CRANFIELD UNIVERSITY

School of Aerospace, Transport and Manufacturing

Pedro Cavestany

Distributed Scene Reconstruction From Multiple Mobile Platforms

PhD Thesis

Academic year 2014/2015

Supervisors:

Dr. Toby Breckon

Dr. Humberto Martínez-Barberá

May 2015

This thesis is submitted in partial fulfillment of the requirements for the degree of *Doctor of Philosophy* (PhD)

Abstract

Recent research on mobile robotics has produced new designs that provide house-hold robots with omnidirectional motion. The image sensor embedded in these devices motivates the application of 3D vision techniques on them for navigation and mapping purposes. In addition to this, distributed cheap-sensing systems acting as unitary entity have recently been discovered as an efficient alternative to expensive mobile equipment.

In this work we present an implementation of a visual reconstruction method, structure from motion (SfM), on a low-budget, omnidirectional mobile platform, and extend this method to distributed 3D scene reconstruction with several instances of such a platform.

Our approach overcomes the challenges yielded by the plaform. The unprecedented levels of noise produced by the image compression typical of the platform is processed by our feature filtering methods, which ensure suitable feature matching populations for epipolar geometry estimation by means of a strict quality-based feature selection. The robust pose estimation algorithms implemented, along with a novel feature tracking system, enable our incremental SfM approach to novelly deal with ill-conditioned inter-image configurations provoked by the omnidirectional motion. The feature tracking system developed efficiently manages the feature scarcity produced by noise and outputs quality feature tracks, which allow robust 3D mapping of a given scene even if - due to noise - their length is shorter than what it is usually assumed for performing stable 3D reconstructions.

The distributed reconstruction from multiple instances of SfM is attained by applying loop-closing techniques. Our multiple reconstruction system merges individual 3D structures and resolves the global scale problem with

minimal overlaps, whereas in the literature 3D mapping is obtained by overlapping stretches of sequences. The performance of this system is demonstrated in the 2-session case.

The management of noise, the stability against ill-configurations and the robustness of our SfM system is validated on a number of experiments and compared with state-of-the-art approaches. Possible future research areas are also discussed.

Acknowledgements

I feel immensely grateful to my supervisor Dr. Toby Breckon, who has constanly followed and directed my progress. His stimulating comments and suggestions throughout this work have been an encouraging motivation in my research. I am also grateful to Dr. Humberto Martínez-Barberá who proposed the distributed aspect of the thesis, and sought the necessary fundings for the realisation of my PhD. I would like to acknowledge as well the work done by Dr. Yifan Zhao on the bureaucratic front during the last stage of the thesis and in the arrangements of the viva.

I want to acknowledge the valuable academic support provided by Dr. Antonio L. Rodríguez, who has always kindly attended my queries related to 3D vision, and whose advice was decisive in developing our SfM approach.

I find very difficult to convey how much, and in how many aspects, I have learned and enjoyed the company and enriching conversations of Dr. Robert Sawko. I am absolute certain that many experiences and ideas that we have shared have made their way through in this thesis. I harbour similar feelings regarding Dr. Juan José Alcaraz, whose bright approaches to problems have enlightened me in many occasions.

During my PhD I have met people so interesting that I am sorry to just list them here. Belén Iglesias, Sarah Bergin, Alex Charlton, Luca Camosi, Marcin and Milena Tracyk, Michał and Gosia Czapinski, Reuben Yarwood, Marianne Hewitt, Morag Skinner... all of them have left somehow their imprint on this work.

Last but definitely not least, I owe a great deal of this thesis to my closest family: my brothers and sisters, for their support (not limited to affection)

and encouragement, but most of all to my parents, who have provided unconditional assistance in all sort of areas. I would need a whole chapter to express their contributions to this thesis.

This work has been supported by a Departmental scholarship of the Applied Math- ematics and Computing Group, of School of Engineering, Cranfield University, and by a grant of the Regional Employment Service of Murcia (Spain), through the Science and Technology Regional Office, Séneca Foundation.

Contents

List of Figures **vii**

List of Tables **xi**

1 Introduction **1**

1.1 Robotics and 3D Vision 1
1.2 Structure From Motion 4
1.3 Objectives and Research Question 9
1.4 Approach Overview 12
1.5 Thesis Outline 12

2 Literature Review **15**

2.1 Robotics and SfM 17
2.1.1 Low-Budget Robots 18
2.1.2 Omnidirectional Robots 19
2.2 Image Noise 21
2.3 Image Features 23
2.3.1 Feature Detectors and Descriptors 23
2.4 The Correspondence Problem 28
2.4.1 Feature Matching 28
2.4.2 Robust Estimation of Epipolar Geometry 32
2.5 Feature Tracking 35
2.6 Motion Recovery 37
2.6.1 SfM in Real Time 38
2.6.2 Epipolar Geometry 39
2.6.3 SfM on Robots and Navigation 42

2.6.4 Ill-Configurations . . . 45
2.6.5 Resection . . . 47
2.7 Structure Triangulation . . . 48
2.8 Bundle Adjustment . . . 50
2.9 Collaborative Perception . . . 54
2.9.1 Distributed Reconstruction and Localisation . . . 55
2.9.2 Different Configurations . . . 57
2.10 Summary . . . 58

3 Structure from Motion on a Single Platform 61
3.1 The Platform: Rovio . . . 62
3.1.1 Noise . . . 66
3.1.2 Small Baselines . . . 67
3.2 Camera Calibration . . . 68
3.3 Image Reception and Preprocessing . . . 71
3.4 Matching Process . . . 73
3.4.1 Recursive Matching Process . . . 74
3.4.2 Spurious Matches Trimming . . . 74
3.4.3 Feature Detection and Matching . . . 77
3.5 Relative Pose Estimation . . . 79
3.6 Feature Tracking System . . . 81
3.7 Global Pose Estimation and Structure . . . 87
3.7.1 Resection . . . 87
3.7.2 3D Structure . . . 88
3.8 Bundle Adjustment . . . 89
3.8.1 Quaternions . . . 89
3.8.2 Local and Global BA . . . 90
3.9 Final Scene Recovery . . . 90
3.9.1 Surface Rendering . . . 92
3.10 Summary . . . 93

4 Structure from Motion across Multiple Platforms **95**

4.1 Multiple Structure from Motion: our Approach 96

4.2 Multiple Reconstruction 98

4.2.1 Loop Closing 99

4.2.2 FAB-MAP 102

4.3 Overlap Management 108

4.3.1 Session $\mathcal{I}_1$ 109

4.3.2 Session $\mathcal{I}_i$, $i > 1$ 110

4.3.3 Distributed Bundle Adjustment 112

4.4 Post-Process 113

4.5 The Case $r = 1$ 114

4.6 Summary 115

5 Performance Evaluation **117**

5.1 Methodology for Evaluation 118

5.2 Single Case 121

5.2.1 Experiments 124

5.2.2 Evaluation of the System 124

5.2.3 Comparison with State of the Art Systems 134

5.2.4 Validation against Benchmark Data-Sets 138

5.2.5 Evaluation against Ground-Truth 139

5.3 Multiple Case 142

5.3.1 Multi-Session Evaluation 143

5.3.2 Loop-Closings in Single Sessions 146

5.4 Summary 147

6 Conclusions **149**

6.1 Contributions 149

6.2 Future Work 151

References **153**

Appendix **177**

A Structure from Motion - a Geometric Overview **177**

A.1 Homogeneous Coordinates 178

A.2 Camera Calibration 180

A.2.1 The Intrinsic Matrix 180

A.2.2 Normalised Coordinates 182

A.2.3 The Camera Matrix 183

A.3 Homographies 184

A.4 The Epipolar Geometry 187

A.4.1 The Fundamental Matrix 188

A.4.2 The Essential Matrix 191

A.4.3 Estimation of the Epipolar Geometry 194

A.4.4 The Essential Space 196

A.4.5 Extraction of R and $\boldsymbol{t}$ 196

A.5 Sequential SfM: the Problem of the Scale. 198

A.5.1 Resection 200

A.6 Reconstruction 201

A.6.1 Triangulation 202

A.7 Bundle Adjustment 203

A.7.1 Reprojection Error 203

A.7.2 Levenberg–Marquardt Algorithm 204

B Algebraic Definitions **207**

B.1 Hat operator and cross product 207

B.2 Line between two points 208

List of Figures

1.1 A 3D reconstruction model 2
1.2 Example of 3D software: *Photo tourism* 4
1.3 Dense 3D reconstruction: DTAM 5
1.4 3D vision application: Artificial lens blur 6
1.5 The mobile robot Rovio 8
1.6 Omni-wheel and possible motion directions 10

2.1 Critical sequences in uncalibrated motion 19
2.2 Feature detection and matching 24
2.3 Robust estimation applied to linear regression 32

3.1 Flowchart of SfM process 62
3.2 Rovio features 63
3.3 A blurred image delivered by Rovio. 64
3.4 Software environment 65
3.5 Mosquito effect 66
3.6 Checkerboad "Tsai grid" used for camera calibration. 69
3.7 Bilateral filtering 72
3.8 Effect of bilateral filtering over feature detection. 73
3.9 Typical distribution of distance between descriptors in a match 75
3.10 Trimming effect of the noise filters on the set of corresponding features . 76
3.11 Error correspondence on the projection of a 3D point 77
3.12 Merging of two features in a *bundle* 82
3.13 Filters employed in the feature tracking system 83
3.14 Feature tracking system: case 1 84

3.15 Feature tracking system: case 2 85
3.16 Feature tracking system: case 3 86
3.17 Histogram of feature track lengths 88
3.18 Flowchart of the post-processing stage 91
3.19 Surface rendering of 3D reconstructions 93

4.1 2D mapping from direct encounters 97
4.2 A general scenario of multiple reconstruction 98
4.3 Example of loop-closure 99
4.4 A false loop-closure 100
4.5 Example of sequence overlapping 101
4.6 Vocabulary of bags-of-words (BOW) 102
4.7 Categories of the training data-set for vocabulary generation 103
4.8 Precision and recall 105
4.9 loop-closure occurrence with the platform used 107
4.10 The initial position problem 108
4.11 BA management of overlaps 113

5.1 Experiments studied in the single case 122
5.2 Camera poses and 3D structures of the four experiments 123
5.3 Structure information on the data-sets referenced 125
5.4 Statistical information on the data-sets referenced 126
5.5 Epipolar lines in *visionlab* and *turntable* sequences 126
5.6 Influence of baseline distance on reprojection error 127
5.7 Comparison of results between SfM process and post-process 128
5.8 Comparison of feature track histograms between SfM process and post-process 129
5.9 Comparison of 3D structures between SfM process and post-process . . . 130
5.10 Computational times in SfM process and post-process 131
5.11 Computational times in the global BA thread 132
5.12 Rendered surface of *turntable* sequence 133
5.13 Point cloud of the *turntable* sequence 134
5.14 Rendered surface of *industrialArea* sequence 135
5.15 Camera poses and 3D structure by state of the art systems 136

5.16 Reconstruction of Leuven Castle by evaluated systems 139
5.17 *engineRoom* Experiment 139
5.18 Validation of the studied cases against ground-truth 140
5.19 Detail of the comparison with Ground-Truth on *engineRoom0* sequence. 141
5.20 Validation of *turntable* sequence against ground-truth 141
5.21 Omnidirectional motions in *pipeline* sequence. 142
5.22 Camera poses reconstructed by state of the art softwares in the *pipeline* sequence. 142
5.23 camera poses and 3D structure of the *engineRoom0* and *engineRoom1* sequences. 144
5.24 Camera poses and 3D structure of the merged sessions. Since all the cameras are joined by a line, there is a line joining the last camera of the first session and the first camera of the second session 145
5.25 Comparison between ground-truth and our results in the merged camera poses of the *engineRoom* experiment. 145
5.26 Camera poses in the sequence *turntable2*. 146
5.27 Loop closing evaluation in the case $r = 1$. 147

A.1 The pinhole camera 178
A.2 The geometric camera model 180
A.3 Central projection 183
A.4 The epipolar geometry 186
A.5 Transfer of projections through homography 187
A.6 The triple scalar product 192
A.7 The triangulation problem 202

LIST OF FIGURES

List of Tables

2.1 Necessary samples to ensure an outlier-free model in robust estimation . 34

3.1 Effects of bilateral filtering 73
3.2 Comparison between feature detectors and descriptors 78
3.3 Evaluation of different query tree structures of FLANN 78
3.4 Computational times taken by FLANN implementations 78
3.5 Possible cases in the feature tracking system 85
3.6 Surviving matches after each filter during the post-process stage 91
3.7 Comparison between SfM process and post-processing 92

4.1 Precision and recall indices for *engineRoom* and *turntable2* sequences. . 106

5.1 Lengths of feature track covered by bundles 129
5.2 Comparison of results with state of the art systems 135
5.3 Ratios regarding the projections generated on *visionlab* sequence. 136
5.4 Comparison of results between our system and state of the art systems . 137
5.5 Ratios regarding the projections generated on *turntable* sequence. 137
5.6 Comparison of the reprojection error given by our system and the state of the art systems on the sequences studied. 138
5.7 Comparison of our system and state of the art systems on the Leuven castle data-set. 138
5.8 Overlap detection in multiple reconstruction 143
5.9 3D structure statistics in multiple reconstruction 146

Chapter 1

Introduction

This thesis addresses research challenges within the domain of robotic visual sensing in specific relation to 3D perception of an environment by one or more robotic platforms.

1.1 Robotics and 3D Vision

Robotics is used in a vast number of fields: industry, military and medicine represent just a small example of the increasing importance that robots are gaining in many aspects of our society. Indeed, robots have recently found their way into the domestic world. As time goes by, more mobile robots are affordable by household budgets, thanks to increasing returns to scale and simpler designs. Alongside the integration of robotics into our domestic environment we additionally see that research in the area of computer vision has been introduced into this domestic sphere as the primary sensing mechanism for such types of robot. Finally, distributed cheap-sensing systems acting as unitary entity have recently been discovered as an efficient alternative to expensive mobile equipment.

By applying mathematical and computational algorithms computer vision extracts information out of images. The range of knowledge that can be retrieved with computer vision is wide, but in this work we will focus on 3D vision. 3D vision is a part of computer vision that tackles the extraction of 3D information from a scene covered by a given set of input images. It also provides information about the relative location of each image within the sequence. Robotics can take advantage of this technique for

Figure 1.1: A 3D reconstruction model attained with computer vision. Source: Olsson et al. (2010).

mapping and navigation purposes (Nister et al. (2004)). We intend to employ these techniques in a distributed fashion.

The mathematical substratum of 3D vision methods is projective geometry, which studies the geometric properties which are preserved under projective transformations. The use of projective geometry started in the Renaissance (14th - 17th centuries), when painters learned how to depict scenes using the laws of perspective, such as vanishing points. Later on, in the era of computers, projective geometry has been employed for rendering real or synthetic scenes or objects in a large variety of applications.

3D vision researchers take the theory of projective geometry in the opposite direction. Instead of projecting a given 3D scene on a 2D image, the postulates of projective geometry are applied to a sequence of images in combination with statistical inference methods and other algorithms from image processing in order to obtain information about where each image was taken, relatively to each other, and about the 3D structure of the scene. This is the inception of the discipline multiple view geometry (MVG), which studies the geometric projective relationships between multiple views of a common environment (Hartley and Zisserman (2004)). Appendix A describes the necessary background in MVG and projective geometry to understand the algorithms used by this work. Fig 1.1 shows the 3D reconstruction of a scene performed out of images with MVG techniques, along with the estimations of the locations where each picture was taken.

Multiple view geometry finds in mobile robots a most proper field of application, and with the increasing use of mobile robots in so many areas, the methods offered by MVG have become of common use in robotics. Indeed, often robots are used in hazardous and inaccessible environments, such as a mine, a pipe or a chamber filled with toxic gas. These premises have boosted the development of mobile and autonomous robots, in order to perform those tasks with precision and without risk to human life. Whereas fixed robots are widespread in industry, mobile robots are still maturing their techniques, mainly because it is very difficult for a mobile robot to maintain a reliable estimation of its location within an unknown (or even known) scene while transiting it. MVG techniques not only allow to create a 3D map of the environment transited, but also provide the relative location of the mobile robot as it moves around. Therefore, researchers have shown great interest in 3D vision applied to mobile robotics as a primary method to perform visual navigation.

Inside the area of mobile robotics, multiple robot systems and swarm systems have recently been also attracting research work (Jeong and Lee (2013); Kim et al. (2010); Radke (2008)). Based on relatively simple rules that each individual has to abide by, the synergetic aggregation of individual robots into a unified group results in complex and nearly inteligent behaviours. Examples of this are present in nature: Many insects, birds and fish take advantage of this mechanism. Given how expensive a fully-equipped mobile robot is, a great deal of research is devoted to develop distributed systems so that it is possible for groups of affordable, commercially available robots accomplish tasks that otherwise would need a high-technologically developed and expensive single robot.

Distributed systems of robots where each one is equipped with a sensor, such as GPS, infrared, ultra-sound sensors or even wireless signal receptor can take advantage of the information gathered by each individual (Chang and Wu (2013); Otsuka et al. (2013); Wendel et al. (2012)). By bringing together the sensor information along with other parameters (location, time, etc.) into a common pool and taking into account the uncertainty involved in the problem, a distributed robotic system (DRS) can extract accurate and robust knowledge about the environment. We will implement this paradigm to explore the application of 3D vision on the field of distributed sensing, since little work has been done in this regard and its explotation can provide results comparable to other

Figure 1.2: *Photo tourism* allows to reconstruct monuments out of images taken in the Internet. Source: Snavely et al. (2006).

much more expensive systems like LIDAR, stereo rigs or depth cameras like Microsoft Kinect (Guan (2006); Zhang (2012)).
This work addresses the problem of obtaining 3D reconstruction and motion information from images streamed from a group of low-budget mobile robots, with the aid of 3D vision techniques. There are mainly two methodologies in 3D vision to obtain the structure and camera information from a set of input images: structure from motion (SfM) and visual simultaneous localisation and mapping (VSLAM). This work implements SfM, and here an overview on VSLAM is given in Chapter 2 for completeness.

1.2 Structure From Motion

The core of this work is based on a well-established research area, Structure from Motion (SfM), which finds its foundations in the works of Longuet-Higgins (1981). SfM is an interdisciplinary technique, combining computer vision and projective geometry, that has attracted significant attention from researchers over the past 25 years (Hartley (1997); Hartley and Zisserman (2004); Horn (1990)). Furthermore, the SfM realisation requires a comprehensive knowledge of advanced linear algebra, due to the geometry involved in the overall process.
As said above, the SfM technique essentially performs the reverse process of image formation: out of a scene projected on 2D images SfM extracts 3D maps of the real scene and the relative positions of those images with respect to each other. In the last decade, thanks to a few significant breakthroughs and hardware improvements, this field has experienced a renaissance and its applications have proliferated in many aspects of image processing. To name a few, augmented reality, hand-eye calibration,

Figure 1.3: Dense reconstruction by Newcombe et al. (2011). Top: Overview of the scene. Bottom Left: four patch local 3D maps stitched into the global reference. Right: the final reconstruction.

remote sensing and image organisation / browsing are examples where SfM is used (Wei et al. (2013)).

Many applications have been found for SfM in the last ten years, and impressive results have been achieved in visual reconstruction. Using the Internet as source of images, it is easy to find databases with thousands of images of monuments such as The Eiffel Tower, the Coliseum or the Big Ben. Snavely et al. (2006) present a system called *Photo Tourism* based on SfM which takes large sets of pictures from Internet photo sharing sites and automatically estimates each photo's location, as well as a global 3D model of the scene. Fig 1.2 shows the Coliseum reconstructed with this software.

Current research is oriented towards the management of large datasets, and *Photo Tourism* is an example of this. This effort is especially interesting to mobile robotics, for the autonomy and reliability of a mobile robot is directly related with the amount of images that it can handle. Cummins and Newman (2011) works with databases of millions of images in an appearance-based automobile navigation system capable of finding loop-closures over a range of 1000 kilometres of transited roads. In the field of reconstruction, Klingner et al. (2013) makes use of the formalisms of SfM to align

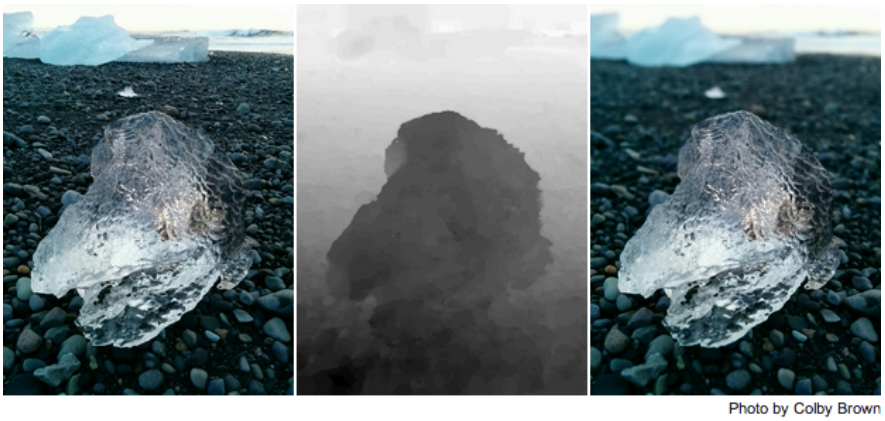


Figure 1.4: Artificial lens blur by Yu and Gallup (2014). Left: one of several input photos. Middle: Depth map. Right: Photo with lens blur.

the camera poses of billions of images taken by the contemporary street view service of Google.

SfM has also been applied over video sequences and images from hand-held cameras. Newcombe et al. (2011), by means of GPU hardware, developed a robust system (called DTAM) which creates in real time dense reconstruction of a scene covered by a hand-held camera, as shown in Fig 1.3. An impressive result with mobile cameras is the work presented by Yu and Gallup (2014). By just taking as input images the different shots taken by a smartphone, whose different locations are only given by accidental motion of the hand, this work manages to extract the depth of the scene and with it the 3D reconstruction. This result is applied to relocate at will the focus in the image and to create synthetic parallaxes. In Fig 1.4 an example of this work is shown.

In the area of robotics the achievements of 3D vision have been remarkable too. 3D vision techniques have been used in the NASA Mars exploration rover mission, where the rover Opportunity performs SfM in order to obtain visual reconstruction models of its environment and for navigation (Maimone et al. (2007)). Based on the work of DTAM, Forster et al. (2014) implements a visual odometry algorithm which runs in real time on embedded systems. This algorithm has successfully been deployed on unmanned aerial vehicles with excellent results.

There have been numerous works on collaborative systems. Werfel et al. (2014) presented a group of termite robots which can build a construction without any human help, by following simple rules, in a similar manner as termites behave. Collabora-

tion between heterogeneous groups of robots has been shown in Mathews et al. (2012), where flying drones communicate with a swarm of ground robots in order to perform difficult tasks. Flight formation and coordination with groups of nano-quadrotors has been achieved by the GRASP Lab, from University of Pennsylvania. Working collaboratively, these drones can pick up and carry heavy weights or create structures.
More specifically in SfM, this method has attracted a significant number of prior studies over the past years (e.g., Snavely et al. (2008*a*); Torr et al. (1998); Wai Yin Leung (2006)). Indeed, SfM is finding its place in many applications (Wei et al. (2013)). SfM replaces other scanning real-world object techniques, like laser scanning. SfM has largely been applied to augmented reality and for animation purposes (Jebara et al. (1999)). Many feature films use SfM when performing special effects and introducing fictitious characters to the scene (Haley-Hermiz et al. (2012)). In many computer vision algorithms, SfM is used as useful intermediate component. For example, in Brostow et al. (2008) segmentation and recognition tasks are developed based on SfM and Torr et al. (1998) uses SfM to guide the feature matching in a video sequence. Projects in computer-human interaction and information encoding have found SfM appropriate for their applications, as SfM helps track face gestures or movements and gives compressed data of a scene through 3D points.
Of all the uses that can be found for SfM, perhaps the most suitable and naturally applied is robotics. Out of the three main necessary competences for a mobile robot to navigate in a given environment, which are:

1. Self-localisation: to know where it is relatively to its surrounding environment, without the aid of neither active nor active external elements (sources of signal, beacons, landmarks, etc).
2. Map-building and its interpretation: to know where the objects in the surrounding environment are with respect to itself, which facilitates to estimate the permitted areas for transiting.
3. Path planning: to estimate the optimal path to move from one point to another, in a given environment.

SfM facilitates greatly the achievement of the first two, regardless the type of scene (either outdoors or indoors), and provides an excellent framework to perform the third

Figure 1.5: The mobile robot Rovio. Right image from Dang and Hundal (2011).

competence, path planning. The self-localisation given by SfM not only locates the robot within a frame of reference, but it also places the robot with its orientation. The map built as a result of the visual reconstruction model created by SfM is a 3D map, so the obstacle-avoidance skill is largely facilitated, as opposed to 2D maps. Moreover, SfM does not require to alter the environment in any way, as it does not need artificial landmarks such as beacons, nor it needs prior information about the scene.

Another advantage of SfM is that this technique does not use any special or expensive sensor. Just an embedded camera is necessary. The minimal requirements of hardware from SfM make it really convenient for mobile platforms, since one of the issues encountered in the design and manufacturing of mobile robots is their cost. In this work we have chosen a low-budget platform for our implementation of SfM. The reason for this is two-fold: first, low-budget robots are being considered for environmental exploration in a distributed fashion; second, this is an opportunity of bringing high technology research algorithms to off-the-shelf mobile robots.

Specifically, the chosen platform evaluated here is the low-budget omnidirectional WowWee Rovio mobile robot (shown in Fig. 1.5) which consists of a wireless network connected low-cost mobile device. Rovio has been designed to be commanded over the Internet and be used for surveillance tasks and as a camera web. The low cost sensor on board possesses its own specific problems with regard to noise characteristics for the task feature identification and matching and this challenge will form part of this work. We will examine the details of this subject in Chapter 3.

We have applied SfM for image-based 3D modelling and autonomous navigation of low-budget mobile robots. Therefore, our work is based on a state of the art technique

which is actively used by the scientific community. While this work copes with specific problems and issues provided by the poor quality of the sensors used and the particular characteristics of the mobile platform, the knowledge of SfM technique and its related fields acquired during this work enables us to address a wide range of areas of image processing in future works.

1.3 Objectives and Research Question

The goal of this work is to achieve a 3D understanding of a given scene by using the image sensors of a group of low-budget mobile robots. This is attained with the implementation of a well known but complex method in computer vision, Structure from Motion (SfM), on a small group of commercial mobile robots (WowWee Rovio). This work, where the particular characteristics of a collaborative system are used to optimise the SfM task, has not yet been addressed, as highlighted by recent surveys in the area (Radke (2008)).

The research question that set out this work is two-fold. Firstly, we consider the implementation of SfM on low-budget, omnidirectional mobile platforms, and the challenges that this type of platform entails: low quality imagery, inter-image ill-configurations. Is it possible to obtain reliable visual odometry of a low-cost, omnidirectional robot, and obtain 3D maps of the scene while it transits along? Secondly, we explore the adaptation of recent work in SfM supported by real-time robust feature point extraction techniques for distributed 3D scene understanding - that is, if multiple robots all see different isolated parts of a given environment and we know approximately where the robots may be relative to each other (with known uncertainty) then collectively can we try to reconstruct the global scene in 3D as they transit in an intersecting search pattern through the environment. This work aims towards the realisation of multi-entity SfM and will combine aspects of image mosaiking, robot motion estimation and multi-robot 3D reconstruction.

The main challenge of this work resides on the chosen platform, which is a low-budget, omnidirectional platform. An omnidirectional (also addressed as holonomic) robot platform has as many actuators as degrees of freedom. In the case of a wheeled robot, which has three degrees of freedom (two normal directions and rotation angle), a robot needs three actuators to be holonomic. Specifically in our work this configuration is achieved

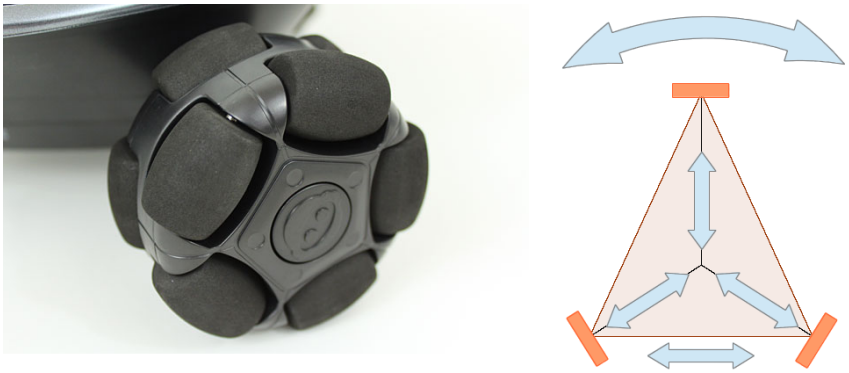


Figure 1.6: Left: omni-wheel detail; Right: possible motion directions for the omnidirectional robot platform.

by three independently commanded wheels, which are able to move almost friction-less along the perpendicular direction to their axis of displacement. This paradigm is represented by the omni wheels (Fig. 1.6). The manoeuvrability provided by this design allows an omnidirectional robot to turn on the spot and move sideways or diagonally while keeping its orientation (Fig. 1.6). Such omnidirectional platform often offers key manoeuvrability characteristics which has recently boosted research on mobile navigation in complex environments.

Under omnidirectional configuration (Fig. 1.5), abrupt changes in robot orientation provoke rapid changes in the camera field of view. This creates frequent difficult epipolar configurations when transiting a given scene. In addition, the low quality of the image sensor and the wireless streaming compression reduces the amount of features detected in the images, and increases their measurement noise.

Sequential Structure from Motion (SfM) techniques have been applied to obtaining robust 3D mapping and self-localisation on mobile robots (Maohai et al. (2006); Ortin and Montiel (2001); Royer et al. (2007)). However, in the case of an omnidirectional robot the facts here discussed (small baseline distance, high measurement noise and feature sparsity) complicate the extraction of the epipolar geometry between image pairs (Jebara et al. (1999); Shakernia et al. (2003); Szeliski and Kang (1994); Vidal and Oliensis (2002)) and render the navigation task challenging.

The first goal of this work, therefore, is to achieve a robust and stable SfM method capable of locating the camera poses and extract as dense as possible 3D information of the scene out of low resolution and noisy images. In addition, this algorithm should

be able to deal with the unique ill-conditioned configurations created by the movement typical of such a holonomic robot.
As an extension of the singular case, SfM applied on a single mobile platform, the eventual co-issue of this work arises:- how to combine the structure recovered by multiple deployed mobile robots within a common environment and merge this information so that the 3D model of the whole environment is obtained. This requires an efficient solution to the challenge of recognising a previous visited areas, called in the literature review loop-closing (Cummins and Newman (2011); Zhang et al. (2010)).
Specifically, the key contributions of this work can be listed as follows:-

- We have overcome the problem of noise created by the low quality of the image sensor and the JPEG compression imposed on the imagery by wireless streaming. JPEG-compression has only been treated in the context of SfM by Torr and Zisserman (1997). By implementing a sequence of filters, our system copes with levels of noise not found in the literature (Gang and Reinhard (2005); Ruiz et al. (2006)).
- We have developed an incremental SfM pipeline (Hartley and Zisserman (2004)) capable of retrieving the relative camera poses under omnidirectional conditions, which lead to ill-conditioned situations (Vidal and Oliensis (2002)). SfM has not been performed on omnidirectional platforms before (Bonin-Font et al. (2008); Fraundorfer and Scaramuzza (2012); Scaramuzza and Fraundorfer (2011)).
- We have developed a novel feature tracking system that manages the scarcity produced by noise and outputs quality feature tracks, which allows a stable 3D mapping of a given scene. Our feature tracking system generalises the work of Rohith et al. (2013) to any type of scenario.
- We have attained distributed reconstructions from multiple instances of the SfM system developed, by applying loop-closing techniques (Cummins and Newman (2011)). This is demonstrated on the 2-session case.
- We have developed a multiple reconstruction system that merges individual 3D structures and resolves the global scale problem with minimal overlaps. Our results extend the work of Zhang et al. (2010) which requires overlaps between stretches of sequences. This is demonstrated on the 2-session case.

The SfM system described in Chapter 3 comprises the contents of Cavestany et al. (2015), which will be published in the proceedings of the international conference of image processing of 2015 (ICIP). We plan to expose the multiple reconstruction system in a subsequent paper.

1.4 Approach Overview

Since this work has two parts clearly distinguished: SfM on a single platform and 3D reconstruction on multiple platforms, we apply two different methodologies for each section of this work:

Single SfM

The approach taken on the retrieval of 3D structure and camera motion of a single mobile platform follows a classical pipeline of a SfM system (see Fig. 3.1), with the addition of a preprocessing of the imagery prior to the matching process step. Subsequently the epipolar geometry is estimated by means of state-of-the-art minimisation methods and resection over the last camera is performed. Here we novelly develop a feature tracking system to make to most of the scarcity of features that this problem poses. The triangulation step is refined via the application of Bundle Adjustment. Finally, semi-dense 3D reconstruction is attained by using commonplace rendering methods.

Multiple Reconstruction

The methodology followed in the estimation of a global 3D map out of 3D structures retrieved by multiple instances of the mobile platform of study is based on minimal loop-closures. More specifically, the loop-closing problem is addressed by an implementation of FAB-MAP (Cummins and Newman (2011)). Once the loop-closures between different sequences are found a global 3D structure and the relative pose of all the cameras involved are calculated by Bundle Adjustment.

1.5 Thesis Outline

This thesis is divided in two main Sections: SfM on a single mobile robot and collaborative SfM. In order to familiarise the reader with the terminology used in this thesis,

Appendix A explains the core elements of the SfM method and the general algorithm that should be followed to attain 3D reconstruction and the relative camera poses of a given sequence of pictures. Chapter 2 goes through the state of the art on 3D reconstruction. All the current techniques used to perform are discussed, as well as different approaches that researches have taken to achieve visual navigation. Additionally, the present contributions towards distributed 3D perception are described. The specific SfM algorithm implemented for a single mobile and omnidirectional platform is explained in detail in Chapter 3. Here the main obstacles encountered on an omnidirectional, low-budget mobile robot are highlighted, and the various implementations proposed are described. Chapter 4 goes into the collaborative multi platform task; now the main concern is to match pictures from different sequences, so that these sequences can be linked and their 3D maps merged. The technique used to close the loops between sequences is explained here. Since we have developed a new system for the application of SfM on mobile robots, it is necessary to validate it and compare it with established methods. This evaluation is carried out in Chapter 5, where also an error analysis is shown. The conclusions drawn from this work and the research areas that remain open in this field are discussed in Chapter 6.

Chapter 2

Literature Review

This work addresses the problem of obtaining 3D reconstruction and motion information from images streamed from a mobile robot. Therefore, it involves two fields of research: robotics and multiple view geometry (MVG). MVG provides tools and methods from which robotic research can take advantage for navigation purposes. Likewise, robotics offers a suitable platform for 3D computer vision research to obtain 3D maps of a scene. There are mainly two methodologies to obtain the structure and camera information from a set of input images: structure from motion (SfM) and visual simultaneous localisation and mapping (VSLAM). Here we give an overview of both of them and present the contributions from the fields of robotics and MVG most relevant for this work.

VSLAM combines the method known as simultaneous localisation and mapping (SLAM), introduced by Smith and Cheeseman (1986) with projective geometry techniques. SLAM is a method extensively used in robotics as it is devised to solve the fundamental robotic problem of navigation and self-localisation. SLAM aims to answer for a mobile robot two questions: *What does the world look like?* and *Where am I?* This is achieved by updating the state of the robot by the perception of its own odometry and of landmarks of a scene, which are used for alternately update the location of the robot and a map of the surroundings. The updating step is performed through the application of the Kalman Filter (KF). KF is an algorithm that produces a statistically optimal estimates of the state of a given system. KF works with the current and previous estimate of a series of measurements. Firstly the current estimate is predicted with the information of the previous estimate (*priori estimate*), and then the current estimate is updated with the information of the last measurement (*posteriori estimate*). KF is

mostly implemented in its non-linear versions, such as Extended Kalman filter (EKF) or the recent Unscented Kalman filter (Wan and Merwe (2001)).

The landmarks used in SLAM can be obtained by a range of sensors, such as LIDAR, GPS, sonar sensors or depth cameras. VSLAM uses instead a camera as a main sensor. Therefore, the landmarks obtained in VSLAM are feature points identified in images taken by the camera, which are easy to track through a sequence of pictures. MonoSLAM (Davison (2003); Davison et al. (2007)) is one of the first reported investigations on monocular VSLAM to achieve real time camera pose estimations and 2D mapping. Subsequent research has tried to reduce the complexity of the problem while increasing the density of scene mapping. Mouragnon et al. (2006*a*) implement a local minimisation of the structure by using Bundle Adjustment (BA) techniques on a mobile robot, where the matching selection of the features is realised with SIMD hardware.

Strasdat et al. (2010) shows a detailed analysis of the relative merits of SLAM approach and BA - expressed in key-frame selection basis - in terms of accuracy and computational cost. This work demonstrates that when the tracking cost is included BA outperforms SLAM in general.

VSLAM is strongly related with SfM. The mapping step can be deployed by a SfM technique. This approach suffers of an important issue, and it is that the accuracy depends heavily on the baseline distance. Tomono (2005) addresses this problem by selecting an appropriate baseline based on criteria for the trade-off between the baseline distance and the number of feature points visible in the images.

VSLAM does not handle as efficiently as SfM image features in 3D space. Since in this work the management of features is crucial, SfM has been implemented in order to obtain 3D reconstruction models and as a solution of the navigation problem in robotics. This chapter addresses the literature review of the SfM process by breaking it down into the different methods that it is composed of. Firstly the contributions on the characteristic points of this work -mobile robotics and noise- are reviewed, so the most relevant works on SfM regarding these fields are revised. Secondly, the main steps of the SfM process are reviewed: feature detection and correspondence matching, feature tracking, motion recovery, triangulation and Bundle Adjustment (BA). We have showed different approaches currently present in the research community for each algorithm, emphasizing those more relevant for our work, i.e. those which deal with noise, ill-configurations or have sparse feature populations. In order to provide a good picture

of the state of the art of certain methods, in some sections the historical evolution of the approaches for a given algorithm has been explained. Finally, recent research on collaborative robots, with special attention to those works which present significant advances towards distributed reconstruction, is also reviewed.

2.1 Robotics and SfM

One of the essential tasks that a mobile robot needs to accomplish in order to interact with its environment and perform activities is autonomous navigation. Many paradigms have been proposed to solve the navigation problem, and many types of sensors have been deployed for this purpose, ranging from inertial sensors to time-of-flight cameras, including laser, infra-red, sonar, GPS, etc (Lobo et al. (1998)). Sensors for Dead-Reckoning have also been used. However, the extereoceptive sensor which has been proven most convenient for mobile navigation is the monocular camera (DeSouza and Kak (2002)). A camera provides great amount of information, it is available to all budgets and easy to integrate in a robot.

Within visual navigation many approaches have been presented to solve the navigation problem, ranging from reactive techniques based on qualitative characteristics extraction and appearance-based localisation to ground plane detection and visual sonar. For example, by handling several thousand features Castle et al. (2008) generate multiple maps for navigation, from which wearable cameras can select the correct local map and localise themselves, in a Augmented Reality (AR) context. SLAM has also been proposed to localise a mobile robot within an environment. However, the implementation of SfM solves simultaneously the self-localisation problem and the map-building requirement, and offers a natural platform to accomplish the path-planning goal. Since autonomous navigation can be defined as the ability of a mobile robot to know at once where it is in an environment, what is the layout of such environment and which path should it take to arrive at a target point, SfM appears as the most suitable available method to solve autonomous mobile navigation. It is for this reason that early on in the mobile robotics era researchers turned to SfM and SLAM in order to find a competent autonomous navigation solution.

Nevertheless, when it comes to actually reconstruct through SfM, many issues need to be addressed. Occluded features, non-smooth motion between frames, blurred pictures

and ambiguous patterns in images make difficult the SfM implementation on mobile robots. Chang and Hebert (2002) tries to overcome these setbacks by using sampling-based representation on feature tracking. Another way to get around the complications at matching is to make use of optical flow. This is a very convenient approach, since the baselines between images on a moving robot are small. Kawanishi et al. (2009) proposes a feature flow model for obtaining matches from separate positions, whereas Pietzsch (2004) uses optical flow to compute the motion.
In this work we introduce a robust SfM procedure for the autonomous navigation of a commercial low-budget robot, equipped with a monocular webcam. The main problems tackled are sparse matching populations produced by the great variance of the noise present in the images and the ill-conditioned configurations provoked by the manoeuvrability of the omnidirectional platform chosen. We intend to lessen the hardware requirements that navigation methods currently impose on mobile robots by introducing robust and light noise filters and robust tracking systems, as well as noise-proof epipolar geometry estimation methods.

2.1.1 Low-Budget Robots

The purpose of this work is to enable robust autonomous navigation on low-budget robots by devising new methods which overcome their image quality, rough odometry and motion limitations and allow the implementation of SfM on them. Mobile robots are already present in many areas (from everyday fields like health care, remote presence to specific fields like search and rescue and ocean data collection). Yet recently a plethora of low-budget household mobile robots have appeared in society. The range of applications where home robots are finding a niche is becoming countless. Pool-cleaning robots, vacuum cleaners, scrubbing robots, gutter wipers, the list of household robots keeps increasing as days go by.
Mobile robots with a more social purpose have also been introduced in the household sphere (Denning et al. (2009)).
The platform chosen for this work is the mobile robot platform Rovio, shown in Fig. 1.5. Rovio is a significantly sophisticated mobile platform manufactured by WowWee that has been designed to be controlled over the Internet and can be used as a mobile webcam. Rovio specifications are described in detail in Chapter 3. Rovio's capabilities have been considered for performing planning motion, by processing detected landmarks on

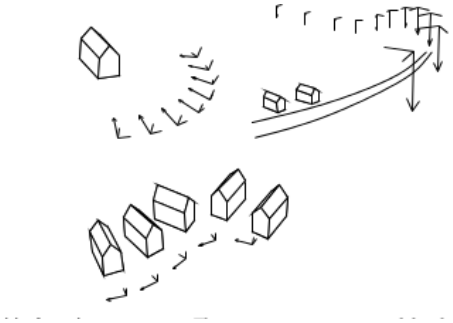


Figure 2.1: Sequences made up of orbital motion, rotation about parallel axes and arbitrary translation and planar motion make unfeasible to obtain Euclidean reconstructions from uncalibrated images. Source: Sturm (1997).

a Support Vector Machine (Dang and Hundal (2011)). Fladung and Mwaura (2011) attach a laser pointer to a Rovio robot to create a 2D map with supervised learning techniques. The retractile arm of the Rovio has been used in Buente et al. (2011) as a non-simultaneous stereo rig in order to achieve a primitive self-localisation and 2D mapping performance.

