between two of the component cameras in order to initialize and operate successfully. For this thesis, the algorithm was modified to use the novel parameterization and initialization methods described in Sections 3.2.2 and 3.3.3, to allow it to work with completely non-overlapping FOV camera cluster configurations.

The representation of the target model point features were changed to be anchored in the first camera keyframe coordinate frame in which they are observed and the point feature update was changed to incorporate the spherical transformation from Section 3.2.2. To support this change, the calculation of the Jacobian was modified to reflect the parameterization changes. Furthermore, the temporary keyframe mechanism at the current cluster pose was added to the BA optimization.

With the changes to the software framework, the system is now able to be initialized using the first set of point feature image measurements, and the target model converges to an accurately-scaled solution when the motion is not degenerate. The pose tracking process is able to run at frame-rate and produce accurate pose estimates for the current cluster pose within the target model.

5.2 Experimental Setup

Four cameras were rigidly attached to a small quadrotor aerial robot, shown in Figure 5.1. The selected cluster configuration is shown in Figure 5.2. Two spherical cameras with ultra-wide FOV of greater than 180 degrees are mounted on either side of the quadrotor looking outwards. Two more cameras are mounted under the helicopter body, one facing directly forwards and the other facing down and back. The second two cameras have a smaller FOV of approximately 150 degrees. The individual cameras are intrinsically calibrated using the Taylor camera model [68]. The extrinsic calibration for the camera cluster was performed using the method described in Appendix C.

All four cameras have a pixel resolution of 752×480, and are synchronously triggered to capture images at the same instant in time. Each camera is connected to the onboard computer via a USB 2.0 connection. While the cameras are capable of capturing images at up to 30 Hz, the bandwidth of the USB 2.0 bus only allows the four cameras to deliver frames at a rate of 7 Hz. In the current implementation of the MCPTAM algorithm, this is the bottleneck as the tracking thread can comfortably run at a faster rate.

An example frame from the modified MCPTAM algorithm running with the four-camera cluster in the indoor lab environment is shown in Figure 5.3. The

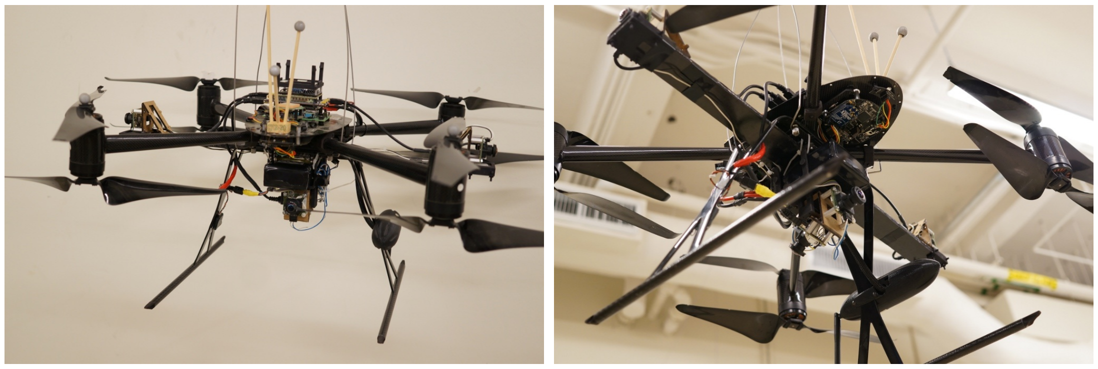

(a) Front view (b) Bottom view

Figure 5.1: For these experiments, four cameras are rigidly attached to a quadrotor aerial vehicle.

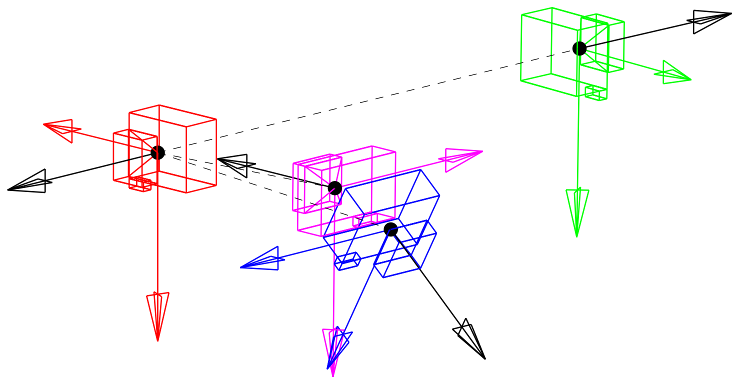
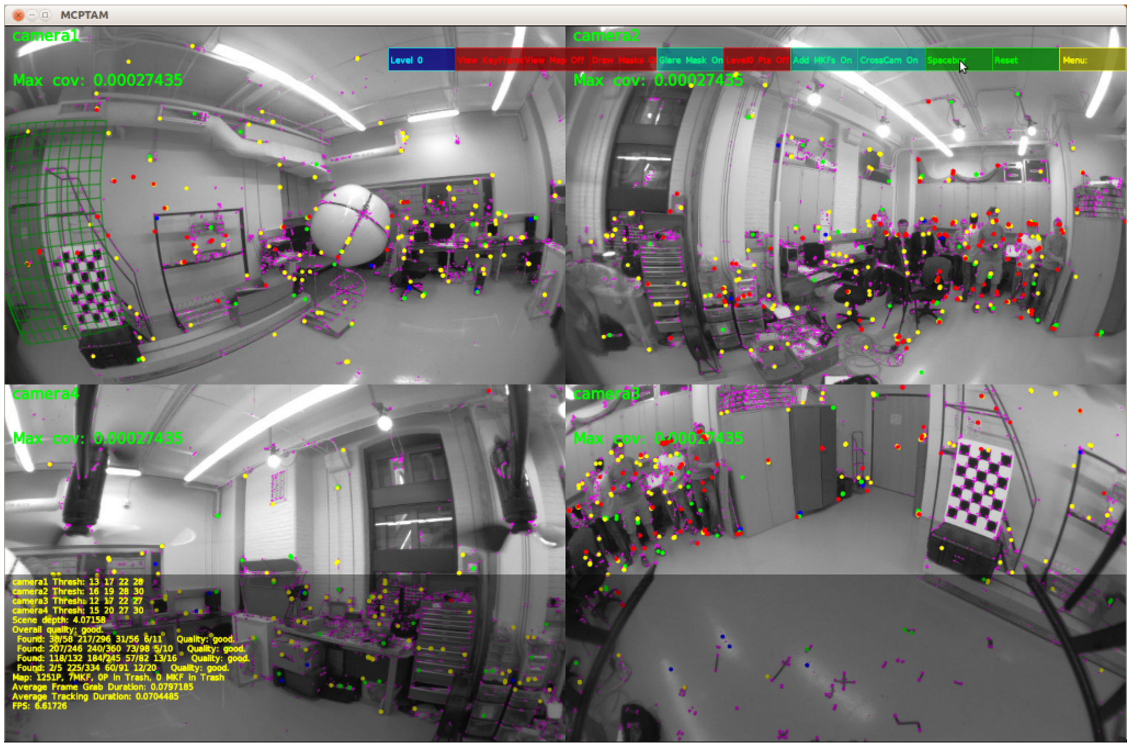


Figure 5.2: The four component cameras of the cluster are fixed to the aircraft with camera 1 (red) and 2 (green) facing outwards to the left and right, camera 3 (blue) looking down and back, and camera 4 (magenta) looking forwards.

colour dots represent the point features detected and tracked at different image pyramid levels.

5.2.1 Cluster FOV Configurations

The four cameras within the cluster have significant overlap in their FOV simply because the viewing angle of the lenses is so extreme. In particular, almost all of the FOV in the front and rear-facing cameras can been seen in the side-mounted wide-angle cameras. In order to demonstrate that the algorithms in this thesis work with or without overlapping FOV, two configurations, shown in Figure 5.4, will be



```
camera1 Thresh: 13 17 22 28
camera2 Thresh: 16 19 28 30
camera3 Thresh: 12 17 22 27
camera4 Thresh: 15 20 27 30
Scene depth: 4.07158
Overall quality: good.
 Found: 38/58 217/296 31/56 6/11   Quality: good.
 Found: 207/246 240/360 73/98 5/10   Quality: good.
 Found: 118/132 184/245 57/82 13/16   Quality: good.
 Found: 2/5 225/334 60/91 12/20   Quality: good.
Map: 1251P, 7MKF, 0P In Trash, 0 MKF In Trash
Average Frame Grab Duration: 0.0797185
Average Tracking Duration: 0.0704485
FPS: 6.61726
```

Figure 5.3: A screenshot of the MCPTAM algorithm running on a cluster of four cameras with FOV overlap. The point features (coloured dots) are tracked and mapped for the indoor lab environment. The spherical distortion from the wide-FOV cameras is evident.

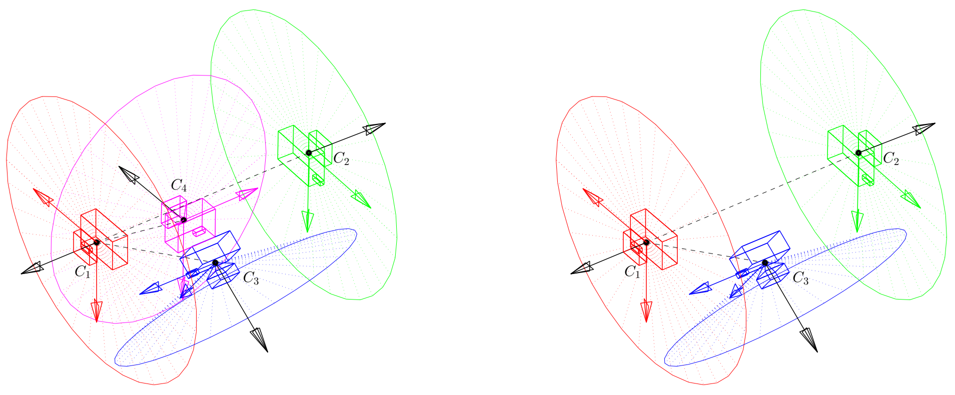
used in the experimental runs that follow:

Overlap (4 cameras) All four cameras use their entire FOV. Some point features will be seen by more than one camera at one time.

Non-overlap (3 cameras) The two side-facing cameras use their entire view, while the rear facing camera only uses a triangle at the bottom which does not overlap with the sides. Point features are only seen by one camera at one time.

5.3 Results

This section presents the pose estimation results for the MCPTAM algorithm running in a series of operating environments with various motion profiles and camera cluster configurations. Both indoor tests against ground-truth measurements from



(a) Overlap (b) Non-overlap

Figure 5.4: The cluster configurations used for the experiments. In (a), four cameras with significant FOV overlap are used. In (b), three cameras are used and the overlapping regions of the images from C_3 are masked out such that there is no common FOV between the cameras.

a Vicon IPS, and outdoor tests in a challenging roof-top environment are presented to demonstrate the accuracy and performance of an estimator based on the parameterization and analysis from this thesis.

5.3.1 Indoor Tests

The Vicon measurements are used to verify the accuracy of the MCPTAM algorithm both with and without overlapping FOV in the component cameras. The Vicon IPS provides high-precision measurements of the six DOF pose of a set of Infra-Red (IR) reflective spheres with respect to a pre-defined Vicon world frame W. The Vicon system software interface allows the user to group a set of IR markers into a single tracked object. The tracked object has a local coordinate frame T in which the positions of the composite markers are known precisely. It is the position and orientation of this local trackable object coordinate frame which is provided by the Vicon system, calculated using the measured positions of the individual IR markers in the workspace at each time step.

The MCPTAM system provides estimates of the position and orientation of the camera cluster with respect to a separate vision world frame M. In this system, the vision world frame is chosen as the position and orientation of the cluster coordinate frame at the beginning of the tracking operation. MCPTAM tracks the position and orientation of the cluster frame U with respect to this initial pose.

Therefore, the pose measurements from the Vicon and MCPTAM systems must be transformed into a common coordinate frame to facilitate any comparison. The calibration process for aligning the two systems is detailed in Appendix C.

It will be confirmed that with sufficient rotational motion, both configurations are able to track the cluster pose accurately, including recovery of the global scale metric. Additionally, two degenerate motions identified in Chapter 4 were used to show situations where the non-overlap configuration is able to accurately recover an up-to-scale solution for the relative motion and structure. These motions are the translation-only case, and the two-camera cluster concentric circle case. However, it is shown that the configuration with overlap is still able to accurately reconstruct the correctly-scaled motion in the degenerate case. This result is used to justify the comparison between the overlap and non-overlap configurations for the later outdoor tests where the Vicon system is unavailable.

Procedure

While the modified MCPTAM algorithm is able to run on live image streams captured from the cluster cameras in real-time, for these experiments the images from the cameras during the quadrotor flight were recorded, along with the measured pose of the Vicon trackable frame. Subsequently, the MCPTAM algorithm was run on the recorded camera images and the resulting pose trajectory from the tracker is captured and compared to the ground truth data from the Vicon measurements. This is done to compare the different cluster configurations on the exact same image data to isolate their effect on the resulting pose estimates.

The MCPTAM algorithm is initialized at the beginning of the recorded motion sequence and keyframes were manually added to the map at regular time intervals on the first run through to ensure that the mapped area is sufficiently observed by several keyframes. MCPTAM is able to automatically add keyframes to the model when it decides it needs to do so. However, this feature was disabled during these tests to ensure uniform coverage of the workspace. Once the image sequence completes, MCPTAM continues to run and the image sequence was restarted. MCPTAM is able to relocalize with respect to the previously observed keyframes and continues to track the cluster pose within the target model. During this second run-through, no new keyframes were added to the target model. As the tracker localizes the cluster pose, it determines whether some of the previously added point features are missing or of poor quality, and if so, will discard them from the model. The resulting target model is stable and provides good localization constraints to track a high-accuracy pose estimate. The image sequence from the motion trajectory is run for a third time, at which point the pose estimates from the tracker are captured for comparison with the Vicon ground truth data.

The system is tested in this way to produce the best possible estimates of the tracked pose of the camera cluster. This simulates the performance of a system when the camera cluster stays in the same area of the environment long enough to reach a steady-state with regards to adding keyframes and refining the target model that it has generated. Due to the iterative nature of the mapping algorithm the performance of the estimate will be diminished during periods of rapid exploration as the computation time of the BA process dictates an upper-bound to the speed at which the map can be updated accurately. This process is performed for the cluster configurations both with and without overlapping FOV.