In a more advanced scope, a visual based navigation algorithm is presented in Begum et al. (2010). Begum *et al.* make use of checkpoints to elaborate a path from a start configuration to a goal configuration while enabling Rovio to avoid obstacles. Rovio is also considered by Karnad *et al.* as a mobile tele-immersion platform to be used in shared tele-presence meetings (Karnad and Isler (2010)).

2.1.2 Omnidirectional Robots

One of the advantages of our platform - which constitutes, at the same time, one of the challenges of this work - is its mobility. It has been mentioned in Chapter 1 that Rovio is a holonomic robot platform. This paradigm is represented by the omni wheels (Fig. 1.6). Such omnidirectional platform often offers key manoeuvrability characteristics with a wide range of application domains.

The manoeuvrability offered by this design allows an omni-directional robot to turn on the spot and move sideways or diagonally while keeping the same orientation. This manoeuvrability allows the platform to change abruptly directions and to take sequences of pictures of a scene in a sideways motion. These are desired features of a mobile platform when reconstructing a scene in a distributed manner. On the other hand,

under omnidirectional configuration abrupt changes in robot orientation provoke rapid changes in the camera field of view. This creates degenerate epipolar configurations when transiting a given scene. In fact, even if the baseline distances are carefully selected, away from ill-conditioned configurations, the sequences of pictures that an omnidirectional platform can take fall in the set *critical motion sequences*, as Sturm demonstrated Sturm (1997), and therefore lead to inherent ambiguities in uncalibrated Euclidean reconstruction. These types of sequences are shown in Fig. 2.1.

The advent of omni wheels have drawn the attention of researchers towards omnidirectional mobile platforms. Since this is a recent paradigm in robotics, a number of design prototypes have been proposed. Several wheel configurations have been tested (Ren and Ma (2013); Rojas and Förster (2006); Safar et al. (2013)) and different types of holonomic-enabling wheels tried out (Chamberland et al. (2010); Udengaard and Iagnemma (2008); Ueno et al. (2009); Ye et al. (2011)). The work presented in Liu et al. (2010) shows in detail the engineering design process for an omnidirectional mobile robot of 4 omni wheels. The design includes a mechanical base, a digital signal processing microcontroller-based control system and the analysis of omnidirectional motions. Interestingly, the work of Liu et al. (2010) shows that preferential directions exist, according to wheel configuration, along which the platform finds least friction. One of the main challenges when operating with holonomic robots is the adequate modelling of their dynamics so as to establish a stable control system. A motion analysis of three and 4 wheel omnidirectional mobile platforms is performed in Oliveira et al. (2009), and a model determined. Udengaard and Iagnemma (2008) studies the dynamics of an omnidirectional robot on rough terrains.

The motion model problem is not trivial as a good model should account for the slippage of the omni wheels. Almeida et al. (2013) characterises the wheel friction and slipping in a differential wheel configuration, and proposes a hierarchical traction control architecture. Williams et al. (2002) studies the omnidirectional mobile robot dynamic slip motion and test the dynamic model derived. Nevertheless, due to frequent slippage the odometry provided by holonomic robots is specially rough. This is shown in Zou et al. (2011), where the accumulation of error along different paths taken by an omnidirectional robot is analysed.

Later in Chapter 3 the odometry provided by our platform is considered to establish a minimum bounding box within which the estimation of a camera pose should fall. The

kinematic model applied over the odometry information follows the principles stated in Ashmore and Barnes (2002). This work uses in the estimation of the kinematics of a three wheel omni-drive mobile platform the concept of induced velocity. The induced velocity of an omni wheel comprises the slippage present in the motion. Therefore Ashmore and Barnes (2002) derives a concise and linear model of the movement which also explains friction sources.

Despite the interest with robotics motion research (DeSouza and Kak (2002); Strasdat et al. (2010)), several challenges remain for computer vision based SfM from an omnidirectional platform (Fig. 1.5). Here we explore the use of such platform for SfM. Since a crucial aspect of the mobile robot considered is that it is a cheap, affordable by household economies, the quality of its sensors are often limited. The first point reviewed therefore is the research on noise, since this is a problem present in this work throughout all the steps of the SfM process.

2.2 Image Noise

One of the main problems that this work has encountered is the levels of noise associated with low cost sensors that are typical of omnidirectional platforms. Since our source of information is imagery, any image noise becomes ubiquitous in all the phases of the process. Therefore filters against noise have been devised along all the steps of the work. In the literature noise is taken as a factor against which methods should be tested to demonstrate their robustness (Ruiz et al. (2006)), but little research has been found that addresses the issue of noise all over their work-flow. Furthermore, the uncertainty considered is usually a Gaussian with $\sigma \leqslant 1$ pixels. In a theoretical and abstract scope, Hartley and Zisserman (2004) copes with noisy measurements explicitly and robust, noise-proof algorithms are described all along their work. Nevertheless, the algorithms shown in Hartley and Zisserman (2004) can not overcome the noisy matching populations that our system produces, and methods need to be devised to reduce the noise to tolerable levels. Thomas and Oliensis (1999) treats the consequences of noisy images on an incremental SfM implementation by estimating the motion error. The motion error is seen as a cause of correlation in the 3D points of the structure. Therefore, by estimating the correlation between 3D world points, Thomas *et al.* retrieves the error on the estimation of camera poses. The motion error is subsequently added in

a Kalman Filter, along with the image error. Onessis *et al.* measure the error by means of covariances, which does not represent uncertainty for SfM universally. In particular, this representation is not valid in situations when the correspondence noise is relatively large with respect to camera baselines (Chang and Hebert (2002)). This is the case of most of inter-image configurations in our sequences. Another approach to noise is to take advantage of special features usually present in the scene such as lines and planes. Gang and Reinhard (2005) stabilise the estimation of the exterior orientation by applying prior knowledge on collinear and coplanar points of the scene to reduce noise. However, this approach does not deal with noise as such, but it applies instead some understanding from the scene to the matching population in order to attain robust estimations. Chang and Hebert (2002) convert the tracking problem and the SfM estimation into a particle filter approach. They create sampled representations of measurements uncertainty (both in tracking measurements and SfM measurements) on a series generated over time, and use this paradigm to propagate SfM uncertainty over a discrete period of time. While this jointly representation of measurements and noise facilitates the tracking of features through occlusions and it is robust against non-smooth motion between frames and presence of noise, it is hardly extendible to a distributed system, where various mobile platforms may share the view of a 3D point.
Noise is also an obstacle that pre-processing algorithms encounter. In order to homogenise shadows and bright parts in a picture of a given scene, Upcroft et al. (2014) uses the proposed illuminant invariant by Ratnasingam and Collins (2010) and RGB images to perform image classification of urban scenes despite challenging variations in lighting conditions. However, this method is quite sensitive if a single RGB channel is over-exposed or under-exposed. This can result in noisy images after the transform especially if compression artifacts are present, like in the case of JPEG compression (Solomon and Breckon (2011)).
The system developed in this work deals with noise in scalable and manageable way. Strict and light filters are devised to overcome the problems of both spuriousness and sparseness of feature matches, and a robust tracking system has been developed for the specific inter-image configurations encountered. Here, in this unexplored case, noisy matches are efficiently trimmed, and the scarcity of surviving feature matches is managed by a novel feature filtering tracking method.

Noise affects principally to the extraction of features, the base of the whole SfM process. In next section the state of the art feature detectors and extractors are reviewed.

2.3 Image Features

The first step in any method which intends to relate different images of the same scene is to find points corresponding to the same world point. These points should have special characteristics as they should be recognisable across different perspective, scaling and lighting changes. It is for this reason why they are called key-points or image features. Image features must be easy to track in a sequence of images, and with this purpose multi-dimensional invariant norms have been devised to describe them uniquely under as many transformations as possible.
In SfM image features are the main source of information that the method takes as input. Corresponding features are used to recover the motion between images, and these very same features are later on back-projected in 3D space with the information of the recovered motion to estimate the 3D point that originated them. It is therefore crucial that features detected in images are of the best quality, since both motion and structure recovery rely on them (and structure doubly). Provided that image information is not readily accessible due to noise presence, it is paramount to apply the most suitable feature detectors and feature descriptors under the conditions of our problem. Here we give an overview of the current algorithms for feature detection and extraction in robot navigation.

2.3.1 Feature Detectors and Descriptors

The robust detection of characteristic features is an important issue in computer vision. Many algorithms rely on the extraction of robust key-points. Therefore, many efforts have been made to develop efficient and powerful methods that identify points unambiguously. These key-points must be easy to identify in images of the same object taken from different points of view, different lighting conditions and even different cameras.
The Fig. 2.2 shows two images of the Eiffel Tower taken from different locations and aspects. It is noticeable that, apart from the fact that lighting conditions have changed, the Eiffel Tower appears to be rotated, scaled and even distorted, this given by perspective laws. We say that it has undergone a projective transformation. If we want

Figure 2.2: Two pictures of Eiffel Tower (left pair), taken under two different perspectives. A good feature detector must find those characteristic points that are distinctly recognisable in both images (right pair). A good feature extractor should define norms with invariant metrics in projective transformations.

to find matches in these two images, we must not only provide suitable key-points, but also give a description of them in such a way that all the aforementioned changes will not modify their characterisation. Therefore, along with every key-point a descriptor is defined. A good feature detector should find special points from each image of Fig. 2.2, with unique descriptors of them, so that it will be possible to robustly identify the same points of the tower in both images.

Schmidt et al. (2013) evaluate the most currently relevant feature detectors and descriptors in the literature in the context of robot navigation. Here we extend this survey, by widening the range of feature detectors selected and incorporating more recent feature detectors. The strategies followed to detect and retain salient points are explained, as well as the constraints they are subject to. This review will help understand what is the most appropriate method to apply in the context of this work.

Harris and Stephens (1988) proposed a method to extract key-points using Hessian matrices and their eigenvalues, which provided some consistency under rotation. Later on, Shi and Tomasi (1994) found that even better results were obtained by simplifying the criteria of corner selection.

Based on these works Lowe Lowe (2004) implemented Scale Invariant Feature Transform (SIFT), a method to find key-points in an image that was invariant to the scale, and intended to be invariant to affine transformation as well. Also, SIFT is invariant to moderate changes in lighting conditions, which is very convenient in many real world computer vision problems. The output of this method is, for each feature, not only

the location and orientation of it, but also a scale in relation to where it was found and additionally a multidimensional descriptor (regularly 128) that tries to describe uniquely the feature point through derivatives. SIFT is a really effective method to localise many prominent features and it provides accurate descriptors to match them with features from other images. The main flaw of SIFT is the computation time. In order to be scale invariant, a number of Gaussian convolutions over different scales of the image are done, along with other computations, which make it impossible to apply SIFT to a streaming video at greater than 10 fps, on commodity hardware.

By contrast, Matas et al. (2004) presented Maximally Stable Extremal Regions (MSER). This work uses extremal regions for wide-baseline stereo matching problem, so instead of finding key-points it detects blobs in images. An extremal region is a set closed under projective transformations and monotonic transformations. MSER presents a near linear complexity in the construction of detectors and a computation speed near frame rate, and it was designed to be affine invariant. MSER also provides descriptors, which are also scale invariant. MSER algorithm extracts from an image I a number of co-variant regions, *Maximally Stable Extremal Regions*. These regions are characterised as stably connected components of some level sets of the image I. In less formal words, MSER can be seen as a method based on *thresholding*. If we classify an image according to the intensity values of its pixels, we could create a sequence of n images $S = \{I_t\}_{t=0}^{t=n}$ where n is the range of intensity values that a pixel can take. I_t is an image in which all the pixels whose intensity values are above a threshold t are white and black otherwise. If we went through the sequence S we would start with a white image, and see appear black regions in I_t as t increases, with these regions growing and merging until the whole image would be black. In this analogy, an extremal region is a connected component of a thresholded image. The set of all connected components of all frames of the sequence S is the set of all *Maximally Stable Extremal Regions*. Even though MSER is really stable and extensively used, it does not work well with images where motion blur is present.

Bay et al. (2006) developed a method for detection and descriptors extraction called Speeded-Up Robust Features (SURF), which aimed at overcoming the main setback of SIFT. SURF is significantly both faster and simpler, and suitable for real-time processes - computational workload. Although its performance does not reach the standards of SIFT, it is usually sufficient for matching tasks (Mikolajczyk and Schmid (2005); Miksik

and Mikolajczyk (2012)). The quickest feature detector in the literature is the method *Features from Accelerated Segment Test* (FAST) by Rosten and Drummond (2006). Based on a simple brightness test on a circle of pixels around a candidate corner, it offers very good results with small image transformations.

The feature detector developed by Agrawal et al. (2008) coined as CenSure, was specifically designed for visual odometry on mobile platforms. CenSure is a Scale-invariant centre-surround detector. It uses a bi-level approximation of the Laplacian of Gaussians (LoG) filter, where the circular mask of LoG is replaced by an approximation that allows to preserve rotational invariance. STAR, a commonplace implementation of Censure, uses a mask that consists of two squares rotated 45 degrees between each other.

Calonder et al. (2010) introduced a feature descriptor, *Binary Robust Independent Elementary Features* (BRIEF). BRIEF was designed for resource-constraint applications, and it provides excellent results coupled with FAST, when processing speed is a concern. BRIEF introduces binary strings as feature descriptors. It applies intensity comparisons on sets of location pairs (x, y) of smoothed image patches. These intensity comparisons are then arranged in binary strings, so that Hamming distance[1] can be used on them to match the descriptors created.

In 2011 Rublee et al. (2011) proposed *Oriented Fast and Rotated BRIEF* (ORB) as an alternative to SIFT and SURF. It is a fusion of FAST key-point and BRIEF descriptor, where some steps have been enhanced. ORB applies Harris corner measure on FAST keypoints, along with a pyramidal approach to produce multiscale-features. The orientation, needed for a feature to be rotation invariant, is given by the vector from the feature to the intensity weighted centroid of a considered patch. The descriptor is formed out of BRIEF descriptors, by "steering" BRIEF descriptors according to a rotation defined by the orientation of the key-point. ORB is a good choice in low-power devices such as smartphones. In this same year Leutenegger et al. (2011) presented *Binary Robust Invariant Scalable Keypoints* (BRISK). BRISK is a feature detector and descriptor extractor based on FAST. BRISK enables scale invariance by analysing the surrounding pixels of the upper and lower octave layers. The descriptor is a 512 bit binary, and it is computed with the weighted Gaussian average over a selected pattern of points near the key-point, equally spaced on concentric circles with the key-point

[1]The Hamming distance between two arrays of the same length is the number of elements at which the corresponging values are different.

feature. In a similar manner as BRISK descriptor, *Fast REtinA Keypoint* (FREAK), developed in 2012 by Alahi et al. (2012), applies a sampling pattern upon the key-point, but the pattern formed by these Gaussian averages is biologically inspired by the retinal pattern in the eye. The pixels being averaged overlap, and are much more concentrated near the key-point.

Amongst these features detectors and descriptors, BRIEF and ORB clearly outperform the rest when there is little in-plane rotation. FREAK on the contrary presents lower accuracy than the others. The SURF descriptor, on the other hand, it is the most influenced by the type of feature detector used. STAR provides good results but it requires a relatively feature-rich environment.

The last feature detector and descriptor with a significant relevance is KAZE, created by Alcantarilla et al. (2012). KAZE is a multi-scale detector and descriptor which works in a non-linear scale space. KAZE uses non-linear diffusion filtering, gradient-dependent equations introduced by Alcantarilla et al. (2012) to reduce the diffusion at the location of edges. This filtering encourages smoothing within a region instead of smoothing across boundaries. Since there are no analytical solutions for the partial differential equations involved in non-linear diffusion filtering, the Additive Operator Splitting numerical methods are used as approximation. Although more expensive to compute than SURF due to the construction of the non-linear space, KAZE outperforms the state of the art techniques both in detection and description. This computational burden was fixed by *Accelerated KAZE* (AKAZE) (Alcantarilla et al. (2013)) by using new numerical methods called Fast Explicit diffusion in the feature detection process. In addition, AKAZE comes with a new descriptor, *Modified-Local Difference Binary*, based on the same principle as BRIEF, but making the grid sub-sampling function of the scale of the feature.

Alternatively to point features lines and edgelets can be used for matching, or along with image features. Lines provide additional information such as planar and orthogonal constraints. The key disadvantage is that they are more difficult to match and track (to start with, they can be more easily occluded). In addition, the extremes of a line segment might not be present in the image.

The abundance of types and variations of feature detectors and descriptors is justified provided the amount of algorithms in image processing that base their performance on image features. Within the scope of this work, image features will be used as matching

candidates in the step described in next section, the correspondence problem. This step crucially feeds all the subroutines the SfM method. It is therefore a principal concern that image features detected are numerous as well as unique.

2.4 The Correspondence Problem

The correspondence problem consists of identifying, in two different images of the same scene, a set of homologous points. The elements of this set, formed of pairs of points - each point of a pair from one of the two images - are called *matches*. The correspondence problem is extensively addressed in 3D vision and even other disciplines. Stitching panoramas, motion estimation, face tracking, video stabilisation and object recognition, to name a few, are examples of applications that make use of correspondence matching (Szeliski (2011)). Correspondence matching is also the basis of the particle image velocimetry measurement technique, which is nowadays widely used to quantitatively measure fluid motion (Santiago et al. (1998)). Generally speaking, matching correspondences becomes necessary when an automatic process intends to establish some relationship over images of a set. The only prior information that feature matching requires to be implemented is that the images considered should show different perspectives of the same environment.

Appendix A describes the SfM method as a geometric derivation of the epipolar relationship between two images. This relationship emerges when both images see the same feature of the 3D world. Therefore, finding correspondences in two frames constitutes the base of the SfM algorithm. It is worth noting that the matching population built up in this step represents in this work the whole source of information with which the SfM system will have to deal in order to attain the camera pose estimation and structure information from a sequence of images. For these reasons the adaptation of the correspondence problem to the SfM process of this work is paramount.

2.4.1 Feature Matching

The correspondence problem is addressed differently according to the overlap between images, relative pose and size of the sequence. If the image sensor is a stereo-camera, and the pair has been rectified, the search for matching is done along the rows of the images. In the case of multiple image batch mosaiking, a common strategy is to attempt to detect

overlapping images using exhaustive matching of image pairs. In robot navigation the images usually are presented sequentially, so the approach should be different. If the motion between consecutive frames is small with respect to the overlap, *optical flow* is usually applied Szeliski (2005). *Optical flow* is the motion pattern generated on an observer by objects in the scene due to the motion of the observer across the scene. Techniques that extract this flow between images perform pixel-based matching (also called *direct methods*), i.e. look for similarities between pixels of images. A suitable error metric to compare images with each other must be chosen in pixel-to-pixel matching, like normalised cross-correlation (NCC) or sum of squared differences (SSD). Once this has been established, a suitable search technique must be devised. The naïve approach is to do a full search (block matching), but more optimal strategies have been introduced to speed up, refine and account for more pronounced changes between images rather than only translational motion.

Optical flow can be divided into two types: dense and sparse. The former performs a per-pixel tracking throughout the whole image, and is computationally consuming, albeit it achieves accurate results. A good example of dense optical flow is Farnebäck (2003). In contrast, sparse optical flow focuses on special key-points that are easy to track and likely to be found on the second image. Sparse optical flow techniques are usually faster than their dense counterparts. One notable sparse technique is that of Kanade-Lucas (KL) (Lucas and Kanade (1981)). KL relies only on local information that is derived from some small window surrounding each of the points of interest. This approach makes KL both computationally efficient and robust. KL is usually applied with the feature detector developed by Shi and Tomasi (1994), and then becomes Kanade-Lucas-Tomasi (KLT) tracker (see Section 2.5). *Optical flow* is a good choice on image sequences where we expect to find a match for a given feature point in its near surroundings and motion between images is small. However it has its own shortcomings that need to be tackled, like the aperture problem (Fraundorfer and Scaramuzza (2012); Szeliski (2005)). Optical flow is not suitable for visual odometry applications due to the quick accumulation of motion error.

As opposed to pixel-based matching methods there are feature-based methods. Two major approaches exist in this group. In image sequences where the expected locations of feature points can be reasonably well predicted it is convenient the implementation of *detect then track* matching methods. This type of methods compares patches using a

translational model between neighbouring frames, and suits best certain motion video sequences. Features are initialised in a single image and then looked for their matchings in next image. However, features tracked along large image sequences suffer great changes. In this case, and also where baseline distances are comparatively big, the most convenient approach is *detect then match*. This approach consists of finding features in images separately and then match them based on local appearance similarity (given by the descriptors). It is in this case where the feature detectors described in Section 2.3 gain relevance, since the transformations undergone by images are greater, and affine invariant feature detectors are necessary (Fraundorfer and Scaramuzza (2012)).

Feature matching tries to find matches among the image features extracted in two images, using some error metrics on their associated descriptors. Typically the error metric is the Euclidean distances L1 or L2, but NCC and SSD are equally suitable here. L1 and L2 are the common choice when the descriptors in use are SIFT or SURF. For binary string based descriptors like ORB, BRIEF or BRISK the Hamming norm should be used, given the nature of these descriptors. The simplest way to accomplish this task is to search, for every feature of the first image, throughout the whole set of features of the second image. To avoid having several features in the second image matched with one feature from the first one, a cross-matching can be done (Zhao and Gao (2006)). The set of matches will then be made up by corresponding features that mutually have each other as best match. However, when there is a shortage of features (due to image noise, not enough overlapping, etc) this technique rules out too many good features. In Section 3.4 we explain other methods to ensure good inter-image matching.

Another mechanism to enforce correct matches, alongside cross-matching, is the distance-ratio test proposed by Lowe (2004). Lowe computes the probability of a correct match as the ratio of the first nearest match of one feature to the second nearest match. All the matches with ratio bigger than a threshold are rejected. Lowe's distance ratio test seeks for unique matches.

The main and obvious disadvantage of brute force matching is that its computational cost grows quadratically with the number of features. To keep matching times low, supervised matching search can be done if there are available other sources of information, like motion models. A structure and motion model will predict, under a constant velocity model, a region where a match for a given feature should be found. Motion

models are usually generated from other sensors, like wheel odometry, inertial measurement unit (IMU), laser or GPS (Maimone et al. (2007)). If structure information is not available, images can still be registered, in a similar manner as in a calibrated stereo rig, and then the search for matches can be done along the epipolar lines.

Many robot mobile platforms have the image sensor as the only extereoceptive sensor. The computational burden imposed by linear search can still be circumvented by using indexing schemes for matching in high dimensional spaces, such as series of 1D binary searches, hash tables or k-dimensional (kd) trees. A widely used library which collects multi-dimensional search algorithms is the Fast Library for Approximate Nearest Neighbour (FLANN), by Muja and Lowe (2009). FLANN can cope with a multidimensional set of points efficiently, by means of creating multidimensional query tree structures according to the characteristics of the query set. There are a number of proposals for tree structures. Arya et al. (1994) developed kd spaces, which build tree structures by splitting the nodes at the median values. With randomised kd trees the approximation of nearest neighbour to the query point is improved by searching simultaneously across a number of randomised trees. Hierarchical k-means tree, included in Muja and Lowe (2009), has every inner node split in k-ways. K-means clustering is then used to classify the data subset at each node. In addition, FLANN can create tree structures as a combination of these three types, or even automatically tune the tree structure indexing to offer the best performance for the dataset provided. Finally, it is also possible to create an index using multi-probe Locality Sensitive Hashing (LSH) Lv et al. (2007).

For the sake of completeness it should be mentioned that there also are in the literature correspondence-free methods (Makadia et al. (2007)). These methods use the whole set of pixels in an image to estimate the relative motion by means of a Harmonic Fourier transform. This approach works very well with low-texture images but it is too expensive in computational terms. Moreover, the extracted motion is not as accurate as with feature-based methods.

The correspondence problem is an open problem in computer vision. Not only there are not exact solutions computationally affordable for feature matching, but also there are a number of unresolved issues that appear when matching images. Examples are: ambiguities created by different instances of a same category, occlusion management and poor-textured surfaces.

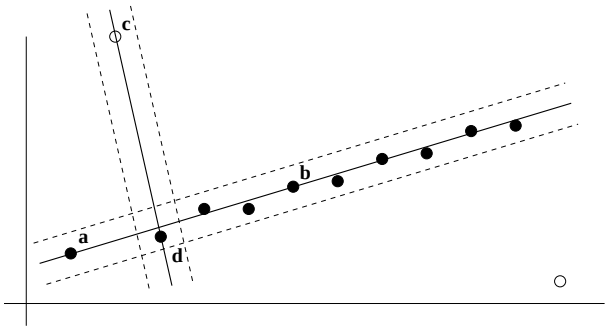


Figure 2.3: In this example of linear regression, where the dotted lines show the margins within which a point is considered inlier with respect to a given model, the minimal model given by the points $c-d$ has many fewer inliers than the minimal model $a-b$, which means that the minimal model $a-b$ will probably be a good model estimation for the whole set of points, and that the point c will most likely be an outlier. Source: Hartley and Zisserman (2004).

2.4.2 Robust Estimation of Epipolar Geometry

The estimation of the epipolar geometry, as it has been exposed in Section A.4, is a difficult task. The estimation of the fundamental matrix is severely affected as soon as outliers are present in the sample. It is therefore of supreme importance to arrive at the epipolar estimation stage with an outlier-less set of matches. Unfortunately, and despite all the measures taken to avoid mismatches, after the matching step a significant part of corresponding features are still outliers (in this work, an average of 25%). Image noise, scene occlusions, image blur, light changes and projective transformations not accounted for by image descriptors corrupt the matching process, and permit mismatches to make their way through to the epipolar estimation phase.

Robust algorithms have been devised to cope with sets populated by outliers. The purpose of these algorithms is two-fold: a) effectively remove present outliers, and b) estimate the sought model for the set considered. If the proportion of outliers is small, there is a family of methods which can detect them and rule them out. As an example, *case deletion* deletes outliers detected and then fits a model to the remaining data. However, in presence of a significant part of outliers *random sample consensus* (RANSAC) (Fischler and Bolles (1981)) has been established as the standard.

RANSAC is a *hypothesis-and-verify* algorithm. It applies iteratively a voting system on the correlation model estimated with a minimal subset randomly taken from the global subset. In each iteration, the minimal model is checked against the rest of the set and

a number of inliers for that model is established, according to some distance threshold. The intuition behind the algorithm is that models estimated from inliers will agree with more elements of the set than models estimated with outliers (see Fig. 2.3). In a classic SfM problem the correlation model to find is a 9×9 matrix (either F or E) and the error metrics used to measure the fitness of a model to the global set is the distance between a feature and the epipolar line generated by its match. This distance is measured by means of the epipolar equation (see Eq. A.4.21), which is usually evaluated with the Sampson distance (first-order approximation) (Sampson (1982)). Other error metrics have been proposed, such as the directional error (Oliensis (2002)). The directional error is the angle created by a ray from the epipole passing through a given feature and the epipolar line.

Since its appearance in 1981, RANSAC has been improved with variants that intend to be more robust against outliers, minimise the number of iterations and acquire more accurate models. PROSAC (Progressive Sample Consensus) (Chum and Matas (2005)) assumes that better match scores are correlated with correct matches, so it orders the tentative correspondences according to this criterion. As a result random samples are initially added from the most "confident" matches, thereby speeding up the process of finding a statistically likely good set of inliers.

Torr and Zisserman (2000) proposed to maximise the log likelihood of the model rather than the number of inliers. This work defines the probability density function of the noise perturbed data for a given true correspondence, and uses it as a starting point to build up a new robust estimation method, MLESAC. MLESAC is based on MSAC (Torr and Zisserman (1997)) which gives a score to inliers as to how well they fit the data, by means of using a simple M-estimator. Preemptive RANSAC (Nistér (2005)) extends this approach. It also selects the hypothesis generated according to their log likelihood but uses pre-emptive scoring of hypotheses to narrow the search of the best fit. In addition, and for real time requirements, the iterations are fixed.

In 2010 McIlroy et al. (2010) presented DESAC, a deterministic scheme for selecting samples. DESAC works in a similar manner to PROSAC, but it also combines ambiguity and performance of previous hypotheses to estimate the probability that a match is correct, and then uses this probability to favour better matches in the estimate of models with minimal sub-sets. In short, DESAC focuses on matches with more probability of being inliers.

Raguram et al. (2009) presented uncertainty RANSAC, which incorporates feature uncertainty. This method performs a pre-selection of putative matches by testing hypothesis with a likelihood ratio and checking the covariance matrix of the hypothesis.

sample size	Proportion of outliers ϵ						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Table 2.1: Number of samples necessary to ensure, with a probability $p = 0.99$, that at least one sample has no outliers for a given size of sammple, s, and proportion of outliers, ϵ. Source: Hartley and Zisserman (2004).

A critical point in RANSAC-like algorithms is the number of parameters necessary to estimate the transformation model between the corresponding features. It has been discussed in Section A.4.3 that an over-parametrisation not only is harmless in the estimation of the geometry between images, but also it usually provides extra advantages, like better fitness in presence of noise. Therefore the measured support would more accurately reflect the true support. However, in robust estimation in presence of outliers the benefits of over-parametrisation are overweighed by using a minimal point set as a model, since the computational cost grows exponentially with the number of parameters of the model, as Table 2.1 shows.

There are alternatives to RANSAC to robustly estimate models in presence of noise. Instead of scoring a model instantiated from a minimal set by the number of points which agree with the model within a threshold, one could score the model by the median of distances to all the instances of the set. The model which scores the least median is selected. This method is the *Least Median of Squares* (LMS), Rousseeuw (1984). Conveniently, LMS does not need prior information about the proportion of outliers in the sample, and no settings of thresholds is required. On the other hand, LMS can not cope with more than 50% of outliers, because the least median would be to an outlier.

In the case of study here the characteristics of the image sequences studied make especially difficult to attain a robust estimation of the geometry between images. The unusual presence of noise produces a high score of outliers in the putative correspondences, and even correct matches can be seen as outliers by good models due to noise. In addition, matches populations are limited by the VGA resolution of images, which forces baselines to be narrow. It is shown in Section 3.1.2 that narrow baselines create ill-configurations, where the estimation of the epipolar geometry is even more unstable than in normal circumstances. We are facing therefore an already unstable problem accentuated by the imposed geometry between frames where the scarce input is highly corrupted by noise. In Chapter 3 the filters implemented at this stage are described in detail. These filters guarantee that only the best inliers are selected as input to estimate the relative pose between frames.
Since the bottle-neck of the problem is the number of features to be tracked, one of the key contributions of this work is the development of a novel feature-tracking system which makes the most of the few survival matches of the severe filters implemented for outlier removal. In next section tracking methods proposed in the literature for sparse matching populations are reviewed.

2.5 Feature Tracking

Feature tracking is the extension problem to finding corresponding features between images. As the name suggests, it consists of tracking a set of features found in an image along a sequence. Feature tracking makes especial sense with sequential algorithms, where it is reasonable to expect that a feature observed in an image will still be present in the next image of the sequence. Unordered set of images require different techniques to figure out in which images a given feature is seen, like loop-closing (see Section 4.2.1). Feature tracking is paramount in SfM schemes, for ulterior methods in the SfM pipeline (such as the *triangulation* procedure and the *Bundle Adjustment* (BA) method, see Sections 2.7 and 2.8) rely their stability on the number of views projected by each 3D point on the image sequence.
The strategy for feature tracking depends on how the correspondence problem has been approached, but in any case feature tracking normally emerges as a result of a concatenation of matching consecutive pairs of images. If a feature x_i of an image I_t is coupled

when matching the pair $\{I_{t-1},\ I_t\}$, and subsequently it is found a correspondence for x_i when matching the pair $\{I_t,\ I_{t+1}\}$, a track for the feature x_i is then created, and it will be maintained as long as consecutive matches are found for x_i. The first feature trackers were of the type KLT, implemented by Shi and Tomasi (1994), in the context of optical flow, but correspondence errors tend to be large with this technique. Rohith et al. (2013) improved the tracking accuracy by discriminating the features in structured images. This work first establishes the geometry between a pair of images with the aid of fast and easily detectable KLT-based features, and then searches for slower and more difficult to track features. The latter features are described with coefficients extracted from the epipolar geometry. Therefore Rohith et al. (2013) grounds its feature tracking system on heterogeneous features and camera geometry based descriptors.

In the last years different formulations to KLT tracker have been introduced. Jin et al. (2003) intend to process the two steps of SfM (feature correspondence and 3-dimensional reconstruction) in one closed loop by matching regions, using photometric deformation models. These regions are defined by the area and normal of planar patches. Jin et al. (2003) estimate the structure and motion via an EKF of the state of a dynamical system. This dynamical system is defined by the normal vectors of matched planar regions, the homographies existent between them and certain factors that account for changes in brightness and contrast in the scene. Jin et al. (2003) claim to integrate visual information in space as well as in time, by using a finitely parametrisable class of geometric and photometric models for the scene. Other approaches exist, like fuzzy logic implementation for noise removal, or to consider feature tracking as a concave programming problem (Maciel and Costeira (2003)). Concave programming is a special case of the general constrained optimisation problem

$$\begin{aligned} &\max_{x \in X} f(x) \\ &\text{subject to } g(x) \leq 0 \end{aligned} \tag{2.1}$$

in which the objective function f is concave and the constraint functions g_j are convex. Global Localisation (GL) scheme can be seen as an alternative to the feature registration problem. GL involves an offline process for space abstraction using features and an online step for feature matching. Dong et al. (2009) represents a good example of this technique. In a parallel-computing scheme, this work constructs an optimal set of

selected keyframes which intend to cover the entire space and at the same time minimise the content redundancy.

If the scope of feature trackers is confined to consecutive images results are inefficient, as occlusions or disruptions in the feature tracks remain neglected. Indistinctive structures, present noise and large image distortion provoke that tracks are split down into several shorter tracks, which affects the stability of the process and creates redundant 3D points in the structure. With appropriate filters for structure redundancy this is not an issue when the number of tracked features is large enough. However, if the matching population is scarce this deficiency develops into a serious obstacle for the reconstruction. This is one of the reasons that have led this work to generate as many quality feature tracks as possible, as it will be shown in Sections 3.4 and 3.6.

Matched features are the source of data that SfM methods take as input in order to extract the relative motion between frames. By relating objects in the world through their projections on different images it is possible to recover the motion between them.

2.6 Motion Recovery

Motion recovery is the first step in the actual SfM method. It consists of the extraction of the relative motion existent between two images of the same scene. As mentioned in Section A.5, it is important to be aware that motion can only be recovered up to scale. Motion between images is usually recovered through homologue features (hence previous sections), although other alternatives exist for feature matching. Firstly state-of-the-art algorithms to achieve real-time motion tracking (and structure recovery) are described as a reference.

In Section A.4 it is explained how the essential matrix E can be extracted if enough matching features in two images are detected. E represents the algebraic form of the epipolar geometry created by the two images considered, and it encodes the relative motion between them. In this section current techniques for epipolar geometry and motion extraction from images are reviewed. We then focus our attention on the specific circumstances of our case of study - i.e. motion recovery on mobile robots, and therefore in a sequential procedure. Ill-configurations and noisy and sparse match populations have also been revised. The planar case is studied, provided its relevance on wheeled mobile platforms.

Feature tracking, motion and structure recovery are strongly related, so it is not possible to talk about one without mentioning the other, and inevitable references to structure estimation will be done in this section. Nevertheless, structure recovery will still be discussed in subsequent parts of this chapter.

2.6.1 SfM in Real Time

The state of the art in SfM can arguably be identified with real-time camera tracking and dense reconstruction on domestic hardware. One of the first works that attained real-time SfM was Mouragnon et al. (2009), who designed a method for motion estimation and 3D reconstruction from a video input. This work describes a whole SfM pipeline on a generic camera. Local Bundle Adjustment, in combination with a suitable key-frame selection, is introduced, which speeds up considerably the optimisation process. The generalisation of the model is achieved by the Pless equation, which is nothing else than the epipolar equation applied to lines which are expressed in Plücker coordinates[2]. In mainly structured scenes, Pollefeys et al. (2007) presented a real-time system which obtained ground-based dense 3D models of urban environments with a stereo-camera. This work focuses their effort on leveraging the redundancy of data to obtain consensus results. The large illumination variations are taken into account by using auto-exposure, and the feature tracking algorithm is adapted to track gain changes across the images. Klein and Murray (2007) (PTAM) supposed a significant step forward in terms of on-line camera tracking with a hand-held camera and commodity hardware. This work estimates the camera pose in a AR workspace while producing detailed maps with thousands of landmarks which can be tracked at frame-rate. This is attained by splitting tracking and mapping into two separate tasks, processing in parallel threads the camera motion and a 3D map of point features. Klein and Murray (2007) base their approach on dense maps of FAST features over a reasonably textured scene.
PTAM performs feature-based camera tracking and as such it has to overcome solid problems typical of point-based systems, such as rotations, occlusions or blurring. Dense reconstruction, when available, avoids all those problems, and with greater accuracy and robustness. This is the paradigm that Newcombe et al. (2011) (DTAM) has taken. DTAM relies on dense, every pixel methods. A textured depth map is generated by the minimisation of a global energy functional in a non-convex optimisation framework. In

[2]Plücker coordinates are the six homogeneous coordinates of a line in projective 3D space.

an interleaved fashion, camera motion is tracked at frame-rate by whole image alignment against the dense texture model. This allows DTAM to keep in place the tracking of the camera even at very high speed of motion. DTAM makes extensive use of GPU hardware, which makes it unsuitable for on-board computers. Engel et al. (2013) decreases the computational demand by representing as a Gaussian probability distribution the inverse depth of each pixel with a minimum image gradient, and then propagating that information over time.

In a similar fashion as DTAM, Forster et al. (2014) (SVO) present a semi-direct monocular visual odometry algorithm which works at pixel level, implemented on a Micro-Aerial-Vehicle (MAV) with impressive results. SVO can process 55 fps on a onboard embedded computer and more than 300 fps on a consumer laptop. SVO uses a probabilistic mapping method to explicitly model outlier measurements in 3D point estimation. The direct method used to track camera motion is based on pixel intensities, saving much time computation, since the feature detection and extraction phases are avoided.

However outstanding are the results for SfM provided by *direct methods*, they relay on certain assumptions which are not always applicable. *Direct methods* mainly confide their robustness in high frame-rate and small baselines, so that some flavour of *optical flow* in VGA resolution can be run on the sequence. Despite all their advantages, *Direct methods* simply can not be employed in certain cases, such as omnidirectional motion with low-quality image sensors (see Section 2.1). The amount of noise that we deal with in this work prevents us from these approaches. Other circumstances might be situations where images are separated by wide baselines, high image resolution constraints or even computational requirements by other threads.

In next sections we review the process of motion recovery, stressing those elements more relevant to our case of study.

2.6.2 Epipolar Geometry

Epipolar geometry extraction is a mature field in image processing. Since its introduction by Longuet-Higgins (1981), it attracted the interest of many researchers, who refined the estimation methods and spread the paradigms for epipolar geometry extraction. Epipolar geometry has been studied in all types of images (high and low resolution, near-planar images, feature-less images, etc.) and types of cameras (central panoramic,

eye-fish, affine cameras, etc). It has been studied over a pair of images, giving rise to the framework described in Appendix A, but also over three images, which produces the trifocal tensor, and even over four images, with the quadrifocal tensor (Hartley and Zisserman (2004)). Epipolar relationships have been produced from a range of common entities between images: points, lines, planes or surfaces can be arguments to the epipolar equation $\boldsymbol{x'}^T \mathrm{E} \boldsymbol{x} = 0$. A plethora of techniques has been proposed for a robust estimation of E, where each one suits different configurations between images. An extensive review on epipolar geometry and the robust extraction of the fundamental matrix can be found in Zhang (1998).

Robust techniques for inter-image motion extraction received great attention by the 3D vision community in the 1990's, and the feature-based case seemed to be settled with the refinement of the 8-point algorithm by Hartley (1997) and the employment of non-linear minimisation methods to SfM. Hartley proposed a normalisation over the matching population which drastically decreases the condition number[3], and the enforcement of the singularity of F by applying SVD. The 8-point algorithm usually serves as bootstrapping for subsequent iterative methods of optimisation. New approaches have appeared later on for rank-2 parametrisation, like those that employ Householder transformations (Wenzel and rainer Grigat (2005)).

This work follows loosely the procedure indicated by the 8-point algorithm, and then refines the results according to a version of the algebraic error minimisation algorithm given by Hartley (1998).

There have been presented in the literature a number of techniques for extracting the relative motion $\{\mathrm{R}, \boldsymbol{t}\}$ from E. The SVD technique was imposed thanks to its slightly superior stability and the development of computationally efficient linear algebra libraries. The properties of E which lead to this technique are elucidated in Huang and Faugeras (1989). However, other techniques have been proposed. Horn (1990) expressed the baseline and orientation in a pair-wise configuration by means of the trace and the matrix of cofactors of E, so that:-

$$\begin{aligned} \boldsymbol{t}\boldsymbol{t}^{\boldsymbol{T}} &= \tfrac{1}{2}\mathrm{Trace}\left(\mathrm{E}\mathrm{E}^{\mathrm{T}}\right)\mathrm{I} - \mathrm{E}\mathrm{E}^{\mathrm{T}} \\ (\boldsymbol{t}\cdot\boldsymbol{t})\,\mathrm{R} &= \mathrm{Cofactors}\,(\mathrm{E})^{T} - [\boldsymbol{t}]_{x}\,\mathrm{E} \end{aligned} \tag{2.2}$$

[3]The condition number measures the sensitivity of the output of a given function to changes in the input arguments. The condition number of a matrix can be expressed by the ratio between the maximal and minimal singular values of the matrix.

Horn was rivalling in use with SVD techniques until early in last decade. Other methods address the extraction of the motion directly from points correspondences and apply alternative techniques such as Semidefinite Programming (SDP) relaxations. Specific methods for particular cases have arisen as well, along with the problem of ending up in a local minimum. Chesi et al. (2002) propose a convex approach for the fundamental matrix estimation, so that the global minima is always found.

Within the literature there are many proposed optimisation algorithms for this problem and a survey can be found in Ma et al. (2001). The minimisation can be aimed at the algebraic error given by the fundamental matrix, or at the geometric distance between the points reprojected and the epipolar lines, of which are notable the gold standard and Sampson methods (Hartley and Zisserman (2004)). Ma et al. (2001) present an optimisation method based on the minimisation of the reprojection error[4] by means of differentiating the rotation and translation matrices and using for minimisation the Newton algorithm (Spang (1962)). Han (2005) tries to improve the Newton method, exploiting the Hessian matrix, and providing a new criterion to choose the step size during the optimisation procedure.

Torr and Zisserman (1997) is of special interest for us since Torr *et al.* analyse the effect of noise created by JPEG compression on the estimation of the epipolar geometry over a set of error metrics. They state that a compression beyond 85% does not permit correct estimations of the fundamental matrix. The dynamics of the JPEG compression will be described in detail in Chapter 3, but suffice here to say that JPEG compression is a lossy compression, where always information is lost, even though the quality factor used for compression is $Q = 100$. A compression of 85% is achieved by setting a quality factor $Q = 30$. A further compression produces a dominating tiling effect in the image space, creating too many spurious features. It should be noted that the biggest decrease of information occurs from $Q = 100$ to $Q = 70$. The work presented by Torr and Zisserman (1997) is, to the best of our knowledge, the only work that treats JPEG-compressed imagery in the context of epipolar geometry recovery. The quality factor Q of our imagery is 100, which does not prevent images from having high levels of noise, since the quality of the CMOS image sensor used (see Section 3.1.1) produces also artifacts

[4]The reprojection error is the distance between the projection on the image of a 3D point reconstructed and the feature which originated that point.

(with variance of $\sigma \sim 10$, see Section 3.1.1). The nature and behaviour of the noise present in this work will be described in detailed in Section 3.1.1.
The work presented here shows how the epipolar relationship is estimated up to inter-pixel error and features successfully tracked with JPEG compressions.

Continuous Case and Motion Field

Another approach to motion recovery is to assume that the camera moves continuously, so that there is just a differential distance between images. This assumption is reasonable in the case of very closely spaced views. Baumela et al. (2000) computes the epipolar geometry by means of differential matrices. SfM is tackled successfully in Kahl (2001) under the continuous case. Here Kahl applies a *Maximum a Posteriori* (MAP) estimator to SfM, in order to get smooth constrains and deal with critical camera motions and local minima.
Motion recovery follows a different paradigm in the case of *direct methods*. Since this approach is employed with small baselines (of magnitudes of pixels) these techniques work instead with the motion field, which can be defined as an ideal representation of 3D motion as it is projected onto a camera image. The motion field $\boldsymbol{v}$ in a point $\boldsymbol{P}$ of the image is given by the camera motion $\boldsymbol{V}$ and the depth Z of the real world points with respect to the cameras, yielding:-

$$\boldsymbol{v} = \frac{Z\boldsymbol{V} - V_z\boldsymbol{P}}{Z^2} \tag{2.3}$$

In this work we focus on a feature-based, sparse matching procedure to recover the motion of cameras, and the epipolar geometry is exploited on this purpose. Therefore such dense motion fields do not belong to the scope of this work and we will not delve into it further. The reader is directed to Szeliski (2011).

2.6.3 SfM on Robots and Navigation

The study and application of SfM methods in this work is oriented towards auto-localisation, navigation and 3D mapping of a scene by a low-budget omnidirectional mobile platform. The technique leading to navigation skills performed with solely visual information is referred to as Visual Odometry (VO). The research presented by Fraundorfer and Scaramuzza (2012); Scaramuzza and Fraundorfer (2011) develop a

documented and complete description of VO process. A current, classical implementation of VO based on SfM on standard CPU hardware is described in detail in Silva et al. (2012).

The term VO was coined by Nister et al. (2004), and it offers an enhancement over the unreliable traditional odometry given by wheel encoders. Due to the characteristics of this work only feature-based VO techniques are applicable here, and we will focus on them.

Nister et al. (2004) presented a whole system for near real-time ego-motion recovery of a mobile platform equipped with different types of camera rigs, emphasizing in stereo rigs. In the single case, Nistér *et al.* introduced the concept of firewall by which views of 3D points of images placed before a given firewall are not triangulated along with views from the other side of the firewall. This simple technique prevents the system from fatal divergences in camera poses that may provoke instabilities and spurious observations. This is because relative camera estimation starts afresh after a firewall is set. However desirable this trick is, it implies the splitting down of many feature tracks, which is a significant lack of efficiency. Our work presents a system stable against spurious views and where features are tracked to their maximum extent which, as opposed in Nister et al. (2004), provides stability to camera motion and structure estimation.

Implementations of VO that intend to outperform conventional odometry must be run in real-time. One possible strategy to work around the intensive computational load demanded by SfM subroutines is to provide the robot with offline maps of the environment. This is the approach taken by Royer et al. (2007). The robot undergoes first through a learning step, in which it is manually guided along a path so that offline 3D map of the scene is built as a batch. With this prior information the robot is capable of self-localising in real time. Davison and Kita (2001) investigate sequential methods for real-time mobile navigation. These approaches should form a representation of the "state" of the system with the information available at present, i.e. the current and previous images. Davison and Kita (2001) explain how this necessarily leads to uncertainty in the measurements, which is usually tackled with Bayesian and propagation algorithms. Another problem that sequential methods encounter is the size of the state, i.e. the number of features to track and triangulate. The size of the state increases as images are processed by the system, and at some point the computational burden will prevent the system from keeping up to date. Therefore real-time VO systems need

to bound the state size with mechanisms like a firewall, aforementioned. MonoSLAM simply limits the system to 100 features. The drift problem is also a serious issue in autonomous navigation. Systems which simultaneously build maps and calculate motion over long sequences commonly fall into the motion drift. The motion drift is made up of consistent errors in the estimated position of the camera relative to a world coordinate frame which lead to equivalent errors in estimated feature positions.
Mouragnon et al. (2006*a*) show an incremental VO algorithm which simultaneously minimises the camera poses and the 3D structure over a fixed number of frames, (Local Bundle Adjustment, LBA) starting with the most recent one. This way Mouragnon et al. (2006*a*) keep the pose uncertainty and the size of the system within computational limits. Their experiments show relatively long sequences with a low drift, which suggests how adequate LBA is in this context. Our implementation of the *Bundle Adjustment* technique is inspired in this idea (Section 3.8.2).

Batch SfM vs Sequential SfM

Once we know how to retrieve the epipolar geometry, the next step is to determine how are we going to process the images, either as they arrive from a video stream or as a complete set. The algorithms which process the images as they are received are called sequential techniques, whilst those that treat the sequence as a whole, batch techniques. Batch techniques usually provide better results, although it is clear that they must be run offline, on a closed group of images. Moons (2008) and Wai Yin Leung (2006) provide a complete explanation of the whole process. Snavely et al. (2008*a*) represents the "batch technique" philosophy taken to the maximum extent, for it receives as input uncalibrated random images from the Internet. Snavely et al. (2008*a*) present a multi-view stereo algorithm that deals with different lighting, scale and clutter as it tries to optimise the matching stage.
Snavely et al. (2008*b*) create a skeletal subset with selected images, by estimating the accuracy of two frame reconstructions and a graph algorithm. Subsequently the rest of images are added to the subset.
One important issue in batch techniques is the number of images. It usually requires a large amount of images to get reliable structure results. This problem has also been addressed in the literature where we see Saxena et al. (2007) reconstruct a given environment with few images by inferring the 3D position of blocks in the image (super

pixels) and using a Markov Random Field (MRF). Agarwal et al. (2006) provide a new method to reconstruct a scene based on few images. This method looks for a global optimisation of the SfM, in a scalable fashion.

Sequential techniques usually require non linear and recursive methods, but they have the advantage of working on an open stream. Therefore, they permit on-line implementation, as we would like our method to be for a robot transiting the environment. The first step is to consider a sequential method as a repetition of the two frames case. This is the line of Lee et al. (2008) and Zucchelli (2002). They recover SfM from a pair of images using optical flow. Zhang (1995), in turn, uses correlation and relaxation methods, with a robust technique (the Least Median of Squares, LMS) to discard false matches, in order to obtain the epipolar geometry of two images. Once the two-frame case is mastered, we can try to make the most of all the information given by previous images, and this approach gives rise to recursive algorithms. In the same way, Beardsley and Torr (1996) updates the structure recursively. Butterfield (1997), in turn, deals with the drift problem, which is a common issue in large sequences.

There are certain inter-image configurations where SfM methods become highly unstable. These situations, named as ill-configurations, should be tackled when performing visual mobile navigation.

2.6.4 Ill-Configurations

There are two types of singular cases wherein the fundamental matrix is determined by a two-parameter family of solutions: a) all the 3D points detected are coplanar and b) the motion is a pure rotation.

If the correspondences belong to a plane then their views are related by a homography (see Section A.3) and therefore $\boldsymbol{x_i'} = \mathrm{H}\boldsymbol{x_i}$. In this situation the epipolar equation yields $\boldsymbol{x'}^T \mathrm{F}\boldsymbol{x} = \boldsymbol{x'}^T \left(\mathrm{FH}^{-1}\right) \boldsymbol{x'} = 0$, which is true if FH^{-1} is skew-symmetric. Therefore the solution for F is any matrix of the form F = SH, where S is skew-symmetric, and can be expressed by a column vector $\boldsymbol{t}$ (see Appendix B.1, Eq. B.1.1). $\boldsymbol{t}$ can take any value, which leads to a two-parameter family of solutions for F (one degree of freedom is irrelevant as F is homogeneous and thus up to scale).

If there is no translation between images the epipolar geometry is not defined, since the camera centres are coincident and there is no epipole. In this case the two images

are related by a homography, and like in a) there will be a two-parameter family of solutions for F.

It is important to be aware of these cases, because configurations "close" to degeneracies - i.e. with small baselines or near-planar scenes - will create numerically ill-conditioned estimations. Moreover, small baselines are likely to occur in a sequence of images streamed by a robot transiting a scene. Key-frame selection is therefore compulsory in VO. Likewise, scenes purely made up of walls or planar surfaces should be avoided.

There is research that copes with this problem. Torr et al. (1998) thoroughly study it, and offer workarounds for these situations. In case b) maintaining correspondences is proposed as alleviation of the problem, so the computation of camera poses is done once translation of the platform is resumed. Their method uses homographies to solve the robust estimation of the correspondence problem. Eun and G. (2007) manage to attain dense reconstruction of near-planar surfaces by a succession of three BA runs combined with 2D registration and a plane + parallax representation.

In our case of study we cannot afford to allow wide baselines between frames. The noisy nature of the images streamed by the platform considered prevents SfM with wide baselines because that would lead to insufficient number of matches. Scarce matching populations would have a severe impact in the epipolar geometry estimation, which would not be able to outrun the noise. With this conundrum, we have been forced to work with small baselines which will result in ill-conditioned equation systems. The method and approach devised to obtain robust estimations in this situations is described in Chapter 3.

One possible approach towards small baselines is to avoid altogether the extraction of the relative motion through the fundamental matrix and make the most of the characteristics of the problem. For small translations, it is possible to accurately recover the rotations by direct computation, and also the translation recovery is insensitive to first-order rotation errors. This is the approach taken by Oliensis (1999) in a multi-frame SfM algorithm. A remarkable achievement in this context is Yu and Gallup (2014). Under the assumptions of small motion, this work extracts dense reconstruction out of a short video taken by an user who intends to hold the camera still. The small displacements produced by accidental motions of the photographer provide enough baseline for them to acquire dense depth maps.

We have not made special assumptions on the motion as our approach intends to be general. Instead we have devised robust filters against noise in matching populations in order to ensure stable motion retrieval.

2.6.5 Resection

This section has discussed hitherto how to retrieve the relative pose between two cameras with matched feature points. However, as it has been discussed in Section A.5, the relative pose estimation is not enough when working with sequences of images. It is necessary to express all the camera poses with respect to a global frame of reference. This problem is called *Global Pose Estimation* (GPE) or the *Perspective-n-Point* (PnP) problem. Since the methods that address this question ultimately estimate the extrinsic parameters of each camera, PnP is sometimes identified with *Exterior Orientation* (EO). EO is the problem of calibrating the camera of a mobile robot when the intrinsic parameters are known. The process of placing a camera in the global frame of reference is called *resection*. Section A.5.1 describes the linear method for resectioning cameras. EO is an old problem in robotics. The methods developed in this area can be classified in 3 groups: a) approximate methods, b) point-based methods and c) projective methods. The approximate methods are based on linear algorithms and can be used when no great accuracy is needed. Examples of approximate methods are the *Direct Linear Transformation* (DLT), formally presented by Abdel-Aziz and Karara (1971), and methods which implement coordinate and spacial transformations. For single image resection the *Church's method* can be used. Fiore (2001) solves the EO problem by using orthogonal decompositions to isolate the depths of the points, so the problem is reduced to an absolute orientation with scale problem, which is solved by SVD. Point-based methods take advantage of geometric properties of points and lines in the image, such as collinearity, coplanarity and coangularity. Finally, projective methods make use of the projective geometry that derives from the mapping from the 3D world to the 2D image plane.

Within the group of projective methods we can differentiate between iterative and non-iterative methods. The former obtain excellent results, which come at the price of extensive computation. The gold standard algorithm minimises the reprojection error (see Section A.7.1) iteratively by applying Levenberg-Marquardt (LM) optimisation.

Here the algorithm is bootstrapped with DLT and a previous normalisation is carried out, in a similar manner as explained in Section A.4.3.
There are non-iterative PnP methods that achieve very good results in a closed manner. Gao et al. (2003) implements Wu-Ritt's zero decomposition algorithm, which gives a complete triangular decomposition for the PnP equation system. Gao et al. (2003) additionally provides criteria to know the number of solutions of the PnP problem. The work of Gao et al. (2003) combines the analytical algorithm and this criteria to produce a new algorithm, CASSC (Complete Analytical Solution with the assistance of Solution Classification), which is claimed to give robust camera pose estimations. Another non-iterative solution is Lepetit et al. (2009). This solution is especially attractive because not only are its results precise, but in addition its computational complexity grows linearly with n. In addition, this algorithm is capable of working with both planar and non-planar configurations. The n 3D points are expressed as a weighted sum of four virtual control points. Subsequently the coordinates of these points are estimated in the camera reference and expressed as weighted sum of the eigenvectors of a 12×12 matrix. The weights are obtained by solving a constant number of quadratic equations.

2.7 Structure Triangulation

The second step of SfM is the estimation of the structure, as it requires the knowledge of the camera poses of the sequence. Based on the mapping from 3D world to the 2D image plane (Eq. A.2.13), triangulation techniques extract the 3D position of the points corresponding to inter-image matched features.
As its name indicates, the triangulation method obtains the position of a 3D point $\boldsymbol{X}$ from the rays that join it with the centres $\boldsymbol{C}$ of each camera that sees it. These rays, which intersect with the image planes at the matched features, create a triangle where all the sides are known and the only unknown vertex is $\boldsymbol{X}$.
The oldest and most basic triangulation method is the *mid-point* method, as mentioned by Hartley and Sturm (1997); Kanatani et al. (2008). Ideally the rays coming from each camera centre intersect at $\boldsymbol{X}$, but due to image measurement noise in practice this is usually not the case. In the *mid-point* method the point triangulation is estimated as the midpoint of the shortest line segment connecting both rays, i.e. the common perpendicular line.

The *mid-point* method can only be applied over a pair of images, and it is neither affine invariant nor projective invariant, since perpendicularity and distance ratios are not preserved by these transformations. Therefore the use of this method is discouraged.

A popular approach to this problem is the *Linear Triangulation* or DLT. This method has been described in detail in Section A.6. DLT establishes a linear system of equations from the relationship $\boldsymbol{x} \sim \mathrm{P}\boldsymbol{X}$, where the symbol "$\sim$" indicates that this relationship is true up to scale. By expressing the former expression in a vectorial equation, a linear system $\mathrm{A}\boldsymbol{X} = 0$ is created for every camera over which $\boldsymbol{X}$ is projected.

The equation $\mathrm{A}\boldsymbol{X} = 0$ is a *Linear Least Squares* problem, and it can be solved by SVD. If the vector $\boldsymbol{X}$ is taken in its homogeneous form, i.e. $\boldsymbol{X} = (x, y, z, k)^T,\ k \in \mathbb{R}$, then the resolution method is called *Linear Eigen Triangulation* (Linear-Eigen). However, if we fix $k = 1$ the linear system is solved using inhomogeneous coordinates and the method becomes *Linear Least Squares* (Linear-LS).

None of the aforementioned methods is projective invariant, although Linear-LS is affine invariant, since the last row of an affine transformation is $(0, 0, 0, 1)^T$ (see Sec. A.3). On the other hand, Linear-LS assumes that the 3D point $\boldsymbol{X}$ is not at infinity, which might not be the case (points at the horizon or vanishing points are at infinity). This is an issue when performing projective reconstruction. These two methods are widely discussed by Hartley and Sturm (1997).

Both methods Linear-Eigen and Linear-LS minimise $\|\mathrm{A}\boldsymbol{x}\|$ which has no geometric meaning. What should be minimised instead is the reprojection error (see Section A.7.1). Hartley and Sturm (1997) propose an iterative method that minimises the L2 norm of the reprojection error, which provides the *Maximum Likelihood Estimate* (MLE) solution. This method does not only improve the results given by the linear methods but also is invariant to projective transformations.

Hartley and Sturm (1997) introduced a close solution when only two cameras are involved. If the noise follows a Gaussian model, this triangulation method is then provably optimal. This method implies to find the roots of a 6-degree polynomial. In this stereo context, Kanatani et al. (2008) argue that this polynomial method has two singularities in the epipoles, and present an iterative method based on the epipolar error, which claim to be just as precise and equally costly.

In case that multiple cameras view the point $\boldsymbol{X}$ LM can be applied, as well as the Sampson approximation, whose results are not as accurate but come with less computation

weight.

Practically all the methods described here have been tested in this work. The iterative methods were discarded due to their computational burden. Both the close form and Kanatani method find a set of matches which minimise the L2 distance to the real matches and exactly satisfy the epipolar constraint, but under the assumption that the fundamental matrix F is correct. This work refines the camera poses as new images are received by the system, and given the noise present is not wise to substitute the matches detected by fictitious matches which fulfil $\boldsymbol{x'}^T \mathrm{F} \boldsymbol{x} = 0$, where F usually does not fully describe the actual relative motion. In addition, the polynomial and Kanatani method are designed for the two view case.

The triangulation method present in the working version is Linear-Eigen, since it is light, allows triangulation from multiple cameras and can be refined as new images are received by the system. Linear-Eigen method has also been easily adjusted to the new feature tracking system devised in this work, so we deem it as the most suitable triangulation method for this system.

2.8 Bundle Adjustment

Bundle Adjustment (BA) is the process by which the results obtained in the SfM method are further refined. Due to noise, inter-image configurations, point singularities and other geometric noise, the camera poses and structure at this stage usually are only close approximations of the true values and can be greatly improved. It is not uncommon that the whole SfM phase serves as a mere initialisation of BA.

As outlined in Appendix A, BA simultaneously minimises both camera poses and structure through a cost function defined by the reprojection error. There has been much research in BA in the look-out of simple and efficient formulation of BA, since it is a key step in 3D reconstruction and one of the bottle necks when real-time reconstruction is intended. Moreover, owing to the characteristics of the BA problem, the matrices involved in this algorithm are sparse, so great computation time and memory can be saved when the sparseness of this method is properly tackled.

Triggs et al. (2000) offer an extent and profound study of the most relevant aspects to take into account when implementing BA. The choice of the cost function affects the efficiency of the problem but also the robustness of the optimisation. Many practical

estimators have been proposed, like Maximum Likelihood (ML), Maximum a Posteriori (MAP) and explicit Bayesian model averaging. The most stable metrics are based on the whole population of views (inliers and outliers), since they allow for the presence of mismatches. The vectorial space of the structure should be chosen according to the nature of the problem. Generally speaking, homogeneous coordinates provide better results, since in projective geometry *infinity* is just another place, which can be smoothly visited by the points during the optimisation process. However, when working with homogeneous coordinates the cost function must ensure gauge invariance, i.e the choice of the coordinate system must not affect the geometry of the structure and camera poses configuration.

Special attention should be given to the parameterisation of camera rotation matrices, so that singularities and regions of uneven coverage are avoided. A common choice is the use of quaternions subject to $\|q^2\| = 1$. The quaternions are a 4-coordinate number system that extends the complex numbers, with special characteristics which make them an excellent choice to parametrise camera rotations (see Section 3.8.1).

BA is the problem of optimising a metric estimator. The parameter space that the estimator uses may be non-linear, so an approximate local model must be defined to linearise the displacements $\delta\boldsymbol{x}$. The art of a robust optimisation relays largely on the choice of the local model, its minimisation, to make sure that the estimate improves over iterations and the criteria for stopping the iterations. A common local model employed is the Taylor series, in its linear (first order methods) and quadratic (second order) versions. The quadratic version involves the use of the gradient vector and the Hessian matrix. Depending on how the local model is formulated the resulting method is the Newton's method, which approximates the local model to the cost function in the surroundings of the minimum point; the Gradient method, which stops the approximation of the cost function at the first derivative of the Taylor series (the gradient vector), or the Levenberg-Marquardt method, which falls in between the former two, as described in Section A.7.

Quality control is an important issue in BA. Diagnostic tests should be used to detect outliers and evaluate the accuracy and reliability of the estimations. Triggs et al. (2000) provide different mechanisms for outlier identification, techniques for acquiring the data and methods for achieving reliability in the results.

Since in each update computation is roughly quadratic in the size of the state vector, it is necessary to limit it. In this sense, the algebraic manipulation of the matrices involved in BA should make the most of the inherent sparseness of the problem in order to bound the number of operations. Cholesky and LDL decomposition of matrices are examples of matrix algebra applied to BA. These decompositions are mostly used when computing matrix inverses. In terms of implementation, BA algorithms take advantage of software libraries designed to efficiently handle memory, fast variable ordering and customised sparse systems solving.[5]

For this purpose the Schur complement and the reduced system that derives from it is used at the update step. The Schur complement is a matrix made up with blocks of the matrix A in the system $A\boldsymbol{X} = 0$ which permits the reduction of the system so that the unknowns can be easily computed by Gaussian elimination. Moreover, the Schur complement allows ignoring selected unknowns during the update. This is also useful when one is interested in updating only the camera poses or only the structure. Another way of simplifying the system is to change the system coordinates so that the transformed Hessian meets certain conditions (such as having the eigenvalues well differentiated) and permits faster operations. This transformation is called preconditioning.

Working on these premisses researchers have been able to optimise and perfect the BA algorithm. Jeong et al. (2010) order the cameras in such a way that allows efficient handling of the amount of blocks that are filled in during the LDL block factorization, and uses block-based preconditioning conjugate Gradient on the reduced camera system. Jeong et al. (2010) claimed to have attained the fastest BA system at the time.

Schur complement greatly simplifies the system, but when working with large sequence of images (an order of thousands), the Schur complement trick is not capable of overcoming the complexity of the problem and fails. By using conjugate Gradients at the Newton step and preconditioners to evaluate its performance, Agarwal et al. (2010) obtain a system that, unlike the Schur complement, scales to larger sequences. In order to deal efficiently with large sequences too, Sibley et al. (2009) derive a relative objective function for BA. With the goal set on attaining really large scale simultaneous localisation and mapping algorithm that operates incrementally in constant time, Sibley et al. (2009) work with a metric-space defined by a connected Riemannian manifold. With a global coordinate frame, BA becomes expensive, mostly at loop-closures, when all the

[5] As an example, BLAS3 is a C++ library developed to specifically meet these requirements.

parameters need to be adjusted. However, with this approach the MLE is constant in time. Since the manifold is a metric space, and distance between points can be computed, it is possible to plan algorithms which are commonly defined over graphs in the first place. In addition, since the manifold is locally Euclidean, it is possible to apply algorithms which require local metrics, like obstacle avoidance. The BA system proposed is therefore locally Euclidean and globally topological, where the frames of reference are connected by the Riemannian manifold. The work of Sibley et al. (2009) is therefore an adaptive relative formulation that can be viewed as continuous sub-mapping approach. Another formulation which tries to minimise the BA algorithm is the differentiation of two different minimisation systems: a global one, responsible for the drifting problem and loop-closures, and a system defined in the surroundings of the last frame in the sequence. This is the approach taken by Holmes et al. (2009), that represents landmarks and camera poses in relative frames and remove temporarily certain measurements, so that BA can be split into a *local* BA with most recent cameras and landmarks, with a fixed computation time, and a *global* BA with all the keyframes, in cubic time, like a standard BA. This produces three important outcomes: a) the local BA allows exploratory map-building to keep pace with camera pose tracking, b) it produces statistically consistent results and c) any update in positions from the global adjustment are immediately incorporated in the local BA. The aforementioned works: Mouragnon et al. (2006*a*) and Mouragnon et al. (2009) set out a similar layout in their BA systems. This work has been inspired by this philosophy in the implementation of BA.

Yet another technique used in the simplification of BA is the elimination of parameters from the problem. Zhang et al. (2006) propose a BA system which does not involve solving for the camera orientations. They eliminate the camera orientation parameters by algebraic manipulation, and then formulate a rotation matrix free cost function for BA, which makes the system more robust since does not depend on rotation matrix disturbances. In this context, Rodríguez (2013) proposes GEA (*Global Epipolar Adjustment*), a high-performance structureless BA correction method based on algebraic epipolar constraints. Thanks to the algebraic nature of the cost introduced, it can be very efficiently optimised, in most cases decreasing into a fraction the time required by BA to obtain the global minimum.

In a context closer to the scope of the present work, Engels et al. (2006) show how BA can be used as a component of a real-time camera tracking system. It describes how a

significant part of BA can be done every time a new frame is added. Working on long sequences, they quantify the failure rate, and demonstrate that BA decreases the rate of gross failures in such a way that decreases the frequency of total failure of the camera tracking.

Regarding software implementations of BA, there are two efficient examples in the literature: Sparse Bundle Adjustment (SBA, by Lourakis and Argyros (2009)) and Sparse Sparse Bundle Adjustment (sSBA, by Konolige (2010)). SBA focuses on a flexible API and looks for efficient performance. It exploits the sparseness common in SfM problems, by splitting the normal matrix into different camera and structure blocks and implementing the sparse Schur complement. sSBA incorporates advances in direct sparse Cholesky solvers, giving as a result a faster BA implementation. However, SBA is easier to integrate in the code and permits the use of customised camera "drivers", as well as the optimisation of uncalibrated images.

2.9 Collaborative Perception

Over the last 15 years a great interest about Multi Robot Systems (MRS) and camera networks has grown amongst researchers. As a result, different paradigms have risen according to the approach of each MRS with regard to aspects as communication, hierarchy of decisions, social behaviour and collaboration. A good taxonomy on MRS can be found on Iocchi et al. (2001). Many issues and possibilities have been opened as the research went ahead. Zhu and Yang (2010) analyse several dimensions of network robot systems, such as robotic sensor networks, swarm intelligence and cooperation amongst robots. Since the robots are acting within a group, they have to reach an agreement about what is to be done next. This consensus problem has been tackled in many ways; Ren et al. (2005) deeply survey the most relevant approaches. Also, many algorithms that are well known on a singular camera become really complex, and with multiple possible solutions, when they are applied to a group. Radke (2008) takes an extensive look in the literature and describes how these algorithms evolve under the distributed system. Some topics of our interest are handled, although the vast majority of them are applied from the static perspective. When our work started, there was no work that had attempted to reconstruct a scene through a team of mobile robots with the aid of image sensors. This statement is supported by Radke (2008).

2.9.1 Distributed Reconstruction and Localisation

The feature that characterises the implementation of a collaborative system over a single system is that a strategy needs to be defined when several platforms interact with the environment. For a system of robots to reconstruct cooperatively a given scene an appropriate decision making algorithm must be defined. Whether the decision should be done in a voting or horizontal system or in a hierarchical configuration should be determined according to the requirements of the problem and specifications of the platforms.

A good example of this question is Chang and Wu (2013) who propose a decision making technique to enhance signal detection and indoor mobile robot positioning for a global positioning satellite system (GNSS) receiver. In a context of mobile robot networks, Zavlanos et al. (2011) provide a theoretical framework for controlling graph connectivity. In order to maximise the mathematical connectivity of a network, Zavlanos et al. (2011) make use of algebraic tools which range from convex optimisation to a subgradient-descent algorithm. Multirobot rendezvous, flocking and formation control are also discussed.

2D-Mapping scenes collaboratively has been approached in many ways. Özkucur and Akin (2010) develop multi-robot map merging for navigation purposes. They implement a SLAM algorithm based on a particle filter (EKF-SLAM) and extend it to *fast-SLAM*. The optimal parameter set is searched by evolutionary strategies. A cohesive literature review about multi-robot systems is done, where they tackle the problem of the initial position, the usage of heuristics and the transformation from one robot location to another. Dias et al. (2013) introduce a cooperative perception framework for multi-robot real-time 3D high dynamic target estimation in outdoor scenarios. A decentralised cooperative perception layer is obtained. They use epipolar constraints for feature matching, feature searching and detection for image processing of robots with low computational power. This framework can be integrated in a Decentralised Data Fusion (DDF) multi-target tracking approach to reduce uncertainty propagation, in the context of data association problem and track initialisation issues. An interesting example of collaborative perception is the work of Palacios-García et al. (2011). They create a system where the robots can learn from the others; from the SIFT descriptors of

objects and their silhouette, the robots learn representations of objects in the environment. Collaborative tasks involved in this algorithm are: strategies for coordination and communication, exploration, mapping and deployment, sensing, surveillance and monitoring. The decision making is realised in a decentralised manner. Kim et al. (2010), based on iSAM, give a relative formulation of the relationship between multiple pose graphs. iSAM, developed by Kaess et al. (2007), is an incremental smoothing and mapping approach for SLAM based on fast incremental matrix factorisation. iSAM provides efficient algorithms to access the estimation uncertainties which enable data association. Kim et al. (2010) extend iSAM for multi-robot mapping based on multiple pose graphs, by using a relative formulation of the relationships between multiple pose graphs. This approach avoids the initialisation problem and is more efficient than global formulation. Data association and loop closing is facilitated through iSAM. Kim et al. (2010) rely on direct encounters (the robots see each other), as opposed to indirect encounters or loop-closures (a robot sees part of the scene already seen by the other one). Additionally, it works with anchors, which are used to put the camera poses in a global frame when an encounter occurs. This work highlights crucial problems in multi-robot mapping, such as consistency, computational efficiency and communication requirements. Guan (2006) addresses three important problems of cooperative 3D vision: reconstruction, motion planning and multiple robot collaboration. Guan locates each robot with feature-based localization and probability framework. Regarding motion planning, collision is avoided between robots and efficient paths are designed for further exploration. The environment modeling is done with passive sensors (stereo cameras) and active sensors (LRS, LIDAR). The information provided by these sensors is then merged. This may be done by registering two different models up to affine transformation based on some parametric models and then use Iterative Closest Point (ICP) method. The localisation of each individual robot is attained through SLAM. Multiple robot localisation is attained by means of encounters between robots. Basically, Guan (2006) works with individual uncertainties, and these uncertainties are transferred between robots when an encounter occurs, with the aid of Monte Carlo Localisation. This approach has some limitations. Knowledge about where the other robots are not present can be extracted if a robot sees a scene without any encounter, and Guan does not make use of this information. The system described assumes that robots are marked appropriately, and the collaboration is passive. Finally, the scheduled path is not changed to help localisation.

Recently the collaborative approach has been used for reconstructing 3D environments. In combination with GPS information, Wendel et al. (2012) merge by alignment 3D dense reconstructions from several UAV into one single map, in a framework tailored for façades and urban scenes. Likewise, Reid et al. (2013) create a multi-robot system designed to navigate, explore and map large-scale urban environments.
Regarding cooperative localisation, a system made up of multiple robots offers different ways to localise each robot with respect to the other robots and ultimately with respect to a global, fixed frame of reference. Breitenmoser et al. (2011) performs relative localisation of mobile devices (quadrotors) by using the monocular vision of the devices and a module which targets markers in the scene. In essence, the target is identified in the image and subsequently the robot position. Working with a fixed distributed network of non-overlapping cameras, Anjum (2011) localises the camera by using trajectory estimation of an object moving around, with a kalman filter. Multiple cameras are then located with an automated camera calibration algorithm. Kato et al. (1999) make use of omnidirectional vision robots to develop a method to identify themselves. This method consists of a *Distributed Robotic System* (DRS), and it is built on the assumption that many large-scale tasks are done better with multiple, simple robots rather than one single, sophisticated machine. Kato et al. (1999) emphasise the importance of the communication between robots, when doing tasks collaboratively. The localisation implemented is relative: each robot locates itself with respect to the society of robots, and for this reason each robot should be readily identified in the environment. A traditional approach in navigation to locate a mobile object is the technique *Dead Reckoning* (DR), which consists of calculating the current position of the mobile platform from previously determined position, and advancing that position based upon speed and course. DR has a significant setback, as it is subject to cumulative errors. Kurazume et al. (1994) propose, as opposed to DR, a method by which a group of robots splits into two and each half serves as landmark to the other alternatively.

2.9.2 Different Configurations

The multiple approach has been implemented on diverse setups of platforms. Jeong and Lee (2013) present *inchBot*, a novel swarm micro-robotic platform, endorsed with stackable hardware and omni-directional motion enabled by flexible spoke wheels. Following the policy stated by Kato et al. (1999), a cooperative search approach of a robot

swarm is presented by Tang and Eberhard (2013). They design a model of the robots with omni-wheels and develop an algorithm based on the behaviour of flight of birds.
A group of robots can take advantage of wireless communication. Otsuka et al. (2013) establish an algorithm for wireless communication between a group of low computational power robots (which imitate insect behaviour), which happen to be omnidirectional and with a similar design to the ones used in this work. The concept of always connected cloud computing can also be used to perform the computational work which is common to a group of mobile devices, leaving the individual updates, much less expensive, to each robot. This is the approach taken by Riazuelo et al. (2014). Taking as a starting point PTAM (Klein and Murray (2007)), the map optimisation phase of the SLAM algorithm is carried out in the cloud, so all the requirements that each platform needs is internet connection. The structure seen by each robot is automatically estimated by their RGB-D cameras. The centralised system provides storage for the map and individual maps that can be used by other robots. This configuration allows a robot to fuse its map with the one in the database.
Since image overlapping is very likely to happen in a team of cameras, feature matching finds a new dimension in a distributed system. According to their data, Ermis et al. (2010) manage to develop a correspondence matching system based upon activity features with better results than SIFT. Regarding tracking, Wang et al. (2010) present a camera network which is able to track mobile objects, by performing a hierarchy of events.
Camera calibration has also been tackled in distributed systems. Kassebaum et al. (2010) estimate the projection matrices of a group of cameras by means of a 3D target of known geometry. The cameras share information to locate themselves within only one coordinate frame. Lobaton et al. (2010), in turn, present a simple representation of an ad hoc camera network that captures topological information about the scene covered, without prior knowledge of each camera location.

2.10 Summary

This chapter has reviewed the state of the art methods that involve all the steps of the SfM process. We have found limitations in the fields of concern of this work, and in this work we propose methods to address them. Specifically:

1. We have seen that research on omnidirectional robots (Oliveira et al. (2009)) addresses dynamic modelling, but to the best of our knowledge no 3D Vision technique has been applied on this platform. We extend the research on 3D vision on mobile platforms, by developing a SfM system on a holonomic robot.

2. Noise is usually used to measure the robustness of a method, and few works try to deal with it within the process (Chang and Hebert (2002); Hartley and Zisserman (2004)), but either do not assume so much noise as in this work or apply methods that are not applicable on distributed reconstruction. We bridge this gap by the integration in a distributed system robust and light-weighted noise filters. Therefore SfM systems that handle noise (Chang and Hebert (2002)) are here extended, since we develop a more general framework that can be used for distributed reconstruction, capable of coping with higher level of noise. This work can also be seen as an extension of Thomas and Oliensis (1999); Vidal and Oliensis (2002) altogether, since the issues of noise and ill-configurations are treated here as a whole.

3. In terms of feature tracking, the feature tracker KLT has recently been improved (Rohith et al. (2013)), but assuming prior information on the imagery. VO techniques use either direct methods (Forster et al. (2014); Newcombe et al. (2011)) to deal with small baselines -inviable with noisy images- or split down the feature tracks (Nister et al. (2004)) to keep computation affordable, which would make the 3D reconstruction unstable. These limitations are addressed in this work with the implementation of a novel feature tracking method with specific filters that manages the scarcity of feature tracks so a stable 3D reconstruction is attained. Our system makes no assumptions on the imagery, deals with high levels of noise ($\sigma \sim 10$) and intends to make the most of the feature tracks generated. In particular, the feature tracking system presented by Rohith et al. (2013) is here generalised to any type of scenario.

4. Regarding distributed reconstruction, there are contemporary investigations which obtain 3D scene visualisations from groups of robots or sequences, but even these works are supported by external tools like the Cloud or RGB-D cameras (Riazuelo et al. (2014)) or rely on long overlaps over sequences, with an expensive

matching method (Zhang et al. (2010)). This system is easily extended to perform distributed reconstruction without the support of external sensors, and requires minimal loop-closures between individual reconstructions to merge them into a global 3D structure. We specifically improve the technique used by Zhang et al. (2010), as our implementation of the loop-closing system (FAB-MAP) does not need long stretches of sequences overlapped and can merge individual reconstructions with minimal common views.

Chapter 3

Structure from Motion on a Single Platform

This chapter describes how Structure from Motion (SfM) is achieved from images streamed by a low-budget mobile platform. From the compressed images transmitted via wireless to the surface rendering of the reconstruction, the methods and algorithms devised to overcome the difficulties encountered are described in detail as a complete pipeline.

First of all, the platform chosen for our experiments is presented and its specifications described. The special characteristics of this mobile robot (omnidirectional, cheap, wireless communication) make it a very good candidate for our research, but these characteristics come with important caveats, such as ill-conditioned inter-image configurations and ever present noise in images.

Once the streamed images are received, the images are preprocessed, in order to alleviate the effect of noise. In the matching section the filters implemented to ensure a good population of matches are described. Following the matching step, the method implemented to extract the epipolar geometry in ill-conditioned inter-image configurations and with noisy and sparse matches is explained. Subsequently the sparsity of the matching population is dealt with by a novel tracking system, specifically designed for this work. In the triangulation and Bundle Adjustment steps we describe the specific arrangements undertaken here. The flowchart of the SfM process applied on a given image I_n is shown in Fig. 3.1.

Finally the post-process, which maximises the population of the point cloud, and the

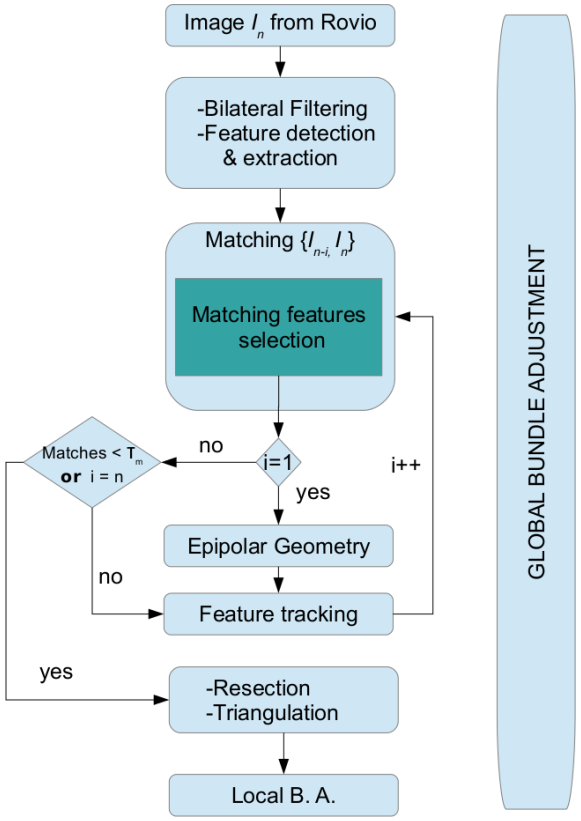


Figure 3.1: Flowchart of the SfM process implemented on a low-budget mobile platform. After a first stage of image preprocessing and feature detection, a recursive matching and feature tracking process take place, wherein the epipolar geometry of I_n is extracted. Subsequently the second part of SfM is executed, where the 3D points are estimated (Triangulation) and the camera motion and structure simultaneously refined (Bundle Adjustment, B.A). On a different thread a global scope B.A. is run.

surface rendering, where 3D filters are applied to smooth the reconstruction, are described.

To properly comprehend all the methods and filters introduced in this chapter first it is necessary to understand the low-cost platform of choice.

3.1 The Platform: Rovio

The experiments carried out in this work have been done on the mobile platform Rovio, introduced in Fig. 1.5. Rovio is a commercial mobile widget manufactured by WowWee that has been designed to be controlled over the Internet and can be used as a mobile webcam.

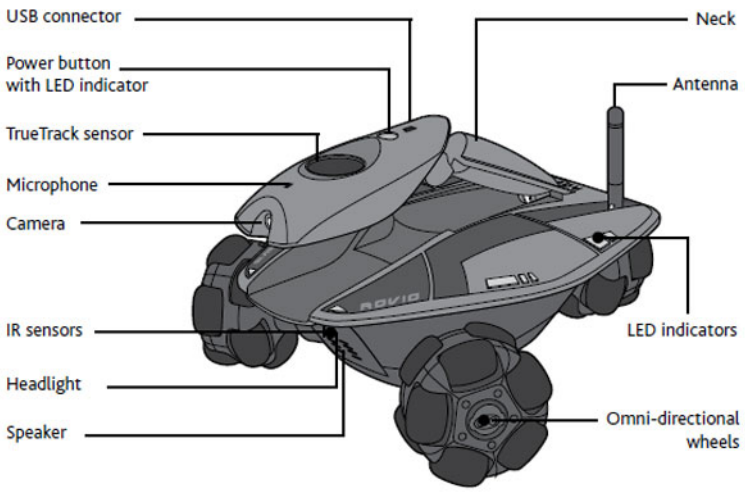


Figure 3.2: Rovio features. The webcam is placed at the tip of the articulated arm. Source: WowWee.

Fig. 3.2 illustrates the main features of Rovio, which are stereotypical for low-cost robots of this class. Provided with Wi-Fi connectivity (802.11b, 802.11g), Rovio is also equipped with an articulated arm. On the tip of the arm a VGA (640×480) camera is placed. In this work the arm has been used in the position shown in Fig. 3.2 since it provides the biggest field of view of all the possible positions of the arm. Rovio is wheeled so that its movement is stable. Moreover its three wheels are arranged in radial axes and each wheel actually includes 10 smaller wheels on its hub, which allow the robot to go any direction and adopt any aspect on the plane with just one turn (see Fig. 1.6). Regarding our research the most important feature is the wireless streaming of images as it makes the platform suitable for a sequential SfM application and enables it to move around freely without the impediment that a cable would create. In addition Rovio has an up facing True Track sensor designed to allow it to estimate its position of a given enclosed environment from a specific base station. On the front there is an Infra-Red (IR) sensor usable for basic obstacle avoidance.

The choice of these robots is suitable as well as challenging. It is suitable given the configuration and features of the Rovio as a low-cost platform. On the other hand, it is also challenging as a standard SfM usually requires good quality images and accurate measurements of the robot movement, both of which the platform lacks.

In face of the reconstruction of the environment by using such a mobile platform, several

Figure 3.3: A blurred image delivered by Rovio.

issues arise. First of all, the odometry supported by Rovio is not reliable. When moving sideways, given the holonomic nature of Rovio, it is likely to skid and have inadvertent changes of orientation. This effect is pronounced by the instability of the torque given by the motors at low speed motion. The odometry provided by the wheel encoders results therefore useless for accurate navigation purposes. Nevertheless, in Section 3.5 the use of the wheel encoders as a support for motion estimation is discussed.