For the first test trajectory, after presenting the pose estimates generated using the stable target model, the pose estimates during the initial run through the image sequence will be presented to demonstrate how the system estimates converge from the initial conditions.

A. Non-Degenerate Case – Translation and Large Rotation

The first motion trajectory tested was a general motion in which there are large translations and rotations within the full workspace of the Vicon system. The magnitudes of the rotation relative to the initial orientation are shown in Figure 5.5.

Using the fixed extrinsic calibration between the camera cluster frame and the IR marker frame found in Section C.2, the estimated pose and the Vicon measurement trajectories were aligned with the scale fixed at unity to compare the true error magnitudes of the position and orientation estimates from the two cluster configurations. Accordingly, the only free variable in the alignment procedure is the world transformation, $\mathbf{T}_M^W$. The resulting aligned trajectories exhibit excellent agreement with the ground truth and are shown in Figure 5.6.

The magnitudes of the error in the position and orientation estimates, in millimetres and degrees, respectively, for both cluster configurations are shown in Figure 5.7. The Root Mean Squared Error (RMSE) of each configuration is shown as the dashed constant line on each graph. The overlap configuration shows an RMSE for position and orientation magnitudes of 5.5 mm and 0.42 deg, respectively. The non-overlap configuration has RMSE values of 9.9 mm and 0.42 deg.

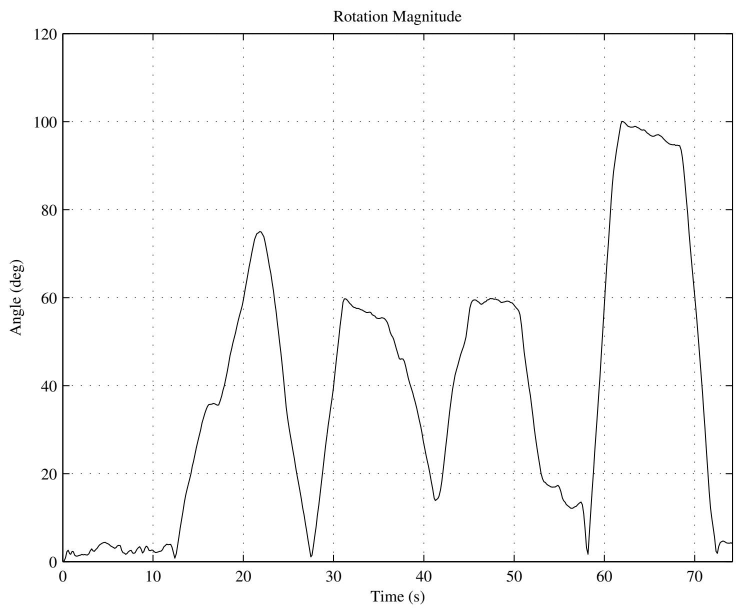


Figure 5.5: The magnitude of the rotation angle experienced by the camera cluster through the large rotation motion trajectory.

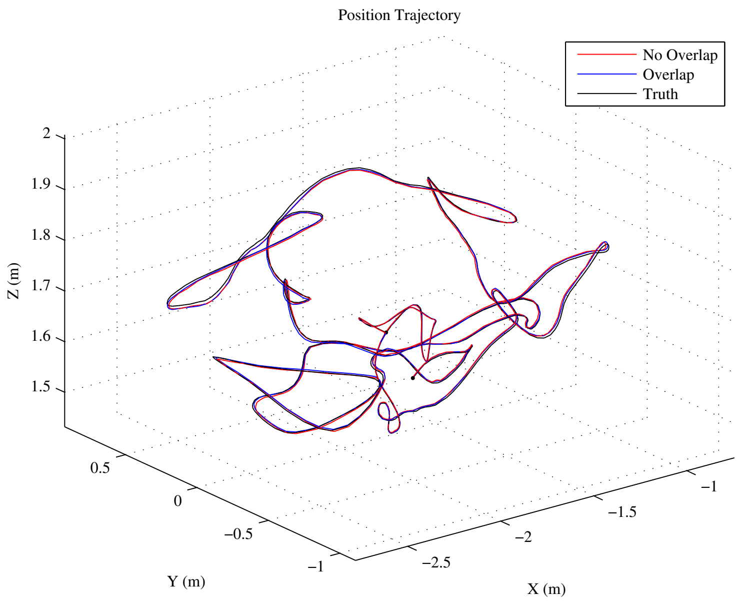


Figure 5.6: The position trajectories of the estimated pose from the overlap (blue) and non-overlap (red) configurations compared against the measured position from the IPS (black).

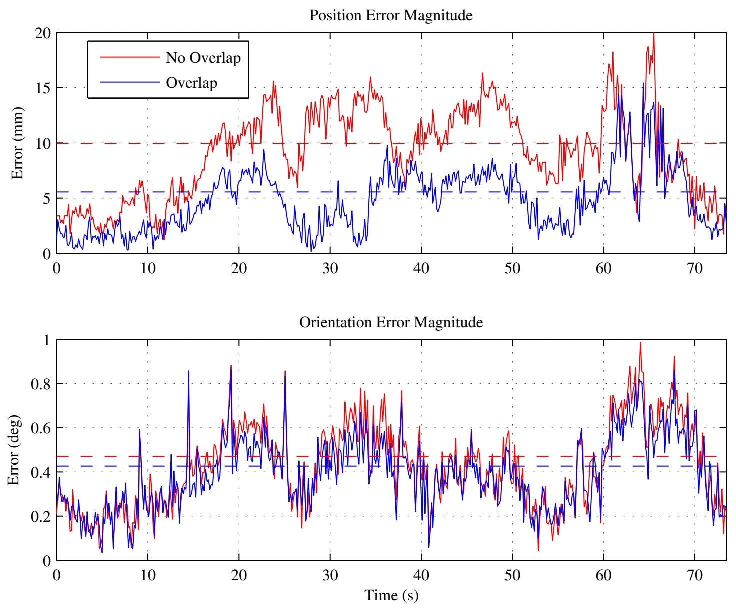


Figure 5.7: The magnitudes of the position and orientation errors for the overlap (blue) and non-overlap (red) configurations. The RMSE for each configuration are shown as the dashed constant lines.

To see how much of the error is associated with an incorrect global scale estimate, the estimated pose trajectories for both cluster configurations are aligned with the Vicon measurements using the trajectory alignment optimization with the scale factor set as variable. This produces a normalized trajectory and isolates the scale factor between the MCPTAM and Vicon pose measurements. The aligned trajectories are shown in Figure 5.8.

When compared to the previous trajectory alignment with the scale factor fixed, the difference with Figure 5.6 is difficult to observe. The scale factor between the estimated pose and Vicon trajectories for the two cluster configurations were found to be 1.003 and 1.012 for the overlap and non-overlap configurations, respectively. These scale factors represent the ratio of the position magnitudes of the Vicon measurements to the MCPTAM position estimates. Accordingly, a scale factor greater than one means that the actual positions are larger than the MCPTAM estimates.

With the error due to incorrect scale removed from the estimates, the magnitudes of the remaining error in position and orientation are shown in Figure 5.9. It is clear that the orientation error magnitudes are almost identical to those with the scale error included, in Figure 5.7. This strongly indicates that both of the camera

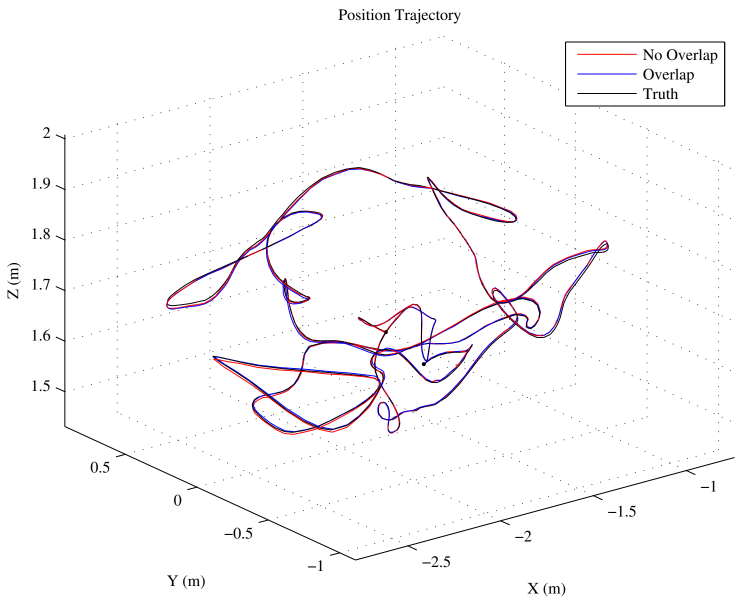


Figure 5.8: The position trajectories, with errors due to incorrect scale estimate removed, of the estimated pose from the overlap (blue) and non-overlap (red) configurations compared against the measured position from the IPS (black).

cluster configurations are able to estimate the up-to-scale solution effectively, even though there may be small errors in the scale recovery. The resulting RMSE for the overlap configuration are 5.1 mm and 0.42 deg, while the non-overlap configuration has RMSE of 4.6 mm and 0.47 deg. This experiment demonstrates that when there is a large amount of rotation in the relative motion, both cluster configurations are able to recover extremely accurate estimates of the cluster pose through the motion sequence.

In the above tests, the target model was allowed to stabilize over several runs of the image sequence to allow it to produce the most accurate map possible to test the pose estimation. With the large rotational and translation motion, the BA thread was able to converge to an accurate, correctly-scaled solution for the map model. To test the performance of the system during the initial phase when keyframes are being added and the solution is uncertain, the pose estimates were collected immediately after start-up for the the two cluster configurations. The magnitudes of the estimated and measured positions are shown in Figure 5.10, along with the ratio of the estimated magnitudes over the Vicon measured position magnitudes. A correct scale value is indicated by a unity ratio.

The abrupt changes in the position estimates are caused by the BA thread pro-

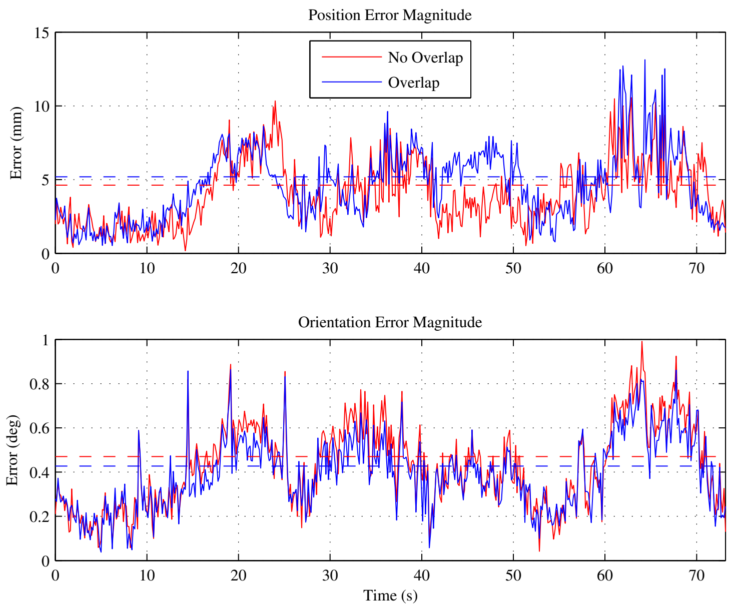


Figure 5.9: The magnitudes of the position and orientation errors, with errors due to incorrect scale estimate removed, for the overlap (blue) and non-overlap (red) configurations. The RMSE for each configuration are shown as the dashed constant lines.

viding a newly updated map model at that time step. Initially, both configurations provide poorly-scaled position estimates when the relative translation and orientation magnitudes are small compared to the initial cluster pose. As the motions evolve, the scale estimates vary but eventually converge to close to unity.

The magnitude of the position errors during this initial phase are shown in Figure 5.11. With respect to Figure 5.5, there is not any significant rotation until approximately 14 seconds into the trajectory. Therefore, the amount of rotation between any keyframes collected up to that point, is small and the motion is near-degenerate (refer to Chapter 4).

After that time, the algorithm must choose to place a keyframe and complete the subsequent BA optimization before the proper scale is resolved. For the non-overlap configuration, an initial scale correction is observed at 14 seconds, followed by another at approximately 23 seconds that leads to greater accuracy due to further rotation between the permanent keyframes in the target model. With the overlap configuration, the positions are more accurate from the start, but the significant scale correction occurs around 18 seconds. As more of the environment is explored and more permanent keyframes are added to the target model, both

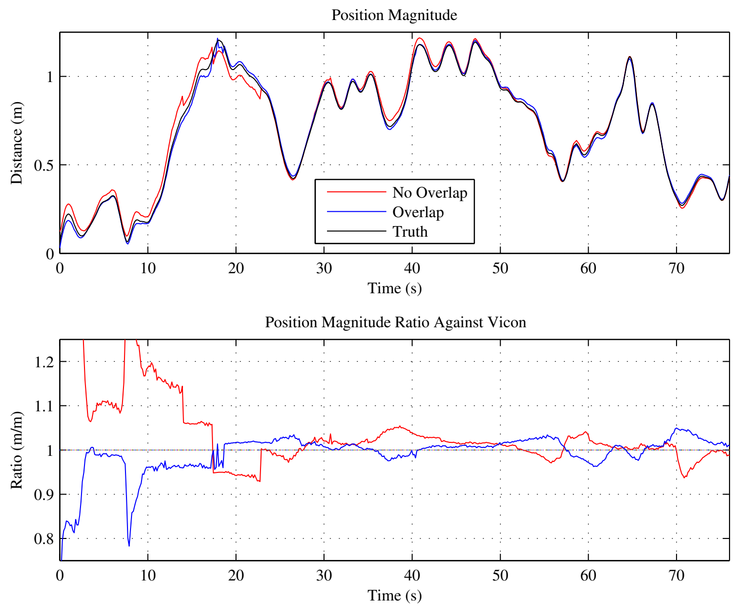


Figure 5.10: The position magnitudes (top) for the two cluster configurations against ground truth. The ratio of the magnitudes (bottom) compared to the IPS position magnitude.