In addition, if the wireless signal is poor the streaming video becomes unsteady. Bad image delivery combined with movement irregularities causes images received to be blurred, mostly when Rovio turns. Fig. 3.3 is an example of a blurred image produced by these factors. In order to avoid images out of focus the reception of the imagery occurs when the platform is in stationary position.

The omnidirectional configuration of the platform allows it to perform rotational motion in the plane around a central axis, with negligible displacement. This characteristic, typical of omnidirectional platforms, produces that the distance between consecutive images (called baseline, see Section A.4) may be small with respect to the field of view of the image, which results in a case of near-degeneracy, as explained in Section 2.6.4. These inter-image ill-configurations render the extraction of the epipolar geometry unstable. Special filters and methods have been devised in Section 3.5 to robustly retrieve the relative motion between images.

Finally, the quality of images streamed by this platform is very poor. Since Rovio is a low cost platform, the image sensor is cheap and prone to noise and saturation.

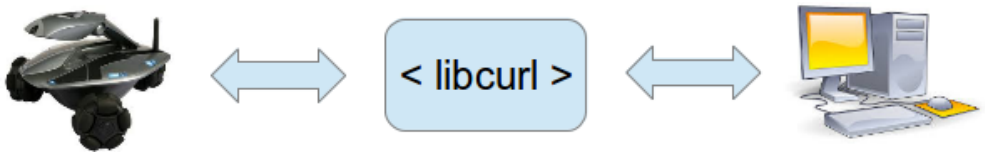


Figure 3.4: Software environment. The Rovio communicates with the desktop via libcurl requests. Source: partially from Wikipedia (2015*b*).

Additionally, in order to allow the wireless streaming of images, JPEG compression [1] is applied to them. Due to the low quality of the image sensor and this severe compression that images undergo, the images received by the system have high levels of noise. This noise has become a great obstacle throughout all the steps of our SfM process. Many methods described in this chapter have been developed to overcome the problem of noise.

Software environment

The robot Rovio can be given an IP address to which a PC can connect either over a WLAN network or via the Internet. All the instructions that the Rovio receives, as well as the packages that it sends, are encapsulated by modules of libcurl (libcurl (2011)), a free and easy-to-use client-side URL transfer library. Libcurl is widely used between clients and servers in the Internet. The libcurl commands may be sent through the API offered by the manufacturer, which is an actual website hosted by the Rovio. Alternatively, in order for researchers to integrate the communication with Rovio in their framework, a C++ API has been developed (Breckon (2011)) that allows the developer to integrate remote commands to the Rovio. Fig. 3.4 shows a scheme of the communication Rovio - PC desktop.

Our work uses the API Breckon (2011) to configure the Rovio and its camera, as well as for receiving the images that are processed by our system.

Despite these challenges, the Rovio platform remains true to the challenges of implementing high-end sensing capabilities, such as SfM on a low-cost platform within a domestic or explorer robotic context. We address each in turn.

[1] The JPEG compression of the images treated in this work follow the JPEG standards, not to be mistaken for JPEG-LS or JPEG 2000

Figure 3.5: Mosquito effect in an image from the platform. On the zoomed in left-side photo the mosquito effect is noticeable around the edge of the chair and boxes.

3.1.1 Noise

Noise has been one of the main issues in this work. Since the input data for the system are the images streamed by the mobile platform, and more specifically the image features detected (Section 3.4), all the methods of the SfM process are exposed to its effects. It is important therefore to understand the origin and nature of noise that we have addressed.

The first cause of noise, apart from the quality of the CMOS sensor used by the camera of the Rovio, is the way images are delivered to our system. As it is described in Section 3.1 our system receives the images from the platform via wireless streaming. This configuration has many advantages when operating with omnidirectional mobile platforms, but it comes with an important shortcoming: the images are JPEG-compressed due to the lack of bandwidth provided by the wireless connection.

JPEG is a commonly used method of lossy compression for digital images. In essence, the compression algorithm transforms the image into a frequency domain by means of the discrete cosine transform, and then quantizates it. The quantization of the image produces loss of information and thus it may introduce severe artifacts such as block boundaries effects near contrasting edges (especially curves and corners). Over a sequence of images these types of artifacts are referred to as *mosquito noise* as the resulting spurious dots, which change over time, resemble mosquitoes swarming around the object (see Fig. 3.5). Mosquito noise has been the main source of noise in our images, being the level of estimated noise $\sigma \sim 10$; that is, given a pixel with colour value x,

the JPEG-compression colour value of the same pixel is given by a gaussian centred in x and with $\sigma \sim 10$.[†] As a comparison, we have tested a cheap webcam (model 10X SUPP), which can be found in any domestic workstation. The level of noise produced by this webcam is of $\sigma = 4.3$.[‡] It is therefore clear that noise impacts significantly on this work.

The first step to reduce noise in images is to reduce the artifacts created by JPEG compression with image filtering. In Section 3.3 the method chosen to alleviate the effect of noise is introduced.

Even though the effect of noise is present throughout the whole process, its impact is most noticeable at the matching step, where image features from different frames are paired. We have reduced the effect of noise by removing noisy matches during the matching process by using restrictive filters. As a result, sparse matching populations are generated. In addition, the low signal-to-noise ratio typical of narrow baselines configurations is pronounced by the presence of noise. Due to this, the retrieval of the epipolar geometry becomes a challenge. These problems have led our work towards the development of a novel tracking system, described in Section 3.6.

3.1.2 Small Baselines

A small relative motion between two images is a double-edged sword. On the one hand, the big overlapping between the two images facilitates large populations of feature matches. This seems convenient since it will be seen that our system suffers from a shortage of matching features. On the other hand, small distance configurations between images leads to a very poorly conditioned SfM problem. In sequences of images with shortage of features, a trade-off in the baselines should be found between the number of tracking feature points to detect and a suitable inter-image distance to ensure a stable relative motion estimation.

We have addressed the problem of small baselines by implementing a robust algorithm for epipolar geometry estimation (Section 3.5), capable of dealing with ill-configurations,

[†] A given pixel may take a value $x \in [0, 255]$ in each of the three channels R, G, B.

[‡] Noise estimation is a research field on its own. We have estimated the noise by taking large series of pictures of a flat, single-coloured surface in disperse light conditions and measuring the variation in colour of a region of the image. This variation has been computed by averaging the histograms of pixel values across the set of images. Other procedures for noise estimation can be found in Liu et al. (2006).

and developing a novel feature tracking system (Section 3.6) which optimises the noise-free features found in the images.
Now that the main challenges of the mobile platform chosen for this work have been introduced, the steps to estimate the 3D structure and motion from this platform under these conditions are described in detail.

3.2 Camera Calibration

Following the guidelines given in Appendix A, the first step to consider when performing 3D vision is the calibration of the camera.
Camera calibration is the process of estimating the parameters of a given camera model. In this case we have parametrised the pin-hole camera model (see Section A.2) by applying image processing methods to images taken by the camera in study. A full camera calibration entails the estimation of the optics of the camera and its pose (orientation and translation) with respect to a fixed coordinated frame. Here only the inner characteristics of the image sensor are tackled, as the estimation of the camera pose is one of the outputs of SfM method and will be studied in depth in Section 3.5.
The projection of rays of light on the image plane is performed through a physical device with a given configuration which should be modeled. In addition, the transducers employed by the sensor image to capture the light distort the light itself. It is therefore necessary, in order to achieve a faithful projection of the world on the image plane, to correct the distortions introduced by the optics and characterise the configuration of the sensor (Hartley and Zisserman (2004)).
The two main optical aberrations which distort the image are radial distortion and tangential distortion. Radial distortion is caused by spherical lenses, much cheaper to manufacture than the ideal parabolic lenses. On pixels away from the centre of the image the non-parabolic behaviour of a spherical lens is more noticeable and creates the barrel effect. Tangential distortions arise from the assembly process of the camera as a whole. Manufacturing defects result in the tangential plane of the lens not being exactly parallel to the imaging plane. Both radial and tangential distortions can be corrected by mapping the pixel grid according to Taylor series (Brown (1971); Fryer and Brown (1986)). Fortunately, the platform used in our experiments has integrated

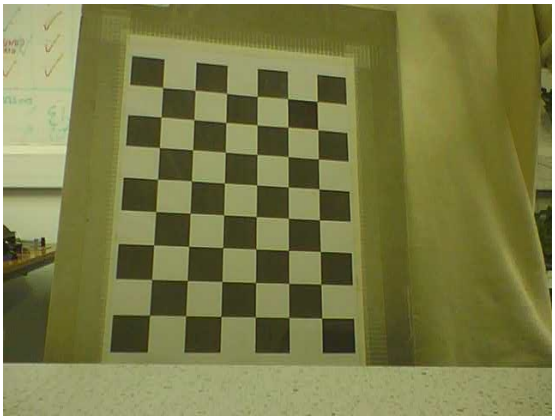


Figure 3.6: Checkerboad "Tsai grid" used for camera calibration.

firmware which corrects the distortions created by the image sensor[2]. Therefore the images received by our system are distortion-free.

The configuration of the camera in our case can thus be simplified to retrieving the distance between the camera centre and the image plane (focal length) and the coordinate frame within which pixels are referenced (the image centre, i.e. the intersection of the optical axis with the image plane, see Fig A.2). The parameters that contain this information are called intrinsic parameters. The intrinsic parameters are encoded by the intrinsic matrix K, whose derivation is explained in Section A.2.

In the literature many techniques are present which aim to calibrate the intrinsic parameters of a camera. It is possible, if the plane at infinity has been identified, to perform auto-calibration over a sequence of an unknown scene of more than 3 images, by extracting the Image of the Absolute Conic (IAC) (Hartley and Zisserman (2004)) which is fixed under similarity transformation. However, although various methods have been proposed for this, it remains quite a difficult problem, since the identification of the plane at infinity itself is a complex task. Faugeras et al. (1992) developed an algorithm for self-calibration which makes use of Kruppa equations in order to link the epipolar transformation to the IAC. Armstrong et al. (1996) propose a self-calibration method for the planar case, which takes advantage of the fixed identities under projective transformations. A Bayesian approach for self-calibration is presented by Qian (2004).

[2]The image sensor integrated is an OmniVision OV7670/OV7171 CMOS

Nevertheless, a more practical approach prevails for sequences of images taken by one camera with constant intrinsic configuration (mainly not variable focal length). This approach implies the use of a calibration object of known geometry over which a calibration algorithm is deployed. Fig. 3.6 shows this calibration object, which follows a checkerboard pattern ("Tsai grid"), designed to obtain the corners positions of the imaged squares with high accuracy. The results provided by such methods have been proven to be accurate and reliable, and since it is only necessary to carry out the calibration process once in a camera lifetime (provided its focal length is constant), this is the most used approach for intrinsic camera calibration.
The first algorithms for calibration rigs were based on the *Direct Linear Transformation* (DLT) method (Abdel-Aziz and Karara (1971)). DLT is a non-iterative algebraic method for solving homogeneous linear systems[3]. We have followed Zhang's technique (Zhang (2000)) to calibrate our system. Out of a sequence of images arbitrarily taken of the calibration rig, Zhang implements a closed-form solution for the homography between the corners of the squares in the Tsai grid plane and their corresponding views in the image plane (see Fig. 3.6). This solution is further refined with a Maximum Likelihood Estimation (MLE) of the views of corners of the chessboard plane over the sequence.
In this work a given sequence of images is always taken by the same camera with fixed optics configuration. Therefore all the methods employed in the SfM pipeline work in normalised coordinates (see Section A.2.2). By embedding the characteristics of the camera in the image coordinates we achieve three goals:

- The methods here exposed are valid for any type of camera.
- Many matrix calculations are simplified since the intrinsic matrix K has already been accounted for in the normalised coordinates.
- The epipolar geometry is now defined by the essential matrix E, which has two less degrees of freedom than the fundamental matrix F and therefore its estimation is less prone to diverge.

The calibration of the camera enables the Euclidean reconstruction of the structure. In the following sections the methods that form the system that obtains 3D reconstruction

[3]In Appendix A we have used DLT algorithm for solving different steps of SfM in ideal conditions, where the procedure of this algorithm has been explained in detail.

and camera motion information out of images released by the platform are described. The first measure to undertake is the alleviation of the noise. This is performed as soon as an image is received by the system, and we have named it as the stage of preprocessing.

3.3 Image Reception and Preprocessing

Section 2.6.3 classifies the types of SfM in batch SfM and sequential (also called incremental) SfM. This work has implemented sequential SfM, that is, the images are processed as they are received by the system and are not reordered within the sequence. Therefore we consider, at any given time, the most recent image received I_n of the overall sequence.

The first task to tackle when an image I_n is received is to decrease the level of noise while maintaining the texture of the objects in the image. An efficient inexpensive method to perform feature preserving noise reduction on each image received is bilateral filtering (Tomasi and Manduchi (1998)). Others filters based on blur effect were considered before bilateral filtering, but the good results given by the latter convinced us of its use. Other filters more sophisticated can be tested (anisotropic diffussion, non-local means, wavelet transforms, etc) to optimise the refinement of the input images. method smooths images while preserving edges, by means of non-linear combination of nearby image values. It takes into account geometric closeness and photometric similarity, and it employs a weighting system in both colour range and distance domains, with no phantom colours produced.

There are 3 parameters that can be set in the bilateral filtering method: the diameter around each pixel that is used during filtering (d), the range in the colour space in which two spatially close pixels will be smoothed into a semi-equal colour (σ_c), and the area of influence between two pixels with a similar colour (σ_s). These parameters should be tuned, as a great area of influence of the filtering method will flatten the surfaces, eliminating noise, but also preventing subsequent methods from detecting feature points. Fig. 3.7 shows the effect of bilateral filtering with the settings used in this system.

To quantify the effect on the level of noise of bilateral filtering is not straightforward. The measurement of noise according to the method described in Section 3.1.1 reveals a decrease of the level of noise of less than 10%, with the settings of the working system.

Figure 3.7: Bilateral filtering. The image has been filtered with settings $d = 3$, $\sigma_c = 50$ and $\sigma_s = 50$. Compare with Fig. 3.5 to appreciate the results.

This is because the effect of bilateral filtering is mainly noticeable in images rich in features (different colours and edges). An indirect way to evaluate the effect of bilateral filtering is to see how it affects the number of feature points and matches detected during the matching step[4].

Fig. 3.8 shows the number of SURF features in a raw image and in a filtered image. Even though more features are detected in the raw images (blue points), the quality of the features in the filtered image is higher and therefore more features remain after the feature selection taken in the matching step (red points). In addition, in the filtered case the remaining features have better quality, so they are better tracked, giving as a result a significant difference in the numbers of the point cloud with respect the non-filtered images, as it appears in Table 3.1, where the overall results over the *visionlab* sequence, in terms of projections and 3D points, are shown. A more greedy bilateral filtering would suppose a further decrement on number and quality of features detected. Depending on the type of sequence (how featured the scene is, type of path taken by the platform), the reconstruction of the scene may fail if bilateral filtering is not applied. The optimal settings of the parameters of the bilateral filtering method are dependent on the feature detector used.

The preprocessing of the images is a first measure taken to tackle the problem of noise. However, more filters, which will trim the still noisy features, are necessary to be de-

[4]All the evaluations performed in this chapter are done over a sequence of 55 images where the platform transits through a vision laboratory in approximately straight direction (*visionlab* sequence).

Figure 3.8: Effect of bilateral filtering over feature detection. The blue points are the initial feature points detected and the red points are the feature points that pass the filters in the matching stage. The feature detector used here is SURF (Bay et al. (2006)). 4927 feature points are initially detected in the raw image (left), whereas only 4328 are detected in the filtered image (right). However, only 354 go through the matching selection process in the raw image and 382 in the filtered image.

	With bilateral filtering	No bilateral filtering
overall projections	100777	41057
overall 3D points	24393	13654

Table 3.1: Effects on the results of applying bilateral filtering. The final projections and 3D points for raw images and filtered images are shown. (SURF features).

ployed in order to ensure a successful application of SfM on this work. These filters are applied during the stage of feature detection and matching.

3.4 Matching Process

This section explains how the feature matching is performed within our framework. Firstly the algorithm followed to match a received image I_n with the rest of the sequence is described. Subsequently we explain the matching filters deployed to overcome the problem of noisy features. Finally the actual feature detectors and matchers considered for this work are introduced and evaluated.

3.4.1 Recursive Matching Process

Given the sequential nature of this system the matching process is performed pairwise between I_n and previous images I_{n-i} of the sequence, with i increasing until either the match population found in the pair $\{I_{n-i}, I_n\}$, $i = k$, is below a given threshold τ_m or the value i is greater than a parameter τ_l, both set empirically. We denote this recursive matching by the expression $\mathcal{D} = \{I_{n-i}, I_n\}_{i=1}^{i=k}, 1 \leq k \leq n$. Fig. 3.1 shows graphically how the recursive matching process works. Note that sequential SfM is, under same conditions, quicker than batch SfM because there is no need to look for an image with overlapping field of view - the assumption being that the nth image overlaps to the $(n-1)th$ image. This procedure assumes that the platform is too far from the places where the frames $I_j, j < n-k$ were taken and there is no loop-closure in a given sequence. The feature tracking system presented in Section 3.6 is designed to work optimally with this recursive algorithm for matching images.

In order for the recursive matching to work, the feature matching between frames should not be contaminated by noisy features. We have devised specific filters that rule out spurious feature correspondences and mismatches.

3.4.2 Spurious Matches Trimming

Three filters have been deployed to rule out noisy correspondences and mismatches, low-quality matches and multiple matching to a single feature. These filters select the matches according to their quality. We assess the quality of a match between two features a and b by the L2 difference of their descriptors, denoted by δ_{ab}.

The main characteristic of the filters applied over the features detected over others present in the literature (Hartley and Zisserman (2004)) is two-fold: a) their simplicity in computation, which makes them have little overhead, and b) their capability of actually trimming the matches populations of spurious features. Spurious features are a critical factor in this work given the high level of noise that is faced. Therefore noise is treated in this work in two phases: in the denoising of the raw image (see Section 3.3) and in the filtering of the matches populations found.

In the first filter only unique matches are considered. The uniqueness of a match is defined by the ratio δ_{ab}/δ_{ac}, where b is the closest matching feature to a, and c is the second closest matching feature, based on the L2 difference of the descriptors. This

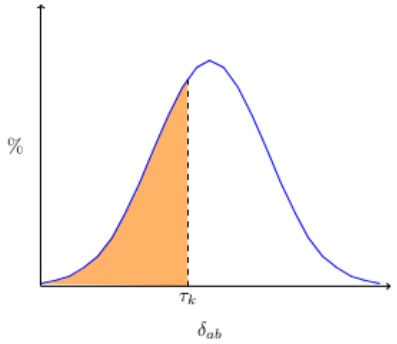


Figure 3.9: The histogram of the δ_{ab} distribution is typically a gaussian curve. The parameter τ_κ establishes the limit for selected matches.

filter was devised by Lowe (2004) and it has been proven to be very efficient in the identification of good matches. The set of matching features created for each pair $\{I_{n-i},\, I_n\}$ that pass this filter is denoted by $S_{(n-i)n}$.

The second filter selects the best matches of the set $S_{(n-i)n}$. This selection is accomplished by taking certain percentile rank, τ_κ of the score on δ_{ab} population over $S_{(n-i)n}$. We assume the noise to be Gaussian, so the histogram of the δ_{ab} population of matches of $S_{(n-i)n}$ will typically form a Gaussian curve, shown in Fig. 3.9. The orange area represents the percentile rank τ_κ. All matches of $S_{(n-i)n}$ whose distance δ_{ab} is greater than $\tau_\kappa \cdot \mu$ (where μ is the average value for the histogram) are discarded. The value τ_κ is set empirically.

To understand the dynamics of this filter the concept of error of correspondence should be introduced. Fig. 3.11 shows the projection of a 3D point $\hat{\boldsymbol{X}}$ to two images at $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{x}}'$. The corresponding image points $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{x}}'$ fulfill the epipolar geometry, as opposed to the measured points $\boldsymbol{x}$, $\boldsymbol{x}'$. The distances d, d' represent the effect of noise, which makes the feature detectors to deviate from the perfect match, due to pixel colour values variation. In absence of noise the measurements $\boldsymbol{x}$, $\boldsymbol{x}'$ would form a perfect match, but if the displacement created by noise is too large they become unsuitable for the epipolar geometry estimation and should therefore be ruled out. By selecting the percentile rank τ_κ of the score on δ_{ab} we establish a correlation between noise in feature space and physical displacement in image space.

Fig. 3.10 shows the trimming effect on the set of corresponding features of the filters applied over these sets. Many real matches between features are discarded due to their low quality.

Figure 3.10: Trimming effect of the noise filters on the set of corresponding features. The first row shows in blue 4396 features detected out of which 1069 matches were selected by Lowe's filter, drawn in green. These matches make up the set $S_{(n-i)n}$. The middle row shows in orange the set $S'_{(n-i)n}$, 446 matches selected by the *percentile* filter superimposed over Lowe's matches. The last row shows in red 340 inliers matches found by RANSAC, which compose the set $S''_{(n-i)n}$.

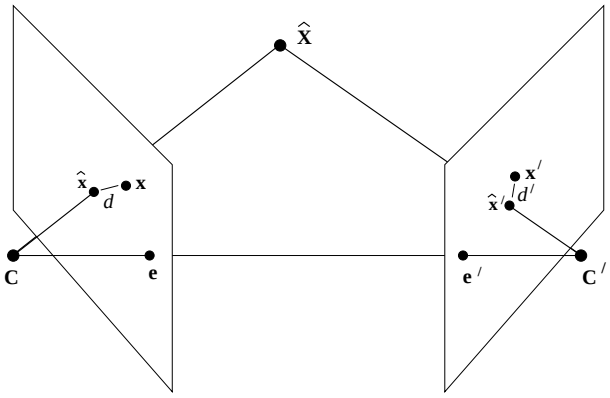


Figure 3.11: Error correspondence on the projection of a 3D point $\hat{\boldsymbol{X}}$. Source: Hartley and Zisserman (2004).

Finally, we enforce one-to-one feature matching between image pairs. This combination of filters counteracts the effect of noise on the feature matching process, but additionally results in a significantly sparse set of feature matches $S'_{(n-i)n}$ from which we then have to perform standard SfM for point-based structure.

3.4.3 Feature Detection and Matching

The SfM algorithm relies its success on finding homologous points across a sequence of images, as Section 2.3 explains. In Section 2.3 a review on the most relevant feature detectors and descriptors is performed. We have evaluated the most appropriate feature detectors for this work, namely SURF (Bay et al. (2006)), SIFT (Lowe (2004)) and ORB (Rublee et al. (2011)). The method developed by Shi and Tomasi (1994) provides good corners to track, but not descriptors for them; MSER is designed for the stereo case, and does not work well with motion blur. The detectors CenSure (Agrawal et al. (2008)) and FAST (Rosten and Drummond (2006)), even though they are quick and light detectors, generate key-points with weak descriptors, provided the noisy characteristics of the images. Optical flow has some restrictions such as its sensitivity to changes in light conditions and to noise. Moreover, it only performs well with small variations of the field of view. These feature detectors were down selected with the criteria given by Schmidt et al. (2013).

Table 3.2 shows a comparison of the detectors evaluated across the filters applied and RANSAC (described in Section 3.5) over the matches between I_n and I_{n-1}. SIFT is the most robust, but it comes at the price of computational time. SURF has been revealed

Descriptors	FLANN	Lowe	Percentile	RANSAC
SIFT	6829.87	1289.69	467.22	358.72
SURF	4303.46	950.78	411.46	306.57
ORB	3717.07	578.56	201.56	157.22

Table 3.2: Comparison between feature detectors and descriptors. Each column shows the number of matches that are present after each filter. The values are the average of matches between I_n and I_{n-1} over the *visionlab* sequence.

as the best balance between lightness and performance. The performance of ORB is much lower in terms of number of matches and quality, which results in half the quantity of surviving matches after RANSAC with respect to SURF. These differences become more apparent when the matches over the set $\left\{S_{(n-i)n}\right\}_{i=1}^{i=k}$ are taken in consideration. Amongst all the feature matchers introduced in Section 2.4.1 we have found to perform best the FLANN implementation of kd-randomised trees, in terms of performance and efficiency. Three different multidimensional query tree structures have been tested: a brute force, a hierarchical k-means tree and the chosen kd-randomised tree. Table 3.3 shows the performance in terms of matches for each implementation. Although the results are very similar, the kd-randomised tree structure gives slightly better results and takes shorter to compute. Table 3.4 shows the time spent by each FLANN implementation. Section 5.2.2 studies in detail the computation time spent in the matching process.

	FLANN	Lowe	Percentile	RANSAC
Randomised kd-trees	4303.46	950.78	411.46	306.57
Hierarchical k-means	4303.46	946.35	407.59	304.19
Brute force	4303.46	939.94	403.54	301.33

Table 3.3: Evaluation on the matching performance of the main multidimensional query tree structures from FLANN. The table shows average values from the *visionlab* sequence.

Computational time (ms)	
kd-randomised	185
hierarchical k-means	760
brute force	1197

Table 3.4: Computational times taken by each FLANN implementation on a given pair of images from sequence *visionlab*.

The outcome of the matching process is a set $S'_{(n-i)n}$ whose matches have been trimmed in order to get a matching set as clean of noise as possible. However, some mismatches may have passed all the filters. These mismatches, *outliers* in the population of $S'_{(n-i)n}$ are dealt with by RANSAC (see Section 2.4.2) during the matching process of the camera I_n with respect to the camera $I_{n-i}, i = 1 \cdots k$.

3.5 Relative Pose Estimation

RANSAC (Section 2.4.2) is the last method used to trim spurious matches and at the same time it gives an estimate of the essential matrix (although this estimate is not used). For every pair of images $\{I_{n-i}, I_n\}_{i=1}^{i=k}$ RANSAC is applied. Subsequent methods used for the relative pose estimation are extremely sensitive to outliers so it is paramount that RANSAC rules out any remaining outlier in $S'_{(n-i)n}$. The parametrising model to which RANSAC tries to fit the population of $S'_{(n-i)n}$ is the essential matrix. Table 3.2 shows that around 20% of the matches that arrive to this stage are ruled out by RANSAC. The metric used to evaluate how a match $\boldsymbol{x} \leftrightarrow \boldsymbol{x'}$ agrees with the model is the epipolar distance, that is the distance from $\boldsymbol{x}$ to the epipolar line generated by $\mathrm{E}\boldsymbol{x'}$ and vice versa (see Section A.4.1). The resulting matching set cropped by RANSAC becomes $S''_{(n-i)n}$.

As pointed out in Appendix A the estimation of the epipolar geometry is initialised by the 8-point algorithm (Hartley (1997)) applied over the set $S''_{(n-i)n}$. The essential matrix given by this method is still far from an acceptable result. The quality of a given normalised[5] essential matrix is evaluated by estimating the algebraic error that it produces, with the epipolar Eq. A.4.21[6].

It can be argued that the odometry of the wheels of the omnidirectional platform may serve as a support for the estimation of the epipolar geometry, and indeed an odometry system based on Ashmore and Barnes (2002) has been developed to be used as a bounding box of the initial estimation of the relative motion of I_n with respect to I_{n-1}. However, the wheel odometry proved unreliable and was taken out of the final system.

[5]Since the essential matrix is a homogeneous matrix, the values of its elements are up to scale. Therefore we normalise them by making their $L2$ norm unitary. This normalisation allows us to compare different essential matrices.

[6]Generally speaking, it is better to evaluate the geometric epipolar error (Ma et al. (2001); Sampson (1982)), but when working with calibrated cameras the algebraic error provides a good metric and it is faster to compute.

A typical algebraic error from the 8-point algorithm is in the range $[10^{-2}, 10^{0}]$ pixel per match. However, in order to extract the motion accurately it is necessary to obtain errors of order no greater than $\backsim 10^{-2}$. The minimisation of the epipolar error is attained by the implementation of the algebraic minimisation algorithm (Hartley and Zisserman (2004)).

The algebraic minimisation algorithm

The 8-point algorithm needs to convert the estimate E' into a singular matrix in order to enforce the singularity constraint, by using SVD (see Section A.4.4). Numerically however this procedure is suboptimal, since each element of E has different importance on the epipolar constraint. An alternative solution is to find the targeted singular matrix E directly and then project it onto the essential space $\mathbf{\Theta}$. This is done by the algebraic minimisation algorithm. This algorithm is only applied to the pair of images I_n and I_{n-1}, as shown in the flowchart of Fig. 3.1.

Let A be the coefficients matrix introduced in Section A.4.3. The 8-point algorithm finds a matrix $\bar{\mathrm{E}}'$ subject to $\|\bar{\mathrm{E}}'\| = 1$ which minimises $\|A\bar{\mathrm{E}}'\|$ ($\bar{\mathrm{E}}'$ is the stacked column vector of E'). We are now interested in a *singular* matrix E subject to $\|\bar{\mathrm{E}}\| = 1$ which minimises $\|\mathrm{A}\bar{\mathrm{E}}\|$. This is not possible to achieve by linear methods, since the constraint $\det(\mathrm{E}) = 0$ is cubic. Nevertheless this problem can be solved iteratively with a simple algorithm.

It is known that any singular 3×3 matrix can be expressed as $\mathrm{E} = \mathrm{M}[\boldsymbol{e}]_x$, with M being a non-singular matrix and $[\boldsymbol{e}]_x$ the skew-symmetric matrix created from the epipole in the first image. Assuming that the epipole $\boldsymbol{e}$ is known, the expression $\mathrm{E} = \mathrm{M}[\boldsymbol{e}]_x$ can be converted into $\bar{\mathrm{E}} = \mathrm{G}\bar{\mathrm{M}}$ where $\bar{\mathrm{M}}$ contains the matrix M in row-major order, and the matrix G is as follows:-

$$\mathrm{G} = \begin{bmatrix} [\boldsymbol{e}]_x & & \\ & [\boldsymbol{e}]_x & \\ & & [\boldsymbol{e}]_x \end{bmatrix} \tag{3.1}$$

Since $\bar{\mathrm{E}} = \mathrm{G}\bar{\mathrm{M}}$, now the minimisation problem consists of finding a matrix $\mathrm{G}\bar{\mathrm{M}}$ subject to $\|\mathrm{G}\bar{\mathrm{M}}\| = 1$ which minimises $\|\mathrm{AG}\bar{\mathrm{M}}\|$. This is a constraint least-squares minimisation problem and can be solved by applying SVD (Hartley and Zisserman (2004)). Note that the solution to this problem is, by definition, a singular matrix.

This algorithm establishes a mapping $\boldsymbol{e} \mapsto \mathrm{A}\bar{\mathrm{E}}$, where $\mathrm{A}\bar{\mathrm{E}} = \epsilon$ is the algebraic error. Therefore, starting from the left epipole of the estimate E' given by the 8-point algorithm, we can iterate to find the final E that minimises the algebraic error. The optimisation of this step can be done by the Levenberg-Marquardt (L-M) method (see Section A.7.2).

To summarise, the algebraic minimisation method splits the optimisation into two parts: the first one finds the singular matrix E that minimises $\mathrm{A}\bar{\mathrm{G}}\bar{\mathrm{M}}$ given the epipole $\boldsymbol{e}$ and the second part iterates the value of $\boldsymbol{e}$ so as to minimise $\|\epsilon\|$. In our implementation the L-M converges in one or two iterations and the whole optimisation problem usually does not require more than two iterations. Note that here the L-M method only optimises three parameters (the coordinates of $\boldsymbol{e}$) but still the algebraic error for all the matches of the set $S''_{(n-i)n}$ is minimised.

The algebraic error given by the optimised matrix E falls in the range $[10^{-5}, 10^{-2}]$. The last steps in the estimation of the relative pose are the projection of E onto $\boldsymbol{\Theta}$ and the extraction of R and $\boldsymbol{t}$, as explained in Sections A.4.4 and A.4.5. The matrices R and $\boldsymbol{t}$ will be further refined in subsequent methods.

The algebraic minimisation algorithm ensures a robust estimation of the relative camera pose of I_n with respect to I_{n-1}. This estimate is done within the recursive matching process which generates the sets $\left\{ S''_{(n-i)n} \right\}_{i=1}^{i=k}, 1 \leq k \leq n$ (see Fig. 3.1). These sets of matching features are now examined and added to the feature tracking system.

3.6 Feature Tracking System

A key problem implicit in all SfM approaches is the feature registration problem, where multiple pair-wise feature correspondences must be merged into a single multiple-view feature tracking, or *bundle* of features for a given 3D point $\boldsymbol{X}$. This situation is shown in Fig. A.7. In Section 2.5 it is emphasized how many works struggle with occlusions and noisy features, and are unable to form feature tracks with non-consecutive matches. The main problem that both batch and sequential methods have is that, when reconstructing long sequences of images, the reconstruction is based on tracking features along consecutive images. Due to noise, occlusions etc. the method can lose track of a feature and if later the same feature is found, the algorithm takes it as a different one, giving as a result that the same feature provides two different 3D points. This is

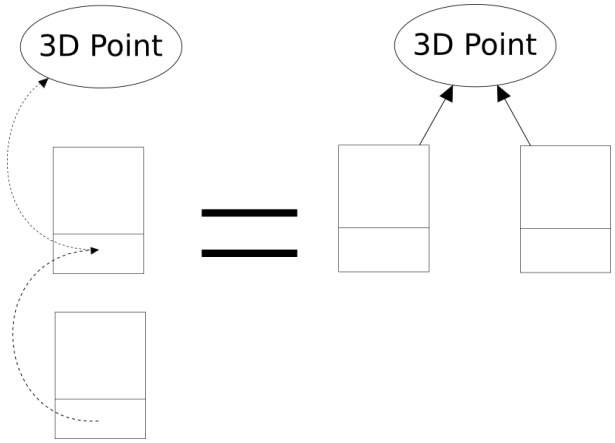


Figure 3.12: Merging of two features in a *bundle*. When a feature is matched to another feature which is already associated to a 3D point, the structure *bundle* automatically links it to this 3D point.

a penalising factor on the drift problem. Zhang et al. (2010) deal with the problem of non consecutive matches along long sequences, with excellent results.

Here, in this unexplored case, where noisy matches have been efficiently trimmed previously, the scarcity of surviving feature matches is managed by a novel feature filtering tracking method. This implementation is achieved by an efficient management of the *bundles* created. A *bundle* can be defined as a structure which links a given 3D point $\boldsymbol{X}$ with its multiple views in several images. In addition, uncertainty is addressed by conservative thresholds. Another crucial role that our feature tracking system accomplishes is to obtain feature tracks of enough length and quality between I_n and previous images so that it is possible to estimate the global pose of I_n with respect to a common frame of reference (see Section A.5).

The rationale behind this feature tracking system is to offer an efficient tracking system when doing recursive matching. If tracks were generated by simply pairwise matching, many potential tracks would be discontinued and therefore lost. Even with recursive matching all the matches generated should be conveniently managed in order to exploit the correspondences.

The input of this algorithm is the sets $\left\{S''_{(n-i)n}\right\}_{i=1}^{i=k}, 1 \leq k \leq n$. Each match from $S''_{(n-i)n}$ is added to the population which will generate the 3D structure, and then it is linked to a 3D point according to certain filters and criteria exposed in this section.

We have devised this novel feature tracking system with three goals in mind:

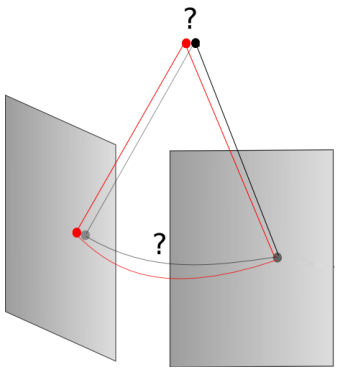


Figure 3.13: The bundles red and black have corresponding features which have been matched (red feature on the left image with black feature on the right image). Filter f_1 checks first whether the black bundle has a feature too on the left image, and in case affirmative, whether this feature is actually the same one as the red feature. f_2 checks whether the 3D point of both bundles are close enough to be considered the same one.

- It should be efficient, that is, it should maximise the length of good quality feature tracks.
- It should be robust, that is, any mismatched track must be avoided.
- It should be dynamic, that is, it should automatically perform 3D merging between existing *bundles*.

Bundles

Three main computational operations should be enabled when efficiently tracking matches over a sequence:-

1. Direct access to $\boldsymbol{X}$ referenced from any feature in its bundle and vice versa,
2. Addition of new features to a track and
3. Merging of existent *bundles*.

In our tracking method we novelly devise bundles as structures similar to linked lists, inspired in Brzeszcz and Breckon (2010) which allows us to efficiently perform these tasks. The nodes of this type of list are linked not to the next node of the list but to the "head" of the list, which contains information about the 3D point. Therefore, when a new feature is added to the bundle of $\boldsymbol{X}$, this specific implementation of bundle will automatically link it to $\boldsymbol{X}$ and through $\boldsymbol{X}$ to the rest of features of the bundle. Fig. 3.12 shows a diagram as to how the merging of two features is done.

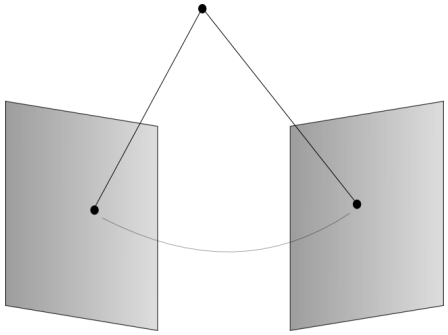


Figure 3.14: Case 1: two bundle-less features are matched. In this case a bundle is created for them.

Feature tracking system filters

Given the sparsity of the 3D point cloud produced by our matching filters it is necessary to properly manage the addition of features to a bundle and the merging between bundles, in order to create sufficient duration feature tracks. This is handled by two filter checks.

The first filter f_1 checks whether two bundles have both features in common images. If this is the case, then a further check is done to verify that the features from both bundles are actually the same. If the filter f_1 is not passed, the features involved are not added to the structure.

For example, if a feature m_a from image I_a is matched with a feature m_b from image I_b, f_1 checks whether the bundle associated to m_b has already a feature from image I_a. The analogue check is done with the bundle associated to m_a. When this is the case it compares the values of the coordinates of the features involved to establish whether they are truly the same feature. If they happen to be different, the bundles to which m_a and m_b are removed from the structure population according to the cases generated by Table 3.5. Otherwise their bundles are merged according to Table 3.5.

This filter ensures that a bundle is linked to one feature per image. Fig. 3.13 shows graphically the mechanism of this filter.

The second filter f_2 compares whether two 3D points p_i and p_j are close enough to be considered the same 3D point. For each axis $\imath \in \{x, y, z\}$ we define $\delta^\imath = \|p_i^\imath - p_j^\imath\|$, and $\mu^\imath = mean\left\{p_i^\imath, p_j^\imath\right\}$. The filter f_2 checks that $\delta^\imath < k \cdot \mu^\imath$. If this is the case they are assumed to be the same 3D point. The value of k is set empirically. This is illustrated in Fig. 3.13 as well.

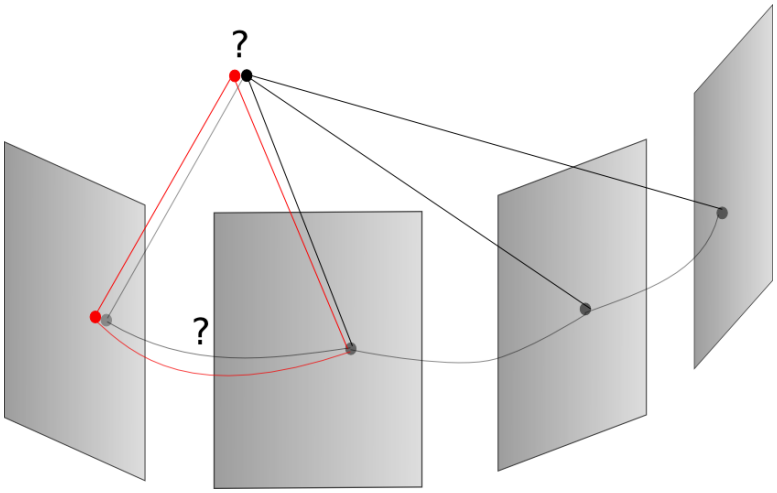


Figure 3.15: Case 2: Addition of a feature to a bundle through a match.

Since the feature tracking system is run within the recursive matching process, the N possible cases that arise when the set ${S''}_{(n-i)n}$ is analysed are multiple. Any feature in a match can be bundle-less (or empty), linked to a bundle which has not been reconstructed yet (so it has no 3D point) or linked to a bundle of a reconstructed 3D point. Therefore six possible cases arise, represented in Table 3.5.

Cases	Feature from image I_{n-i}			Feature from image I_n		
	Empty	No 3D	Yes 3D	Empty	No 3D	Yes 3D
1	X			X		
2	X				X	
3	X					X
4		X			X	
5		X				X
6			X			X

Table 3.5: Possible situations between features when processed by the feature tracking system. "Empty" means that the feature has not been added to any bundle yet. "No 3D" means that the feature belongs to a bundle which has not been reconstructed yet. "Yes 3D" means that the feature is linked to a reconstructed bundle.

The actions taken in every case are:

Case 1: A brand new 3D point $\{0, 0, 0\}$ (whose actual value will be estimated in the triangulation step, Section 3.7) and its bundle are created and added to the structure

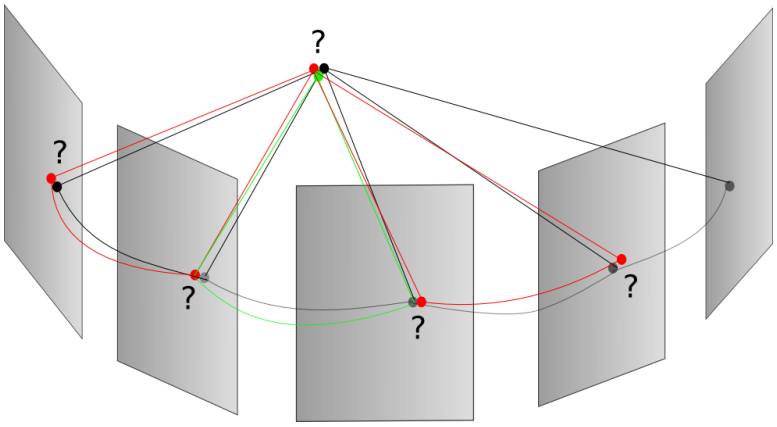


Figure 3.16: Case 3: Merging of two existent bundles (red and black) via a match (shown in green).

population. At this point the bundle is composed of the two matching features. In Fig. 3.14 this case is shown.

Cases 2-5: Here filter f_1 is applied. In case of success the bundle-less feature (in cases **2** and **3**), or the 3D point-less feature (in cases **4** and **5**) is added to the another bundle. Otherwise, the new feature is discarded. These cases are shown in Fig. 3.15.

Cases 6: Apart from applying filter f_1, additionally, the filter f_2 is applied. If the pair of bundles passes this last filter, they are merged into one bundle. The resulting 3D point is estimated as a weighted average between the two 3D points of the original bundles. The weights are the number of times that each bundle has undergone the triangulation step. In other words, the weighting factors when merging two bundles are the number of views that a bundle has minus one, since each time a view is added to a given bundle, its corresponding 3D point is estimated again by triangulation. Therefore, this weighting sum assumes that the position of a 3D point will be more fixed and likely the longer that its bundle is. Fig. 3.16 illustrates this case.

The matches which fail to pass filter f_1 are taken out of the set of matches $S''_{(n-i)n}$. As a result, the feature tracking system also rules out matches which would potentially destabilise the reconstruction of the 3D structure and the refinement of the cameras. However, those matches that do not pass filter f_2 are not discarded, since we assume

that these cases are mismatches between features which belong to healthy bundles. The set of matching features which come out of the feature tracking system becomes now ${S'''}_{(n-i)n}$.

The specific creation and management of the structure of bundle, along with the filters associated to it, allows us to obtain precise camera poses and a reliable point cloud out of sparse matches populations (in our experiments, at this stage an average image has 755 views of 3D points, with 3.56 projections per 3D point). Fig. 3.17 shows the histogram of feature tracks of the *visionlab* sequence, at this stage (blue line) and at the stage of post-process (red line, see Section 3.9). Note that nearly 60% of the feature tracks have no more than 4 features.

Through specific filters and a novel noise resilient feature tracking method, we have developed a novel feature tracking system which handles the inter-bundle relationships via robust and light filters. Over the structure population increased with the matches from the sets $\left\{{S'''}_{(n-i)n}\right\}_{i=1}^{i=k}$, now it is possible to place the camera poses with respect to the global reference frame, and update the 3D structure.

3.7 Global Pose Estimation and Structure

This is the last step of the SfM process as such. Here the global pose estimation of the camera I_n is estimated and the 3D structure updated with the incorporation of the set of bundles from $\left\{{S'''}_{(n-i)n}\right\}_{i=1}^{i=k}$.

3.7.1 Resection

The introduction of the sets $\left\{{S'''}_{(n-i)n}\right\}_{i=1}^{i=k}$ increases the structure population and widens the range of the bundles. With this new information the scale of the camera pose of $I_n, n \geq 3$, is adjusted to be coherent with the rest of the sequence. This refinement is performed with the resection method proposed in Lepetit et al. (2009). As it has been explained in Section 2.6.5, this non-iterative solution has a computational complexity which grows linearly with n.

Here, as in recovering the 3D structure, the use of bundles ease the implementation greatly. The resection of the camera I_n is done if there are enough features from the set ${S'''}_{(n-1)n}$ which have been reconstructed (see Section A.5). Since at this stage the set ${S'''}_{(n-1)n}$ has not been reconstructed yet (the reconstruction is the next step of

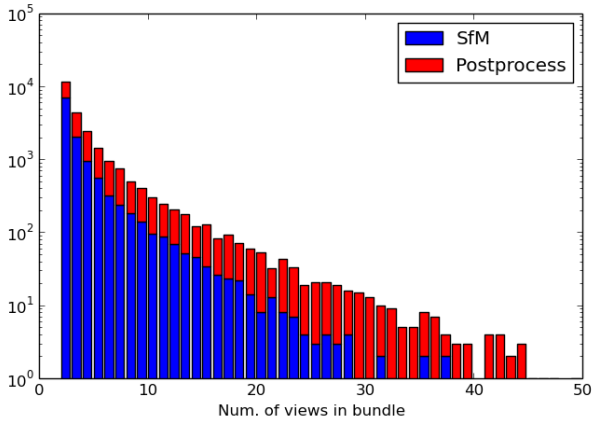


Figure 3.17: Histogram of feature track lengths over the *visionlab* sequence. The *Post-process* step is explained in Section 3.9.

the SfM pipeline), it will only be possible to apply resection if we find a subset of features $q_{(n-1)n} \in S'''_{(n-1)n}$ that in turn belong to the set $S'''_{(n-2)(n-1)}$. Since the set $S'''_{(n-2)(n-1)}$ was reconstructed when the image I_{n-1} was received by the system, it yields that the subset $q_{(n-1)n}$ is made up of reconstructed features (see Section A.5). This subset is easily found by searching in $S'''_{(n-1)n}$ for features linked to reconstructed bundles. In the *visionlab* sequence the set $q_{(n-1)n}$ contains in average 335.06 views. It has been observed empirically that the minimum size of $q_{(n-1)n}$ for the resection method to give accurate results is of 20 elements.

In the case that there are less than 3 features in the subset $q_{(n-1)n}$, (shortage of matches can be caused by a feature-less part of the scene, a sharp turn of the robot which changes abruptly the field of view, etc.) it is assumed that the module of $\boldsymbol{t_n}$ is the same as the module of $\boldsymbol{t_{n-1}}$. This will be corrected at the BA phase. It should be noted that this situation is rare to occur in our experiments.

With the camera I_n located in the same coordinate frame as the rest of the sequence, now it is possible to apply triangulation to the population of 3D points (or bundles).

3.7.2 3D Structure

Once the global camera poses have been calculated the triangulation process over the updated point cloud takes place, where the new 3D points are estimated and those whose bundles have been increased are recomputed.

The triangulation method followed here is the same as described in Section A.6, that is, a variant of the *Direct Linear Transformation* (DLT) method (Abdel-Aziz and Karara (1971)), called Linear-Eigen method (see Section 2.7). Linear-Eigen is applied as an initial estimate that will be refined during BA. In addition to the advantages mentioned in Section 2.7, this method generalises very easily if the 3D point is seen by more than two views. This property conveniently suits our approach, since we then can make use of the structure of bundles to fill in the matrix A of Eq. A.6.51 very rapidly, by just looking up the views linked to a given bundle.

3.8 Bundle Adjustment

The last stage of the 3D reconstruction involves the application of Bundle Adjustment (BA), where camera poses and 3D points are simultaneously optimised by minimising the reprojection error function cost. This work runs the implementation of Lourakis and Argyros (2009) (SBA) which efficiently applies L-M minimization method (see Section A.7.2) by exploiting the sparseness of the SfM problem. In Section A.7.2 it is explained how the damped version of the normal equations (these are the equations that result from minimising the cost function A.7.54) gives to L-M method its stability and rapidness. The normal equations, however, are greatly sparse, due to the lack of interaction between parameters for different 3D points and cameras. SBA designs and develops a customised variant of the L-M algorithm which takes into account the pattern of zeroes in the normal equations by not making operations on zero elements. Here again the implementation of bundles as structures to store the 3D points and their views result of great advantage. Amongst other structures, the array v_{ij} from Eq. A.7.54 is filled in very quickly thanks to the internal design of bundles.

3.8.1 Quaternions

In Section 2.8 it is emphasized the importance of the parametrisation of camera rotation matrices. SBA employs quaternions. In mathematics, the quaternions are a number system that extends the complex numbers. They can be represented as the sum of a scalar and a vector, and it is mainly this feature that makes quaternions commonly used in geometry to express rotations and changes of reference, as a quaternion can confine the same information as a rotation matrix with 4 values instead of 9, the number of

elements of the rotation matrix. Their use saves computation time and memory, as well as the need to handle singular points and angles, as is required when using Euler angles.

3.8.2 Local and Global BA

We employ BA in two scopes, locally and globally, as Mouragnon et al. (2006*b*) and Engels et al. (2006) propose. The local BA is conducted within the process pipeline, as a last refining step on the new camera I_n and 3D points . The global BA is executed in a different thread to the sequential pipeline over the whole point cloud and the last n cameras poses (see Fig. 3.1). The parameter n is set empirically. Every m images processed by the system the global BA considers all the cameras of the sequence in the optimisation. The consequences of the interaction between the main thread and the global BA thread is shown in Section 5.2.2, when evaluating computation times taken by each stage of the SfM process.

When the first two cameras of the sequence are optimised a cheirality test (Hartley and Zisserman (2004)) is done over the 3D point cloud and the camera poses, to check that no point is behind its camera. This test has already been done during the extraction of the matrices R and $\boldsymbol{t}$ (see Section A.4.5), but if only two cameras are reconstructed the BA method may have inverted the orientation of the camera poses with respect to the 3D point cloud, since at this stage there are not enough restrictions to force the 3D points to be in front of the cameras.

After both pair-wise BA and global BA, the 3D structure undergoes a filter based on reprojection error (see Section A.7.1). Each 3D point is reprojected on the camera plane of its views; if the distance from the projection to the measurement of the view is larger than a threshold τ_r, the 3D point is taken out of the reconstructed point cloud. τ_r is set empirically. This filter guarantees that the 3D structure is clean of outliers, which is essential in order to refine the camera poses.

After all the images of the sequence have been processed, BA is applied over the whole point cloud and camera poses as one last refinement.

3.9 Final Scene Recovery

The combination of limited camera resolution, image noise and small baselines inherent within the use of an omnidirectional mobile platform forces our core SfM method to be

highly selective over the quality of matches. This produces a sparse scene reconstruction resulting in a sparse 3D point cloud of scene surfaces compared to traditional SfM approaches (Mouragnon et al. (2006*b*)).

Descriptors	FLANN	Lowe	Percentile	RANSAC
SIFT	6705.69	2196.16	1945.96	1555.18
SURF	4225.22	1637.35	1348.65	998.73

Table 3.6: Surviving matches after each filter during the post-process stage

In order to provide a dense surface reconstruction (e.g. as shown in Fig. 3.19) a variant of the SfM pipeline is run as a data post-process. This variant makes use of the estimated camera poses and the previously extracted features. Since the motion is fixed, there is no inherent risk in now including noisy matches, and we can relax the thresholds of the match quality filters. Particularly, τ_u is more benign and there is no selection over the score on δ_{ab}. In addition, the thresholds imposed on RANSAC are more relaxed as well. Furthermore, in this stage there is no pair-wise BA, and the global BA method only acts over the 3D structure, leaving the camera poses intact. The flowchart of the post-processing of an image I_n is shown in Fig. 3.18.

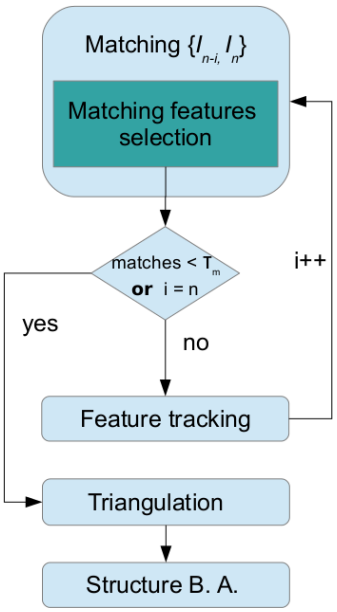


Figure 3.18: Flowchart of the post-processing of image I_n.

This arrangement produces a point cloud whose population is increased up to over 200% in terms of recovered 3D scene surface points (See Fig. 3.17). Note that 4,303 features are extracted by SURF from an average image, and the final point cloud has 1,675 views per image, which gives 38% of efficiency over the total features extracted per image. Table 3.6 shows the analogous results to Table 3.2 for SURF and SIFT descriptors in the post-process phase. In comparison with Table 3.2 the number of matches generated now is much larger. The values in Table 3.6 are the average of matches between I_n and I_{n-1} over the *visionlab* sequence. The increase on the number of matches that pass each filter is notable. Obviously, many more features and 3D points are eventually discarded by the feature tracking system (which maintains the level of filtering), but the final results are still as much as twice the density of 3D point population before the post-process stage. These results are shown in Table 3.7, where the feature detector used is SURF. The matches deleted by f_1 in Table 3.7 account for the trimming realised by the feature tracking method.

	% deleted by f_1	3D points deleted by reproj. error	Overall 3D points	Overall projections of 3D points
SfM process	15.49	3563	11982	40623
Post-process	33.06	15558	24393	100777

Table 3.7: Comparison between the results after SfM process and after post-processing the matches.

3.9.1 Surface Rendering

It is possible to render surface reconstructions out of the point cloud given by our SfM process. Here the 3D structure is cropped and filtered by applying two filters: a radius outlier removal method and a k-nearest neighbour distance filter. These statistical techniques are present in Rusu and Cousins (2011). In order to render the reconstruction, normals are estimated and smoothed based on Moving Least Squares (MLS) surface reconstruction method (Alexa et al. (2003)). Subsequently the structure is further triangulated and surfaces reconstructed using a Poisson method (Kazhdan et al. (2006)). An example of the results of these methods is shown in Fig. 3.19. A sample of the *visionlab* sequence is shown in Fig. 5.1.

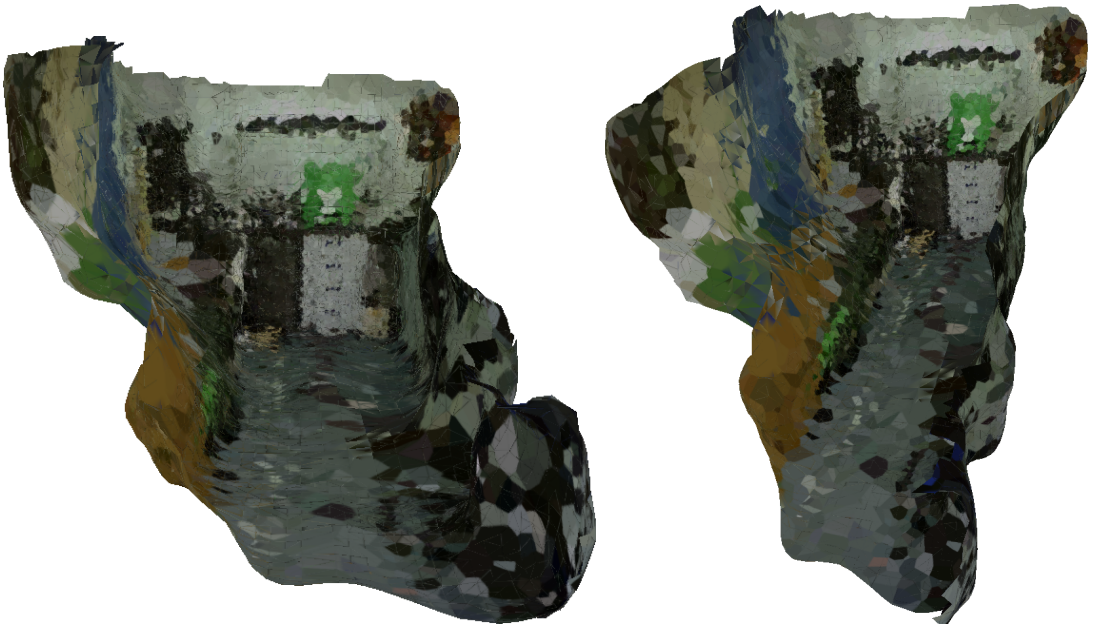


Figure 3.19: The 3D reconstruction of the *visionlab* sequence, rendered.

3.10 Summary

In this chapter the methods employed to realise SfM on a low-budget omnidirectional mobile platform have been explained. This is to the best of our knowledge the first time that SfM method is successfully applied on a omnidirectional platform.

The first measure taken has been to reduce the high level of noise produced by the JPEG compression that wireless streaming imposes over the images. This is done sequentially by applying bilateral filtering and selecting the corresponding points between images according to strict filters, which harshly trim of outliers the feature matching population. We mainly use Lowe's ratio (Lowe (2004)) and a novelly devised filter, *percentile* filter, which selects certain percentile rank τ_κ of the score on δ_{ab} over a set of corresponding features. This combination of filters allows us to treat levels of noise of $\sigma \sim 10$. The level of noise that researchers usually process is $\sigma \sim 2$ (Ruiz et al. (2006); Thomas and Oliensis (1999)). Moreover, our filtering methods enables the system to deal with inter-image ill-configurations provoked by the omnidirectional nature of the mobile platform chosen.

With feature matches free of outliers it is possible to retrieve the epipolar geometry, although it is still necessary to implement robust estimation, in order to be robust

against ill-configurations potentially created by the omnidirectional idiosyncrasy of the mobile platform. Therefore a robust epipolar geometry estimation algorithm has been implemented.

The shortage of matches originated at the filtering stage would result in too short feature tracks along the sequence, so a novel feature tracking method has been developed to make the most of the surviving matches. This method creates feature tracks of length and quality good enough to ensure the global pose estimation of the cameras and the stable estimate of the 3D structure. All this is achieved thanks to the use of *bundles*, specifically devised structures similar to dynamic lists which allow an efficient management of the 3D points and the views on which they are projected. Even though the majority of tracks (60%) are 3 or 4 images long, they are precise enough as to obtain stable 3D reconstructions. This result supports the hypothesis that out of sparse matching populations our feature tracking system effectively creates feature tracks of good quality, since it is usually assumed that feature tracks must be at least of 4 images long (Chang and Hebert (2002); Zhang et al. (2010)). We understand that this feature tracking system, in combination with the noise filters implemented, extends the state of the art on SfM on mobile platforms.

As a final step of the 3D reconstruction Bundle Adjustment (BA) is implemented, in two scopes: pair-wise BA when a given image is processed and global BA over the sequence in a different thread.

Once the camera poses have been accurately estimated, in a post-process phase the 3D structure is enlarged by running the matching process with much more relaxed filters. Optionally the point cloud can be treated in order to obtain a rendered surface.

The SfM process on a single mobile platform is the algorithmic core over which the 3D reconstruction on a group of mobile platforms is built up. Next chapter describes how this multiple reconstruction has been attained.

Chapter 4

Structure from Motion across Multiple Platforms

In this chapter the problem of distributed reconstruction over omnidirectional mobile platforms is tackled, and a system capable of retrieving the 3D global map of a scene transited by r robots is presented.

First the strategy employed in this work to solve this problem is introduced, emphasizing the main challenges usually encountered here by the researchers. The implementation of this strategy, based on distributed matching (more specifically, loop-closing) is subsequently described. Special highlight deserves the algorithm used for finding loop closures, FAB-MAP (Cummins and Newman (2011)).

The overlaps between different reconstructions found during the loop-closing stage should be conveniently managed, using organically the methods presented in Chapter 3 to deal with the problems inherent to the platform used (noise, ill-configurations, see Section 3.4). On a higher layer of computation, once an overlap is found, the merging of multiple reconstructions - performed as they are built up - into one global structure is explained, along with the particular distributed approach for the BA algorithm.

Every Multi Robot System (MRS) needs a defined strategy which will condition the behaviour or pattern in the motion of the mobile platforms as well as the interactions between themselves. The strategy of this work can be outlined with two terms: it is designed to be *simple* as well as *general*. In addition, the approach taken in our system can be applied at no price on single sequences where the platform revisits parts of a scene, generating loop-closures.

4.1 Multiple Structure from Motion: our Approach

There are mainly two problems that should be addressed when performing SfM from multiple platforms: the initial position problem and map merging.

The first problem ultimately consists of allocating a common reference frame for all the platforms involved, in order to establish the motion of each robot relative to the rest of robots and make possible to transform from one robot location to another, as highlighted by Breitenmoser et al. (2011). This problem means to overcome the constraint imposed over the first camera of being at origin (see Section A.5). This constraint is unimportant for a single SfM process but can only be applied to one instance in multiple SfM. The naïve approach is to know exactly where each robot starts moving, but this is unfeasible. Each individual should somehow share some information about its whereabouts - in a global or common reference - with the rest of the group. This is usually attained by direct encounters (Kato et al. (1999); Kurazume et al. (1994)) although loop-closing could be used for this purpose. Anjum (2011) makes use of Kalman filters in order to keep track of the global trajectory of each robot. The work of Anjum (2011) shows that the main challenge here is the adequate management of uncertainty, produced by the propagation of error that occurs when distributed estimates are calculated from multiple single measurements.

Fig. 4.1 shows a result by Kim et al. (2010), where collaborative simultaneous localisation and mapping (SLAM) is achieved by means of direct encounters. Based on incremental smoothing and mapping (iSAM[1]), a pose graph representation of the SLAM problem, Kim et al. (2010) obtains global 2D maps out of individual implementations of SLAM. Fig. 4.1 shows this result: superimposed on the layout map of the explored environment, the 2D maps generated by two different mobile platforms (a ground robot and a quadrotor) are drawn in blue and red, respectively. The green lines indicate the occurrence of direct encounters.

The second problem, map merging, is closely related to the first problem. Here a method should be devised to merge the maps from each individual robot. In Section 2.9 three techniques are mentioned: iterative closest point (ICP)[2], distributed feature matching

[1] iSAM, developed by Kaess et al. (2007), is an incremental smoothing and mapping approach for SLAM based on fast incremental matrix factorisation.

[2] ICP performs point cloud registration by iteratively revising the transformation necessary to adjust two different point clouds.

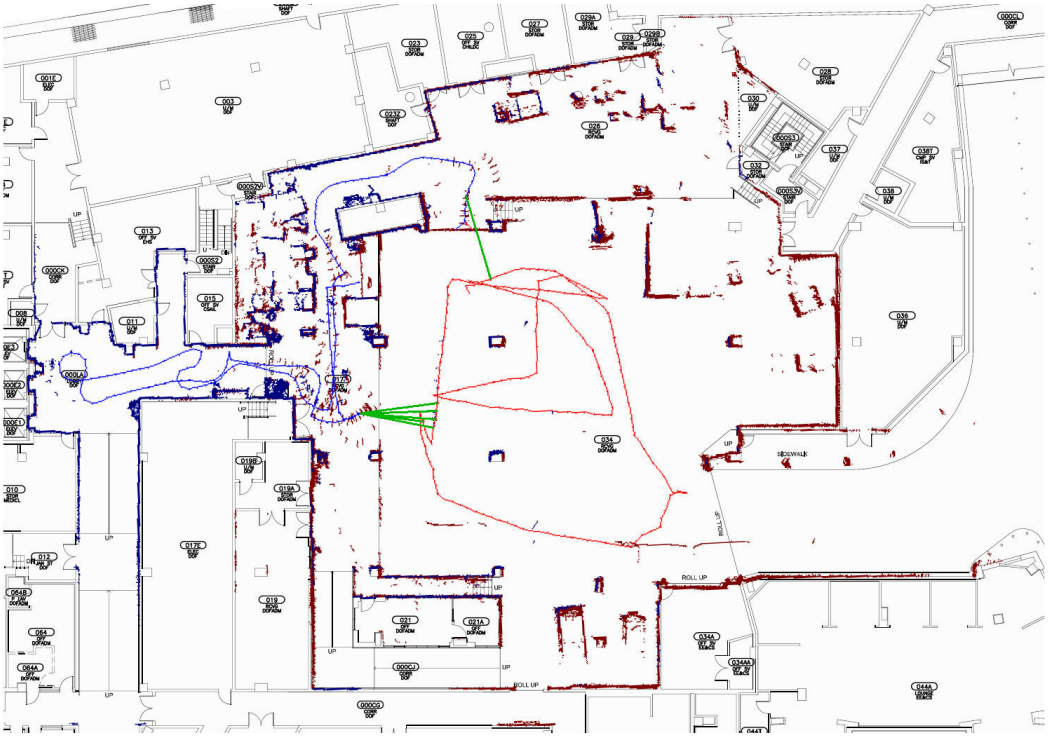


Figure 4.1: 2D mapping from direct encounters between a wheeled robot and a quadrotor. In blue the 2D map created by the wheeled robot, in red the 2D map created by the quadrotor. The green lines show where the encounters occurred. This result is from Kim et al. (2010).

and loop-closing. Here the main problem is to rescale all the individual 3D maps into one single scale. This only can be done if the camera poses are referenced in the same frame of coordinates. Further, collaborative reconstruction has challenges inherent to the multi-agent idiosyncrasy of the problem: uncertainty management, decision making, coordination, communication, motion planning, etc (Riazuelo et al. (2014); Wendel et al. (2012)).

The distributed system developed in this work offers a solution to the problems of the initial position and map merging, in a way which intends to be simple as well as general. The research work on collaborative mapping is mostly based on SLAM approaches (or related methods, such as iSAM) and direct encounters. Our algorithm achieves global camera positioning and map merging by finding loop-closures between sequences taken from different robots, or the same robot in different moments. Our algorithm is simple as it is based on common feature matches between reconstructions,

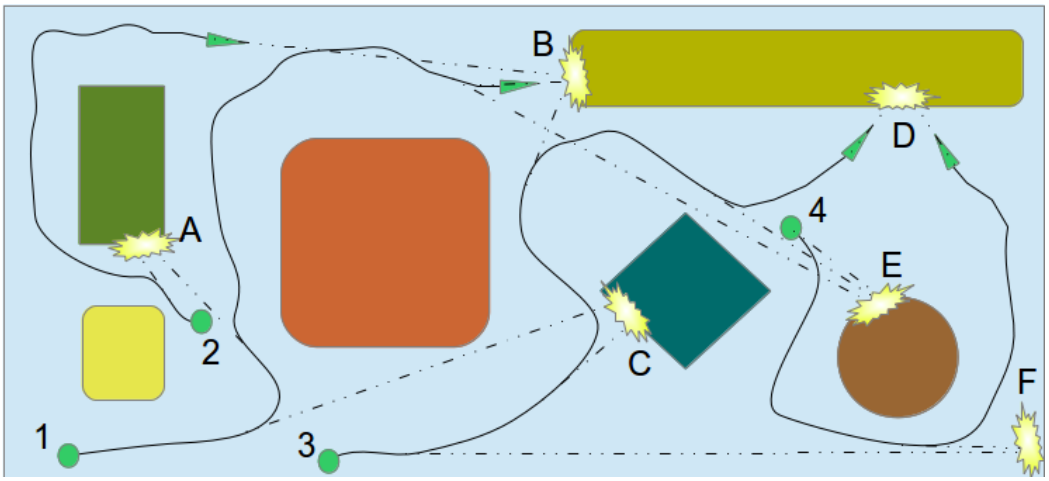


Figure 4.2: A general scenario of multiple reconstruction. Here $r = 4$. There are 6 overlaps (A-F), each one involving 2 or 3 different sessions.

and the addition of camera poses from different structures is efficiently handled during the Bundle Adjustment (BA) stage. On the other hand it is general since it does not rely on encounters. This means that it can be applied to any kind of mobile platform. In addition, it is valid for both multi-robot and multi-session reconstructions. In this work we study the multi-session case, which means that an experiment in the multiple case is, in practical terms, a sequence of single experiments. Therefore the deployment of the Rovio follows the same manner as in Chapter 3 in all aspects.

4.2 Multiple Reconstruction

Even though the system which performs multiple SfM is based on the SfM process described in Chapter 3, which treats images sequentially, here we take a different procedure. To begin with, the sequence taken by each robot is stored as a different *session*. Therefore a set $\mathcal{S} = \{\mathcal{I}_i\}_{i=1}^{i=r}$ is defined in multiple reconstruction, where each session is composed of n_i images, $\mathcal{I}_i = \{I_q\}_{q=1}^{q=n_i}$, and the total number of images is $m = n_1 + n_2 + \cdots + n_r$. Fig. 4.2 shows a general layout of a multiple reconstruction system. It is important to remark here that all the sessions share the same storage variable for the 3D structure.

First of all, the images of all the sessions are preprocessed and their features and descriptors are extracted, in parallel mode according to the number r of sessions.

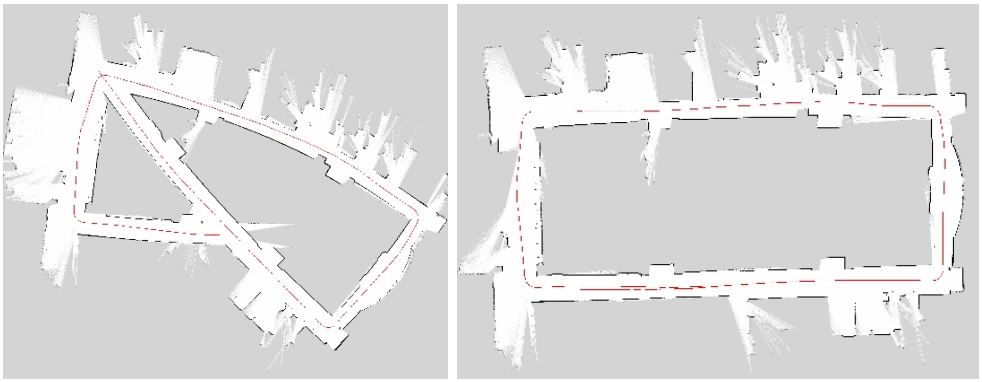


Figure 4.3: Example of loop-closure. When a loop-closure is produced the drift of the whole loop (left image) can be corrected and the mapping is shown correctly (right image). This result is from Kaess and Dellaert (2005).

The preprocessing is performed by means of the bilateral filtering method (Section 3.3) .

Before each session is processed, it is necessary for every session to share some information with the others so that it will be possible during the processing of each session to establish a common reference frame for all the sessions. This is achieved by finding loop-closures (or *overlaps*) across sessions.

4.2.1 Loop Closing

Loop closing is the problem of finding in a sequence, giving a query image, the image which overlaps best with it. Loop closing is applied mainly in the context of SLAM, when a robot revisits an area already transited. In this situation the robot should:- 1) realise that it has already been there, 2) find the image that corresponds to the first transition so that it can be matched to the image of the second transition and the old landmarks can be recognised back in view. These two steps are necessary or the camera position will not be estimated correctly, and the drift problem, inherent to the SLAM technique, will not be solved. The difficulty in finding a loop-closure comes from the increase of uncertainty as the robot goes through a large loop and the inconsistency created in the extended kalman filter (EKF) due to linearisation errors. Fig. 4.3 shows an example of the effects of a loop-closure on the camera pose estimations of a large loop taken by a mobile platform.

The loop closing problem is still considered an open problem, even though much research

Figure 4.4: A false loop-closure assigned by FAB-MAP, described in Section 4.2.2. Source: Cummins and Newman (2011).

has been devoted to this in recent years. It is considered an open problem because there are cases where the most robust algorithms fail, as it is shown in Fig. 4.4. The challenge lies in the difficulty of identifying a given scene under different lighting conditions (and atmospheric elements when outdoors) and perspectives, and to differentiate the key features of a scene from objects which are accidental or temporary.

In the scope of SLAM, Latif et al. (2013) present a consensus-based approach to robustly place recognition over time. It can correct wrong loop-closures, works in an incremental fashion and handles multi-session. Lepetit and Fua (2006) developed a key-point-based approach, which formulates wide-baseline matching between keypoint of query images and model images (training set). By formulating the problem as a classification problem, and implementing the solution with randomised trees, Lepetit and Fua (2006) achieve a big run-time computational reduction. Williams et al. (2011) devise the relocalisation method, by which the camera poses are relocalised relative to the map structure of the scene. This method extends the image classifier given by Lepetit and Fua (2006) to a system that learns the appearance of a given image patch at each map feature. These learned landmarks enable the relocalisation of a camera in case of failure tracking or a loop closure, while preserving the map integrity.

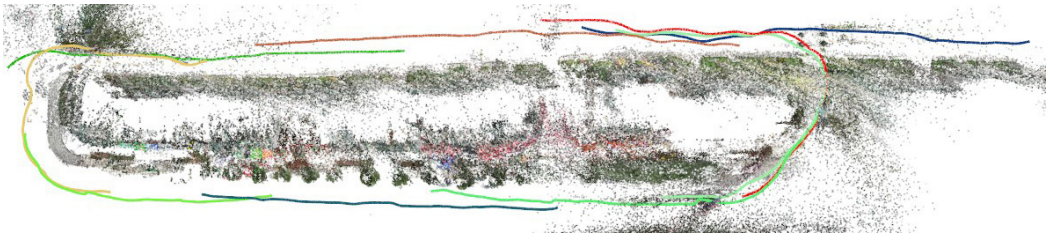


Figure 4.5: A result from Zhang et al. (2010). Note that different sessions need to overlap along a stretch of images so that feature tracks can be matched.

Another solution for loop-closing is to establish global descriptors. Shvarts (2013) proposes the use of GIST, a global descriptor based on the amplitude spectrum of the Fourier transform of an image, as a way to perform loop-closing and identify similar images in a database. Shvarts (2013) implements a method that compares the global descriptors of big databases in a multi-stage filtration procedure. Subsequently Shvarts (2013) introduces a new algorithm for merging maps based on neural networks.

There are investigations that have proposed methodologies to perform non-consecutive matching, which can be seen as a variant of loop-closing. Zhang et al. (2010) address the non-consecutive feature point tracking problem and propose an effective method to match interrupted tracks when performing SfM. The domain of Zhang et al. (2010) is on sequences of VGA images. It addresses the consecutive matching by selecting putative correspondences with Lowe's ratio (see Section 2.4.1) and applying a constrained spatial search with planar motion segmentation. This is realised by estimating several homographies between pairs of images and then rectifying them according to these transformations. This two-pass approach multiplies the number of long feature tracks. The matching between non-consecutive images is done in two steps. Firstly, a matching matrix for all the images of the sequence is computed. In order to generate the matching matrix a vocabulary tree of feature tracks descriptors is constructed for fast image indexing. Subsequently a hierarchical K-means approach is applied on the vocabulary tree to cluster the feature tracks of the sequence. The values of the matching matrix are the measurements of the overlapping confidence based on the feature track descriptors similarity cast by the clustering of the vocabulary tree. In the second step rectangular regions containing the brightest (which denote most overlapping confidence) pixels in the matching matrix are used for detecting overlapped subsequences. Therefore Zhang et al. (2010) can track features from multiple videos with overlapping subsequences.

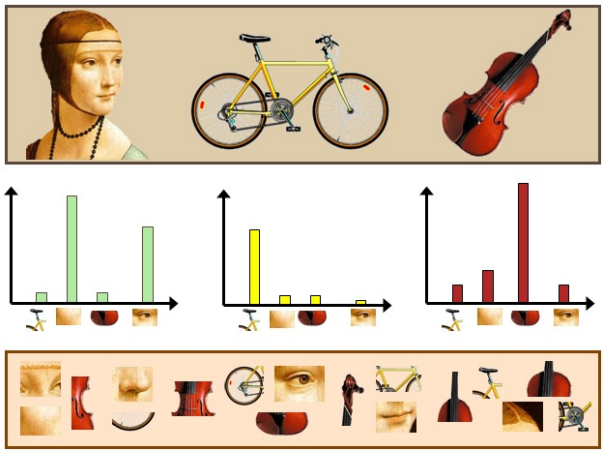


Figure 4.6: By capturing characteristic regions of objects, a vocabulary of bags-of-words (BOW) is generated. Source: Anon. (2015)

However attractive the results of Zhang et al. (2010) are, their loop-closing method relies on overlapping *subsequences* of images, rather than images themselves, since the matching entity used in the loop-closure step is feature tracks. If two sequences of images have only few scattered overlapping images, like in Fig. 4.2, they will not produce similar feature tracks and the multiple reconstruction will not be possible. Our method is capable of merging different 3D reconstructions with a minimal number of overlaps, even a single overlap (see Chapter 5). Fig. 4.5 shows how different sessions overlap along subsequences.

The method which has demonstrated to be best fitted to address the loop-closing method, according to this criteria, and for large sequences of datasets, is Fast-Appearance Based MAPping (FAB-MAP) method, by Cummins and Newman (2011), which is able to operate with hundreds of thousands of images. We have addressed the loop-closing problem with FAB-MAP.

4.2.2 FAB-MAP

FAB-MAP is a system developed for appearance-based place recognition. FAB-MAP compares images of locations that have already been visited against new images, and outputs the probability that the new image is a location re-visited, as well as an estimate of the probability of being at a new, not visited location. The only input of FAB-MAP is the camera images. This system has been devised as a "appearance-only SLAM", so that

Figure 4.7: A summary of the 67 categories of the data-set used as training set. Source: Quattoni and Torralba (2009).

a map can be built up taking as a reference the loop-closures found by FAB-MAP. This system has been tested on a data-set of more than 1000 km of road and over 103.000 omnidirectional images, at a frame rate of 2Hz (Cummins and Newman (2011)).

FAB-MAP works with Bags Of Words (BOW) and therefore it requires a vocabulary created from a data-set of training data. A BOW is a sparse vector of occurrence counts of a given vocabulary of local image features, created from a training data-set. Given a query image, this image can be quantized using BOWs from the vocabulary and then identified with a category present in the vocabulary. Each category is associated to a certain histogram profile produced by the BOWs. Fig. 4.6 illustrates this process: each picture (bust of a lady, a bicycle and a violin) is decomposed in local image features and these features matched to a given vocabulary of features. Then for every image a histogram is created with the number of occurrences. This histogram provides information as to which category each image belongs, as shown in Fig. 4.6. In FAB-MAP each image of a sequence defines a category itself. When a new image is received, SURF features are extracted. These features are then segmented according to the vocabulary, which produces *visual words.* This vocabulary is built up by clustering all the features from the training data-set. The Voronoi regions[3] of the cluster centres establish the set of features that will be associated to a visual word.

BOW tend to be related to themselves. For example, the BOW corresponding to a drawer will normally appear along with a BOW of a table. These dependencies between BOWs are captured by a tree-structured Bayesian network, using the Chow-Liu algorithm (Chow and Liu (2006)).

[3]In a given area where there is a set of scattered points, the Voronoi region of a given point is the area where the distance to this point is shorter than to any other.

The estimation whether a query image is a new location or whether it belongs to a scene already seen by the system (and the determination of the scene) is performed in FAB-MAP by applying a conditional Bayesian rule to the BOWs that the query image generates and the BOWs of the rest of the previously observed system. The mapping created by FAB-MAP can be seen therefore as a recursive Bayes estimation system.
In order to deal with data-sets of thousands of images FAB-MAP makes use of inverse term weighting and geometric verification. Given a sequence of images (each one has originated a number of BOWs), inverted index is the number of images that a BOW appears in. The geometric verification consists of verifying a rough geometric transformation between the new observation and the images that result with most probability of being loop-closures. This verification is done via RANSAC.
This work has used the OpenCV (Bradski (2000)) implementation of FAB-MAP, openFABMAP (Glover et al. (2012)). Therefore, a vocabulary of BOWs and a Chow-Liu tree have been built up based on OpenCV methods. Our implementation differs from OpenCV implementation in that the steps of cluster centres creation and estimation are parallelised with Threading Building Blocks (TBB)[4], in order to save computation time during the training phase.
The choice of the training set is important. It should contain images similar to those of the sequences to reconstruct but not from the same scene[5]. In order to cover all the possible scenarios the training set chosen for our system was taken from an extensive database developed by MIT with 67 categories and more than 15.000 images, used for indoor scene recognition (Quattoni and Torralba (2009)). Fig. 4.7 shows the 67 categories of the data-set. We have created two training sets out of the MIT data-set: *training-set.5M* and *training-set.10M*, with BOWs from 300 and 600 images respectively, taken homogeneously from the MIT data-set.
The output given by our implementation is a multiple array $\mathcal{T}$, with three levels of depth. This array contains, for every image I_q^i of every session $\mathcal{I}_i$, an array of images $\mathcal{L}_q^i = \{I_s\}_{s=1}^{s=m-n_i}$) with which I_q^i overlaps. Note that a given image may have overlaps with several images from different sessions. However, in practice $\mathcal{T}$ will be largely sparse. For example, in Fig. 4.2 the maximum number of overlaps per session is 4.

[4]TBB is a C++ template library which simplifies the implementations of applications running in parallel. See `https://www.threadingbuildingblocks.org/`

[5]See Cummins and Newman (2008) and `https://code.google.com/p/openfabmap/wiki/Usage`

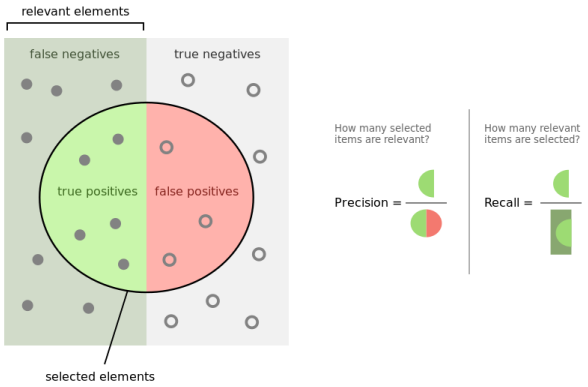


Figure 4.8: Precision and recall explained graphically. From Walber (2015).

These overlaps will be used in multiple reconstruction to establish the loop-closures and merge 3D maps from reconstructions of several mobile platforms. An example of loop-closure produced in our experiments is shown in Fig. 4.9.

Since Glover et al. (2012) is applied sequentially on the sessions, the overlaps will be made between images I_s from session $\mathcal{I}_i$ with images I_q from session $\mathcal{I}_j$, $j < i$. The overlaps are in this direction because posterior sessions are queried against previous sessions, which have already been processed by Glover et al. (2012). We are interested in the overlaps happening in the opposite direction, in order to meet the common reference requirement. Section 4.3 shows that, after a camera pose from image I_q from session $\mathcal{I}_j$ is estimated, it is checked whether it has overlaps I_s from sessions $\mathcal{I}_i$ that have not been reconstructed yet ($j < i$), so that I_s can be estimated in the same frame of reference as I_q. In this case I_s will become a seed for the reconstruction of session $\mathcal{I}_i$. In order to obtain this configuration after the execution of Glover et al. (2012) the matrix of overlaps is inverted.

The detection of loop-closures is usually evaluated by means of two metrics: precision and recall, which are usually represented in graphs as precision vs recall. These terms can be defined as:-

- **Precision** It refers to the fraction of relevant instances that have been retrieved (*True Positives*, TP) over the subset of retrieved samples (*True Positives* and

False Positives, FP). It can be expressed with the ratio:-

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.1}$$

- **Recall** It refers to the fraction of relevant instances that have been retrieved (TP) over the subset of relevant instances (*True Positives*) and *False Negatives*, FN). It can be expressed with the ratio:-

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.2}$$

Precision is a measure of the quality of a system in retrieving instances of the target class (loop-closures in our case) and recall is a measure of the quantity of instances of the target class that have been retrieved over the subset of relevant instances. Fig. 4.8 shows graphically these two concepts applied on a set of elements.
The graph precision vs recall shows how accuracy decreases when more images are retrieved by the system. The number of images retrieved (TP + FN) is regulated by a threshold, τ_l. We have evaluated our implementation of Glover et al. (2012) on two experiments, each of a different kind: *engineRoom* and *turntable2* sequences. Fig. 5.17 and Fig. 5.26 show respectively a sample of each sequence. These two experiments have been described and studied in detail in Chapter 5. The sequence *engineRoom* consists of two sessions of 75 and 72 images respectively, with 10 overlaps between them. In the sequence *turntable2* the platform turns around a pile of objects, drawing two perfect circles. Each lap consists of 95 images. Table 4.1 shows the indices of precision and recall for these two experiments.

	Precision	Recall
engine room	1	0.1
turntable	0.90	0.89

Table 4.1: Precision and recall indices for *engineRoom* and *turntable2* sequences.

It is crucial in multiple reconstruction that the loop-closures detected are not false positives, because a false positive can make the 3D reconstruction unstable. Nevertheless a wrong overlapping between sessions can be ruled out during the matching and pose

Figure 4.9: Example of loop-closure between two images taken in different sessions from the platform used.

estimation processes, but this is not desirable. For this reason the parameters that govern the behaviour of Glover et al. (2012) are set in this case very conservatively. The result is apparent in Table 4.1. The precision in the *engineRoom* sequence is the unit (no false positives), but it comes at the price of a low recall.