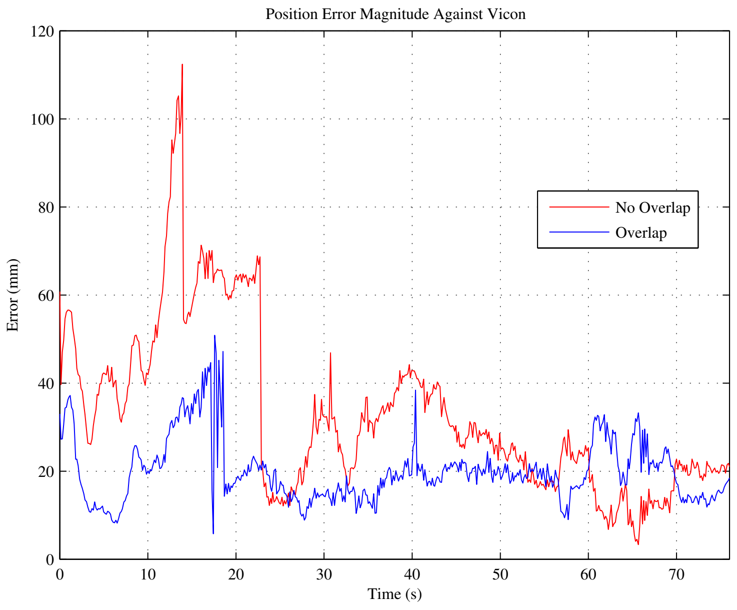


Figure 5.11: The error of the position magnitude during the initial motion for the two cluster configurations compared against ground truth.

solutions become more accurate as the run progresses.

B. Near-Degenerate Case – Translation and Small Rotation

The second motion profile selected was one in which the amount of relative rotation was kept minimal. The quadrotor vehicle was flown with a constant heading angle in yaw and the only rotation through the trajectory was due to subtle pitching and rolling motions to generate the translational motion. The maximum rotation angle relative to the initial cluster pose was approximately 9 degrees, while the translations were again large enough to cover the entire Vicon workspace. The magnitude of the orientation changes in the motion sequence are shown in Figure 5.12, as measured by the Vicon system.

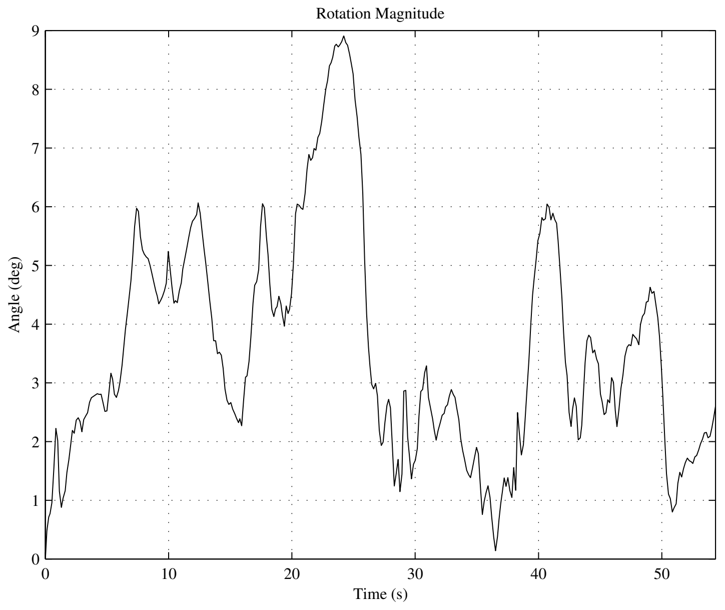


Figure 5.12: The magnitude of the rotation angle experienced by the camera cluster through the small rotation motion trajectory.

Motion with small rotation is a scenario close to degeneracy for the non-overlap configuration, as discussed in Chapter 4, and the solution will be insensitive with respect to global scale. In the presence of measurement noise, the global scale will still converge, but the scale metric recovered may be incorrect. This motion scenario was captured using the quadrotor aerial vehicle in-flight to demonstrate how the common case of maintaining a constant heading angle may lead to poor scale recovery since the pitching and rolling motions are small.

The same processing procedure is carried out as with the previous motion trajectory, and the aligned trajectories with the Vicon measurements are shown in Figure 5.13.

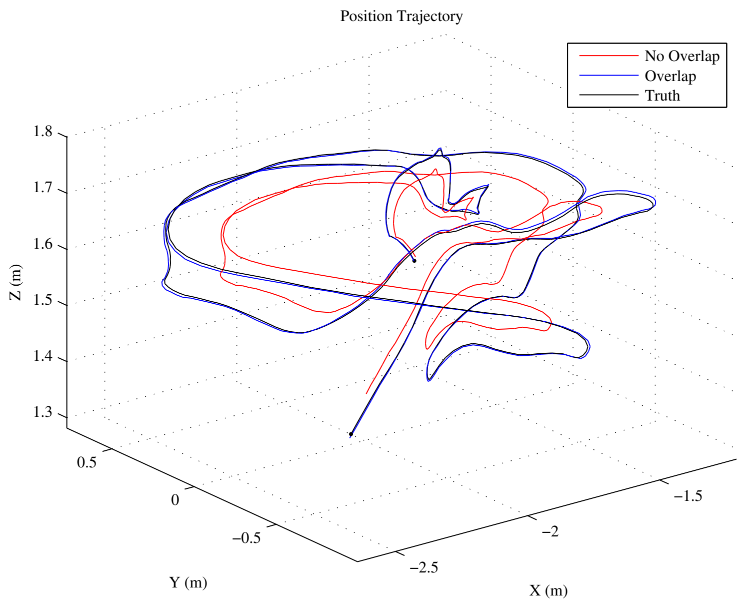


Figure 5.13: The position trajectories of the estimated pose from the overlap (blue) and non-overlap (red) configurations compared against the measured position from the IPS (black).

It is immediately apparent that the non-overlap configuration is unable to recover the correct global scale of the solution. However, the overlap configuration, due to the use of inter-camera correspondences, is still able to resolve the scale despite the small rotational motion. It is noteworthy that the recovered trajectory from the overlap configuration does not agree with the Vicon measurements as well as with the previous motion with large rotation. It is likely because the previously large rotational motion placed additional constraints on the solution which helped to more accurately recover the map model scale even when the FOV overlap is accounted for.

As expected, the magnitudes of the position and orientation errors are large for the non-overlap configuration, and still relatively accurate, by comparison, for the overlap case, as seen in Figure 5.14. The RMSE for the non-overlap configuration are 125 mm and 0.21 deg, while those of the overlap configuration are 8.5 mm and 0.20 deg. It is clear that the overlap configuration is still able to recover an accurate solution when the non-overlap configuration experiences degenerate motion. As a

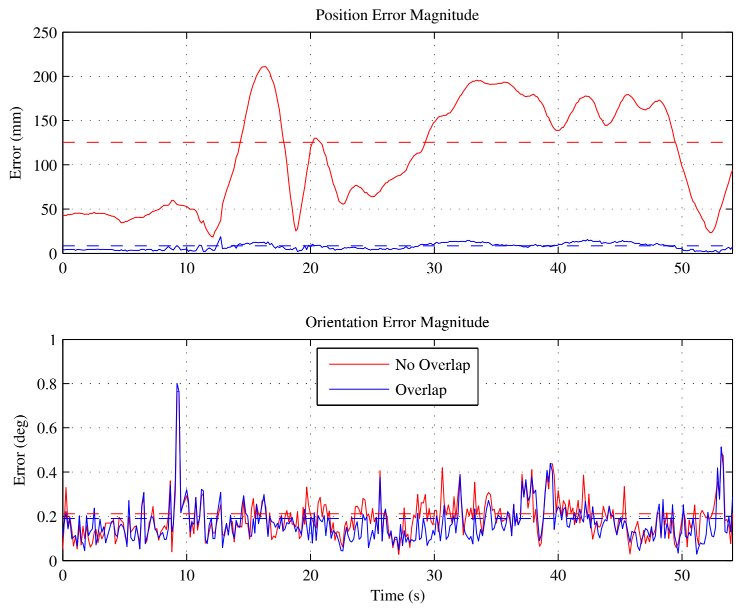


Figure 5.14: The magnitudes of the position and orientation errors for the overlap (blue) and non-overlap (red) configurations. The RMSE for each configuration are shown as the dashed constant lines.

result, the overlap scenario will be used to verify the solution of the non-overlap configuration for the outdoor test cases that follow in Section 5.3.2.

Most of the position error generated using this motion profile is associated with an incorrect scale estimate. When the trajectories are aligned with the scale factor variable, the resulting position trajectories show good agreement, as seen in Figure 5.15.

The scale factors identified for the non-overlap and overlap configurations are 1.276 and 0.9874, respectively. The remaining position errors are reduced after the removal of the component related to scale and indicate that both configurations recover an accurate up-to-scale solution. These pose error magnitudes are shown in Figure 5.16.

As before, the orientation error magnitudes are similar to those found previously, but the position errors are significantly reduced, particularly for the non-overlap case, to result in an RMSE of 4.9 mm. The overlap configuration has a position RMSE of 4.3 mm.

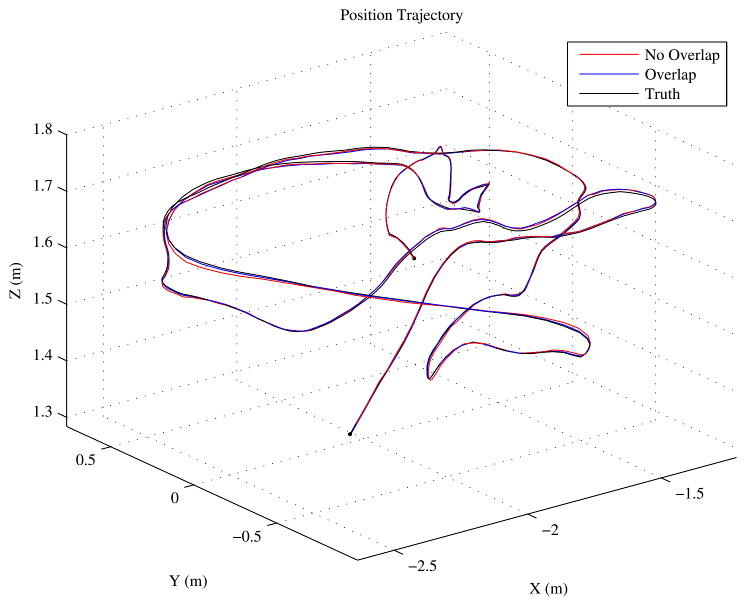


Figure 5.15: The position trajectories, with errors due to incorrect scale estimate removed, of the estimated pose from the overlap (blue) and non-overlap (red) configurations compared against the measured position from the IPS (black).

C. Degenerate Case – Concentric Circles

The second degenerate motion profile tested for the non-overlap configuration was a two-camera cluster moving in concentric circles with a common centre collinear with the two camera optical centres, as described in Chapter 4. For this test, only the two side-mounted cameras were used in the configuration, as shown in Figure 5.17.

The cluster was attached to the ceiling with a string and allowed to hang with the cameras pointed towards the ceiling and floor with their optical axes collinear with the string. The cluster was swung like a pendulum so that the camera centres moved on the concentric spheres with a centre at the attachment point on the ceiling. A diagram of the experimental setup is shown in Figure 5.18.

With this setup, any pair of poses of the cluster motion result in the camera centres moving along concentric circles with the centre at the ceiling attachment, and therefore, the entire motion is degenerate and the scale is ambiguous. This is true for any amount of rotation. The trajectory for the pendulum motion contains a large amount of rotation as shown in Figure 5.19.

The MCPTAM algorithm was run using this motion profile and the target map

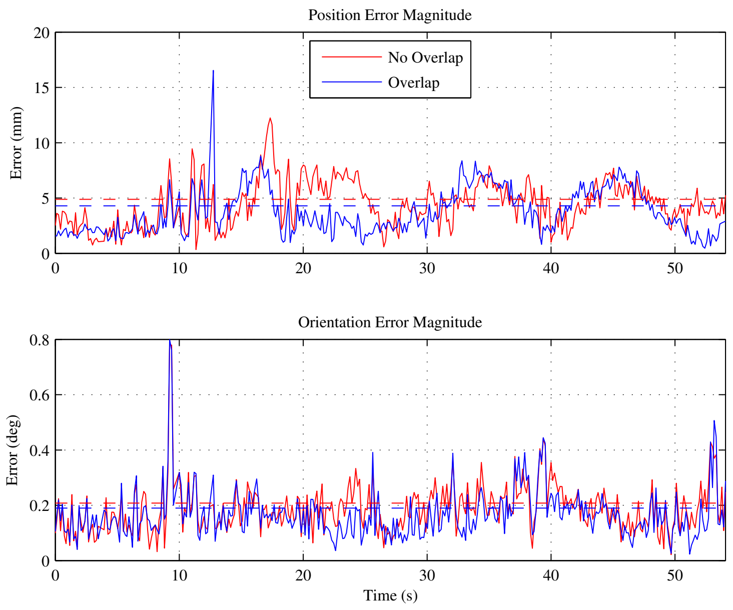


Figure 5.16: The magnitudes of the position and orientation errors, with errors due to incorrect scale estimate removed, for the overlap (blue) and non-overlap (red) configurations. The RMSE for each configuration are shown as the dashed constant lines.

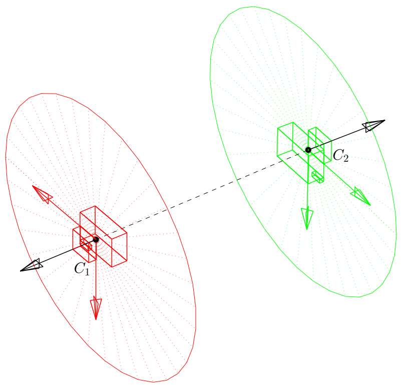


Figure 5.17: The two-camera cluster composed of the two sideways facing cameras. There is no overlap in the camera FOVs.

model was allowed to converge. As expected, the recovered scale estimate was incorrect, but when the estimated and measured trajectories were aligned with a variable scale factor, the solution showed good up-to-scale accuracy, as can be seen in Figure 5.20.

In this case, the scale factor correction was found to be 0.8736, which represents

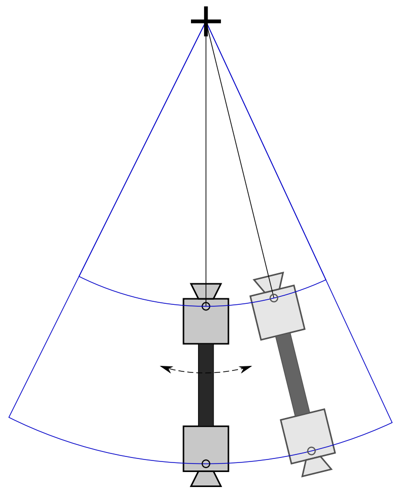


Figure 5.18: The two-camera cluster was attached by a string to the ceiling. The motion was restricted to swing like a pendulum in the concentric spheres centred at the ceiling point.