Another case is a mobile platform which revisits areas of an environment in a single sequence. Here there is no need of estimating the camera pose of the overlap I_s (it will be estimated as in the single case), so the matrix of overlaps is not inverted. Therefore the overlaps in this case happen to have previously been estimated. In Section 4.5 it will be explained that overlaps between images of the same session produce new matches (and *bundles*), but no camera pose is estimated when processing them. Moreover, in this case overlaps are most likely to happen in sub-sequences, so that a subset of consecutive images are matched against another subset of also consecutive images. As a consequence of this the precision here is lower, whereas the recall is significantly higher, as Table 4.1 shows.

Our implementation of Glover et al. (2012) is executed after the feature descriptors have been extracted from all the sessions $\mathfrak{I}_i$, $1 \leq i \leq r$. After the loop-closing phase all the overlaps between different sessions have been found and each session is processed roughly following the pipeline described in Chapter 3, but with alterations in order to take into account the overlaps, as algorithms 1 and 2 describe.

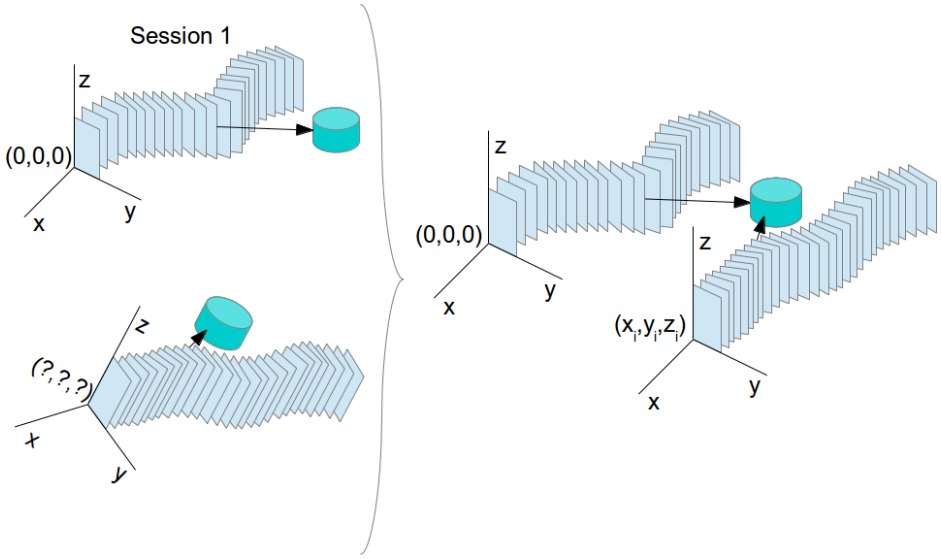


Figure 4.10: Session 1 is always fixed, with the first camera at origin and with no rotation. (top left). However, it is not possible to allocate session i (bottom left) unless an overlapping is found and the sessions bound together (right).

4.3 Overlap Management

With the feature vectors extracted from all the sessions and the overlaps between sessions found, each session is now executed in a similar manner as in the single SfM process (see Chapter 3). The main variation we introduce here is the position I_h, $1 \leq h \leq n_i$ within the sequence of images $\mathcal{I}_i = \left\{I_q^i\right\}_{q=1}^{q=n_i}$ where the SfM processing of each session will start.

A sequence of images $\mathcal{I}_i$ has to be allocated within a global reference frame in order to be reconstructed (see Section A.5). In the case of a single session ($r = 1$), this is naturally fixed by imposing the camera pose of the first image I_1^1 to be at origin with no rotation. However in the multi-session case this requirement becomes apparent, since in general each mobile robot will start to move around a scene in a different location. Therefore it is not possible to assign to the first camera of each session $\mathcal{I}_i$ the origin location, if all the sessions are to be allocated in the same coordinate frame. It is therefore necessary to find an alternative way to place the camera poses of session $\mathcal{I}_i$ with respect to at least one of the j previous sessions, $1 \leq j < i$. This is precisely the problem that overlaps are used for in this work. Fig. 4.10 illustrates this situation.

Owing to the global coordinate frame issue, the algorithm for multiple reconstruction

devised here distinguishes between the session $\mathcal{I}_1$ and the sessions $\mathcal{I}_i$, $1 < i \leq r$.

Algorithm 1 Process of session $\mathcal{I}_1$

Require: set of images $\mathcal{I}_1 = \{I_q\}_{q=1}^{q=n_1}$
Ensure: 3D reconstruction of $\mathcal{I}_1$, overlaps $\mathcal{L}_q^1$ reconstructed.
  for all $I_q \in \mathcal{I}_1$ **do**
    if $I_q > 1$ **then**
      $I_q \leftarrow$ recursive matching against set $\mathcal{D} = \{I_{q-k} \cdots I_{q-1}\}$
      $I_q \leftarrow$ epipolar geometry, resection, triangulation, BA
      add I_q to $\mathcal{D}$
      check $\mathcal{T}(1, q)$ for loop-closures $\rightarrow \mathcal{L}_q^1 = \{I_s\}_{s=1}^{s=v}$, $1 \leq v \leq m - n_1$
      for all $I_s \in \mathcal{L}_q^1$ **do**
        $I_s \leftarrow$ recursive matching against set $\mathcal{D}$
        $I_s \leftarrow$ epipolar geometry, resection, triangulation, BA
      end for
    else
      $I_1 \leftarrow$ origin
    end if
  end for

4.3.1 Session $\mathcal{I}_1$

The first session is reconstructed normally, setting the first camera I_1^1 in the origin, without loss of generality. For each camera I_q^1 of session $\mathcal{I}_1$, $q > 1$, the SfM process now takes place.

The main difference to the execution of SfM between the singular case and the multiple case comes from the methods that are run at a time. The preprocessing and feature extraction steps have already been applied, so now I_q^1 goes through the remaining methods of the SfM pipeline (see Fig. 3.1). The multiple reconstruction implementation makes use of a shared library built from the methods of the singular case, and an appropriate structure of inherited classes is devised. As a result, matching process, relative and global pose estimation, triangulation and local BA (how global BA has been executed here will be discussed in next section), are run by the very same methods as in the single SfM process.

After the application of SfM on I_q^1, I_q^1 is added to the set $\mathcal{D}$ of reconstructed images, and the element $(1, q)$ of $\mathcal{T}$ is looked up to check whether I_q^1 has overlaps with ulterior sessions $\mathcal{I}_i$, $1 < i \leq r$. In the general case I_q^1 will have an array of overlaps $\mathcal{L}_q^1 = \{I_s\}_{s=1}^{s=v}$,

$1 \leq v \leq m-n_1$ (although usually $v \ll m-n_1$). In this case, each overlap I_s is processed and matched against the set $\mathcal{D}$, so that the recursive matching is performed now on the set $\left\{I^1_{q-k}, I_s\right\}_{k=0}^{k=t}$. I_{q-t} will be the image with which I_s has less matches than a threshold τ_m (see Section 3.4). Hence epipolar geometry of I_s is estimated with respect to I^1_q, common bundles are generated between I_s and session $\mathfrak{I}_1$, and I_s is resected with reference to $\mathfrak{I}_1$, as shown in Alg. 1. Overall, the camera pose of I_s is defined with respect to the global coordinate frame of session $\mathfrak{I}_1$. Once I_s has been fixed in relation to $\mathfrak{I}_1$, the next element of $\mathcal{L}^1_q$ is processed in exactly the same way, being matched against the set $\mathcal{D}$. This algorithm has been written in pseudocode in Alg. 1.

The reconstructions of the overlaps $\mathcal{L}^1_q$ in the session $\mathfrak{I}_1$ allow posterior sessions to allocate their camera poses in a unique coordinate frame, so that a global 3D structure out of multiple sessions can be obtained.

4.3.2 Session $\mathfrak{I}_i$, $i > 1$

The difference in the execution of $\mathfrak{I}_1$ with $\mathfrak{I}_i$, $1 < i \leq r$ resides on which image of the sequence is taken as the "first camera", that is, which image will fix the coordinate frame of the sequence. The image I^i_h from sequence $\mathfrak{I}_i$ with the smallest index whose camera pose has already been estimated (by means of a loop-closure with I^j_q from sequence $\mathfrak{I}_j$, $j < i$.) will be appointed as the starting point to process the sequence $\mathfrak{I}_i$.

Since the "starting point" in session $\mathfrak{I}_i$ is I^i_h, now the SfM process should be run in two sequences: $\left\{I^i_{h+1} \cdots I^i_{n_i}\right\}$ and $\left\{I^i_1 \cdots I^i_{h-1}\right\}$. The second sequence will be addressed in reverse order.

Along with the global reference frame problem, there is the scale issue. In order for the 3D structure from $\mathfrak{I}_i$ to be coherent with the rest of sessions, it does not suffice to locate a camera I^i_h of $\mathfrak{I}_i$ with respect to its overlap I^j_q from a previous sequence. With solely that information the rest of the cameras of $\mathfrak{I}_i$ will be placed in general with an arbitrary scale, rendering the multiple reconstruction infeasible, and BA methods would not be able to produce coherent structures out of point clouds with different scale. It is therefore necessary to fix the scale of $\mathfrak{I}_i$ according to prior sequences.

As discussed in Sections A.5 and 3.7 a camera pose is integrated with the rest of camera poses of a given sequence by resection. In multiple reconstruction we solve the problem of the scale by applying this technique as well. This can be done by just looking up, out of the set of matches from the pair $\left\{I^j_q, I^i_h\right\}$, those which belong to bundles that have

Algorithm 2 Process of session $\mathfrak{I}_i, 1 < i \leq r$

Require: set of images $\mathfrak{I}_i = \{I_q\}_{q=1}^{q=n_i}$
Ensure: 3D reconstruction of $\mathfrak{I}_i$, overlaps $\mathcal{L}_q^i$ reconstructed.
  for all $I_q \in \mathfrak{I}_i$ **do**
    if $I_q^i \rightarrow$*estimated* **then**
      $I_h^i \leftarrow I_q^i$
      break
      {The flag *estimated* indicates whether an image has been reconstructed.}
    else
      continue
    end if
  end for
  for all $\{I_q, q > h\} \in \mathfrak{I}_i$ **do**
    $I_q^i \leftarrow$ recursive matching against set $\mathcal{D} = \left\{I_{q-k}^i \cdots I_{q-1}^i\right\}$
    $I_q^i \leftarrow$ epipolar geometry, resection, triangulation, BA
    add I_q^i to $\mathcal{D}$
    check $\mathcal{T}(i,q)$ for loop-closures $\rightarrow \mathcal{L}_q^i = \{I_s\}_{s=1}^{s=v}$, $1 \leq v \leq m - n_i$
    for all $I_s \in \mathcal{L}_q^i$ **do**
      $I_s \leftarrow$ recursive matching against set $\mathcal{D}$
      $I_s \leftarrow$ epipolar geometry, resection, triangulation, BA
    end for
  end for
  for all $\{I_q, q < h\} \in \mathfrak{I}_i$ **do**
    $I_q^i \leftarrow$ recursive matching against set $\mathcal{D}' = \left\{I_{q+1}^i \cdots I_{q+k}^i\right\}$
    $I_q^i \leftarrow$ epipolar geometry, resection, triangulation, BA
    add I_q^i to $\mathcal{D}'$
    check $\mathcal{T}(i,q)$ for loop-closures $\rightarrow \mathcal{L}_q^i = \{I_s\}_{s=1}^{s=v}$, $1 \leq v \leq m - n_i$
    for all $I_s \in \mathcal{L}_q^i$ **do**
      $I_s \leftarrow$ recursive matching against set $\mathcal{D}'$
      $I_s \leftarrow$ epipolar geometry, resection, triangulation, BA
    end for
  end for

already been reconstructed (see Section 3.7). However, in order to maximise the link between $\mathfrak{I}_i$ and $\mathfrak{I}_j$, I_h^i is recursively matched to the images $\left\{I_{q-k}^j\right\}_{k=0}^{k=t}$ until the matches between I_h^i and I_{q-t}^j are less than a threshold τ_m.

The sequence is reconstructed sequentially, with the first camera being I_h^i. Analogously to the procedure in the first session, after I_q^i has been processed the element (i, q) of $\mathfrak{T}$ is looked up in the search for overlaps with I_q^i. In case affirmative the overlaps $\mathcal{L}_q^i$ are treated in the same manner as in Section 4.3.1.

It may occur that session $\mathfrak{I}_i$ has more images $I_{h'}$, $h' > h$, already reconstructed from overlaps with previous sessions other than I_h^i. In this case the epipolar geometry and resection of $I_{h'}$ is skipped during the SfM process.

After the image $I_{n_i}^i$ has been processed, there remains to be reconstructed the images $\left\{I_1^i, \cdots I_{h-1}^i\right\}$.

Reverse Reconstruction

The set of images $\left\{I_1^i, \cdots I_{h-1}^i\right\}$ are reconstructed in the same manner as the rest of the sequence, with the only qualification that they are processed in reverse order, starting in I_{h-1}^i and finishing in I_1^i. Now the set of reconstructed images $\mathcal{D}$ is built up backwards, and the recursive matching set is $\left\{I_q^i, I_{q+k}^i\right\}_{k=1}^{k=t}$, with $1 \leq q < h$ and $1 \leq t < n_i$. As always in multiple reconstruction, when I_q^i has been reconstructed the element (i, q) of $\mathfrak{T}$ is looked up in the search for overlaps with I_q^i. The Alg. 2 describes the steps followed to reconstruct session $\mathfrak{I}_i$.

Overlaps management is an integral part of the multiple reconstruction system developed in this work. Another aspect important in this system is how BA has been applied over multiple sessions, which creates a significant difference between the singular case and the multiple case. The correct application of BA in the multiple case is paramount not only to optimise the structure and camera motion, but also to enable the optimisation of all the sessions and the overlaps between them.

4.3.3 Distributed Bundle Adjustment

The BA methods are the pool where the reconstruction of all sessions are merged, along with all the cameras. This merging is possible thanks to the overlaps, which provide a link between sessions and allow BA to find a coherent scale for the whole set.

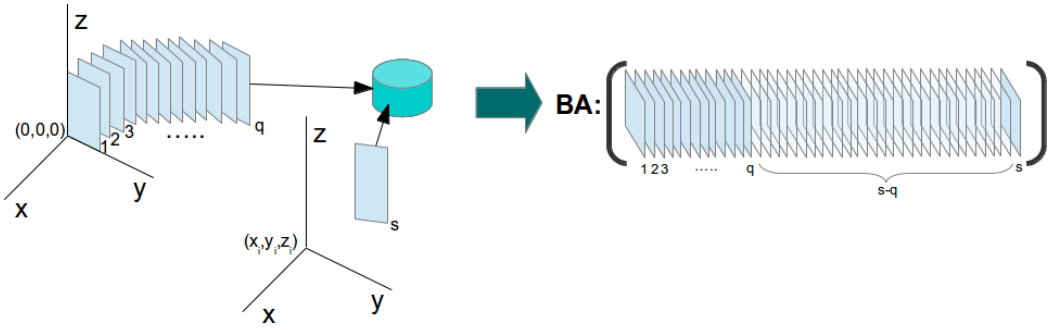


Figure 4.11: When an overlap is found ($id_q \leftrightarrow id_s$), BA methods add to the optimisation set $\{s-q\}$ fictitious images, in order for BA to be able to operate over the overlap. The actual $\{s-q\}$ images will be added in later on. This operation does not affect neither the efficiency nor the optimisation of the real images added to BA.

In a similar way as in the singular case, here we apply pair-wise BA every time an image is processed (either an image I_q^i from a given sequence or an overlap I_s) and equally, in a different thread, a local BA after each image is processed. Every m images processed, a global BA is executed.

Specifically in the multiple case, every session stores their 3D point structure in the same global storage reference, which is in turn taken by the BA methods as input. Regarding the camera poses, the BA methods address all the cameras by a global index, with no distinction for the session a camera comes from. The camera poses are added sequentially to BA as they are processed, so when an overlap with global index id_s is added for optimisation to the stack of camera poses, where the last added camera has a global index id_q, $\{s-q\}$ cameras are added before Id_s, without penalising the optimisation process of BA. This situation is shown in Fig. 4.11.

After each application of BA a reprojection error-based filter is run on the whole structure. In addition, a global BA is executed on the whole set before the post-process step.

4.4 Post-Process

In a similar way to the singular case, after all the sessions have been processed, a post-process step is performed. The post-process pipeline follows the same flowchart as depicted in Fig. 3.18, with the qualification introduced by the overlaps between sessions. Therefore, for a given I_q^i from sequence $\mathcal{I}_i$, the post-process phase is first run on all its

overlaps $\mathcal{L}_q^i$ and subsequently on I_q^i, as part of the sequence $\mathfrak{I}_i$. Alg. 3 describes the algorithm performed in post-process.

Algorithm 3 Post-Process of session $\mathfrak{I}_i, 1 \leq i \leq r$

Require: set of reconstructed images $\mathfrak{I}_i = \{I_q\}_{q=1}^{q=n_i}$
Ensure: Semi dense 3D reconstruction of $\mathfrak{I}_i$ and overlaps $\mathcal{L}_q^i$.
  for all $I_q \in \mathfrak{I}_i$ **do**
    check $\mathfrak{T}(i,q)$ for loop-closures $\rightarrow \mathcal{L}_q^i = \{I_s\}_{s=1}^{s=v}$, $1 \leq v \leq m - n_i$
    for all $I_s \in \mathcal{L}_q^i$ **do**
      $I_s \leftarrow$ recursive matching against set $\mathcal{D} = \left\{I_{q-k}^i \cdots I_{q-1}^i\right\}$
      $I_s \leftarrow$ triangulation, BA
    end for
    $I_q^i \leftarrow$ recursive matching against set $\mathcal{D} = \left\{I_{q-k}^i \cdots I_{q-1}^i\right\}$
    $I_q^i \leftarrow$ triangulation, BA
  end for

In contrast with the application of BA in multiple reconstruction, and since the cameras are fixed during this phase, now a different BA instance is created for every session, which will only optimise the structure of that session. Still the whole 3D structure will be shared by each and every session.
The specific implementation of the BA methods and the management of overlaps that is taken in our distributed reconstruction system produces enables our system to address single cases where the platform transits already visited areas.

4.5 The Case $r = 1$

The main difference that a loop-closure between frames of the different sessions (multiple case) has with respect to a loop-closure between frames of the same session (single case) is that in the single case the overlaps are always images that have already been reconstructed, since in this case the matrix of overlaps has not been inverted (see Section 4.2.2). Therefore, if I_q (whose camera pose is known) has an overlap with I_s, $s < q$, which has previously been reconstructed, there is no camera pose to be estimated. Furthermore, since I_s has already been estimated, it has been added to the BA methods and should not be added again. In this case the overlap is used to link I_s and I_q by means of the matches between them, so that when Global BA is run their camera poses will be refined further, and any drift will be corrected.

The only action to take here, therefore, is to ensure that the overlap I_s is not added neither to the pair BA nor to the Global BA methods if I_s belongs to the same session as I_q. Note that this measurement is transparent for the multiple reconstruction case. In Chapter 5 we have evaluated this case on the *turntable2* sequence.

4.6 Summary

This chapter has described in detail how distributed scene understanding has been achieved in this work.

After a general review on different techniques for realising multiple reconstruction, we justify the use of loop-closing techniques as a simple approach that at the same time intends to be general. Different works that make use of loop-closing are revised and the technique of our choice, FAB-MAP, is thoroughly explained.

Our implementation of FAB-MAP, Glover et al. (2012), outputs for every image of every *session* covered by each mobile robot, whether there is an overlap with any image of other posterior session. The system exposed here manages these overlaps in order to link the sessions together and enable 3D map merging, which is effectively done by the BA methods especially arranged for this purpose. Our approach enables the reconstruction of singular sequences where the platform transits areas already visited.

Once all the camera poses of all sessions have accurately been estimated and a sparse 3D structure obtained, each session goes through a post-process phase that, similarly to the post-process of Chapter 3, multiplies the number of features.

The distributed system developed in this work from multiple instances goes beyond the state of the art work in multi-session reconstruction. Specifically, the presented system extends the work of Zhang et al. (2010), which requires overlaps between stretches of sequences, whereas we attain 3D merging with minimal overlapping between sessions. In chapter 5 we show that our system resolves the scale problem between sessions with only one overlap so that they can be referenced with respect to a global coordinate frame.

The performance and results of our implementation of multiple reconstruction, along with the results of our single SfM system will be discussed in Chapter 5.

Chapter 5

Performance Evaluation

This chapter evaluates the performance of the system devised in this work. First the methodology for the evaluation of the system is described. Subsequently the SfM process developed here is tested in both the singular and multiple cases.

With respect to the single reconstruction, results from experiments in different environments are shown, and the evaluation methodology is applied in these experiments. In addition, the performance of our system is compared with other state of the art systems, especially on the points where we believe have been improved by our work.

This system has also been validated by employing it on the reconstruction of known benchmark data-sets and by comparing it with state of the art software. Additionally, ground-truth validation has been carried out.

The distributed reconstruction system is also assessed. The visual 3D reconstruction and camera poses from multiple reconstruction experiments are shown, and the quality of our implementation of FAB-MAP on finding loop-closures evaluated.

In addition to multiple sessions, the distributed SfM system developed in this work can also be applied to a single mobile robot which revisits part of a scene. An experiment of this kind is discussed to show the performance of the multiple reconstruction system in these cases.

The first step for evaluating a given system is the definition of a methodology. This methodology should be clearly designed in order to provide some objective metrics that could be reproduced when assessing other systems for fair comparison.

5.1 Methodology for Evaluation

The adequate definition of the methodology for evaluation of a system is important because it needs to assess correctly the parameters that a given work optimises. Beardsley and Torr (1996) introduces several statistics and metrics to evaluate and compare SfM systems, focusing on the trifocal sensor. In the context of BA algorithms, Triggs et al. (2000) devotes a section to quality control, and offers various methods to evaluate the internal and external reliability of a BA method, how to analyse the sensitivity of a system and how to perform model selection tests.

This work develops a SfM system on mobile robots with special characteristics (noisy images, challenging feature tracking, omnidirectional motion) which creates specific challenges on the fields of correspondence matching, feature tracking and epipolar geometry estimation. All these aspects are closely related. First we will establish how to evaluate our system globally, and then each of the aforementioned fields will be addressed.

A SfM system is assessed by checking the accuracy of the camera poses estimated with respect to a ground-truth. If the motion estimation is correct, and a good triangulation method is used (see Section 2.7) the 3D reconstruction will be reliable. However, the ground-truth of a given data-set in not always readily available, and alternative methods should be found[1]. The most common method to jointly assess both motion and structure estimates is by measuring the reprojection error given by the 3D reconstruction on the camera poses. The derivation of the reprojection error is given in Section A.7.1.

In the evaluation of our results we have used the *Root Mean Square* (RMS) of the reprojection error over the whole SfM system. The RMS of a set of values is the square root of the arithmetic mean of the squares of these values. In Appendix A the reprojection error given by a 3D point $\boldsymbol{X_j}$ over the view $\boldsymbol{x_{ij}}$ is defined in the Eq. A.7.52. Expressing the Euclidean distance in mathematical terms, Eq. A.7.52 yields:-

$$\epsilon_{ij} = \sqrt{(\overline{x}_{ij} - x_{ij})^2 + \left(\overline{y}_{ij} - y_{ij}\right)^2} \tag{5.1}$$

Therefore, the RMS applied on the reprojection error over the whole 3D structure,

[1] We have evaluated our system against ground-truth in Section 5.2.5.

introduced in Eq. A.7.53, yields:-

$$\text{RMS} = \sqrt{\frac{\sum_{j=0}^{j=ni=m} \sum_{i=0} v_{ij} \left((\overline{x}_{ij} - x_{ij})^2 + (\overline{y}_{ij} - y_{ij})^2 \right)}{2l}} \tag{5.2}$$

where

$$l = \sum_{j=0}^{j=ni=m} \sum_{i=0} v_{ij} \tag{5.3}$$

Eq. 5.3 accounts for the sum of all the projections on the set of m cameras from each of the n 3D points of the structure. As described in Section A.7.1, v_{ij} denotes the binary variables that equal 1 if the *jth* 3D point is visible in image i and 0 otherwise.
For the sake of clarity we have normalised Eq. 5.2 so the measurement evaluated here is:-

$$\text{RMS}_N = 1000 \times \text{RMS} \tag{5.4}$$

The reprojection error explains how well the 3D structure fits with the camera poses estimated. Usually, the combined information of the reprojection error, the number of 3D points, the total number of views or projections, and a visual inspection of the point cloud and camera poses are good enough to assess qualitatively how good a 3D reconstruction is. The number of 3D points deleted during the filtering process is valuable if we are interested in assessing the filtering performance of the algorithm.
However, it is possible that a poorly estimated reconstruction gives a low reprojection error if the feature tracks are short enough. In fact, a 3D point reconstructed with only 2 views will always give 0 reprojection error. This statement can be gathered from Section A.6.1 and specifically from Eq. A.6.50. Indeed, if $n = 2$, the linear system given in Eq. A.6.51 is a square system of independent equations so it is determinate compatible and therefore with a unique exact solution, regardless the accuracy of the 2 camera poses that form the linear system. Hence the 3D point $\boldsymbol{X}$, exact solution of Eq. A.6.51, will always cast null reprojection error, but its 2 views will be taken into account in the denominator of Eq. 5.2, artificially modifying downwards the total reprojection error of a given 3D structure. For this reason the reprojection error of all

the data-sets in this chapter has been evaluated taking into account only 3D points whose *bundles* have more than 3 views, $n > 3$.

Conversely, a 3D reconstruction with a large proportion of long feature tracks and low reprojection error is likely to be accurate, since only good camera poses will throw low reprojection error over multiple views of a given 3D point.

In presence of noise, the global reprojection error does not suffice as a measurement of the quality of a SfM system. Very well localised cameras can throw a high reprojection error if the views are contaminated with noise. Our evaluation takes this aspect into account and comparative analyses of 3D points and their projections, along with the camera pose produced by our SfM system have been studied and discussed.

It should be noted that for a given choice of parameters of our SfM system and machine precision the results yielded by our implementation does not vary with different executions, both in the single case and the multiple case. Therefore, we have not shown the variance of the different results shown in this chapter since it is null. This conclusion is plausible since there is not random operation during the workflow of our SfM system. We have observed that the optimal setup of parameters is practically invariant to the type of experiment or machine, and therefore we have deemed unnecessary to perform a quantitative sensitivity analysis.

We also evaluate the performance of our feature tracking system. This has been done in a similar way as with noise, but a study on feature track lengths and feature track histograms has been performed in Section 3.6, and compared the results given by our system and state of the art systems.

The implementations chosen for comparison are Changchang (2011) and AgiSoft (2014), two state of the art implementations. AgiSoft (2014) is a commercial package (version 1.1.0) from AgiSoft LLC, used in other research works (Verhoeven (2011); Verhoeven et al. (2012)). The software Changchang (2011) is a renowned GUI application for 3D reconstruction using SfM (Changchang (2013); Changchang et al. (2011)).

Additionally, we can evaluate our system with clean, noise-free data-sets, so that it can be compared against Changchang (2011) and AgiSoft (2014) by observing their performance on bench-mark data-sets. This has been done in Section 5.2.4.

One important element in the evaluation of our noise filters and feature tracking system are histograms of feature tracks, so the number of *bundles* with different amount of views can be visualised. This type of histograms is a measurement of the efficiency

of the feature tracking system of a given SfM process. In addition, it indicates the robustness of a system against noise.

One of the claims of this work is its robustness against ill-configurations created by the omni-directional motion of the mobile platforms studied. This is evaluated by conducting experiments with critical motions, such as sideways and diagonal motion (*pipeline* sequence), rotations around a point placed in front of the robot (*turntable* sequence) and near-pure rotations (*pipeline* sequence).

Multiple Reconstruction

In the evaluation of the multiple reconstruction performed by our system we have evaluated the RMS of the reprojection error and confirmed that loop-closings detected are correct. Generally speaking, we have followed the methodology for results evaluation described in contemporary works in this field (Kim et al. (2010); Özkucur and Akin (2010); Reid et al. (2013); Riazuelo et al. (2014), where the merged reconstructions are visualised.

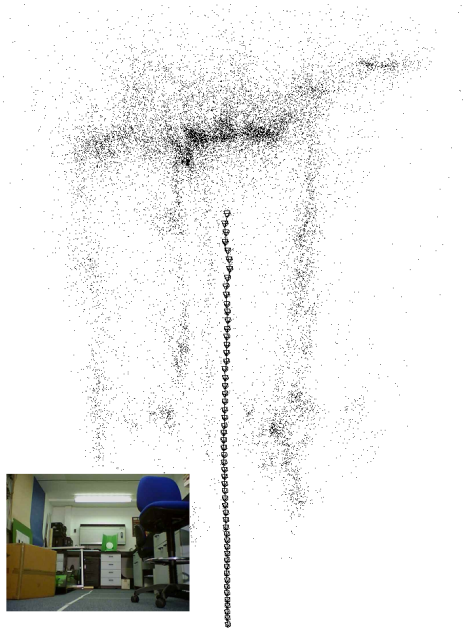
Now that the methodology for the evaluation has been established the results of our system will be shown.

5.2 Single Case

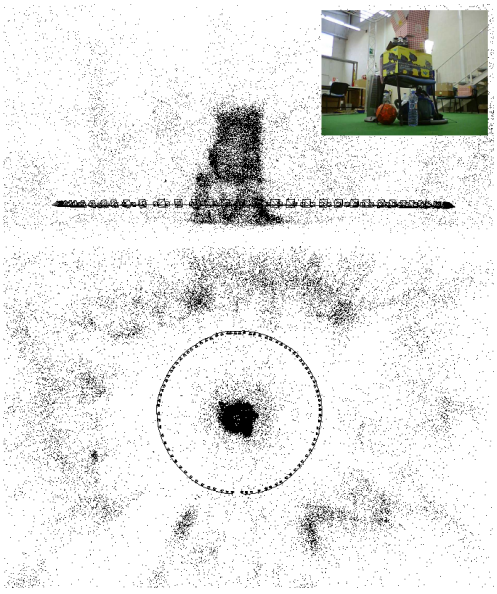
This section discusses the results achieved in the sequential SfM process described in Chapter 3. First we evaluate the 3D reconstructions given by our system. The validation of our system is then done by comparing it with Changchang (2011) and AgiSoft (2014). Afterwards the performance of our SfM process is evaluated in benchmark data-sets and compared with Changchang (2011) and AgiSoft (2014). Finally the robustness of our system against ill-configurations is shown, in comparison with Changchang (2011) and AgiSoft (2014), along with a ground-truth validation.

Since this work has been applied on low-budget mobile platforms, the experiments conducted are an important aspect of it. Here we show 4 experiments from different environments. These experiments are intended to cover enough scenarios as to properly evaluate the performance of our implementation.

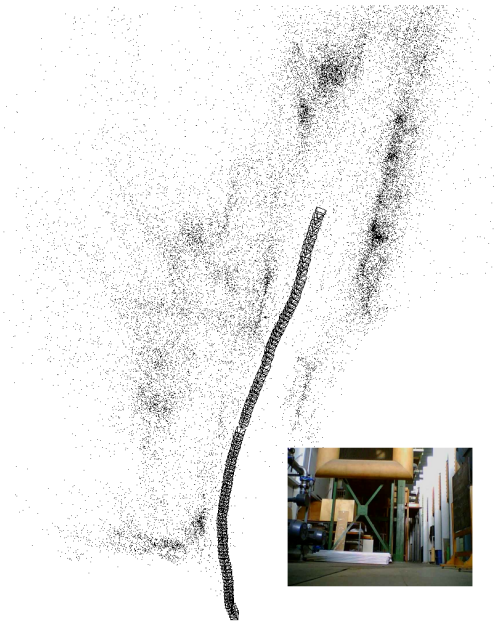
Figure 5.1: Four experiments conducted. From left to right, samples of *visionlab*, *turntable*, *industrialArea* and *pipeline* sequences.



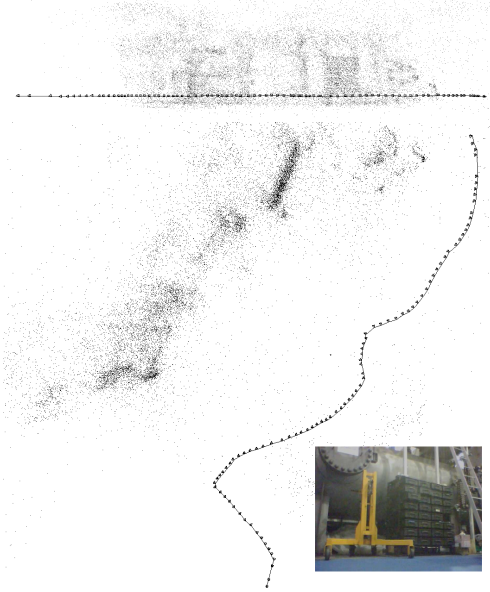
(a) Camera poses and point cloud of the *visionlab* sequence.



(b) Camera poses and point cloud of the *turntable* sequence. The top picture shows that the circle created by the cameras is perfectly planar.



(c) Camera poses and point cloud of the *industrialArea* sequence.



(d) Camera poses and point cloud of the *pipeline* sequence. The top picture shows a side point of view, where the planar nature of the motion is highlighted.

Figure 5.2: Camera poses and 3D structures of the four sequences *pipeline*, *turntable*, *visionlab* and *industrialArea*.

5.2.1 Experiments

In order to validate the SfM system developed in this work we considered the evaluation of four characteristic experiments. We have chosen four because we reckon that with these experiments the most representative types of motion of the Rovio are covered. The results yielded by these experiments allow us to measure the efficiency and performance of our system. Specifically, these experiments are: a transition over a laboratory of vision, a turntable sequence, a motion along an industrial area, and another one along a pipeline system. Fig. 5.1 shows a sample of each sequence.

The first experiment, *visionlab*, is composed of 55 images. Here the robot takes an approximately straight path until it reaches the wall of the laboratory.

The second experiment, *turntable*, is a sequence of 87 images taken in circle around several objects piled up in the centre of the circle. Here the platform realises motions that only omnidirectional robots can perform, i.e. it moves sideways and rotates around the centre of the scene at the same time.

The third experiment, *industrialArea*, has been taken in an area dominated by industrialised items. It is made up of 98 images. The robot takes a slight curve as it goes forward passing along different elements.

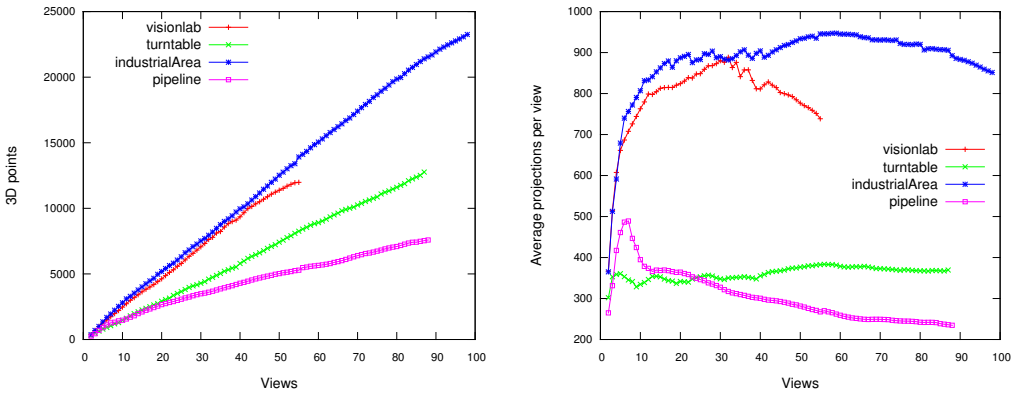
The fourth experiment, *pipeline*, has been made in an environment plenty of tubes and cylindrical elements. The platform describes a long path where the platform realises various omni-directional motions. The sequence takes 88 images. This sequence contains the most challenging types of motion. The platform moves diagonally, laterally (while keeping the orientation of the camera), performs rotations while moving forward, realises abrupt changes of directions, etc. Most of these manoeuvres create ill-configurations (see Section 2.6.4). We tackle this aspect of the reconstructions in Section 5.2.5.

All these experiments have been conducted using SURF detectors and descriptors.

The experiments have been run on an Intel i5 -3317U at 1.7GHz, 4 GB RAM. As a first result, Fig. 5.2 shows the camera poses and 3D structure of each sequence.

5.2.2 Evaluation of the System

This section evaluates the SfM process over the 4 sequences exposed in Section 5.2.1. Here we evaluate the performance of our system according to the methodology described in Section 5.1.



(a) Evolution of the 3D points in the structure with the number of views, on the experiments referenced.

(b) Evolution of the projections per view with the number of views, on the experiments referenced.

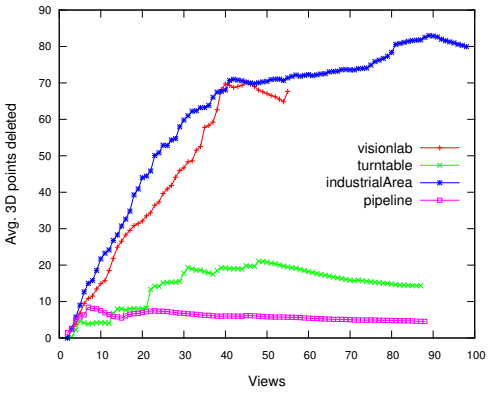
Figure 5.3: Structure information on the data-sets referenced

Fig. 5.3 shows statistical information about the data-sets referenced. Note that the sequences where the robot moves predominantly forward (*visionlab* and *industrialArea*) the acquisition ratio of 3D points (Fig. 5.3a) is higher than the sequences where the robot moves mainly sideways (*pipeline* and *turntable*), since in the former the field of view changes more gradually. In all cases it is noticeable that the acquisition rate is mainly linear.

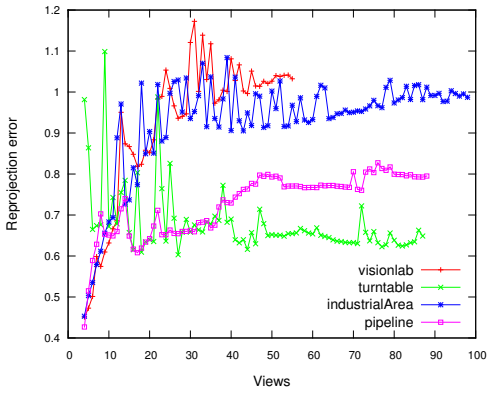
Similar behaviour is observed in Fig. 5.3b, where the average projections are stabilised at higher rate in the sequences *visionlab* and *industrialArea* than in the sequences *pipeline* and *turntable*. In fact, Fig. 5.3b can be seen as a sort of derivative of Fig. 5.3a with respect to the views.

One way to measure the accuracy of the 3D structure is to evaluate how many 3D points are discarded by the reprojection error filters (see Section 3.8.2). This measurement is shown in Fig. 5.4a. The evolution of the reprojection error per view with the number of views is shown in Fig. 5.4b. Again we observe a similar pattern as in Fig. 5.3. The reason for this is that in lateral motion the camera poses are more rigidly fixed since the configuration of the cameras generates less uncertainty during the triangulation step, as shown in Fig. 5.6. More precise camera poses only result in better computed 3D structure and therefore less outliers and reprojection error.

The type of motion in each sequence also explains the different behaviour of the data from Fig. 5.3 and Fig. 5.4. In a lateral motion the epipolar lines tend to be horizontal,



(a) Evolution of the 3D points deleted per view with the number of views, on the experiments referenced.



(b) Evolution of the average reprojection error with the number of views, on the experiments referenced.

Figure 5.4: Statistical information on the data-sets referenced

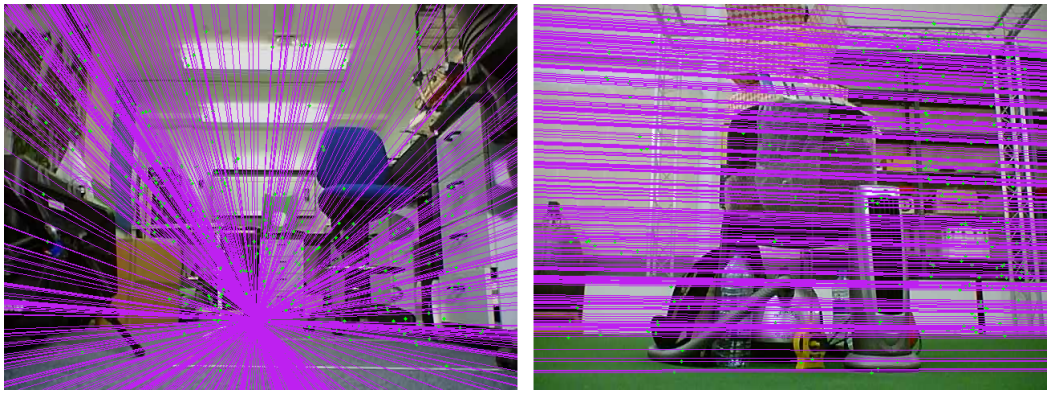


Figure 5.5: Epipolar lines in *visionlab* (left) and *turntable* (right) sequences.

as the epipoles are placed far away from the images, as shown on the right image of Fig. 5.5. In a forward motion, on the other hand, the epipoles are usually in the image and the epipolar lines create rays passing through the epipoles. This is shown on the left image of Fig. 5.5. The different configurations of the epipolar lines affect the matching selection process (during RANSAC). This can be seen with the help of Fig. 5.5. The distance between a given feature and its corresponding epipolar line (the epipolar distance) varies differently with the orientation of the epipolar line, according to the type of motion. In a forward motion (see Fig. 5.5, left) a slight variation in an epipolar line (which ultimately means a rotation around the epipole) can provoke a great increase in the epipolar distance if the feature is far enough from the epipole. This phenomenon does not occur in lateral motions, where the epipoles are usually far away from the image (see Fig. 5.5, right). An epipolar line will hardly change its orientation if it turns around an epipole. This produces more stable reconstructions on sequences with lateral motion and consequently less reprojection error. Additionally, the precision of the triangulation method is more sensitive to forward camera motion than lateral camera motion, as Fig. 5.6 shows.