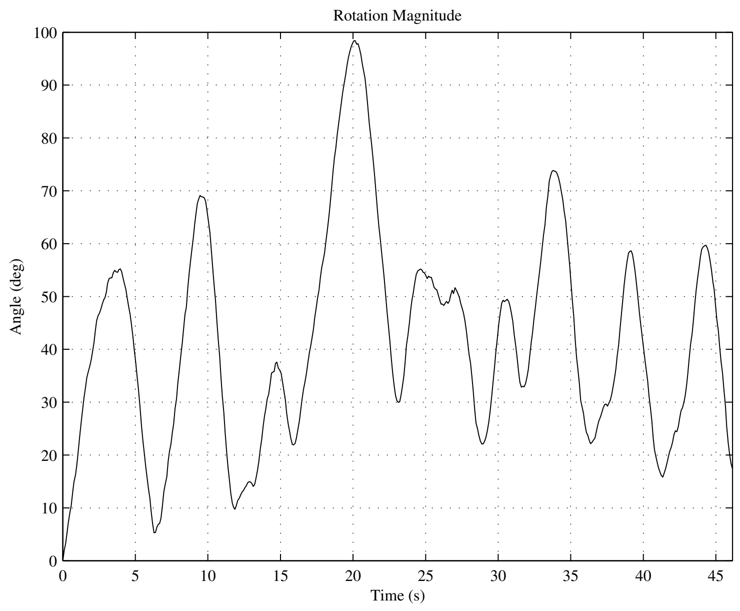


Figure 5.19: The magnitude of the rotation angle experienced by the camera cluster through the pendulum swing concentric circles motion trajectory.

a significant inaccuracy in the recovered scale metric. Once the position error due to the incorrect scale is removed, the remaining position errors are small, as shown in Figure 5.21. These results validate the degenerate motion analysis from Chapter 4.

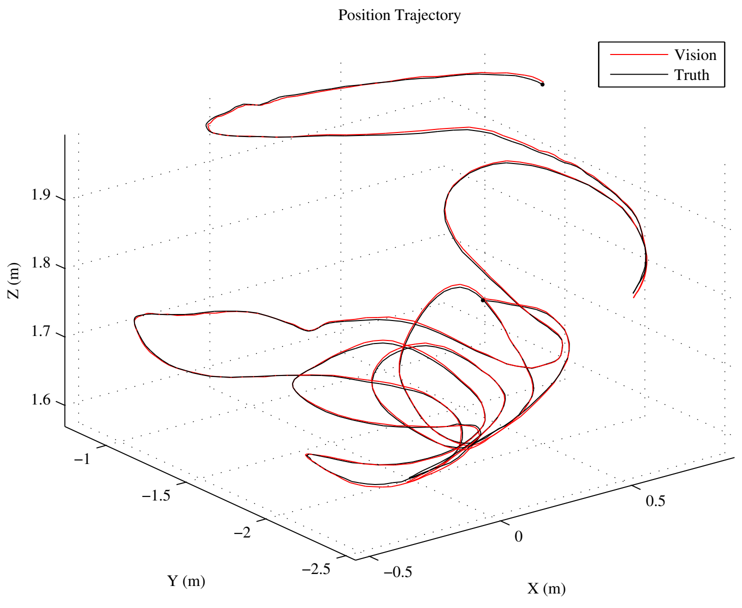


Figure 5.20: The position trajectories, with errors due to incorrect scale estimate removed, of the estimated pose from the two-camera cluster (red) configuration compared against the measured position from the IPS (black).

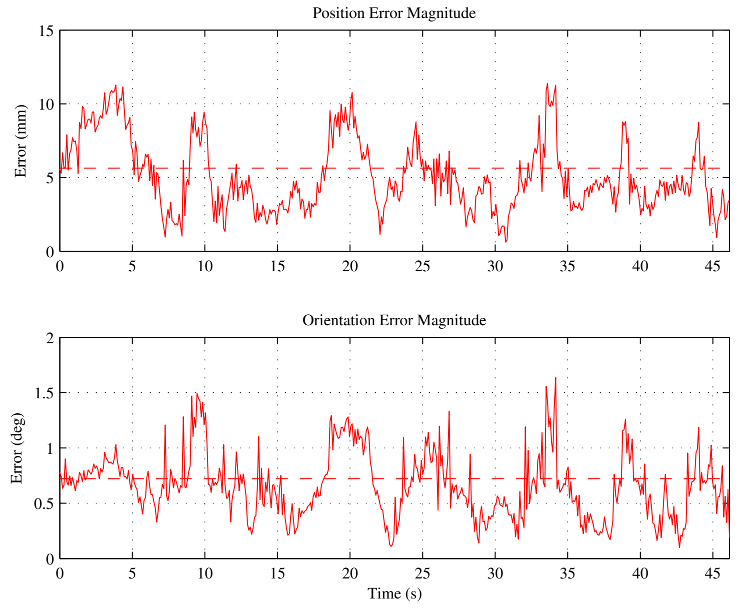


Figure 5.21: The magnitudes of the position and orientation errors, with errors due to incorrect scale estimate removed, for the two-camera cluster (red) configuration. The RMSE is shown as the dashed constant lines.

Table 5.1: Indoor Test Results

Motion	**Config.**	**Scale**	**RMSE**	
			Pos. (mm)	**Rot. (deg)**
Large Rotation	Overlap	1	5.5	0.42
		1.003	5.1	0.42
	Non-overlap	1	9.9	0.47
		1.012	4.6	0.47
Small Rotation	Overlap	1	8.5	0.20
		0.9874	4.3	0.20
	Non-overlap	1	125	0.21
		1.276	4.9	0.21
Pendulum Swing	Non-overlap	0.8736	5.6	0.72

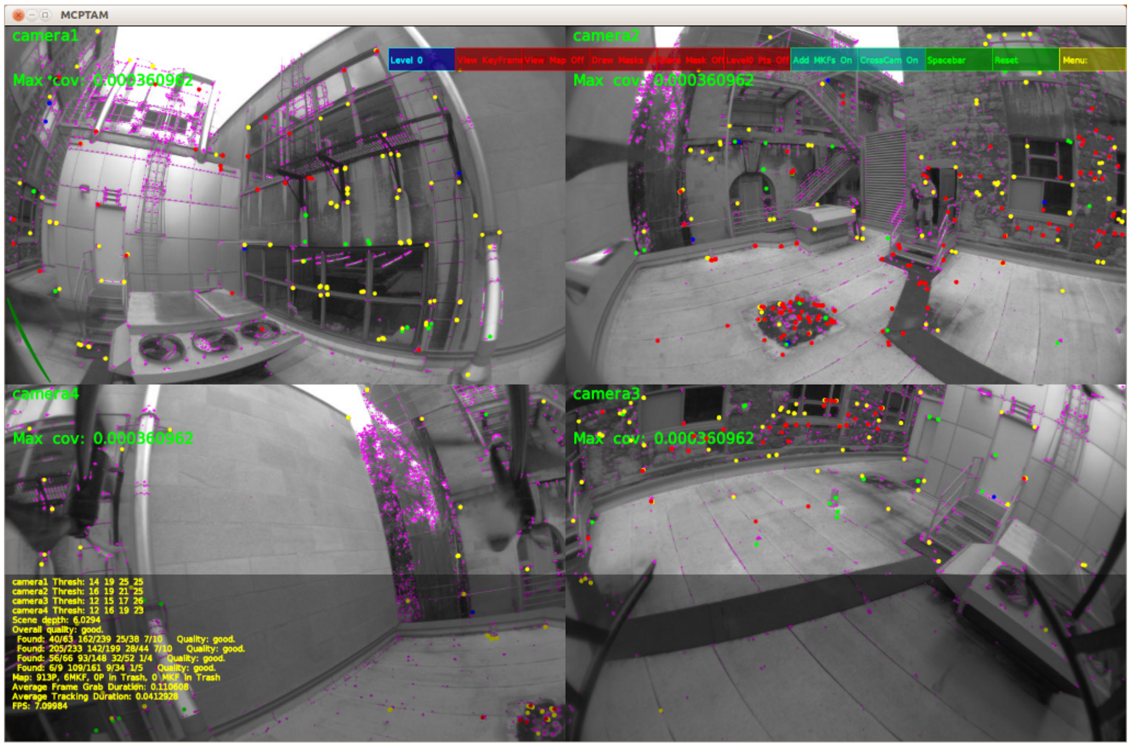
Summary

The results from the indoor test cases are summarized in Table 5.1. These experiments demonstrate that the new MCPTAM system is able to accurately estimate the current cluster pose, including the global scale metric, with both the overlap and non-overlap configurations, providing there is sufficient orientation change in the motion trajectory. Additionally, when the motion is degenerate for the non-overlap case, it is still able to recover an accurate up-to-scale solution despite the global scale being ambiguous. When the motion becomes non-degenerate, the scale can then be recovered accurately. The overlap configuration was able to recover an accurately-scaled solution despite the non-overlap motion degeneracies and therefore, will serve as the reference trajectory measurements for the outdoor test in the next section.

5.3.2 Outdoor Tests

The previous indoor tests allowed the MCPTAM system performance to be confirmed using the Vicon IPS providing ground truth pose measurements. To demonstrate the applicability of the proposed algorithm and implementation in real-world applications, the MCPTAM system was tested in an outdoor roof environment to verify that the system was capable of operating in a larger workspace with natural lighting conditions and difficult visual landmarks. Unfortunately, the Vicon IPS cannot operate outside due to the sunlight overpowering the reflections from the passive IR markers. As a result, the non-overlap cluster configuration was directly compared to the overlap configuration to determine the quality of the pose tracking algorithm when using non-overlapping FOV. The previous indoor test cases showed that the overlap configuration was able to recover an accurate relative motion trajectory, including the proper global scale, even when the non-overlap configuration experienced degenerate motion and could only recover an accurate up-to-scale solution.

In this test, the quadrotor vehicle was flown around the large roof area and the cameras observed point features on the surrounding walls and ground. The environment can be seen in the example frame from MCPTAM during the execution with the overlap configuration, shown in Figure 5.22. Some of the walls were quite smooth and provided poorly textured surfaces which were devoid of any usable point features. This is a challenging scenario for any vision algorithm, particularly when the collective FOV is narrow, since the track can easily be lost when not enough features are visible in the camera images. However, the large collective FOV for the proposed cluster system allows the estimator to track any available point features since they are visible in at least one of the component cameras.



```
camera1 Thresh: 14 19 25 25
camera2 Thresh: 16 19 21 25
camera3 Thresh: 12 15 17 26
camera4 Thresh: 12 16 19 23
Scene depth: 6.0294
Overall quality: good.
 Found: 40/63 162/239 25/38 7/10    Quality: good.
 Found: 205/233 142/199 28/44 7/10    Quality: good.
 Found: 56/66 93/148 32/52 1/4    Quality: good.
 Found: 6/9 109/161 9/34 1/5    Quality: good.
Map: 913P, 6MKF, 0P in Trash, 0 MKF in Trash
Average Frame Grab Duration: 0.110608
Average Tracking Duration: 0.0412928
FPS: 7.09984
```

Figure 5.22: A screenshot of MCPTAM running with the overlap cluster configuration in the outdoor roof environment. The set of available point features is sparse in certain directions due to lack of texture. However, the large collective FOV of the cluster is able to track any features available to prevent tracking failure.

The yaw angle heading of the quadrotor was varied during the flight to ensure there was sufficient orientation change in the cluster trajectory and allow the non-overlap configuration to avoid degenerate motions. A plot of the magnitude of the rotation angle through the roof flight, as measured by the overlap configuration is shown in Figure 5.23.

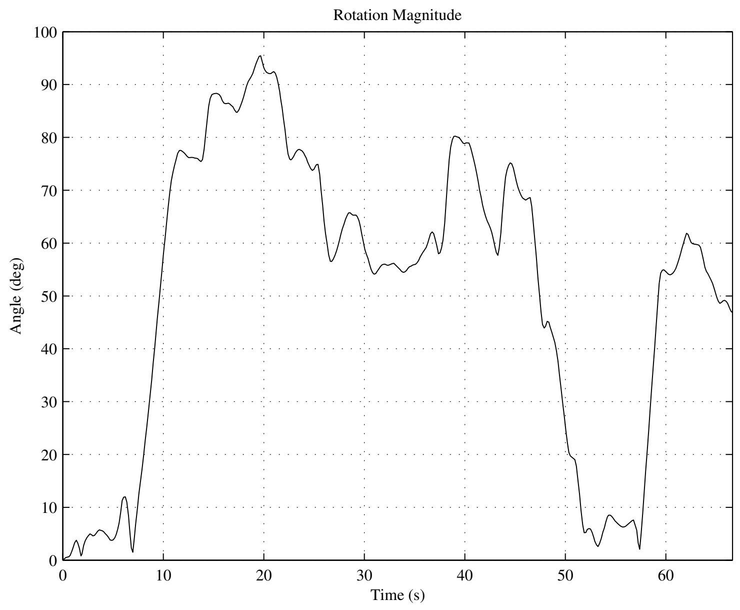


Figure 5.23: The magnitude of the rotation angle experienced by the camera cluster through the outdoor roof motion trajectory.

The pose trajectories for the two configurations are computed by the MCPTAM algorithm using the same processing procedure as in the previous indoor tests (refer to Appendix C), and aligned by fixing the scale factor to unity and the body transformation to identity, $\mathbf{T}_U^T \equiv \mathbf{I}_{4\times4}$. This was done since both estimators use the same primary camera as the cluster coordinate frame. The world transformation still needs to be estimated since the algorithm was started at slightly different time steps for the two configurations. Both cluster configurations were able to track the relative pose consistently through the motion and the resulting aligned trajectories show good agreement, as shown in Figure 5.24.

The estimated global scale from non-overlap configuration is slightly larger than that recovered by the overlap case. The magnitudes of the position and orientation errors are shown in Figure 5.25. Both configurations recover similar pose trajectories, and the RMSE between them are 24 mm and 0.15 deg.

When the pose estimates from the non-overlap configuration are aligned with

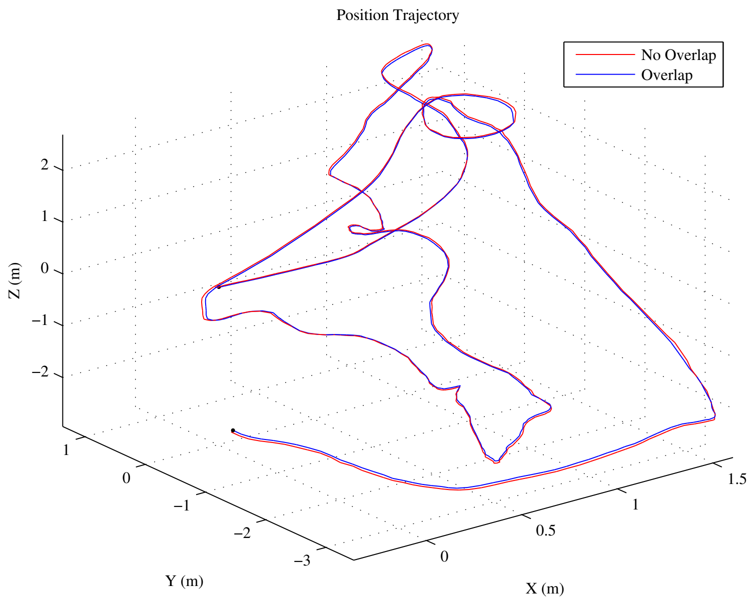


Figure 5.24: The position trajectories estimated by the non-overlap (red) and overlap (blue) configurations for the outdoor roof flight.

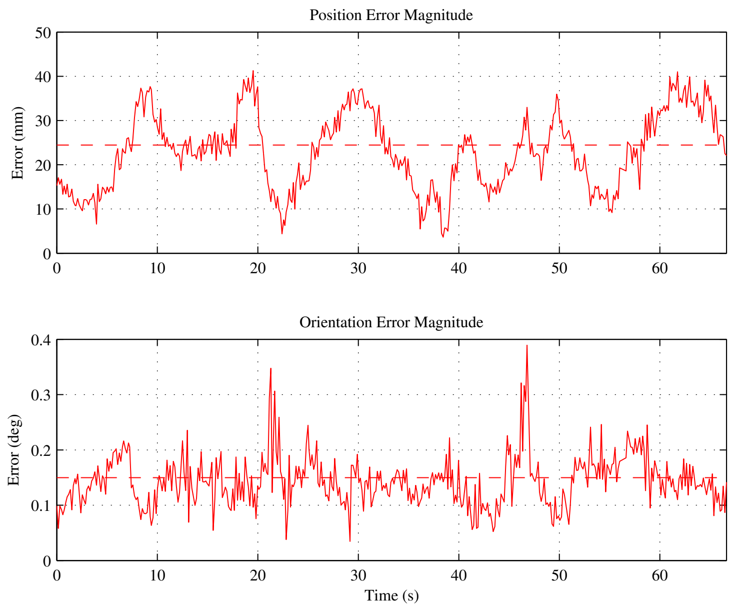


Figure 5.25: The magnitudes of position and orientation errors for the estimates from the non-overlap configuration compared to the overlap configuration. The RMSE is shown as the dashed constant lines.

those from the overlap case and the scale factor is allowed to vary, it is observed that most of the position error is due to a small disagreement in the estimated scale. After removing the error component related to the incorrect scale estimate, the resulting position trajectories are shown in Figure 5.26.

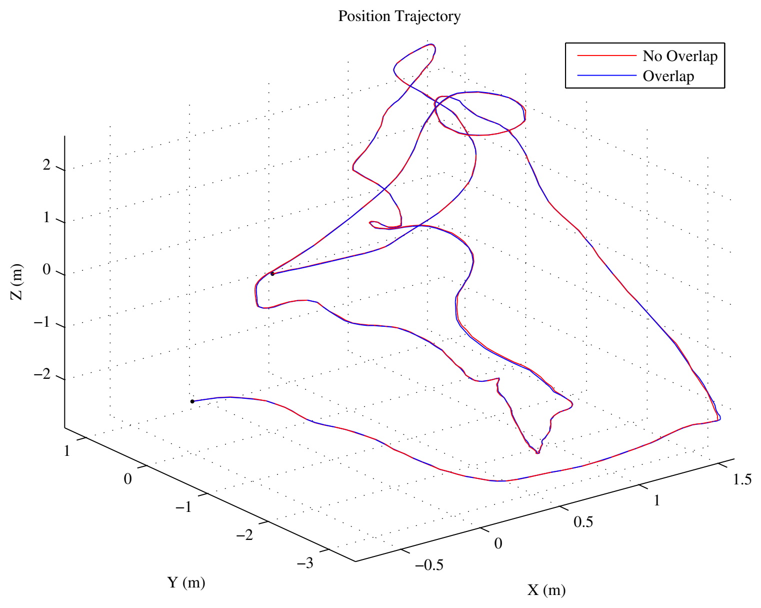


Figure 5.26: The position trajectories, with errors due to incorrect scale factor removed, estimated by the non-overlap (red) and overlap (blue) configurations for the outdoor roof flight.

The scale factor between the two configurations was found to be 0.9899. With the overlap configuration assumed to recover the correct scale metric, the non-overlap configuration is able to recover the correct scale within approximately 1% of this value. The magnitudes of the pose errors after removing the scale-error component are shown in Figure 5.27. They are significantly reduced to an RMSE of 9.0 mm for the non-overlap configuration.

The outdoor test results are summarized in Table 5.2. These results show that the two camera configurations produce consistent estimates of the pose trajectory of the camera cluster as the quadrotor moves through the outdoor roof-top environment despite the challenging visual environment.

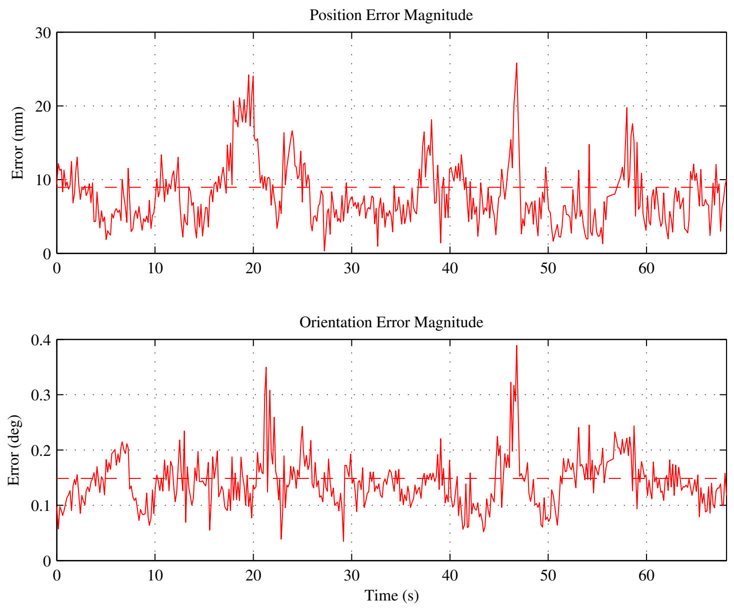


Figure 5.27: The magnitudes of position and orientation errors for the estimates, with errors due to incorrect scale factor removed, from the non-overlap configuration compared to the overlap configuration. The RMSE is shown as the dashed constant lines.

Table 5.2: Outdoor Test Results

Scale	**RMSE**	
	Pos. (mm)	**Rot. (deg)**
1	24	0.15
0.9899	9.0	0.15

5.4 Summary

In this chapter, a full experimental implementation of the algorithms and analyses in Chapters 3 and 4, was presented. A four-camera cluster was constructed, calibrated, and mounted onto an actual quadrotor aerial vehicle. The MCPTAM algorithm was modified to use the parameterization and initialization scheme from this thesis, allowing for multicamera clusters using cameras both with or without overlapping FOV to successfully track and model the target object or environment after being initialized by only the information in the first set of camera images. The proposed algorithm is able to run at real-time rates on camera images collected during motions in both indoor and outdoor environments using natural image features.

The accuracy of the modified MCPTAM algorithm was demonstrated by comparing the pose estimates from cluster configurations with and without FOV overlap to ground truth pose measurements from a Vicon IPS. It was confirmed that providing the degenerate motions from Chapter 4 were avoided, the non-overlap configuration was able to estimate the relative pose of the cluster and target model to sub-centimetre and sub-degree accuracy. Furthermore, when the motion was degenerate – when the relative rotation between keyframes was small, or a two-camera cluster moved the in the concentric circle degeneracy – the non-overlap configuration was still able to successfully track an accurate up-to-scale solution.

Finally, the performance of the estimator was presented for a challenging outdoor roof-top environment where sections of the environment were sparsely populated by usable point features. It was shown that the large collective FOV of the cluster configurations allowed the algorithm to maintain observations of the available point features and successfully track the cluster pose trajectory. The scale estimates of both the overlap and non-overlap configurations were similar, showing that the non-overlap configuration operates as well as a cluster with FOV overlap which was able to make sparse stereo correspondences at each time step.

Chapter 6

Conclusions

The work presented in this thesis includes the design and analysis of a novel real-time relative pose and target model estimation algorithm using calibrated non-overlapping FOV camera clusters suitable for precise robotic interaction tasks. In these camera arrangements, the FOV of the individual cameras need not overlap in space at the same point in time, and the large collective FOV makes effective use of the available pixels, constraining the localization and resolving rotation-translation ambiguities present with other camera configurations. As a result, the position and orientation of the cluster coordinate frame is accurately recovered, including the global scale metric when the camera centres have non-zero baselines between them, and despite no prior knowledge of the moving target object or environment.

A full review of the multicamera cluster motion estimation algorithms in the literature, presented in Chapter 2, identified the strengths and weaknesses of the state-of-the-art methods. Most current methods capable of working with completely non-overlapping FOV solve the visual odometry problem, which is particularly vulnerable to accumulating scale error in the motion increments, leading to inaccuracies from solution drift. Currently, there are no other localization systems capable of successfully running from initialization with a completely unknown target model, using a cluster with non-overlapping camera FOV.

This work fills the void by presenting a complete SLAM estimation framework in Chapter 3. The camera cluster is modelled as a set of perspective cameras rigidly attached in a common coordinate frame. From this base, the system measurement model was formulated for the image space coordinates of the point features comprising the rigid-body target model. The state was represented using the $\boxplus$-manifold to encapsulate the topological structure of the rigid-body motion space, while estimation is performed using an iterative least-squares optimization method.

Additionally, a new parameterization for the point feature position update was proposed, applying a spherical perturbation in the bearing and radial directions for the vector. Isolating the effect of global scale into the single radial parameter allows the system to converge in the bearing directions, then correct the depth when the information is available through the motion.

The full SLAM algorithm was presented in which the estimation of the current relative pose of the cluster and target model is run in parallel with the full nonlinear BA process. The pose tracking process uses the most recent target model from the BA process to localize the current cluster pose within the target model, taking the target model as fixed, and only estimating the current pose. This reduces the state space dimension and allows the process to run in real-time at the frame rate of the cameras. Concurrently, the BA process runs, triangulating the point features within a small subset of the relative poses selected as keyframes. A novel parallel initialization methodology was proposed that allows the system to successfully converge to an accurate solution despite no overlap in the component camera FOV, and therefore, large uncertainty on the initial point feature positions. This mechanism allows the proposed algorithm to overcome the limitations of the current methods in the literature and operate with completely non-overlapping camera FOV.

The non-overlapping FOV multicamera clusters using the proposed algorithm are able to provide accurate localization measurements as long as the solution to the optimization problem is not degenerate. The set of configurations for which the solution is under-constrained is larger than that of other camera setups with FOV overlap. Accordingly, the configurations of the cluster cameras, relative motion, and target model structure which result in a degenerate solution were identified in Chapter 4. The two-camera cluster with non-overlapping FOV system, observing a set of point features over two keyframes was considered for both the coupled and decoupled scenarios (refer to Section 2.2.1) using an optimization algorithm with a cost function based on least-squares image space error. The complete, systematic approach was presented and resulted in confirmation of previously-known concentric circle degenerate configurations, as well as identification of new configurations, including completely planar systems, and cluster translations along a quintic surface defined by the system geometry.

These analyses demonstrate that the set of degenerate configurations is significant and includes several common robotic motion scenarios, such as straight-line or constant-radius turns for a car-like robot, or heading-angle hold for a quadrotor robot. For these applications, the two-camera non-overlapping FOV cluster may not be an appropriate choice for the sensor configuration. This point reveals the importance of identifying the degenerate configurations of clusters with more cameras, observing more point features, to investigate what happens to the degeneracies in these cases. This subsequent analysis is proposed for future work in Section 6.1.2.

The proposed framework was implemented in software and hardware on an aerial robot platform, in Chapter 5, by modifying the MCPTAM algorithm. The system is able to provide accurate real-time estimates of the relative motion of the camera cluster and target environment. The performance of the algorithm was confirmed through a series of experiments comparing the estimated motion from the algorithm against measurements collected from a high-precision IPS for a variety of cluster configurations, through a set of relative motion profiles.

The relative motion trajectories were specifically selected to include testing the performance of a non-overlapping FOV cluster when the motion was at or near degenerate. Accordingly, the cluster was subjected to non-degenerate motion with a large amount of translation and rotation, near-degenerate motion with large translations but small rotations, and a two-camera cluster was swung in a pendulum motion, exercising the known degenerate motion of the cluster cameras moving in concentric circles.

For the non-degenerate motion, both the clusters with and without FOV overlap were able to accurately estimate the relative motion to within sub-centimetre and sub-degree precision. However, for the degenerate motions, the cluster with non-overlapping FOV was able to accurately estimate an up-to-scale solution but converged to an incorrect global scale. In the case of the hovering aerial robot, holding a constant heading angle is a common flight scenario. It was found that the slight pitching and rolling motions were insufficient to resolve the scale accurately, and therefore, more deliberate rotational motions need to be commanded for a cluster with completely non-overlapping FOV to resolve the scale. If this motion is not possible, it may be necessary to include some FOV overlap in the cluster cameras to resolve the scale ambiguity.