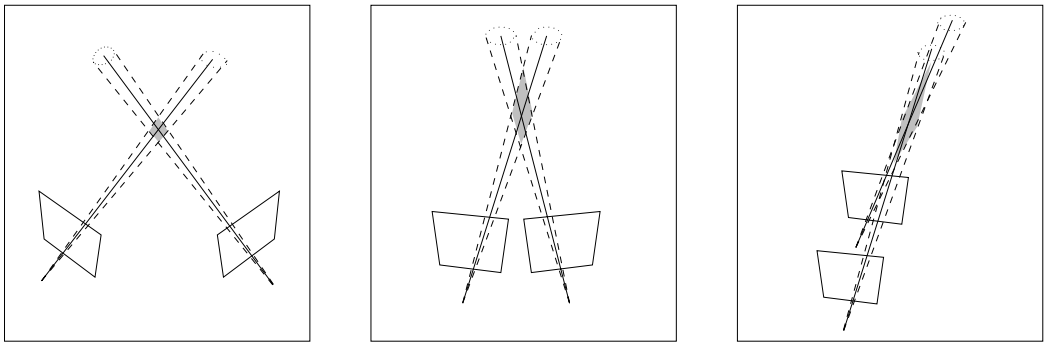


Figure 5.6: The error in the measurement of a view affects differently in the uncertainty area of the 3D point reconstruction, depending on the length of the baseline, and this in turn affects the reprojection error. Source: Hartley and Zisserman (2004).

Post-Process

The effect of post-processing the 3D structure obtained during the SfM process has already been discussed in Section 3.9. Here we delve into it in more detail, and offer a comparison between the results given by the SfM process and the post-process in the *visionlab* and *turntable* sequences. These sequences have been chosen as representative

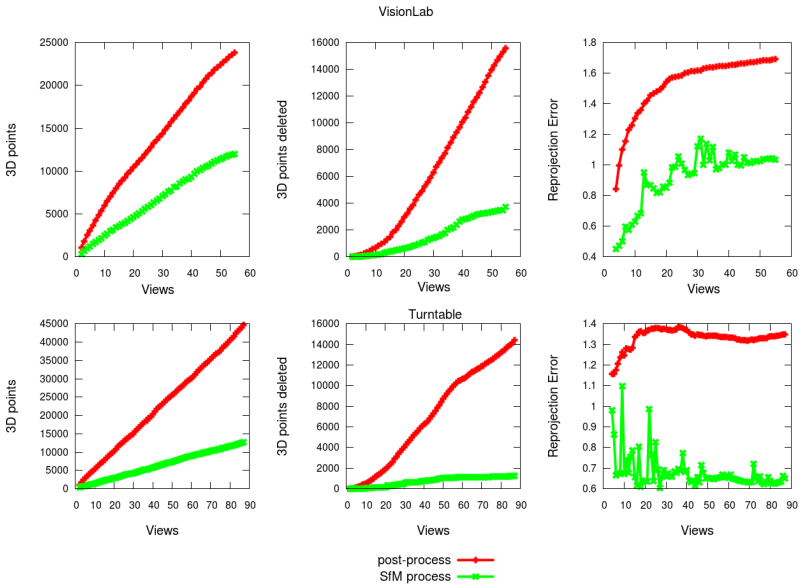


Figure 5.7: Comparison between SfM process and post-process performance on the *visionlab* and *turntable* sequences.

of experiments where the platform performs a mainly straight motion (*visionlab*) and a lateral motion (*turntable*).

The post-process that is implemented in our system affects primarily to the number of matching features that are generated. Since now all the matching filters are relaxed, many more matching features are populated across images, and as consequence more feature tracks and 3D points are generated. However, in order to maintain the quality of the reconstructed structure, the reprojection error filters are as strict as during the SfM process, which produces as a result the removal of many more 3D points and a slight increase of the total reprojection error. Fig. 5.7 shows a comparison between the two phases. The behaviour of the data is similar to Fig. 5.4 and Fig. 5.3. One interesting point to note is that in the *visionlab* sequence the reprojection error in the post-process phase evolves logarithmically, since now the camera poses are fixed and the only source of error are noisy points. We can see signs of this algorithmic pattern in the reprojection error of the SfM process, although it presents local peaks of high reprojection error since at this stage camera poses are simultaneously been optimised along with the structure. The peaks appear when at some point the configuration of the camera poses are suboptimal. The reprojection error in the *turntable* sequence tends to

be more constant, but the phenomenon of the peaks can still be appreciated.

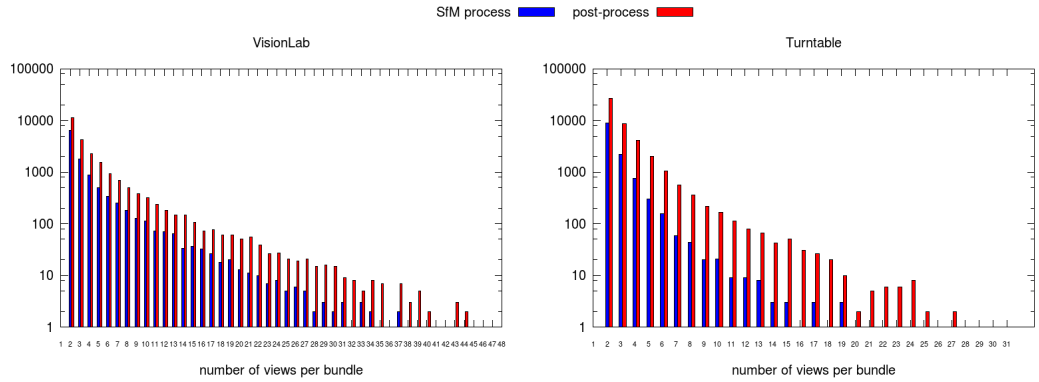


Figure 5.8: Comparison of feature track histograms between the SfM process and the post-process in the *visionlab* and *turntable* sequences.

Fig. 5.8 shows how feature track lengths have been increased at the post-process phase. The number of bundles with less than 5 views is tripled after post-process. This ratio increases on bundles that contain more views.

Fig. 5.8 also serves to illustrate the effect of noise on the system. Due to noise, the majority of bundles (and therefore the majority of 3D points) have short feature tracks. Table 5.1 shows the percentage of projections of the whole 3D structure covered by bundles with different length, differentiating between bundles with no more than four views and bundles with more than four views.

% of views seen by bundles over the total number of views				
	bundles of 2, 3 or 4 views		bundles of more than 4 views	
	visionlab	*turntable*	*visionlab*	*turntable*
SfM process	82.14	94.95	17.86	5.05
Post-process	75.53	89.18	24.47	10.81

Table 5.1: Percentage of views seen by bundles on *visionlab* and *turntable* sequence. More than 75% of the total projections of the 3D structures are covered by bundles with length no superior to 4 views.

It is clear from Table 5.1 how the type of motion taken by the platform affects the reconstruction outcome. The proportion of bundles with more than 4 views in the *visionlab* sequence is roughly three times as big as in the *turntable* sequence, given the straight path taken in *visionlab*, which creates bigger overlaps between images.

The fact that our system manages to obtain 3D reconstructions with low reprojection error shows the effectiveness of the feature tracker system devised. We manage to obtain matches which, even though most of them create short feature tracks, are of sufficient quality as to generate accurate 3D structures and camera poses. It is clear that longer feature tracks produce higher precision in 3D points locations, and in standard imagery is easy to find 4 or 5 views per 3D point. In the *visionlab* sequence an average 3D point is seen by 3.53 views after the SfM process and by 4.12 after post-processing.
Fig. 5.9 shows graphically the difference between the 3D structure of *visionlab* sequence before the application of post-processing and afterwards.

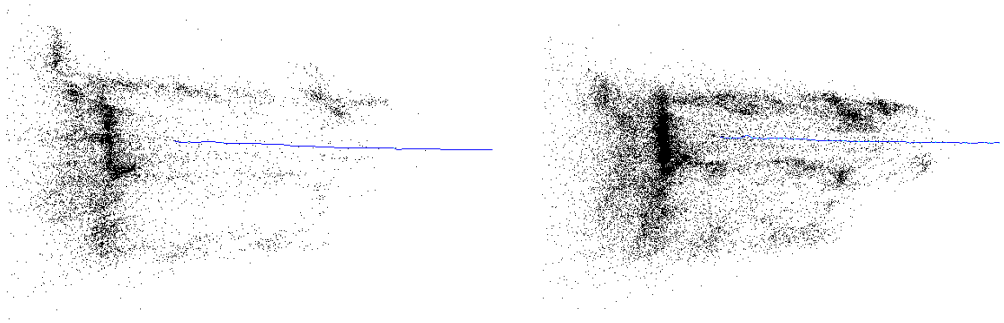


Figure 5.9: Graphical comparison on the *visionlab* sequence between the point cloud before applying post-process (left) and afterwards (right)

Computation Times

Computation times in SfM is an important aspect to consider. Fig. 5.10 shows the computational times in each of the steps for the SfM process and for the post-process phases (see Fig. 3.1 and Fig. 3.18), on the *visionlab* sequence.
In terms of computation the SfM process is mainly dominated by the recursive matching process (within which the feature tracking system is run) and, to a lesser extent, BA. This is mainly due to the noise presence in the sequence, which forces a big optimisation effort to both feature tracking system and BA algorithms. In addition, given the recursive nature of the feature tracking system, the execution time of the matching step tends to increase linearly with the number of cameras. As an average, the combination of steps given by receiving the image, preprocessing, feature detection and description (with the noise filters within), reprojection error filters, resection and triangulation contribute with 17.7% of execution time of an image. As average, it takes 7 seconds to

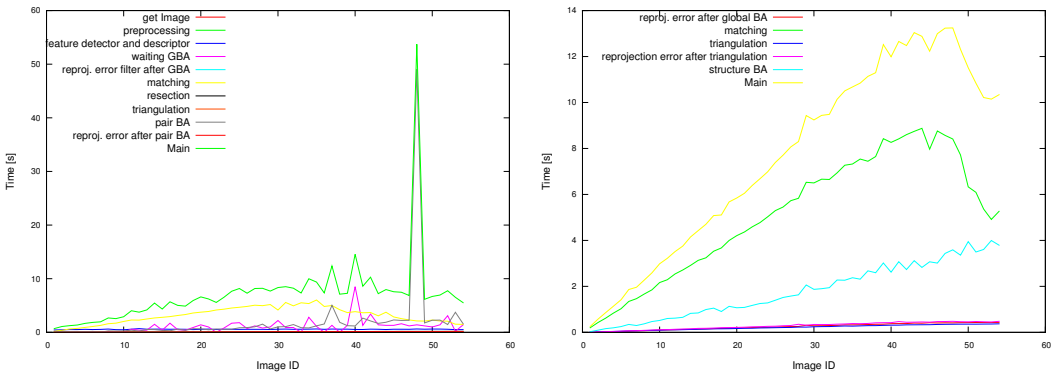


Figure 5.10: Computational times in SfM process (left) and post-process (right) phases. The term "Main" is the aggregation of all the steps of the process.

process an image. There are peaks of times scattered along the sequence (the most prominent at image 49) but they are caused by the stop criteria of the BA methods rather than a bad camera pose estimate (this specifically happens at image 49). There are four criteria for SBA to stop an optimisation (Lourakis and Argyros (2009)):-

- The magnitude of the gradient takes a value smaller than a given threshold ε_1;
- The relative variation of the solution is smaller than a threshold which involves a parameter ε_2;
- The value of the residual $(\overline{\boldsymbol{x}} - \boldsymbol{x})$ drops below a threshold ε_3;
- The relative reduction in the value of the residual drops below threshold ε_4;
- The algorithm iterates a given number of times.

These parameters are set to very conservative values in these experiments. The number of maximum iterations set for pair-wise BA optimisation is 1000, and the thresholds are set up to precision machine[2]. If, for example, the iteration parameter is lowered to 100 then computation times are noticeably decreased without compromising accuracy in many cases. This is precisely the case in image 49, where the BA algorithm keeps iterating for more than 50 seconds. The other parameters ε_1, ε_2, ε_3 and ε_4 can be adjusted to achieve lower times according to each sequence.
In the graph for the SfM process phase in Fig. 5.10 there is a term, "waiting GBA" which occasionally reaches significant values over the total spent time. This term reflects the

[2]This configuration has been chosen because in this work the criterion of robustness and precision of the system has prevailed over computation time optimisation.

interaction between the main thread and the thread of the global BA (see Section. 3.8.2). Since both threads access the same data (3D points and camera poses), the thread for the global BA that was triggered when image I_n was processed should be finished before the feature tracking method of image I_{n+1} process starts, as this method works with bundles, which are linked to 3D points. Depending on the necessity of refining the 3D structure and camera poses, sometimes the process of image I_{n+1} may have to wait for global BA to finish in order to resume the process execution. As with the local BA algorithm, the parameters for the global BA on these experiments are very conservative and can be optimised if an improvement on computation times is required.

The computation times for the global BA thread are shown in Fig. 5.11. It can be appreciated that it follows the same patten as the term "waiting GBA" in Fig. 5.10 (left). Data acquisition time (copying the data from the main thread) and data extraction time (pouring the data on the main thread) are negligible compared with the BA process. In average the global BA thread takes 1.6 seconds to optimise each image.

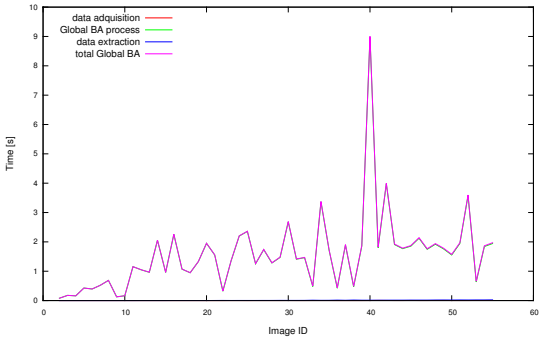


Figure 5.11: Computational times in the global BA thread. Here "Main" practically collides with "Global BA process".

Since in the post-process camera poses are fixed, the evolution of the times needed for the process of each image is much smoother than in SfM process, where the evolution is irregular as the cameras are to be refined. This smooth evolution is shown in Fig. 5.10 (right). Now there are many more features to process and the time not used in finding features and refining the cameras is spent in matching recursively and refining the position of the 3D points generated. Here again the main contributing steps are matching and BA. An average image is processed in 7.6 seconds in this phase. Overall, the *visionlab* sequence was processed in 13 minutes and 26 seconds.

Figure 5.12: 3D surface of the object placed in the centre around which the platform in *turntable* sequence revolves.

The matching step ratio slightly drops towards the end of the sequence, both in SfM process and post-process stages. The reason for this behaviour is that the platform is approaching the wall of the laboratory and no new feature points are detected.

Rendered Surfaces Reconstructions

The number of 3D points detected during post-process enables the rendering of surface reconstructions. We have obtained these 3D reconstructed surfaces by the application of commonplace statistical methods, as described in Section 3.9.1. Here we show results from *turntable* and *industrialArea* sequences which, along with Fig. 3.19, help visualise how the reconstructed surfaces are obtained.

As a comparison with the rendered 3D surface of the *turntable* sequence, Fig. 5.13 shows the point cloud generated by the post-process phase.

Fig. 5.12 and Fig. 5.14 show the 3D surface for the *turntable* and *industrialArea* sequences, respectively. In Fig. 5.12, despite the texture of the objects being fine-grained, the smoothing surface and rendering algorithms (specifically, Poisson method, by Kazhdan et al. (2006)) applied on the structure do not differentiate between objects and tend to create a unified surface out of the structure. In addition, edges and corners are smoothed, which makes the 3D visualisation lose sharpness. Despite these limitations, surface rendering is a useful tool for 3D Mapping, as it can be appreciated in Fig. 5.14.

The key features of the scene are clearly identifiable, and the 3D map is smooth enough to make possible for the robot to establish paths in the environment.
Different aspects of the performance of our system have been discussed in this section. We can now evaluate it with respect to state of the art systems present in the literature, so a more complete evaluation is performed.

Figure 5.13: Point cloud of the *turntable* sequence

5.2.3 Comparison with State of the Art Systems

This section validates the system devised. This validation is performed by comparing, according to Section 5.1, our results with results given by Changchang (2011) and AgiSoft (2014). The datasets chosen for this validation are *visionlab* and *turntable*, since each one represents a type of motion: forward and sideways, respectively.
The evaluation system that calculates the reprojection error has been taken from the library QVision (Rodríguez López et al. (2012)), which is capable of reading formats of 3D reconstructions files used by SBA (which our system generates), bundler (from Snavely et al. (2006)) and the extension for recontruction files that Changchang (2011) and AgiSoft (2014) produce as output. By using Rodríguez López et al. (2012) we ensure a fair evaluation of the reprojection error for all the systems.
Table 5.2 compares the final reprojection error, the total projections and the total 3D points generated by each of the systems evaluated.

[3]The reprojection error given by our evaluation system on the bundler file produced by AgiSoft (2014) is 40.57, but we have reasons to assume that this is due to a bad configuration of the bundler file generated by AgiSoft (2014). Therefore we have optimised with SBA the structure given by AgiSoft (2014) to eliminate the effect of the bundler file, giving as a result 1.32.

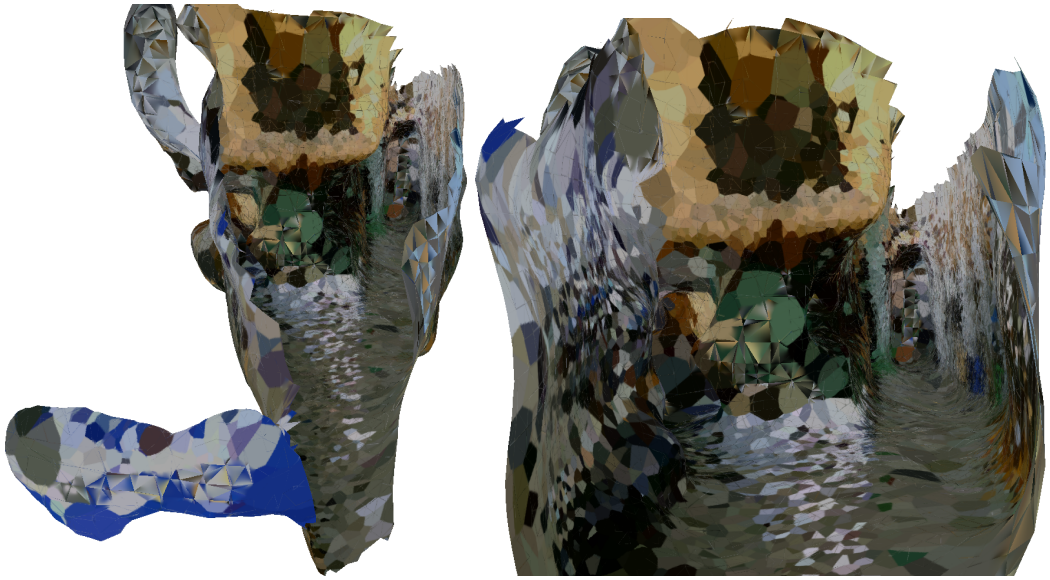


Figure 5.14: 3D surface of the area covered by the platform in the *industrialArea* sequence.

sequence visionlab				
Method	Final reproj. error	Total Projections	Total 3D points	3D points Deleted
Our SfM process	1.64	98888	23865	15481
Changchang (2011)	3.75	24370	2643	no inform.
AgiSoft (2014)	1.32[3]	20325	6273	no inform.

Table 5.2: Comparison of results between our system, Changchang (2011) and AgiSoft (2014) on the data-set *visionlab*.

Even though our system gives a slightly greater reprojection error than AgiSoft (2014), it outperforms AgiSoft (2014) in terms of 3D points found and projections (i.e. the quantity of 3D scene information recovered given the same scene image samples as input). Changchang (2011) performs poorly in this sequence, as its reprojection error shows. Moreover, Fig. 5.15 shows that its cameras are not well aligned, being the first ones very distant between each other. This inter-camera distance is decreased as the sequence goes on.

Table 5.3 shows the ratio of projections with respect to the number of points and cameras in each system. It is usually a sign of good quality matches when a reconstruction is

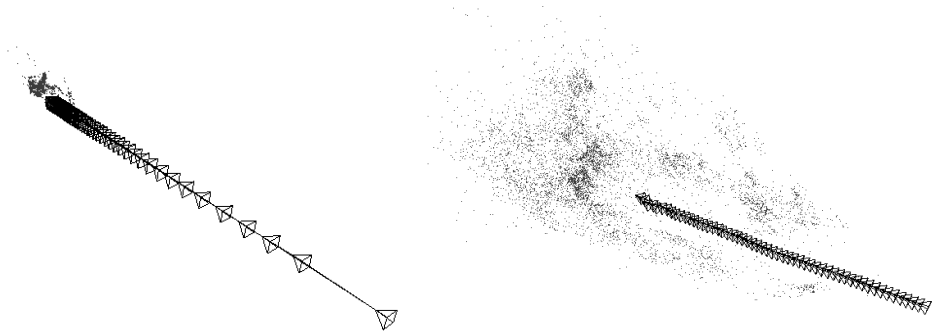


Figure 5.15: Camera poses and point cloud generated by Changchang (2011) (left) and AgiSoft (2014) (right) on the data-set *visionlab*

attained with a few projections per point (Chang and Hebert (2002); Zhang et al. (2010)), because if there is no need of many cameras viewing a point in order to fix it with a low reprojection error, it means that the selected features are precise and free of noise. AgiSoft (2014) manages to obtain a lower reprojection error, but our system finds many more projections per camera. Changchang (2011), however, finds twice as many projections per point with high reprojection error, which leads us to assume that Changchang (2011) does not have any strategy for feature filtering.

System	projections/point	projections/camera
Our SfM process	4.14	1797
Changchang (2011)	9.22	443
AgiSoft (2014)	3.24	369

Table 5.3: Ratios regarding the projections generated on *visionlab* sequence.

Table. 5.4 shows the comparison between the systems evaluated on the data-set *turntable*. Here Changchang (2011) and AgiSoft (2014) outperform our system in terms of accuracy of the structure with respect to the cameras. However, our system still finds many more features and points, providing twice as many as the others. In Section. 5.2.5 we show how the camera poses estimated by our system adjust to the ground-truth value. As with the *visionlab* sequence, Table 5.5 compares indices relative to the number of projections found by each system. Here we can see why the reprojection error given by

[4]Similarly to Table. 5.2, this value is obtained by optimising the structure with SBA. Our evaluation system gives an initial reprojection error of 24.86.

sequence turntable				
Method	Final reproj. error	Total Projections	Total 3D points	3D points Deleted
Our SfM process	1.34	130070	44701	14422
Changchang (2011)	0.77	77703	18988	no inform.
AgiSoft (2014)	1.16[4]	77191	21824	no inform.

Table 5.4: Comparison of results between our system, Changchang (2011) and AgiSoft (2014) on the data-set *turntable*.

our system is higher, provided the low ratio of projections per point. The better results obtained here by Changchang (2011) suggest that Changchang (2011) deals better with image configurations similar to stereo configurations.

System	projections/point	projections/camera
Our SfM process	2.91	1495
Changchang (2011)	4.09	893
AgiSoft (2014)	3.53	887

Table 5.5: Ratios regarding the projections generated on *turntable* sequence.

The results in the other sequences are similar: Changchang (2011) is the system with most projections per point, our system detects most projections per camera, with the reprojection error of all the systems being of the same order (Table 5.6). It should be noted that the reprojection error tends to increase with the number of 3D points. The reprojection error given by our system before the post-process phase (and therefore when the camera poses are already fixed) is lower than the reprojection error given by Changchang (2011) and AgiSoft (2014), as shown in Table 5.6[5]. In the sequences *visionlab* and *industrialArea* the projections and 3D points obtained by our system before post-process already double the results from Changchang (2011) and AgiSoft (2014). In the sequences *pipeline* and *turntable* our system before post-process has found as many as the other systems, but with much less number of projections. This indicates how well our system behaves when the platform performs a forward move, which is the usual motion taken by a mobile robot.

[5]Here we have followed the procedure of previous tables for the estimation of the reprojection error thrown by AgiSoft (2014).

Reprojection Error				
System	visionlab	turntable	industrialArea	pipeline
Before post-process	1.02	0.62	0.98	0.79
After post-process	1.64	1.34	1.62	1.57
Changchang (2011)	3.75	0.77	1.14	1.36
AgiSoft (2014)	1.32	1.16	1.60	1.53

Table 5.6: Comparison of the reprojection error given by our system and the state of the art systems on the sequences studied.

This section has compared our system with state of the art systems on data-sets taken by the mobile platform of study. For a complete comparative, it is adequate to validate our system on popular benchmarks present in the literature.

5.2.4 Validation against Benchmark Data-Sets

This section shows the performance of our system on a benchmark present in the literature. The data-set chosen is Leuven castle (Pollefeys (2004)), a sequence of 28 images, where the camera describes a sideways motion around this castle.
Table 5.7 shows the performance of the systems tested. The pattern followed is the same as with the sequences of the mobile platform used.

System	3D points	Projections	Reproj. Error
Our SfM process (Sections 3.2 - 3.8)	17660	55305	0.40
Our post-process (Section 3.9)	37032	152888	0.83
Changchang (2011)[6]	13015	85720	0.43
AgiSoft (2014)	1990	16547	0.29

Table 5.7: Comparison of our system and state of the art systems on the Leuven castle data-set.

Fig. 5.16 shows the reconstructions obtained by our system, Changchang (2011) and AgiSoft (2014).
Up this point the validation of our system has relied in the quality of the structure. However, a very important aspect in SfM applied to mobile robots is the positioning of the camera poses. Therefore, an evaluation and comparison with respect to ground-truth is necessary to properly validate our system.

[6] Changchang (2011) only can align 25 cameras in this sequence.

Figure 5.16: Reconstruction of Leuven Castle made by our system (left), Changchang (2011) (centre) and AgiSoft (2014) (right).

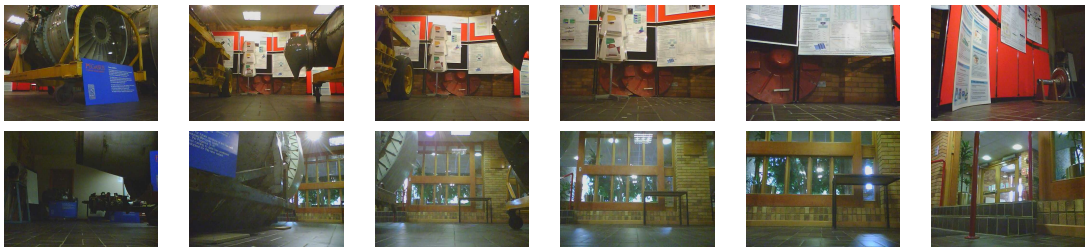


Figure 5.17: Samples of the *engineRoom0* (first row) and *engineRoom1* (second row) data-sets.

5.2.5 Evaluation against Ground-Truth

This section compares the accuracy obtained on the camera poses by our system with the ground-truth, and in one experiment we compare it along with the motion given by Changchang (2011) and AgiSoft (2014). In addition, the robustness of our system against omnidirectional motions is shown and compared to the response of other systems.

The sequences used for ground-truth validation are called *engineRoom0* and *engineRoom1*. These sequences are taken in the engine display room of the Whittle building of Cranfield University. The sequence *engineRoom0* is made up of 75 images, in which the robot describes an 'S' going in between of two turbine engines. The sequence *engineRoom1* has 72 images, and the path described is similar, taken along a different area of the room. Fig. 5.17 shows some samples taken from these sequences.

The ground-truth was set by labeling on the floor the points where the robot should take the pictures, and measuring the coordinates (x, y) of each point. In this regard, the precision of the ground-truth can be assumed to be within 1 cm.

Fig. 5.18 shows the camera poses of our system[7], Changchang (2011), AgiSoft (2014) and

[7]In these sequences, the feature detector and descriptor used is SIFT.

the ground-truth on both sessions. Changchang (2011) only is able to reconstruct the cameras $[12-75]$ of *engineRoom0* and the cameras $[18-62]$ of *engineRoom1*. Fig. 5.18 shows the estimated tracks on the plane $X-Y$. The differences of the systems tested with the ground-truth on the planes $X-Z$ and $Y-Z$ are negligible.

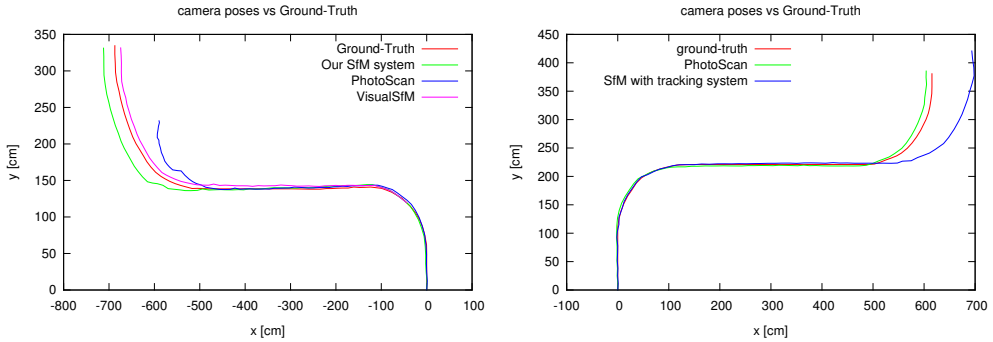


Figure 5.18: Comparison of the systems evaluated with ground-truth on the datasets *engineRoom0* (left) and *engineRoom1* (right). Note that on sequence *engineRoom1* Changchang (2011) is not included, due to the lack of cameras reconstructed by this system.

In *engineRoom0* all the systems (except for Changchang (2011), which does not reconstruct the 11 first cameras) keep a precise odometry until the second curve of the sequence, where they diverge. Fig. 5.19 shows a detail of the sequence.

It can be noticed that in sequence *engineRoom0* our system outperforms AgiSoft (2014). Changchang (2011) adjusts better to the ground-truth but it is not capable of reconstructing the 11 first cameras. However, in sequence *engineRoom1* AgiSoft (2014) adjusts to the ground-truth more faithfully than our system, which enlarges the scale of the inter-camera distances.

We have also validated the *turntable* sequence against ground-truth. This experiment was taken in a similar way as the *engineRoom* sequences. A perfect circle was created with the aid of a string and labels were stuck along the perimeter, where the platform was placed when taking pictures. Therefore, the shape of the loop drawn by the camera poses should be compared with an ideal circle. This comparison is shown in Fig 5.20, where the ground-truth circle is in red.

The optimisation of our system for omnidirectional motion has been shown along this chapter with the discussed results, specifically in Section 5.2.3. One sequence where the robustness of our system is demonstrated specially is *pipeline*, in which the platform

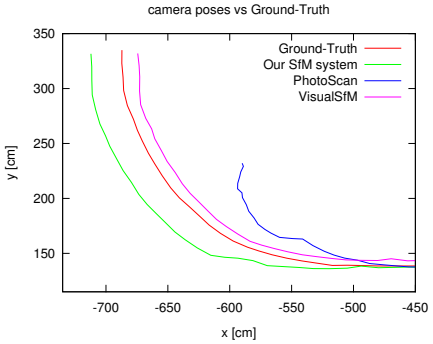


Figure 5.19: Detail of the comparison with Ground-Truth on *engineRoom0* sequence.

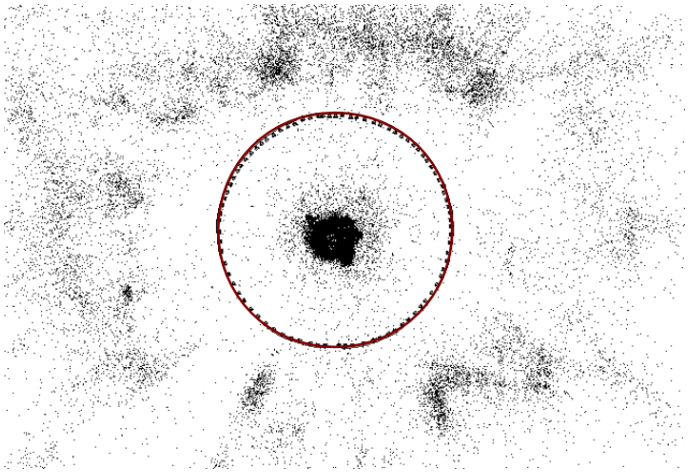


Figure 5.20: Validation of *turntable* sequence against ground-truth

realises various motions typical of omnidirectional robots. Fig. 5.21 shows two sub-sequences of the pipeline sequence. It can be appreciated in the sub-sequence on the left that the robot moves forward as it changes its orientation. Changchang (2011) can not reconstruct these camera poses within its reconstruction model. AgiSoft (2014) does locate correctly the cameras, but with a less dense 3D structure. On the right sub-sequence of Fig. 5.21 the platform performs a rotation as it moves sideways, and then it changes abruptly of direction. Here Changchang (2011) estimates the camera poses, but again with less 3D points.

For comparison purposes the camera poses reconstructed by Changchang (2011) and AgiSoft (2014) in the *pipeline* sequence are respectively shown in Fig. 5.22a and 5.22b. Note that in Fig. 5.22a the first 8 cameras of the sequence have not been reconstructed.

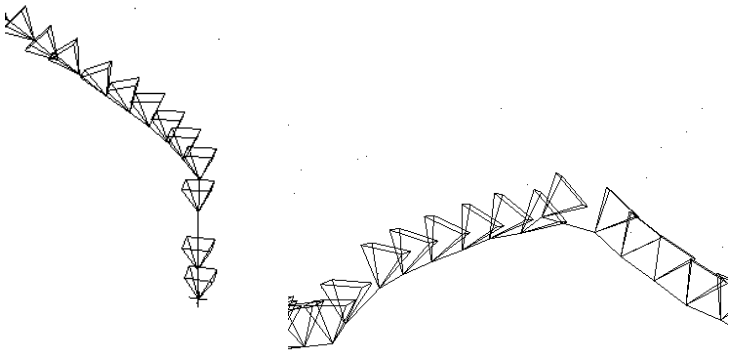
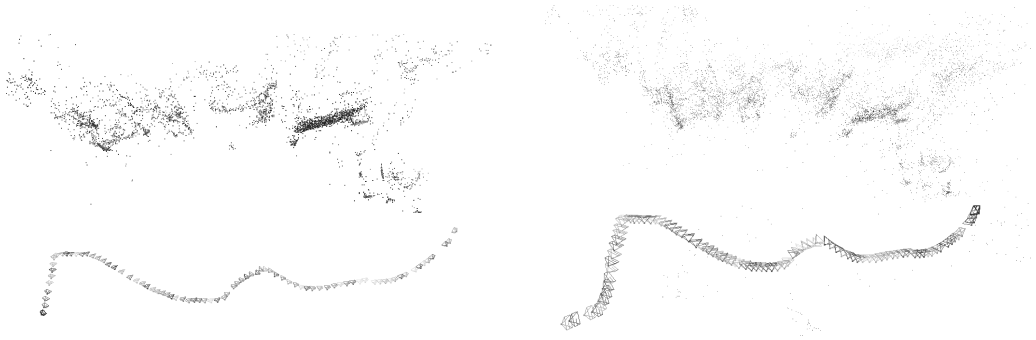


Figure 5.21: Omnidirectional motions in *pipeline* sequence.



(a) Camera poses reconstructed by Changchang (2011) in the *pipeline* sequence.

(b) Camera poses reconstructed by AgiSoft (2014) in the *pipeline* sequence.

Figure 5.22: Camera poses reconstructed by state of the art softwares in the *pipeline* sequence.

Compare the results of Fig. 5.22 with Fig. 5.2d.

The single case has been evaluated and validated in this section. With these results we can now show how multiple reconstruction is performed by our system.

5.3 Multiple Case

This section evaluates the results achieved in the multiple reconstruction system described in Chapter 4. This is done by showing the 3D information and camera poses estimated by our multiple reconstruction system on two experiments[8]. These experiments are the *engineRoom* sequence (described in Section 5.2.5) and an experiment

[8]In these experiments the feature detector used has been SIFT, as it is possible to take advantage of the parallelisation of the feature detection step (see Section 4.2).

conducted on the same scenario as the *turntable* sequence, where the robot completes two laps around the objects piled up in the centre. This last experiment will serve as a validation of our system for processing single reconstructions where the platform revisits areas, generating loop-closures.
Apart from showing the merged reconstructions, the estimated loop-closures are compared to the actual ones, and the RMS of the reprojection error for each experiment, along with other data concerning the 3D visual reconstruction, are shown. A validation against ground-truth is performed in the case of the *engineRoom* sequence.

5.3.1 Multi-Session Evaluation

The real overlaps between sequences *engineRoom0* and *engineRoom1*, the detected overlap and the matched correspondences found with the overlap detected (by means of recursive matching) are shown in Table 5.8. These matched correspondences will be all the common information between sessions that our multiple reconstruction system will have in order to find a global reference frame and a global scale for the 3D structures and the camera poses of the two sessions of the experiment.

	Real Overlaps	Detected Overlaps	Matches found by recursive matching
engineRoom0	[70-75]	[74-76]	[74-76] = 73 [73-76] = 63
engineRoom1	[76-86]		[72-76] = 40 [71-76] = 17

Table 5.8: Real overlaps between *engineRoom0* and *engineRoom1*, and detected overlaps. The matched correspondences found with the overlap [74-76] by recursive matching are also shown.

Fig. 5.23 shows the 3D maps generated by each session in the engine display room, and Fig. 5.24 shows the merged 3D map and camera poses of these sessions.
In Fig. 5.23 we can appreciate that each session is reconstructed with different scales. In Fig. 5.24 both sessions have practically the same scale. Our multiple reconstruction system resolves the problem of the scale with only one overlap, which generates barely one hundred common 3D points between both sessions (see Table 5.8). The proportion of common 3D points with respect to the total number of reconstructed 3D points

(more than 200.000, see Table 5.9) is negligible, and shows how efficiently the overlaps are managed by our system.

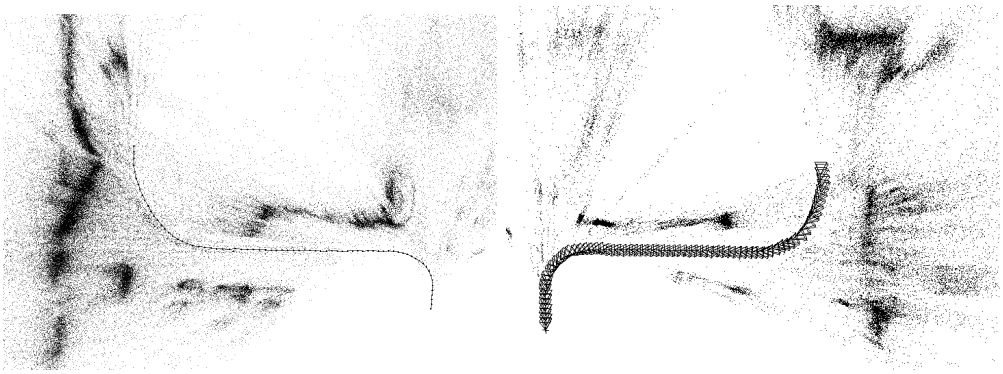


Figure 5.23: camera poses and 3D structure of the *engineRoom0* and *engineRoom1* sequences.

Fig. 5.25 compares the ground-truth camera poses by the estimated ones. Although towards the end of the second session the scale is not maintained and the distances between cameras are greater, the adjustment to the ground-truth is faithful in the rest of the sequence. The second session accumulates the drift created in the first session, but the distance between the reconstructed sessions are the same as in the ground-truth. The behaviour of the camera poses at the end of the second sequence can be explained by the dynamics of the recursive matching. The camera poses of the second session keep the scale of the first session so long as they have some bundle in common with the overlapping camera (in this case, the image 76) during the SfM process. Recursive matching connects the bundles of the image 76 with the subsequent images of session 2 up to the image 90 (image 90 is located in the middle of the first turn). Therefore, from image 90 on, images lose gradually their connection with the overlap, which produces a change in the scale in the last images of session 2. Note that the scale s_2 that minimises the reprojection error in session 2 is different to the optimal scale s_1 in session 1, and therefore camera poses in session 2 tend naturally to s_2 owing to the effect of BA methods, if there is no constraint present.

Table 5.9 shows the reprojection error given by each sequence separately and by the merged map, along with other statistics of the 3D reconstructions.

Figure 5.24: Camera poses and 3D structure of the merged sessions. Since all the cameras are joined by a line, there is a line joining the last camera of the first session and the first camera of the second session

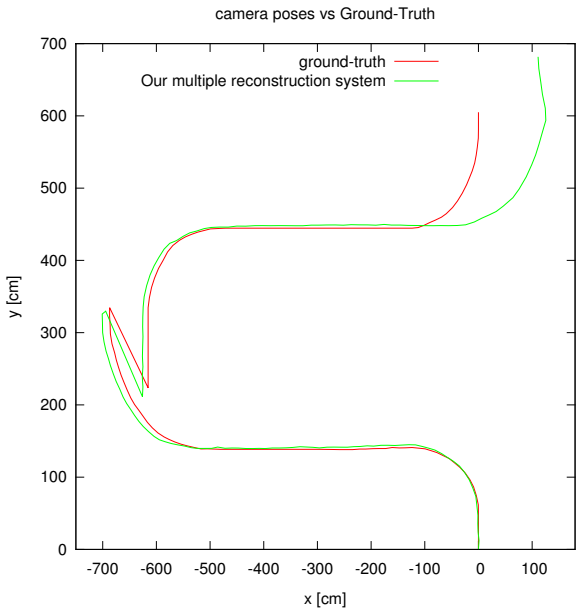


Figure 5.25: Comparison between ground-truth and our results in the merged camera poses of the *engineRoom* experiment.

	Projections	3D points	3D points Deleted	Reproj. Error.
engineRoom0	362756	104379	57957	1.57
engineRoom1	155809	45449	no inform.	1.18
engineRoom merged	688.798	209918	117614	1.52

Table 5.9: 3D structure statistics given by the sequences of the engine display room and of the merged map of them.

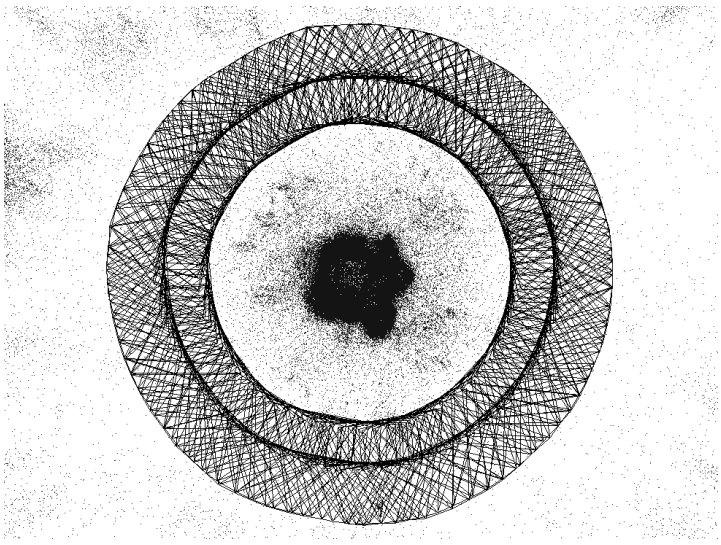


Figure 5.26: Camera poses in the sequence *turntable2*.

5.3.2 Loop-Closings in Single Sessions

The multiple reconstruction system devised in this work can also be used for finding loop-closures in single sessions, as described in Section 4.5. We have proved this by doing an experiment on the same scenario as in the *turntable* sequence. Here the robot realises two complete laps around the objects in the centre, so that multiple loop-closures happen along the 189 images of the data-set.