The performance of the algorithm was also confirmed outdoors in a challenging roof-top environment for cluster configurations both with and without FOV overlap. Several areas of the environment had only sparse point features for the estimator to localize against. However, the wide collective FOV in both cluster configurations allowed some cameras to robustly observe and track the available point features even when the other camera images were only sparsely populated. This was especially advantageous when exploring new areas of the environment as old, well-constrained point features could be observed in some cameras while the others made observations of new features at novel positions in the target model.

The work included within this thesis provides a novel algorithm capable of accurately tracking the relative pose of a non-overlapping FOV camera cluster and unknown target object, including the proper global scale, providing that the relative motion is not degenerate. The fundamental understanding of the structure of these degenerate configurations is enhanced by the presented analyses. Additionally, the experiments demonstrate that the algorithm was able to run on consumer hardware at real-time rates to produce accurate pose estimates in real environments. When the motion profiles are degenerate, the pose estimates are inaccurate in a predictable way, with the vast majority of the position error accounted for by the incorrect scale metric.

In summary, the main contributions claimed in this work are:

- A novel formulation for the multicamera cluster SLAM problem using $\boxplus$-manifolds for state space representation.
- A new point feature position update parameterization based on spherical coordinates to allow the system to converge to the proper global scale from uncertain initial estimates.
- A novel initialization scheme enabling the estimates to accurately converge despite no overlap in the camera FOV within the cluster.
- A novel analysis identifying configurations leading to solution degeneracy in the two-camera non-overlapping FOV cluster using an iterative nonlinear least-squares optimization process. This work is currently under review for publication [84].
- A real-time implementation of the proposed algorithms, mounted and run on a quadrotor aerial vehicle, and shown to produce highly-accurate pose estimates compared against high-precision IPS measurements.

6.1 Future Work

This section presents ideas for developing the methods and analyses from this thesis for further investigation.

6.1.1 Multicamera Cluster Framework

An interesting extension to the framework would be to allow for the relative poses of the cameras within the cluster to change through time. Each camera would have a known kinematic chain within the camera cluster frame. It could be as simple as a camera mounted on a gimbal, up to multiple cameras each mounted on the end-effector of its own robotic manipulator. The system could then be designed to actively change the camera configuration in order to help with the localization and modelling estimates, including:

- Bringing the FOV to overlap when the scale is insensitive due to small motions,
- Looking outwards covering the whole visual sphere with as large a baseline between the cameras as possible for high-accuracy localization and search,
- Avoiding concentric circular motion for a two-camera cluster,
- Arranging cameras for maximizing information-gain during inspection tasks.

An ideal application of this algorithm would be to self-reconfigurable robots [78]. Each module, or a subset of the modules could be fitted with simple embedded cameras. Each camera would then have a fixed kinematic chain to the others in the cluster. Such a system would represent the ultimate flexibility in the camera cluster sensor. The reconfigurable robot could then map and localize itself relative to a moving target object or environment. The study of robot configuration selection is then extended to the items mentioned in the list above. As an example, when the system wants to observe a target object accurately, the robot could assemble itself into a long arc to observe the object at right angles. Additionally, the task of reassembly could use the cameras to search for, locate, and dock with other robots or modules.

Two cameras could be brought very close together such that the images are similar and the stereo-matching takes place close in the image-space. Then, the baseline is increased and the correspondence is tracked through time to create the dense map. With the ability to control the baseline actively, the stereo-matching problem could be simplified.

6.1.2 Degenerate Configurations

Future research will determine the degenerate configurations for different sets of cluster cameras and point feature scenarios. First, the effect of using more than six features will be pursued to see if the degenerate quintic of cluster translations is diminished when additional features are added. The degenerate motions of the cluster such as no rotation, or concentric camera centre circles may remain, but the reduction would be significant. As most modern systems would observe possibly hundreds of point features at each keyframe, this scenario may be practically more relevant.

Next, when the number of cameras within the cluster is increased to three or more, it seems less likely, or impossible, that all cameras will travel in concentric circles between the two keyframes. Indeed, for four non-coplanar cameras, it is impossible for all of the camera centres to move in concentric circles with a common centre. The exact degenerate configurations for a cluster with more than two cameras is important to study since adding these additional cameras may further diminish the effect and provide a better localization solution which accurately captures global scale.

In a similar vein, this thesis does not consider the case when a feature point is initially observed in one camera at the first keyframe, then subsequently observed in a different camera at the second keyframe. This is a practically important case that occurs when the cluster motion is sufficient for one camera to observe an area previously observed by another. It may also relieve the constraint that the cluster must experience some rotation in order to determine the global scale metric. If possible, this would extend the usability of the camera cluster for localization to scenarios where the cluster orientation does not change significantly, but the translation and the camera FOVs are such that a set of features are likely to be seen in more than one camera at different points in time.

An important consideration for a system like MCPTAM is the selection of keyframe locations for the mapping process. There is a direct trade-off between including a sufficient number of keyframes to cover the trajectory with enough point features to prevent the tracker from getting lost, and the computational requirements that scale cubically in the number of keyframes. It is possible, though unlikely, that the subset of keyframes used in the mapping phase are chosen at poses which result in the BA being insensitive to the scale metric. The resulting map would have a poor scale estimate and will adversely affect the accuracy of the localization. It would be more ideal if a potential keyframe location could be quickly evaluated prior to adding it to the map to quantify the constraints that it would place on the scale metric. Then, the keyframes could be selected such that the scale estimate is improved compared to random keyframe selection. The MCPTAM system already includes several heuristic measures that determine when to add a keyframe to the map, the addition of this metric would add another factor to consider in the selection process.

The degeneracy measure would follow directly from the analysis in Chapter 4. Each measurement contains some amount of information about the scale. It is not clear whether it is more beneficial to have a large number of measurements with low scale-sensitivity, or a small number of features that are more sensitive to scale. It is also possible to evaluate the keyframe pose independent of the point feature measurements. In general, the metric needs to be capable of quantifying and comparing how close two keyframes are to being pair-wise degenerate.

6.1.3 Software Implementation

For the practical implementation of the MCPTAM algorithm, an interesting addition would be to have the tracking process maintain a queue of a certain size filled with the previous sets of images collected from the cluster cameras at the preceding time steps. Each frame would have a set of associated metrics related to pose in the world, number of known point features observed, etc. The algorithm can then analyze the trends in these metrics to determine when to add a new keyframe to the map. A simple example of how this would improve the operation is to consider a situation where the cluster is moving away from the area covered by the rest of the keyframes. If not enough point features from the target model are located in the camera images, the tracking thread cannot localize the cluster with respect to the target and the system is lost. However, with a queue of previous camera images, the pose tracking process could request that a frame from two time steps ago, when it was not lost, be added to the model. This would add new point features that may still be visible to the cluster cameras so that tracking can be reacquired and maintained.

A second simple example would be in efficiently selecting keyframes to maximally constrain the solution. In the case of non-overlapping FOV, it is advantageous to have observation rays to point features be as close to perpendicular as possible. By watching the trend for orientation changes in the frame buffer, the previous frame with the largest orientation magnitude can be selected as the ideal keyframe.

It would also be interesting to see if an algorithm could be found which would select keyframes similar to, or better than a human operator. It is unclear whether it would be best to use a deterministic approach, or whether to employ some form of machine learning algorithm to analyze the metrics and balance the computational cost to decide when it is best to place a keyframe.

A final possible direction for future research involves closing the control loop using the generated pose estimates from the algorithm to perform a relative positioning task. In the experiments from Chapter 5, the pose estimation was passive and did not affect the motion trajectory of the quadrotor aerial vehicle. Integrating the modified MCPTAM algorithm into the autonomous control architecture is an important step to enable real-time precision control for the quadrotor platform. With the estimates of the relative pose, the quadrotor can be commanded to fly relative motion trajectories, such as those suggested in Chapter 1, including landing on or inspecting a marine vessel.

When using the completely non-overlapping FOV cluster configurations, considerations will need to be made for how to deal with incorrectly-scaled motion estimates, particularly in the initial phase of the flight when the target model has not had a chance to fully converge. Despite this, the ability to track and position relative to moving target objects will vastly expand the applicability of robot platforms.

APPENDICES

Appendix A

⊞-Manifolds

This appendix presents a more formal definition of the ⊞-manifold in the style of Hertzberg *et al.* [33]. Definitions of the relevant mathematical objects are presented, along with some important properties of these manifolds. For a more rigorous treatment of the subject and proofs of the presented properties, the reader is referred to the original work [33].

First, some supporting definitions for general manifolds are presented:

Definition 1 [60] (Coordinate Chart) A coordinate chart is the pair (U,φ) consisting of an open set $U \subset \mathcal{S}$, along with the map $\varphi : U \to V$, where $V \subset \mathbb{R}^n$.

Definition 2 [60] (Homeomorphism) For open subset $U \subset \mathcal{S}$, a map $\varphi : U \to V$ is a homeomorphism if φ is bijective and continuous on U and $\varphi^{-1} : V \to U$ is continuous on V, where $\varphi^{-1}(\varphi(\mathbf{x})) = \mathbf{x}$, $\forall \mathbf{x} \in U$.

A mapping is homeomorphic if both it and its inverse are continuous.

Definition 3 [33] [60] (Smooth Function) For $\mathcal{S} \subset \mathbb{R}^s$, a function $\mathbf{f} : \mathcal{S} \to \mathbb{R}^n$ is smooth in $\mathbf{x} \in \mathcal{S}$ if there exist derivatives with respect to $\mathbf{x}$ up to arbitrary order.

Definition 4 [33] (Manifold) A smooth manifold is a pair $(\mathcal{S},\mathcal{F})$, or simply $\mathcal{S}$, consisting of a connected set $\mathcal{S} \subset \mathbb{R}^s$ and a family of coordinate charts $\mathcal{F} = (U_\alpha, \psi_\alpha)_{\alpha \in A}$ which cover $\mathcal{S}$, each with an open set $U_\alpha \subset \mathcal{S}$ and homeomorphism $\varphi_\alpha : U_\alpha \to V_\alpha$ of U_α to the open subset $V_\alpha \subset \mathbb{R}^n$, on the condition that if $U_\alpha \cap U_\beta \neq \emptyset$,

$$\varphi_\alpha \circ {\varphi_\beta}^{-1} : \varphi_\beta(U_\alpha \cap U_\beta) \to \varphi_\alpha(U_\alpha \cap U_\beta) \tag{A.1}$$

is a smooth function.

Informally, a manifold is a topological space where at every point in the manifold, there is a local neighbourhood that is homeomorphic with $\mathbb{R}^n$.

Now, the definition of the $\boxplus$-manifold defines a manifold with the addition of the two operators for manipulating and comparing points within the manifold.

Definition 5 [33] ($\boxplus$-Manifold) A $\boxplus$-manifold is a quadruple $(\mathcal{S},\boxplus,\boxminus,V)$, denoted as just $\mathcal{S}$, consisting of a subset $\mathcal{S} \subset \mathbb{R}^s$, operators

$$\boxplus : \mathcal{S} \times \mathbb{R}^n \rightarrow \mathcal{S}, \tag{A.2}$$

$$\boxminus : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}^n, \tag{A.3}$$

and an open neighbourhood $V \subset \mathbb{R}^n$ of $\mathbf{0} \in \mathbb{R}^n$. The mappings for the operators satisfy,

- $\forall \boldsymbol{\delta} \in \mathbb{R}^n$, $\boldsymbol{\delta} \mapsto \mathbf{x} \boxplus \boldsymbol{\delta}$ is smooth on $\mathbb{R}^n$,
- $\forall \mathbf{x}, \mathbf{y} \in \mathcal{S}$, $\mathbf{y} \mapsto \mathbf{y} \boxminus \mathbf{x}$ is smooth on U_x, where $U_x = \mathbf{x} \boxplus V$.

Every $\mathbf{x} \in \mathcal{S}$ is subject to the following axioms:

a) identity element,

$$\mathbf{x} \boxplus \mathbf{0} = \mathbf{x}, \tag{A.4}$$

b) onto on $\mathcal{S}$,

$$\mathbf{x} \boxplus (\mathbf{y} \boxminus \mathbf{x}) = \mathbf{y}, \quad \forall \mathbf{y} \in \mathcal{S}, \tag{A.5}$$

c) one-to-one on V,

$$(\mathbf{x} \boxplus \boldsymbol{\delta}) \boxminus \mathbf{x} = \boldsymbol{\delta}, \quad \forall \boldsymbol{\delta} \in V, \tag{A.6}$$

d) Lipschitz condition,

$$\|(\mathbf{x} \boxplus \boldsymbol{\delta}_1) \boxminus (\mathbf{x} \boxplus \boldsymbol{\delta}_2)\| \leq \|\boldsymbol{\delta}_1 - \boldsymbol{\delta}_2\|, \quad \forall \boldsymbol{\delta}_1, \boldsymbol{\delta}_2 \in \mathbb{R}^n, \tag{A.7}$$

where $\|\cdot\|$ is the vector p-norm [39] such that for $\mathbf{a} \in \mathbb{R}^n$,

$$\|\mathbf{a}\|_p = (|a_1|^p + \cdots + |a_n|^p)^{\frac{1}{p}}, \quad 1 \leq p < \infty, \tag{A.8}$$

and

$$\|\mathbf{a}\|_\infty = \max_i \ |a_i|, \tag{A.9}$$

With this definition, two important properties for the $\boxplus$-manifolds are presented next. For the least-squares optimization methods, the $\boxminus$ operator must allow for a comparison to create the concept of a distance on the manifold.

Induced Metric [33] The $\boxminus$ operator defines a metric $d_{\mathcal{S}} : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}$ by,

$$d_{\mathcal{S}}(\mathbf{x}, \mathbf{y}) := \|\mathbf{y} \boxminus \mathbf{x}\|, \quad \forall \mathbf{x}, \mathbf{y} \in \mathcal{S}. \tag{A.10}$$

The state space for the camera cluster system is composed of many keyframe poses and point feature positions. Each of these components uses its own $\boxplus$-manifold and the full state is constructed using the Cartesian product.

Cartesian Product $\boxplus$-Manifolds [33] The Cartesian product of two $\boxplus$-manifolds $\mathcal{S}_1$ and $\mathcal{S}_2$ is a $\boxplus$-manifold $\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2$, with $V = V_1 \times V_2$ and,

$$\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \boxplus \begin{bmatrix} \boldsymbol{\delta}_1 \\ \boldsymbol{\delta}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \boxplus_{\mathcal{S}_1} \boldsymbol{\delta}_1 \\ \mathbf{x}_2 \boxplus_{\mathcal{S}_2} \boldsymbol{\delta}_2 \end{bmatrix} \tag{A.11}$$

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} \boxminus \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{y}_1 \boxminus_{\mathcal{S}_1} \mathbf{x}_1 \\ \mathbf{y}_2 \boxminus_{\mathcal{S}_2} \mathbf{x}_2 \end{bmatrix}, \tag{A.12}$$

for $\mathbf{x}_1, \mathbf{y}_1 \in \mathcal{S}_1$, $\mathbf{x}_2, \mathbf{y}_2 \in \mathcal{S}_2$, $\boldsymbol{\delta}_1 \in \mathbb{R}^{n_1}$, and $\boldsymbol{\delta}_2 \in \mathbb{R}^{n_2}$.

As a result, the manifolds for each component can be treated separately and use their own $\boxplus$ and $\boxminus$ operators independently.

When defining the $\boxplus$ and $\boxminus$ operators for the state manifolds, it is useful to recognize that the groups $SO(3)$ and $SE(3)$ are Lie groups [33].

Definition 6 [49] (Lie Group) A Lie group is a smooth manifold $\mathcal{M}$ with smooth multiplication map $\mathbf{m} : \mathcal{M} \times \mathcal{M} \rightarrow \mathcal{M}$, and smooth inversion map $\mathbf{i} : \mathcal{M} \rightarrow \mathcal{M}$, such that for $\mathbf{x}, \mathbf{y} \in \mathcal{M}$,

$$\mathbf{m}(\mathbf{x}, \mathbf{y}) = \mathbf{x}\mathbf{y}, \tag{A.13}$$

and

$$\mathbf{i}(\mathbf{x}) = \mathbf{x}^{-1}. \tag{A.14}$$

As compact, connected Lie groups, there exist functions, called the exponential and logarithm maps, which are locally diffeomorphic maps between $\mathbb{R}^n$ and the Lie group [33]. These maps are used to define the $\boxplus$ and $\boxminus$ operators for the $\boxplus$-manifolds.

Appendix B

Previous Software Implementations

This appendix describes the software implementations for the Parallel Tracking and Mapping (PTAM) and Multi-Camera Parallel Tracking and Mapping (MCPTAM) algorithms. It is meant to provide the baseline for the modifications implemented for the work in this thesis, described in Section 5.1.1.

B.1 Parallel Tracking And Mapping (PTAM)

The PTAM framework was designed to work with a monocular camera and operates by decoupling the tasks of pose tracking and structure mapping in the SLAM algorithm into two parallel processes, suitable for small Augmented Reality (AR) workspaces. The tracking thread finds the current pose of the camera with respect to the point features in the environment map. The map points are fixed during the tracking so that the iterative nonlinear optimization only solves for the localization states for the current camera pose. This assumption allows the tracking thread to operate quickly since the dimension of the state space is dramatically reduced compared with the full SLAM solution. When the tracking thread determines that the current map is insufficient to maintain enough observations of point features in the environment, it sends the collected image, the point feature measurements, and the estimated current pose to the mapping thread to be added to the map as a new keyframe.

The mapping thread accepts keyframes from the tracking thread, then performs local and global BA on the keyframe poses and the point features within the map.

When there are spare cycles, the mapping thread will look through the previous keyframes to try and discover new point features in the map, or recover previously-failed measurements. When the map has converged, the updated map is passed back to the tracking thread to allow it to continue to localize with respect to the improved map estimate. This thread, due to the complexity of the computation for the nonlinear BA process, runs at a much lower rate than the tracking thread. The system can continue successfully as long as the supplied map is recent enough to contain point features observed in the current camera image.

One of the principle difficulties encountered by decoupling the tracking and mapping problems is that of initialization. The tracker process requires a good map to which it will localize the camera pose, while the mapping algorithm needs at least two keyframe poses with point feature measurements, and a reasonable initial estimate, to generate a map of the target structure. To resolve the stand-off, PTAM implements a completely separate initialization phase to seed the mapping thread with an initial condition and boot-strap the mapping process. On start-up, a set of point features are tracked in the 2D image-space across a short sequence of camera images as the camera is translated across the target environment. When a suitable baseline has been established, the operator presses a key to tell the algorithm to calculate the relative position and orientation of the first and last camera frames, as well as triangulate the depths to the tracked point features. This estimate is generated using the two-frame five-point VO method [61] and provides a rough initial condition to seed the mapping process. Since the system uses a single camera, the global scale of the solution is ambiguous. PTAM constrains the system scale by imposing a baseline length between the first two keyframes of 10 cm. While almost always incorrect, it does allow a consistent scale to be maintained.

A major advantage of the PTAM framework is the robust point image processing and feature matching front-end. An image pyramid is constructed, for each incoming camera frame, by sub-sampling the image successively by a factor of two. Point features are identified at each pyramid level in a camera image using the FAST corner detector [65, 66] and a descriptor is generated using the small image patch surrounding the interest point. At subsequent frames in the tracking algorithms, the previous point features are searched for, and matched against, using a pre-warped version of the descriptor patch compared with zero-mean Sum of Squared Difference (SSD) scores.

B.2 Multi-Camera PTAM (MCPTAM)

The MCPTAM system by Harmat *et al.* [30], is a modified version of the PTAM software for use with multiple heterogeneous wide-angle FOV lens central cameras rigidly connected together into a cluster. The algorithm has been completely integrated into the Robot Operating System (ROS) development environment.

The camera model of PTAM is replaced by a Taylor omnidirectional camera model [68] capable of representing devices with FOV greater than 180 degrees. To accommodate for the large radial distortions in these types of cameras, MCPTAM modifies the point feature patch-warping and matching process to search on the image-space epipolar arcs.

The PTAM BA back-end has been completely replaced by an optimization algorithm using the flexible g^{2}o [47] framework. Harmat *et al.* parameterize the rigid cluster configuration for the component cameras in a similar way to that presented in Chapter 3, as a collection of Keyframes (KF) with known position and orientation with respect to a base Multi-Keyframe (MKF), which itself has a position and orientation in the world frame which must be estimated. In this parameterization, the KFs correspond to the individual camera coordinate frames, C_i, and the MKFs correspond to the keyframes, K_k. The point features are parameterized using their Cartesian coordinates with respect to a common world coordinate frame, not with respect to the individual anchor coordinates frames, as in this thesis.

Unlike the PTAM algorithm, the MCPTAM system is capable of recovering the correct global scale of the motion and structure by taking advantage of overlap in the FOV of the component cameras. When a point feature is observed in two cameras at the same point in time, the depth of that feature can be accurately triangulated using the known calibration between the cameras. Along these lines, the image-based map initialization phase of PTAM is replaced by constructing an initial map of point features observed and triangulated within the intersection of the overlapping FOV of the cluster cameras at the first time step. Using this initialization method, the previous MCPTAM was confined to using clusters with sufficient FOV overlap to build a usable map model that allowed the system to subsequently track the initial motion.

Appendix C

System Calibration

This appendix presents the methodology for determining the extrinsic calibration of the cluster cameras with respect to the cluster coordinate frame. Additionally, the process for calibrating the locations of the tracking markers used by the Vicon Indoor Positioning System (IPS), with respect to the cluster frame is detailed. The latter calibration is necessary to compare these ground-truth measurements against the estimates generated by the proposed algorithm to verify their accuracy.

C.1 Extrinsic Calibration of Camera Cluster

The extrinsic calibration parameters for the relative poses of the cameras within the cluster were found using the method of Harmat *et al.* [30]. The method proceeds as follows. One of the cluster cameras is positioned to view an object with point features at known relative locations to each other. A checkerboard is used, where the size of the squares are precisely measured a priori. The image processing algorithm is able to identify the corners of the squares on the board. When the system recognizes the checkerboard pattern in the image, the user presses a button and these points are added to the target model at the measured positions. These points are then observed and tracked using the PTAM methodology until sufficient translation has occurred. At that time, new natural point features are added to the target model and triangulated to find their estimated position. The system proceeds in the manner of MCPTAM as the camera is moved around within, and maps, the environment. While the first camera continues to track its motion, the next camera in the cluster is brought to view the checkerboard pattern and the user again presses a button to signal that it is the same checkerboard being observed as that seen by the original camera. As a result, the second camera is localized into the same target model generated by the first camera and positioned with respect to the checkerboard model. The second camera is then also able to observe the other natural point features in the target object.

This process repeats until all of the cameras are brought into the same target model frame. It is important to ensure that all of the cameras observe the entire environment from as many different viewpoints as possible, as this constrains the calibration solution and increases the resulting accuracy. When the motion is complete, the user begins the calibration process. The poses of the cameras within the cluster are then optimized concurrently with the keyframe poses and point feature positions to minimize the image reprojection error for the camera measurements. The global scale of the solution is constrained since the checkerboard provides point features at known relative positions with respect to each other.

The results of the optimization are the positions and orientations of each component camera coordinate frame with respect to the selected primary cluster coordinate frame. The MCPTAM algorithm will use these extrinsic parameters to describe the configuration of the camera cluster in the tracking and mapping processes. The relative poses of the four cameras in the cluster used by the current experiments are again shown in Figure 5.2.

C.2 Calibration of IPS and MCPTAM Frames

The Vicon IPS provides high-precision measurements of the six DOF pose of the defined tracked object frame, T, containing the set of IR reflective spheres. This pose is measured with respect to a Vicon world frame W at each time step. The MCPTAM system provides estimates of the position and orientation of the camera cluster frame, U, with respect to a separate vision world frame M. In order to compare the two measured and estimated trajectories, they must be aligned into a common coordinate frame. This section describes the trajectory alignment procedure used in the experiments for this thesis.

The two world frames for the Vicon and vision systems, W and M, respectively, are fixed with respect to each other at all time. In addition, since the tracking markers are fixed with respect to the cluster cameras, the position and orientation of the trackable frame with respect to the cluster frame is similarly fixed for all time. The cycle of transformations relating all of these quantities together can be seen in Figure C.1.

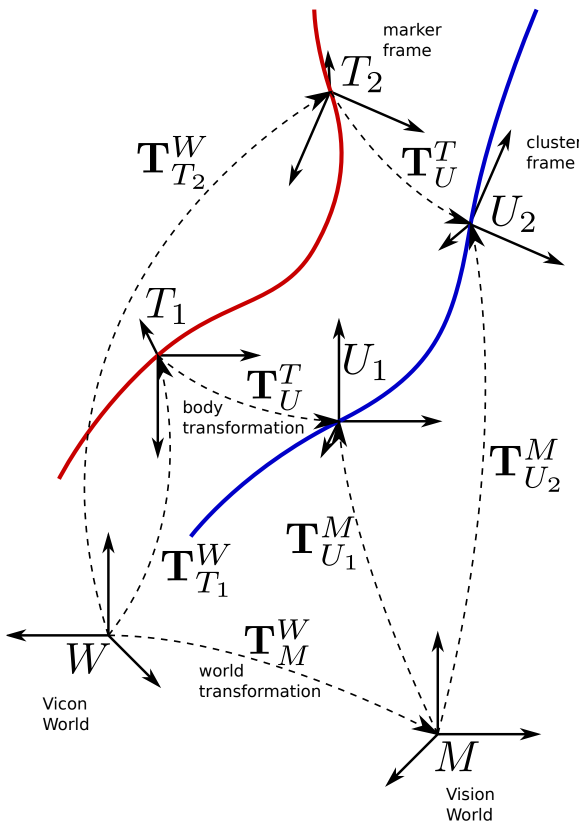


Figure C.1: The transformation chains relating the measurements of the pose trajectory by the Vicon IPS, $\mathbf{T}_{T_k}^{W}$, and MCPTAM, $\mathbf{T}_{U_k}^{M}$. To compare the trajectories, they must be aligned by determining $\mathbf{T}_{M}^{W}$ and $\mathbf{T}_{U}^{T}$

The motion quantities $\mathbf{T}_{T_k}^{W}$ and $\mathbf{T}_{U_k}^{M}$ correspond to the measured and estimated poses from Vicon and MCPTAM systems, respectively, at time step k. The remaining two transformations, $\mathbf{T}_{M}^{W}$ and $\mathbf{T}_{U}^{T}$, are constant and must be determined by a calibration process using a set of measured and estimated pose pairs from Vicon and MCPTAM. The transformation between the two world frames, $\mathbf{T}_{M}^{W}$, may change from run-to-run if the MCPTAM algorithm is started at different locations within the Vicon workspace. However, the body-fixed transformation, $\mathbf{T}_{U}^{T}$, can remain constant across different runs as long as the IR marker locations and the primary camera pose within the cluster are maintained.

Similar to Harmat *et al.* [30], these transformations are calibrated through a trajectory alignment process. First, the set of Vicon measurements, $\mathbf{T}_{T_k}^{W}$, and MCPTAM pose estimates, $\mathbf{T}_{U_k}^{M}$, at all time steps, k, are aligned in time. Then, the parameters of the world and body-fixed transformations are optimized as elements in $SE(3)$, as well as a common scale factor s to convert the coordinate transformations into similarity transformations,

$$\mathbf{S}_M^W = \begin{bmatrix} s\boldsymbol{\mathcal{R}}_M^W & \mathbf{t}_M^W \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in Sim(3), \tag{C.1}$$

$$\mathbf{S}_U^T = \begin{bmatrix} s\boldsymbol{\mathcal{R}}_T^U & \mathbf{t}_T^U \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in Sim(3). \tag{C.2}$$

For each time step in the trajectories, the error is calculated as the residual coordinate transformation around the transformation loop,

$$\mathbf{T}_W^W = \mathbf{S}_M^W \mathbf{T}_{U_k}^M \big(\mathbf{T}_{T_k}^W \mathbf{S}_U^T\big)^{-1} \tag{C.3}$$

$$= \begin{bmatrix} \boldsymbol{\mathcal{R}}_W^W & \mathbf{t}_W^W \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}, \tag{C.4}$$

which, in the absence of noise, should be identity. Note that the transformation is not a similarity transformation since the scale factor s cancels out of the rotational part of $\mathbf{T}_W^W$, but still affects $\mathbf{t}_W^W$. The error vector for the time step is found using the logarithm operator on $SE(3)$,

$$\bar{\mathbf{z}}_k = \log_{SE(3)}\big(\mathbf{T}_W^W\big). \tag{C.5}$$

This new calibration system is designed to allow any or all of the two $SE(3)$ transformations and one scale factor to be fixed during the trajectory alignment. Allowing the scale factor to vary facilitates comparing the accuracy of an up-to-scale solution and isolate the errors associated with an incorrect scale estimate.

The camera cluster was translated and rotated all around the Vicon tracking workspace to excite and constrain all of the pose DOFs during the image collection. It is important to have as much motion as possible to allow the MCPTAM algorithm to obtain accurate pose estimates after generating a stable accurately-scaled map through triangulation on wide baselines with sufficient orientation changes, as well as to ensure large varied rotations since the calibration is unable to localize the relative position of the coordinate frames if all of the rotations are about a common axis [7]. In the latter case, the distance between the coordinate frames along the axis of rotation is ambiguous.

The MCPTAM algorithm is run on the collected images multiple times and is allowed to generate a stable target map model. The resulting pose estimates from the tracking thread are collected and the trajectory is aligned with the Vicon measurements using the above optimization with the scale parameter fixed at unity,

$s \equiv 1$. This results in an extrinsic calibration for the position and orientation of the camera cluster frame with respect to the IR marker frame. It is now possible to directly compare the measurements from the Vicon system with the pose estimates from the MCPTAM tracking thread.

References

[1] S Agarwal, N Snavely, S M Seitz, and R Szeliski. Bundle adjustment in the large. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 2, pages 29–42, 2010.

[2] S Agarwal, N Snavely, I Simon, S M Seitz, and R Szeliski. Building Rome in a day. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 72–79, October 2009.

[3] M Agrawal. A Lie algebraic approach for consistent pose registration for general Euclidean motion. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1891–1897, 2006.

[4] T Bailey and H Durrant-Whyte. Simultaneous localization and mapping (SLAM): part II. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

[5] P Baker, C Fermuller, Y Aloimonos, and R Pless. A spherical eye from multiple cameras (makes better models of the world). In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 576–583, 2001.

[6] H Bay, A Ess, T Tuytelaars, and L Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[7] J Brookshire and S J Teller. Extrinsic calibration from per-sensor egomotion. In *Robotics: Science and Systems Conference (RSS)*, 2012.

[8] A A G Carrera and A J Davison. Lightweight SLAM and navigation with a multi-camera rig. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, pages 77–82, 2011.

[9] W Y Chang and C S Chen. Pose estimation for multiple camera systems. In *Proceedings of the International Conference on Pattern Recognition (ICPR)*, volume 3, pages 262–265, August 2004.

[10] J Civera, A J Davison, and J M M Montiel. Dimensionless monocular SLAM. *Pattern Recognition and Image Analysis*, 4478:412–419, 2007.

[11] J Civera, A J Davison, and J M M Montiel. Inverse depth to depth conversion for monocular SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2778–2783, 2007.

[12] J Civera, A J Davison, and J M M Montiel. Inverse depth parametrization for monocular SLAM. *IEEE Transactions on Robotics*, 24(5):932–945, 2008.

[13] B Clipp, J H Kim, J M Frahm, M Pollefeys, and R Hartley. Robust 6DOF motion estimation for non-overlapping, multi-camera systems. In *Proceedings of the IEEE Workshop on Applications of Computer Vision (WACV)*, pages 1–8, January 2008.

[14] A I Comport, R Mahony, and F Spindler. A visual servoing model for generalised cameras: Case study of non-overlapping cameras. In *Proceedings of the IEEE International Conference of Robotics and Automation (ICRA)*, pages 5683–5688, May 2011.

[15] J Crassidis and F Markley. Attitude estimation using modified Rodrigues parameters. In *Proceedings of the Flight Mechanics/Estimation Theory Symposium*, pages 71–83, 1996.

[16] Y Dai, M He, H Li, and R Hartley. Factorization-based structure-and-motion computation for generalized camera model. In *Proceedings of the IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pages 1–6, 2011.

[17] A J Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1403–1410, 2003.

[18] A J Davison, A G Cid, and N Kita. Real-time 3D SLAM with wide-angle vision. In *Proceddings of the IFAC Symposium on Intelligent Autonomous Vehicles*, 2004.

[19] A J Davison, I D Reid, N D Molton, and O Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007.

[20] M Deans and M Hebert. Experimental comparison of techniques for localization and mapping using a bearing-only sensor. *Lecture Notes in Control and Information Sciences*, 271:395–404, 2000.

[21] L Deng, W J Wilson, and F Janabi-Sharifi. Decoupled EKF for simultaneous target model and relative pose estimation using feature points. In *Proceedings of the IEEE Conference on Control Applications (CCA)*, pages 749–754, 2005.

[22] M Dissanayake, P Newman, S Clark, H Durrant-Whyte, and M Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[23] H Durrant-Whyte and T Bailey. Simultaneous localization and mapping (SLAM): part I. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006.

[24] E Eade and T Drummond. Scalable monocular SLAM. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 469–476, 2006.

[25] C Fermuller and Y Aloimonos. Observability of 3D motion. *International Journal of Computer Vision*, 37(1):43–63, 2000.

[26] D A Forsyth and J Ponce. *Computer vision: a modern approach.* Prentice Hall, 2003.

[27] A Gelb. *Applied optimal estimation.* The MIT Press, 1999.

[28] M D Grossberg and S K Nayar. A general imaging model and a method for finding its parameters. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 108–115, 2001.

[29] P Gupta, N da Vitoria Lobo, and J J Lariola Jr. Markerless tracking and gesture recognition using polar correlation of camera optical flow. *Machine Vision and Applications*, 24(3):651–666, 2013.

[30] A Harmat, I Sharf, and M Trentini. Parallel tracking and mapping with multiple cameras on an unmanned aerial vehicle. In *Proceedings of the International Conference on Intelligent Robotics and Applications*, volume 1, pages 421–432, 2012.

[31] R Hartley and A Zisserman. *Multiple view geometry in computer vision.* Cambridge University Press, 2003.

[32] R Hermann and A Krener. Nonlinear controllability and observability. *IEEE Transactions on Automatic Control*, 22(5):728–740, 1977.

[33] C Hertzberg, R Wagner, U Frese, and L Schröder. Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds. *Information Fusion*, 14(1):57–77, 2013.

[34] C Hu and L Cheong. Linear quasi-parallax SfM using laterally-placed eyes. *International Journal of Computer Vision*, 84(1):21–39, 2009.

[35] T Hui and R Chung. Determining shape and motion from non-overlapping multi-camera rig: A direct approach using normal flows. *Computer Vision and Image Understanding*, 117(8):947–964, 2013.

[36] M Kaess and F Dellaert. Visual SLAM with a multi-camera rig. *Tech. Rep. GIT-GVU-06-06*, February 2006.

[37] M Kaess and F Dellaert. Probabilistic structure matching for visual SLAM with a multi-camera rig. *Computer Vision and Image Understanding*, 114(2):286–296, 2010.

[38] T Kazik, L Kneip, J Nikolic, M Pollefeys, and R Siegwart. Real-time 6D stereo visual odometry with non-overlapping fields of view. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1529–1536, 2012.

[39] H K Khalil. *Nonlinear systems.* Prentice Hall, Third edition, 2002.

[40] J H Kim, M J Chung, and B T Choi. Recursive estimation of motion and a scene model with a two-camera system of divergent view. *Pattern Recognition*, 43(6):2265–2280, 2010.

[41] J H Kim, R Hartley, J M Frahm, and M Pollefeys. Visual odometry for non-overlapping views using second-order cone programming. In *Proceedings of the Asian Conference on Computer Vision*, volume 2, pages 353–362, 2007.

[42] J H Kim, H Li, and R Hartley. Motion estimation for nonoverlapping multi-camera rigs: Linear algebraic and L_∞ geometric solutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):1044–1059, June 2010.

[43] J S Kim, M Hwangbo, and T Kanade. Spherical approximation for multiple cameras in motion estimation: Its applicability and advantages. *Computer Vision and Image Understanding*, 114(10):1068–1083, 2010.

[44] J S Kim and T Kanade. Degeneracy of the linear seventeen-point algorithm for generalized essential matrix. *Journal of Mathematical Imaging and Vision*, 37(1):40–48, May 2010.

[45] G Klein and D Murray. Parallel tracking and mapping for small AR workspaces. In *Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 225–234, 2007.

[46] K Konolige. Sparse sparse bundle adjustment. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 102.1–102.11, August 2010.

[47] R Kümmerle, G Grisetti, H Strasdat, K Konolige, and W Burgard. g2o: A general framework for graph optimization. In *Proceddings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2011.

[48] G H Lee, F Fraundorfer, and M Pollefeys. Motion estimation for self-driving cars with a generalized camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[49] J M Lee. *Introduction to smooth manifolds.* Springer, Second edition, 2013.

[50] H Li, R Hartley, and J H Kim. A linear approach to motion estimation using generalized camera models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, June 2008.

[51] H C Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[52] D G Lowe. Object recognition from local scale-invariant features. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2:1150–1157, 1999.

[53] Y Lu, J Z Zhang, Q M J Wu, and Z N Li. A survey of motion-parallax-based 3-D reconstruction algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 34(4):532–548, 2004.

[54] C Madhusudan. Error analysis of the Kalman filtering approach to relative position estimation using noisy vision measurements. Master's thesis, University of Waterloo, 1992.

[55] J McPhee. A unified graph-theoretic approach to formulating multibody dynamics equations in absolute or joint coordinates. *Journal of the Franklin Institute*, 334(3):431–445, 1997.

[56] M Meilland, A I Comport, and P Rives. Dense visual mapping of large scale environments for real-time localisation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4242–4248, 2011.

[57] M Montemerlo, S Thrun, D Koller, and B Wegbreit. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 18, pages 1151–1156, 2003.

[58] E Mouragnon, M Lhuillier, M Dhome, F Dekeyser, and P Sayd. Generic and real-time structure from motion using local bundle adjustment. *Image and Vision Computing*, 27(8):1178–1193, 2009.

[59] R M Murray, Z Li, and S S Sastry. *A mathematical introduction to robotic manipulation*. CRC Press, 1994.

[60] H Nijmeijer and A van der Schaft. *Nonlinear dynamical control systems*. Springer-Verlag, 1990.

[61] D Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–770, June 2004.

[62] T Oskiper, R Kumar, J Fields, and S Samarasekera. Vehicle 3D pose tracking using distributed aperture sensors. In *Proceedings of SPIE*, volume 6230, page 62301X, 2006.

[63] R Pless. Using many cameras as one. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–587–593, June 2003.

[64] M E Ragab and K H Wong. Multiple nonoverlapping camera pose estimation. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 3253–3256, September 2010.

[65] E Rosten and T Drummond. Fusing points and lines for high performance tracking. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1508–1511, October 2005.

[66] E Rosten and T Drummond. Machine learning for high-speed corner detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 1, pages 430–443, May 2006.

[67] T Sato, S Ikeda, and N Yokoya. Extrinsic camera parameter recovery from multiple image sequences captured by an omni-directional multi-camera system. *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 326–340, 2004.

[68] D Scaramuzza. *Omnidirectional vision: from calibration to robot motion estimation.* PhD thesis, ETH Zurich, 2007.

[69] G Schweighofer and A Pinz. Fast and globally convergent structure and motion estimation for general camera models. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 147–157, 2006.

[70] G Schweighofer, S Segvic, and A Pinz. Online/realtime structure and motion for general camera models. In *Proceedings of the IEEE Workshop on Applications of Computer Vision (WACV)*, pages 1–6, 2008.

[71] G Sibley, C Mei, I Reid, and P Newman. Adaptive relative bundle adjustment. In *Robotics: Science and Systems Conference (RSS)*, pages 1–8, 2009.

[72] E W Y So, T Yoshimitsu, and T Kubota. Divergent stereo visual odometry for a hopping rover on an asteroid surface. In *Proceedings of the International Symposium on Space Artificial Intelligence, Robotics, and Automation for Space (i-SAIRAS)*, 2010.

[73] J Solà. Consistency of the monocular EKF-SLAM algorithm for three different landmark parametrizations. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3513–3518, 2010.

[74] J Solà, A Monin, M Devy, and T Vidal-Calleja. Fusing monocular information in multicamera SLAM. *IEEE Transactions on Robotics*, 24(5):958–968, October 2008.

[75] H Stewenius and K Astrom. Structure and motion problems for multiple rigidly moving cameras. *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 252–263, 2004.

[76] H Stewenius, D Nister, M Oskarsson, and K Astrom. Solutions to minimal generalized relative pose problems. In *Proceedings of the Workshop on Omnidirectional Vision*, October 2005.

[77] H Stewenius and M Oskarsson. Camera platforms for localization and map building. In *Proceedings of the Symposium on Image Analysis, SSBA*, 2004.

[78] K Stoy, D Brandt, and D J Christensen. *Self-Reconfigurable Robots: An Introduction*. The MIT Press, 2010.

[79] H Strasdat, A J Davison, J M M Montiel, and K Konolige. Double window optimisation for constant time visual SLAM. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2352–2359, 2011.

[80] H Strasdat, J M M Montiel, and A J Davison. Visual SLAM: Why filter? *Image and Vision Computing*, 30(2):65–77, 2012.

[81] P Sturm. Multi-view geometry for general camera models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 206–212, 2005.

[82] S Thrun, W Burgard, and D Fox. *Probabilistic robotics*. The MIT Press, 2005.

[83] G Tong, X Pang, N Ye, Z Jiang, and H Zhang. A precise spherical camera model based on multi-camera system. *Journal of Computational Information Systems*, 9(3):897–905, 2013.

[84] M J Tribou, S L Waslander, and D W L Wang. Scale recovery in multicamera cluster SLAM with non-overlapping fields of view. Submitted to *Computer Vision and Image Understanding*, August 2013.

[85] R Valkenburg and N Alwesh. Calibration of target positions using conformal geometric algebra. In *Guide to Geometric Algebra in Practice*, pages 127–148. Springer London, 2011.

[86] M R Walter, R M Eustice, and J J Leonard. Exactly sparse extended information filters for feature-based SLAM. *The International Journal of Robotics Research*, 26(4):335–359, 2007.

[87] J Weng and T S Huang. Complete structure and motion from two monocular sequences without stereo correspondence. In *Proceedings of the IAPR International Conference on Pattern Recognition*, volume 1, pages 651–654, 1992.

[88] N L White. Grassmann-Cayley algebra and robotics. *Journal of Intelligent and Robotic Systems*, 11(1-2):91–107, 1994.

[89] W J Wilson, C C W Hulls, and G S Bell. Relative end-effector control using Cartesian position based visual servoing. *IEEE Transactions on Robotics and Automation*, 12(5):684–696, 1996.