Similarly to the *engineRoom* sequences, in this experiment every point where the robot should take a picture was labeled on the floor, according to a circular perimeter. Therefore the 94 images of the second lap should overlap with 94 images from the first lap. Our system detects correctly 84 overlaps, incorrectly 8 and fails to assign an overlap on 2 images.

In Fig. 5.26 the estimated camera poses and part of the 3D structure are shown. The

second lap is superimposed over the first lap, as it is expected to happen.

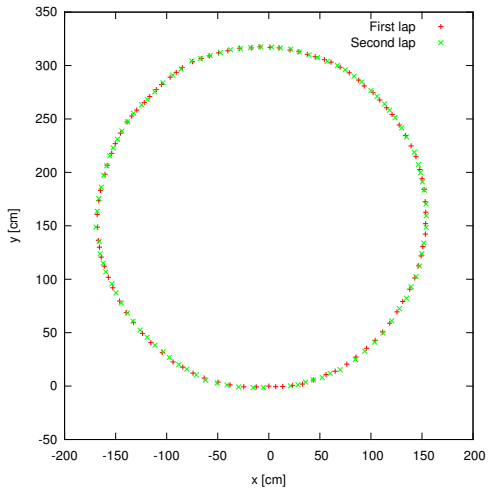


Figure 5.27: Loop closing evaluation in the case $r = 1$.

Fig. 5.27 shows the adjustment of the camera poses on the second lap (in red) to the camera poses in the first lap (green). Ideally, each camera from the second lap should collide with its corresponding camera from the first lap, since the images in the second lap were taken on the same spots as in the first. Note that the first camera of the first lap is at $(0, 0)$ (next to the first camera of the second lap) and that the second lap has 94 cameras, leading one camera of the first lap unmatched.

5.4 Summary

This chapter has evaluated and validated the performance of our system. A methodology of evaluation has been established so that the results can be reproduced and our system can be fairly compared with other softwares.

The single case has been studied over four experiments which intend to cover a range of scenarios and types of motion. Different indices to measure the quality of the reconstructions have been discussed. These indices show how our SfM system overcomes the problem of high noise on JPEG images (Torr and Zisserman (1997)). The performance of the filters devised and the feature tracking system developed is apparent when our system is compared with state of the art softwares (AgiSoft (2014); Changchang (2011)). With similar levels of reprojection error, our system obtains better results in terms of 3D structure and projections.

5. PERFORMANCE EVALUATION

The capability of our system to process ill-configurations created by omnidirectional motion is shown throughout the evaluation of the four experiments exposed. Nevertheless we have shown details of estimated camera poses that specifically correspond to omnidirectional motions.

The reliability of our system on estimating camera poses (and therefore performing visual odometry) has been validated by comparing our results against ground-truth data-sets. In addition, a study on the contribution of the post-process phase, as well as the computational times taken by each stage have been done.

The management of loop-closures found by our multiple reconstruction system has also been evaluated with two representative experiments. We have shown the efficiency of our system in merging different 3D maps and sequences of cameras with minimal overlaps. This result contrasts with the methodology of Zhang et al. (2010), which needs sub-sets of sequences to merge different sequences. Here we show a comparable multiple achievement using lesser constraints on the formulation of the input image set from multiple concurrent robots within the environment against the prior work of Zhang et al. (2010).

Chapter 6

Conclusions

In this chapter the summary and conclusions of this work are detailed. We first list the our contributions and then highlight future developments that can extend the present work.

6.1 Contributions

House-hold mobile devices can take advantage of 3D vision techniques for their navigation and mapping. We believe we have made a contribution towards this goal by completing the two research questions stated in Chapter 1: the development of a SfM system over low-cost omnidirectional motion robot and its extension to a multiple instances in order to obtain distributed 3D reconstruction.

A full SfM process system for low-budget omnidirectional platforms

The first research question, which challenges the accomplishment of SfM on low-quality imagery and inter-image ill-configurations, has been answered with the development and implementation of a SfM pipeline on a low-cost omnidirectional platform.
We have developed a full SfM system which deals with the main problems derived from a low-cost omnidirectional robot: all possible types of motions are covered, including those which generate pathological epipolar configurations. Low quality sensors are handled by addressing the noise created by them and the resulting scarce matching populations. More specifically, the achievements of our single SfM system can be summarised in three aspects: ill configurations, noise and Scarce matching populations.

Ill configurations:- The omnidirectional nature of the platform chosen for our experiments may produce pathological inter-image configurations, which are aggravated by the ever present noise. This imposes ill-conditioned problems for the estimation of the epipolar geometry (Vidal and Oliensis (2002)) which, in combination with the noise filters and the feature tracking system, is overcome by robust estimators of the relative pose between consecutive images and a efficient application of BA techniques. SfM has not been performed on omnidirectional platforms before (Bonin-Font et al. (2008); Fraundorfer and Scaramuzza (2012); Scaramuzza and Fraundorfer (2011)).

Noise:- Noisy sequences, produced by the inevitable JPEG compression that occurs on the wireless streaming of the images are addressed with light but efficient filters, which cope with levels of noise not found before in the literature (Gang and Reinhard (2005); Ruiz et al. (2006); Torr and Zisserman (1997)).

Scarce matching populations:- The scarcity of features generated by ruling out noisy corresponding points is coped with by a novel feature tracking system, which handles the sparse populations of matches and any remaining noise by an efficient management of the *bundles* of features that every 3D point creates. This feature tracking systems is integrated in an incremental SfM pipeline (Hartley and Zisserman (2004)), and generalises the work of Rohith et al. (2013) to any type of scenario.

A multi-session reconstruction system

We extend this SfM system for group of robots by means of an efficient use of the loop-closures created between. With this distributed system we address the second research question, which seeks to attain collaboratively 3D reconstructions from multiple platforms transiting a given scene.

The distributed reconstruction system devised merges the individual 3D maps of each robot into a single one by means of finding the loop-closures between the scenes viewed by each robot (Cummins and Newman (2011)). In terms of multiple reconstruction, our system resolves the global scale problem with minimal overlaps. This is obtained thanks to the common bundles created between different sessions when matching the loop-closures. With these bundles we are able to transmit the scale of a session into another, which leads to a global scale throughout all the sessions. We demonstrate the performance of this system in the two-session case. This result extends the work

of Zhang et al. (2010) which requires overlaps between stretches of sequences. Our multiple reconstruction system shows comparable results by using lesser constraints on the formulation of the input image set from multiple concurrent platforms within the environment against the prior work of Zhang et al. (2010).

6.2 Future Work

In this section we revise the aspects where we believe the present work can be extended and improved.

Improvement the of feature tracking system

The feature tracking system developed in this work can be improved further to be more tolerant to noise. If the filter f_1, described in Section 3.6, detects in an image two different features belonging to the same bundle, it deletes the bundles involved. It would be more optimal if it only removed the features detected. In addition, if the filter f_1 had a threshold to allow two features which are close enough to be assumed the same, it would account better for noisy measurements, and the feature tracking system would be more robust against noise.

Blur detection for Key-frame selection

Due to the specific motion of the robot and lags that may be produced in the wireless network, a significant amount of images streamed by the platform are blurred or are incomplete. Blur detection (an open problem in image processing, see Koik and Ibrahim (2013)) can be applied to the streamed images in order to rule out faulty images and apply SfM directly over the streaming video.

SfM in real time

As described in Section 2.6, there exist methods which attain real time reconstructions, most of them based on GPU hardware (Forster et al. (2014); Klein and Murray (2007); Newcombe et al. (2011)). This work has not been optimised for real time requirements, and we believe that GPU could be used to speed up processes, such as matching and feature tracking. Additionally, a more profound study of the multi-thread configuration

of this system will help to achieve a more concurrent efficiency between the main thread and the global BA thread.

Wheel odometry and wireless signal

Although the wheel odometry realised by the platform is highly unreliable, it is possible to extract some information out of it if an appropriate Kalman filter is applied to it (Wan and Merwe (2001)). This could be used for fixing a bounding box on the possible locations of a given camera, saving computation time to the optimisation methods.
Equally, the Rovio can provide a rough measurement of the signal of the wireless present in a given place. We could take advantage of the present literature on this field (Herrero and Martínez (2011)) in order to ease the camera pose estimation.

Distributed reconstruction

The multiple reconstruction performed in this work can be extended in many ways. To begin with, multi-robot reconstruction can be effectively achieved by implementing direct encounters recognition (Kato et al. (1999); Kim et al. (2010); Kurazume et al. (1994)). This can again be performed by applying FABMAP (Cummins and Newman (2011)) on the platforms or another machine learning technique. In addition, our system can easily be extended to enable loop-closures within a given session (i.e. any robot of a group can revisit areas of the scene, and the system process those loop-closures within the multiple reconstruction system).
Furthermore, collaborative strategies can be developed to optimise the area covered by each robot(Zavlanos et al. (2011)) . There are many aspects in this field to be tackled: decision planning, motion planning, the consensus problem, communication between robots (Kato et al. (1999); Ren et al. (2005); Zhu and Yang (2010)), etc.

References

Abdel-Aziz, Y. I. and Karara, H. M. (1971), Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry, *in* 'Proceedings of the Symposium on Close-Range Photogrammetry', American Society of Photogrammetry, Falls Church, VA, pp. 1–18. 47, 70, 89, 200

Agarwal, S., Chandraker, M., Kahl, F., Kriegman, D. and Belongie, S. (2006), 'Practical global optimization for multiview geometry', *Computer Vision-ECCV 2006* pp. 592–605. 45

Agarwal, S., Snavely, N., Seitz, S. and Szeliski, R. (2010), Bundle adjustment in the large, *in* 'Computer Vision - ECCV 2010', Vol. 6312 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 29–42. 52

AgiSoft, L. (2014), 'Agisoft photoscan, version 1.1.0', `http://www.agisoft.com/`. 120, 121, 134, 135, 136, 137, 138, 139, 140, 141, 142, 147

Agrawal, M., Konolige, K. and Blas, M. (2008), Censure: Center surround extremas for realtime feature detection and matching, *in* D. Forsyth, P. Torr and A. Zisserman, eds, 'Computer Vision - ECCV 2008', Vol. 5305 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 102–115. 26, 77

Alahi, A., Ortiz, R. and Vandergheynst, P. (2012), FREAK: Fast Retina Keypoint, *in* 'IEEE Conference on Computer Vision and Pattern Recognition', IEEE Conference on Computer Vision and Pattern Recognition, Ieee, New York. CVPR 2012 Open Source Award Winner. 27

Alcantarilla, P., Bartoli, A. and Davison, A. (2012), Kaze features, *in* A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato and C. Schmid, eds, 'Computer Vision - ECCV

2012', Vol. 7577 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 214–227. 27

Alcantarilla, P., Nuevo, J. and Bartoli, A. (2013), Fast explicit diffusion for accelerated features in nonlinear scale spaces, *in* 'Proceedings of the British Machine Vision Conference', BMVA Press. 27

Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D. and Silva, C. T. (2003), 'Computing and rendering point set surfaces', *IEEE Transactions on Visualization and Computer Graphics* **9**(1), 3–15. 92

Almeida, J., Dias, A., Martins, A., Sequeira, J. and Silva, E. (2013), 'Distributed active traction control system applied to the robocup middle size league', *International Journal of Advanced Robotic Systems* **10**, 1–12. 20

Anjum, N. (2011), 'Camera localization in distributed networks using trajectory estimation', *JECE* **2011**, 13:13–13:13. 57, 96

Anon. (2015), 'Classification by bags of words', Computer Vision Community. 102

Armstrong, M., Zisserman, A. and Hartley, R. (1996), Self-calibration from image triplets, *in* 'Computer Vision-ECCV 1996', Springer, pp. 1–16. 69

Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R. and Wu, A. Y. (1994), An optimal algorithm for approximate nearest neighbor searching in fixed dimensions, *in* 'Proc. in ACM-SIAM symposium on discrete algorithms', pp. 573–582. 31

Ashmore, M. and Barnes, N. (2002), Omni-drive robot motion on curved paths: The fastest path between two points is not a straight-line, *in* B. McKay and J. Slaney, eds, 'AI 2002: Advances in Artificial Intelligence', Vol. 2557 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 225–236. 21, 79

Baumela, L., Agapito, L., Bustos, P. and Reid, I. (2000), 'Motion estimation using the differential epipolar equation', *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000* pp. 840–843. 42

Bay, H., Tuytelaars, T. and Van Gool, L. (2006), SURF: Speeded up robust features, *in* 'Proc. in European Conference in Computer Vision', pp. 404–417. 25, 73, 77

Beardsley, P. and Torr, P. (1996), '3D model acquisition from extended image sequences', *Computer Vision-ECCV 1996* . 45, 118

Begum, A., Lee, M. and Kim, Y. (2010), A simple visual servoing and navigation algorithm for an omnidirectional robot, *in* 'Human-Centric Computing (HumanCom), 2010 3rd International Conference on', pp. 1–5. 19

Bonin-Font, F., Ortiz, A. and Oliver, G. (2008), 'Visual Navigation for Mobile Robots: A Survey', *Journal of Intelligent and Robotic Systems* **53**(3), 263–296. 11, 150

Bradski, G. (2000), 'The OpenCV Library', *Dr. Dobb's Journal of Software Tools* . 104

Breckon, T. (2011), 'Rovio API C++ Class Library', `http://breckon.eu/toby/software/`. 65

Breitenmoser, A., Kneip, L. and Siegwart, R. (2011), A monocular vision-based system for 6d relative robot localization, *in* 'Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on', pp. 79–85. 57, 96

Brostow, G. J., Shotton, J., Fauqueur, J. and Cipolla, R. (2008), Segmentation and recognition using structure from motion point clouds, *in* 'Proceedings of the 10th European Conference on Computer Vision: Part I', pp. 44–57. 7

Brown, D. C. (1971), 'Close-range camera calibration', *Photogrammetric Engineering* **37**(8), 855–866. 68

Brzeszcz, M. and Breckon, T. (2010), 'Towards real-time video mosaicing using feature driven image correspondences'. 83

Buente, J., Sharma, R. and Mejía, D. (2011), 'Visual slam using rovio', `http://www.cs.cornell.edu/Courses/cs4758/2011sp/`. 19

Butterfield, S. (1997), Reconstruction of extended environments from image sequences, PhD thesis, University of Leeds. 45

Calonder, M., Lepetit, V., Strecha, C. and Fua, P. (2010), Brief: Binary robust independent elementary features, *in* K. Daniilidis, P. Maragos and N. Paragios, eds, 'Computer Vision - ECCV 2010', Vol. 6314 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 778–792. 26

Castle, R., Klein, G. and Murray, D. (2008), Video-rate localization in multiple maps for wearable augmented reality, *in* 'Wearable Computers, 2008. ISWC 2008. 12th IEEE International Symposium on', pp. 15–22. 17

Cavestany, P., Rodríguez López, A. L., Martínez Barberá, H. and Breckon, T. (2015), Improved 3d sparse maps for high-performance sfm with low-cost omnidirectional robots, *in* 'Image Processing, 2015 IEEE International Conference on'. 12

Chamberland, S., Beaudry, E., Clavien, L., Kabanza, F., Michaud, F. and Lauria, M. (2010), Motion planning for an omnidirectional robot with steering constraints, *in* 'Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on', pp. 4305–4310. 20

Chang, C.-L. and Wu, B.-H. (2013), 'A dynamic cooperative scheme with multiple antennas for indoor mobile robot localization', *Abstract and Applied Analysis* . 3, 55

Chang, P. and Hebert, M. (2002), Robust tracking and structure from motion with sample based uncertainty representation, *in* 'Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on'. 18, 22, 59, 94, 136, 191, 205

Changchang, W. (2011), 'Visualsfm: A visual structure from motion system', `http://ccwu.me/vsfm/`. 120, 121, 134, 135, 136, 137, 138, 139, 140, 141, 142, 147

Changchang, W. (2013), 'Towards linear-time incremental structure from motion', 3DV. 120

Changchang, W., Agarwal, S., Curless, B. and Seitz, S. (2011), Multicore bundle adjustment, *in* '2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 3057–3064. 120

Chesi, G., Garullli, A., Vicino, A. and Cipolla, R. (2002), 'Estimating the fundamental matrix via constrained least-squares: A convex approach', *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(3), 397–401. 41

Chow, C. and Liu, C. (2006), 'Approximating discrete probability distributions with dependence trees', *IEEE Trans. Inf. Theor.* **14**(3), 462–467. 103

Chum, O. and Matas, J. (2005), Matching with prosac - progressive sample consensus, *in* 'Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on', Vol. 1, pp. 220–226 vol. 1. 33

Cummins, M. and Newman, P. (2008), 'Fab-map: Probabilistic localization and mapping in the space of appearance', *The International Journal of Robotics Research* **27**(6), 647–665. 104

Cummins, M. and Newman, P. (2011), 'Appearance-only slam at large scale with fab-map 2.0', *Int. J. Rob. Res.* **30**(9), 1100–1123. 5, 11, 12, 95, 100, 102, 103, 150, 152

Dang, H. and Hundal, J. (2011), 'Robot visual mapper', `http://www.cs.cornell.edu/Courses/cs4758/2011sp/`. 8, 19

Davison, A. J. (2003), Real-time simultaneous localisation and mapping with a single camera, *in* 'Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2', ICCV '03, IEEE Computer Society, Washington, DC, USA, pp. 1403–. 16

Davison, A. J. and Kita, N. (2001), Sequential localisation and map-building in computer vision and robotics, *in* 'Revised Papers from Second European Workshop on 3D Structure from Multiple Images of Large-Scale Environments', SMILE '00, Springer-Verlag, London, UK, UK, pp. 218–234. 43

Davison, A. J., Reid, I. D., Molton, N. D. and Stasse, O. (2007), 'Monoslam: Real-time single camera slam', *IEEE Trans. Pattern Analysis and Machine Intelligence* **29**, 2007. 16

Denning, T., Matuszek, C., Koscher, K., Smith, J. R. and Kohno, T. (2009), A spotlight on security and privacy risks with future household robots: Attacks and lessons, *in* 'Proceedings of the 11th International Conference on Ubiquitous Computing', Ubicomp '09, ACM, New York, NY, USA, pp. 105–114. 18

DeSouza, G. N. and Kak, A. C. (2002), 'Vision for mobile robot navigation: a survey', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **24**(2), 237–267. 17, 21

Dias, A., Almeida, J., Silva, E. and Lima, P. (2013), Multi-robot cooperative stereo for outdoor scenarios, *in* 'Autonomous Robot Systems (Robotica), 2013 13th International Conference on', pp. 1–6. 55

Dong, Z., Zhang, G., Jia, J. and Bao, H. (2009), Keyframe-based real-time camera tracking, *in* 'Computer Vision, 2009 IEEE 12th International Conference on', pp. 1538–1545. 36

Engel, J., Sturm, J. and Cremers, D. (2013), Semi-dense visual odometry for a monocular camera, *in* 'Computer Vision (ICCV), 2013 IEEE International Conference on', pp. 1449–1456. 39

Engels, C., Stewenius, H. and Nister, D. (2006), Bundle adjustment rules, *in* 'Photogrammetric Computer Vision'. 53, 90

Ermis, E. B., Clarot, P., Jodoin, P. and Saligrama, V. (2010), 'Activity based matching in distributed camera networks', *IEEE transactions on image processing* **19**(10), 2595–2613. 58

Eun, T. and G., M. (2007), '3-D metric reconstruction and registration of images of near-planar surfaces', *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on* pp. 1 –8. 46

Farnebäck, G. (2003), Two-frame motion estimation based on polynomial expansion, *in* 'Proceedings of the 13th Scandinavian Conference on Image Analysis', SCIA'03, Springer-Verlag, Berlin, Heidelberg, pp. 363–370.
URL: *http://dl.acm.org/citation.cfm?id=1763974.1764031* 29

Faugeras, O. (1993), *Three-dimensional computer vision: a geometric viewpoint*, MIT press. 180

Faugeras, O., Luong, Q.-T. and Maybank, S. (1992), Camera self-calibration: Theory and experiments, *in* G. Sandini, ed., 'Computer Vision - ECCV 92', Vol. 588 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 321–334. 69

Fiore, P. (2001), 'Efficient linear solution of exterior orientation', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **23**(2), 140–148. 47

Fischler, M. A. and Bolles, R. C. (1981), 'Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography', *Commun. ACM* **24**(6), 381–395. 32

Fladung, S. and Mwaura, J. (2011), 'Rovio augmented vision mapping project', `http://www.cs.cornell.edu/Courses/cs4758/2011sp/`. 19

Forster, C., Pizzoli, M. and Scaramuzza, D. (2014), 'Svo: Fast semi-direct monocular visual odometry', *Proc. IEEE Intl. Conf. on Robotics and Automation* . 6, 39, 59, 151

Fraundorfer, F. and Scaramuzza, D. (2012), 'Visual odometry : Part ii: Matching, robustness, optimization, and applications', *Robotics Automation Magazine, IEEE* **19**(2), 78–90. 11, 29, 30, 42, 150

Fryer, J. G. and Brown, D. C. (1986), 'Lens Distortion for Close-Range Photogrammetry', *Photogrammetric Engineering and Remote Sensing* **52**(1), 51–58. 68

Gang, L. and Reinhard, K. (2005), Structure from motion in the presence of noise, Technical report, The University of Auckland. 11, 22, 150

Gao, X.-S., Hou, X.-R., Tang, J. and Cheng, H.-F. (2003), 'Complete solution classification for the perspective-three-point problem', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **25**(8), 930–943. 48

Glover, A., Maddern, W., Warren, M., Reid, S., Milford, M. and Wyeth, G. (2012), Openfabmap: An open source toolbox for appearance-based loop closure detection, *in* 'Robotics and Automation (ICRA), 2012 IEEE International Conference on', pp. 4730–4735. 104, 105, 106, 107, 115

Guan, L. (2006), Sensor-Based Cooperative Multi-Robot 3D Environment Reconstruction, *in* 'Integrative Paper, UNC Chapel Hill (Fulfilling Master's Degree requirement)', Citeseer. 4, 56

Haley-Hermiz, T., Connelly, P., Gasper, A., Scalone, V., Sceusa, N. and Staehler, C. (2012), Augmented cinematography: A look at the use of augmented reality in film production, pp. 214–223. 7

Han, Y. (2005), Newton type algorithm on Riemannian manifolds applied to robot vision, and suggestions for improvement of its performance, *in* 'Vision, Image and Signal Processing, IEE Proceedings', Vol. 152, IET, pp. 275–282. 41

Harris, C. and Stephens, M. (1988), A combined corner and edge detection, *in* 'Proceedings of The Fourth Alvey Vision Conference', pp. 147–151. 24

Hartley, R. I. (1997), 'In defense of the eight-point algorithm', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(6), 580–593. 4, 40, 79, 194, 195

Hartley, R. I. (1998), Minimizing algebraic error, *in* 'Proceedings of the Sixth International Conference on Computer Vision', ICCV '98, IEEE Computer Society, Washington, DC, USA, pp. 469–. 40

Hartley, R. I. and Sturm, P. (1997), 'Triangulation', *Computer Vision and Image Understanding* **68**(2), 146–157. 48, 49

Hartley, R. I. and Zisserman, A. (2004), *Multiple view geometry in computer vision*, Cambridge University Press. 2, 4, 11, 21, 32, 34, 40, 41, 59, 68, 69, 74, 77, 80, 90, 127, 150, 180, 183, 184, 186, 187, 194, 196, 200

Herrero, D. and Martínez, H. (2011), 'Fuzzy mobile-robot positioning in intelligent spaces using wireless sensor networks', *Sensors* **11**(11), 10820–10839. 152

Holmes, S., Sibley, G., Klein, G. and Murray, D. (2009), A relative frame representation for fixed-time bundle adjustment in sfm, *in* 'Robotics and Automation, 2009. ICRA '09. IEEE International Conference on', pp. 2264–2269. 53

Horn, B. K. (1990), 'Relative orientation', *International Journal of Computer Vision* **4**(1), 59–78. 4, 40

Huang, T. S. and Faugeras, O. D. (1989), 'Some properties of the e matrix in two-view motion estimation', *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(12), 1310–1312. 40

Iocchi, L., Nardi, D. and Salerno, M. (2001), 'Reactivity and deliberation: a survey on multi-robot systems', *Balancing reactivity and social deliberation in multi-agent systems* **2103**, 9–32. 54

Jebara, T., Azarbayejani, A. and Pentland, A. (1999), '3D structure from 2D motion', *Proc. in Signal Processing Magazine, IEEE* **16**(3), 66–84. 7, 10

Jeong, D. and Lee, K. (2013), Inchbot: A novel swarm microrobotic platform, *in* 'Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on', pp. 5565–5570. 3, 57

Jeong, Y., Nister, D., Steedly, D., Szeliski, R. and Kweon, I.-S. (2010), Pushing the envelope of modern methods for bundle adjustment, *in* 'Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on', pp. 1474–1481. 52

Jin, H., Favaro, P. and Soatto, S. (2003), 'A semi-direct approach to structure from motion', *The Visual Computer* **19**(6), 377–394. 36

Kaess, M. and Dellaert, F. (2005), A Markov chain Monte Carlo approach to closing the loop in SLAM, *in* 'IEEE Intl. Conf. on Robotics and Automation, ICRA', Barcelona, Spain, pp. 645–650. 99

Kaess, M., Ranganathan, A. and Dellaert, F. (2007), isam: Fast incremental smoothing and mapping with efficient data association, *in* 'Robotics and Automation, 2007 IEEE International Conference on', pp. 1670–1677. 56, 96

Kahl, F. (2001), 'Euclidean reconstruction and auto-calibration from continuous motion', *Conference on Computer Vision, IEEE International* **2**, 572–577. 42

Kanatani, K., Sugaya, Y. and Niitsuma, H. (2008), 'Triangulation from two views revisited: Hartley-Sturm vs. optimal correction', *Proc. 19th British Machine Vision Conf.* pp. 173–182. 48, 49

Karnad, N. and Isler, V. (2010), A multi-robot system for unconfined videoconferencing, *in* 'Robotics and Automation (ICRA), 2010 IEEE International Conference on', pp. 356–361. 19

Kassebaum, J., Bulusu, N. and Feng, W. (2010), '3-D Target-based distributed smart camera network localization', *IEEE transactions on image processing* **19**(10), 2530–2539. 58

Kato, K., Ishiguro, H. and Barth, M. (1999), Identifying and localizing robots in a multi-robot system environment, *in* 'Intelligent Robots and Systems, 1999. IROS'99. Proceedings. 1999 IEEE/RSJ International Conference on', Vol. 2, pp. 966–971 vol.2. 57, 96, 152

Kawanishi, R., Yamashita, A. and Kaneko, T. (2009), 'Estimation of camera motion with feature flow model for 3D environment modeling by using omni-directional camera', *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems* pp. 3089–3094. 18

Kazhdan, M., Bolitho, M. and Hoppe, H. (2006), Poisson surface reconstruction, *in* 'Proc. in the Fourth Eurographics Symposium on Geometry Processing', pp. 61–70. 92, 133

Kim, B., Kaess, M., Fletcher, L., Leonard, J., Bachrach, A., Roy, N. and Teller, S. (2010), Multiple relative pose graphs for robust cooperative mapping, *in* 'Robotics and Automation (ICRA), 2010 IEEE International Conference', pp. 3185–3192. 3, 56, 96, 97, 121, 152

Klein, G. and Murray, D. (2007), Parallel tracking and mapping for small ar workspaces, *in* 'Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on', pp. 225–234. 38, 58, 151

Klingner, B., Martin, D. and Roseborough, J. (2013), Street view motion-from-structure-from-motion, *in* 'Computer Vision (ICCV), 2013 IEEE International Conference on', pp. 953–960. 5

Koik, B. T. and Ibrahim, H. (2013), A literature survey on blur detection algorithms for digital imaging, *in* 'Artificial Intelligence, Modelling and Simulation (AIMS), 2013 1st International Conference on', pp. 272–277. 151

Konolige, K. (2010), Sparse sparse bundle adjustment, *in* 'Proceedings of the British Machine Vision Conference', BMVA Press, pp. 102.1–102.11. doi:10.5244/C.24.102. 54

Kurazume, R., Nagata, S. and Hirose, S. (1994), Cooperative positioning with multiple robots, *in* 'Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on', pp. 1250–1257 vol.2. 57, 96, 152

Latif, Y., Cadena, C. and Neira, J. (2013), 'Robust loop closing over time for pose graph slam', *Int. J. Rob. Res.* **32**(14), 1611–1626. 100

Lee, D., Merrell, P., Wei, Z. and Nelson, B. E. (2008), 'Two-frame structure from motion using optical flow probability distributions for unmanned air vehicle obstacle avoidance', *Machine Vision and Applications* **21**(3), 229–240. 45

Lepetit, V. and Fua, P. (2006), 'Keypoint recognition using randomized trees', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **28**(9), 1465–1479. 100

Lepetit, V., Moreno-Noguer, F. and Fua, P. (2009), 'EPnP: An Accurate O(n) Solution to the PnP Problem', *International Journal of Computer Vision* **81**(2), 155–166. 48, 87

Leutenegger, S., Chli, M. and Siegwart, R. (2011), Brisk: Binary robust invariant scalable keypoints, *in* 'Computer Vision (ICCV), 2011 IEEE International Conference on', pp. 2548–2555. 26

libcurl (2011), 'http://curl.haxx.se'. 65

Liu, C., Freeman, W. T., Szeliski, R. and Kang, S. B. (2006), Noise estimation from a single image, *in* 'Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1', CVPR '06, IEEE Computer Society, Washington, DC, USA, pp. 901–908. 67

Liu, Y., Zhao, J., Apple, J., Frank, T., Saylor, M. and Siegel, T. (2010), 'An autonomous omnidirectional robot', *Journal of Robotics* . 20

Lobaton, E., Vasudevan, R., Bajcsy, R. and Sastry, S. (2010), 'A distributed topological camera network representation for tracking applications', *IEEE Transactions on Image Processing* **19**(10), 2516–2529. 58

Lobo, J., Marques, L., Dias, J., Nunes, U. and de Almeida, A. (1998), Sensors for mobile robot navigation, *in* A. de Almeida and O. Khatib, eds, 'Autonomous Robotic Systems', Vol. 236 of *Lecture Notes in Control and Information Sciences*, Springer London, pp. 50–81. 17

Longuet-Higgins, H. C. (1981), 'A computer algorithm for reconstructing a scene from two projections', *Nature* **293**(5828), 133–135. 4, 39

Lourakis, M. I. A. and Argyros, A. A. (2009), 'SBA: A Software package for generic sparse bundle adjustment', *ACM Trans. Math. Software* **36**(1), 1–30. 54, 89, 131

Lowe, D. G. (2004), 'Distinctive image features from scale-invariant keypoints', *International Journal of Computer Vision* **60**, 91–110. 24, 30, 75, 77, 93

Lucas, B. D. and Kanade, T. (1981), An iterative image registration technique with an application to stereo vision, *in* 'Proceedings of the 7th International Joint Conference on Artificial Intelligence', pp. 674–679. 29

Lv, Q., Josephson, W., Wang, Z., Charikar, M. and Li, K. (2007), Multi-probe lsh: Efficient indexing for high-dimensional similarity search, *in* 'Proceedings of the 33rd International Conference on Very Large Data Bases', VLDB '07, VLDB Endowment, pp. 950–961. 31

Ma, Y., Košecká, J. and Sastry, S. (2001), 'Optimization criteria and geometric algorithms for motion and structure estimation', *International Journal of Computer Vision* **44**(3), 219–249. 41, 79

Ma, Y., Soatto, S., Kosecka, J. and Sastry, S. (2003), *An invitation to 3D vision: from images to geometric models*. 182, 196

Maciel, J. and Costeira, J. (2003), 'A global solution to sparse correspondence problems', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **25**(2), 187–199. 36

Madsen, K., Nielsen, H. B. and Tingleff, O. (2004), 'Methods for non-linear least squares problems (2nd ed.)'. 203, 204

Maimone, M., Cheng, Y. and Matthies, L. (2007), 'Two years of visual odometry on the mars exploration rovers', *Journal of Field Robotics* **24**(3), 169–186. 6, 31

Makadia, A., Geyer, C. and Daniilidis, K. (2007), 'Correspondence-free structure from motion'. 31

Maohai, L., Bingrong, H. and Ronghua, L. (2006), Novel method for monocular vision based mobile robot localization, *in* 'Proc. in International Conference on Computational Intelligence and Security', Vol. 2, IEEE, pp. 949–954. 10

Matas, J., Chum, O., Urban, M. and Pajdla, T. (2004), 'Robust wide-baseline stereo from maximally stable extremal regions', *Image and vision computing* **22**(10), 761–767. 25

Mathews, N., Christensen, A., O'Grady, R. and Dorigo, M. (2012), Spatially targeted communication and self-assembly, *in* 'Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on', pp. 2678–2679. 7

McIlroy, P., Rosten, E., Taylor, S. and Drummond, T. (2010), 'Deterministic sample consensus with multiple match hypotheses', *Procedings of the British Machine Vision Conference 2010* pp. 111.1–111.11. 33

Mikolajczyk, K. and Schmid, C. (2005), 'A performance evaluation of local descriptors', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**, 1615–1630. 25

Miksik, O. and Mikolajczyk, K. (2012), Evaluation of local detectors and descriptors for fast feature matching, *in* 'Pattern Recognition (ICPR), 2012 21st International Conference on', pp. 2681–2684. 25

Moons, T. (2008), '3D Reconstruction from multiple images, part 1: principles', *Foundations and Trends in Computer Graphics and Vision* **4**(4), 287–404. 44

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F. and Sayd, P. (2006*a*), 'Monocular Vision Based SLAM for Mobile Robots', *18th International Conference on Pattern Recognition (ICPR'06)* pp. 1027–1031. 16, 44, 53, 199

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F. and Sayd, P. (2006*b*), Real time localization and 3d reconstruction, *in* 'Proc. in Computer Society Conference on Computer Vision and Pattern Recognition', Vol. 1, IEEE, pp. 363–370. 90, 91

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F. and Sayd, P. (2009), 'Generic and real-time structure from motion using local bundle adjustment', *Image and Vision Computing* **27**(8), 1178–1193. 38, 53

Muja, M. and Lowe, D. G. (2009), Fast approximate nearest neighbors with automatic algorithm configuration., *in* 'VISAPP (1)', pp. 331–340. 31

Newcombe, R. A., Lovegrove, S. and Davison, A. (2011), Dtam: Dense tracking and mapping in real-time, *in* 'Computer Vision (ICCV), 2011 IEEE International Conference on', pp. 2320–2327. 5, 6, 38, 59, 151

Nister, D. (2004), 'An efficient solution to the five-point relative pose problem', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **26**(6), 756 –770. 195

Nistér, D. (2005), 'Preemptive RANSAC for live structure and motion estimation', *Machine Vision and Applications* **16**(5), 321–329. 33

Nister, D., Naroditsky, O. and Bergen, J. (2004), Visual odometry, *in* 'Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on', Vol. 1, pp. I–652–I–659 Vol.1. 2, 43, 59

Oliensis, J. (1999), 'A multi-frame structure-from-motion algorithm under perspective projection', *International Journal of Computer Vision* **34**(2-3), 163–192. 46

Oliensis, J. (2002), 'Exact two-image structure from motion', *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(12), 1618–1633. 33

Oliveira, H. P., Sousa, A. J., Moreira, A. P. and Costa, P. J. (2009), 'Modeling and assessing of omni-directional robots with three and four wheels', *Contemporary Robotics-Challenges and Solutions* . 20, 59, 205

Olsson, C., Enqvist, O. and Kahl, F. (2010), Stable structure from motion using rotational consistency, Technical report, Centre for Mathematical Sciences, Lund University. 2

Ortin, D. and Montiel, J. M. M. (2001), 'Indoor robot motion based on monocular images', *Robotica* **19**, 331–342. 10

Otsuka, A., Nagata, F. and Okino, T. (2013), Measurement of dynamic sampling period of multi mobile robots system with wireless communication, *in* 'Mechatronics and Automation (ICMA), 2013 IEEE International Conference on', pp. 1141–1146. 3, 58

Özkucur, N. and Akin, H. (2010), Cooperative multi-robot map merging using fast-slam, *in* J. Baltes, M. Lagoudakis, T. Naruse and S. Ghidary, eds, 'RoboCup 2009: Robot Soccer World Cup XIII', Vol. 5949 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 449–460. 55, 121

Palacios-García, A. C., Muñoz Meléndez, A. and Morales, E. (2011), Learning concepts with multi-robot systems, *in* J. Cetto, J.-L. Ferrier and J. Filipe, eds, 'Informatics in Control, Automation and Robotics', Vol. 89 of *Lecture Notes in Electrical Engineering*, Springer Berlin Heidelberg, pp. 253–265. 55

Philip, J. (1998), 'Critical point configurations of the 5-, 6-, 7-, and 8-point algorithms for relative orientation', *TRITA-MAT-1998-MA* **13**. 195

Pietzsch, T. (2004), Application of a monocular camera as a motion sensor for mobile robots, PhD thesis, University of Dresde. 18

Pollefeys, M. (2004), 'Leuven Castle image sequence', `http://www.cs.unc.edu/~marc/index.html`. [Online; last accessed 3-March-2015]. 138

Pollefeys, M., Nistér, D., Frahm, J.-M., Akbarzadeh, A., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S.-J., Merrell, P., Salmi, C., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewénius, H., Yang, R., Welch, G. and Towles, H. (2007), 'Detailed real-time urban 3D reconstruction from video', *International Journal of Computer Vision* **78**(2-3), 143–167. 38

Qian, G. (2004), 'Bayesian self-calibration of a moving camera', *Computer Vision and Image Understanding* **95**(3), 287–316. 69

Quattoni, A. and Torralba, A. (2009), Recognizing indoor scenes, *in* 'Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on', pp. 413–420. 103, 104

Radke, R. J. (2008), A survey of distributed computer vision algorithms, *in* 'Aghajan (Eds.), Handbook of Ambient Intelligence and Smart Environments', pp. 1–21. 3, 9, 54

Raguram, R., Frahm, J.-M. and Pollefeys, M. (2009), Exploiting uncertainty in random sample consensus, *in* 'Computer Vision, 2009 IEEE 12th International Conference on', pp. 2074–2081. 33

Ratnasingam, S. and Collins, S. (2010), 'Study of the photodetector characteristics of a camera for color constancy in natural scenes', *JOSA A* **27**(2), 286–294. 22

Reid, R., Cann, A., Meiklejohn, C., Poli, L., Boeing, A. and Braunl, T. (2013), Cooperative multi-robot navigation, exploration, mapping and object detection with ros, *in* 'Intelligent Vehicles Symposium (IV), 2013 IEEE', pp. 1083–1088. 57, 121

Ren, C. and Ma, S. (2013), Dynamic modeling and analysis of an omnidirectional mobile robot, *in* 'Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on', pp. 4860–4865. 20

Ren, W., Beard, R. W. and Atkins, E. M. (2005), A survey of consensus problems in multi-agent coordination, *in* 'American Control Conference, 2005. Proceedings of the 2005', IEEE, pp. 1859–1864. 54, 152

Riazuelo, L., Civera, J. and Montiel, J. (2014), 'C2tam: A cloud framework for cooperative tracking and mapping', *Robotics and Autonomous Systems* **62**(4), 401 – 413. 58, 59, 97, 121

Rodríguez, A. L. (2013), Algebraic epipolar constraints for efficient structureless multiview motion estimation, PhD thesis, Computing Science Faculty. University of Murcia, Spain, Campus Universitario de Espinardo. CP30100. Murcia, Spain. 53

Rodríguez López, A. L., Ortuño Sánchez, A. and López de Teruel Alcolea, P. A. (2012), 'QVision: Computer Vision Library for Qt', `http://qvision.sourceforge.net/`. [Online; last accessed 2-March-2015]. 134

Rohith, M., Rhein, S., Lu, G., Sorensen, S., Mahoney, A., Eicken, H., Ray, G. and Kambhamettu, C. (2013), Iterative reconstruction of large scenes using heterogeneous feature tracking, *in* 'Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on', pp. 407–412. 11, 36, 59, 150, 205

Rojas, R. and Förster, A. G. (2006), 'Holonomic control of a robot with an omnidirectional drive', *KI - Künstliche Intelligenz* **20**(2), 12–17. 20

Rosten, E. and Drummond, T. (2006), Machine learning for high-speed corner detection, *in* 'European Conference on Computer Vision', Vol. 1, pp. 430–443. 26, 77

Rousseeuw, P. J. (1984), 'Least median of squares regression', *Journal of the American statistical association* **79**(388), 871–880. 34

Royer, E., Lhuillier, M., Dhome, M. and Lavest, J.-M. (2007), 'Monocular vision for mobile robot localization and autonomous navigation', *International Journal of Computer Vision* **74**(3), 237–260. 10, 43

Rublee, E., Rabaud, V., Konolige, K. and Bradski, G. (2011), Orb: An efficient alternative to sift or surf, *in* 'Computer Vision (ICCV), 2011 IEEE International Conference on', pp. 2564–2571. 26, 77

Ruiz, A., Lopez-de Teruel, P. E. and Fernandez-Maimo, L. (2006), Practical planar metric rectification, *in* 'Proc. BMVC', pp. 60.1–60.10. 11, 21, 93, 150

Rusu, R. B. and Cousins, S. (2011), 3D is here: Point Cloud Library (PCL), *in* 'Proc. in International Conference on Robotics and Automation'. 92

Safar, M., Watanabe, K., Maeyama, S. and Nagai, I. (2013), Tip-over prevention for a holonomic omnidirectional mobile robot with adwcs using sgcmg, *in* 'Mechatronics and Automation (ICMA), 2013 IEEE International Conference on', pp. 704–709. 20

Sampson, P. D. (1982), 'Fitting conic sections to "very scattered" data: An iterative refinement of the bookstein algorithm', *Computer Graphics and Image Processing* **18**(1), 97 – 108. 33, 79

Santiago, J. G., Wereley, S. T., Meinhart, C. D., Beebe, D. J. and Adrian, R. J. (1998), 'A particle image velocimetry system for microfluidics', *Experiments in Fluids* **25**(4), 316–319. 28

Saxena, A., Sun, M. and Ng, A. Y. (2007), '3-D Reconstruction from sparse views using monocular vision', *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on Computer Vision* pp. 1–8. 44

Scaramuzza, D. and Fraundorfer, F. (2011), 'Visual odometry [tutorial]', *Robotics Automation Magazine, IEEE* **18**(4), 80–92. 11, 42, 150

Schmidt, A., Kraft, M., Fularz, M. and Domagala, Z. (2013), 'The comparison of point feature detectors and descriptors in the context of robot navigation', *Journal of Automation, Mobile Robotics & Intelligent Systems* **7**(1). 24, 77

Shakernia, O., Vidal, R. and Sastry, S. (2003), Structure from small baseline motion with central panoramic cameras, *in* 'Proc. in Computer Vision and Pattern Recognition Workshop', Vol. 7, pp. 83–89. 10

Shi, J. and Tomasi, C. (1994), Good features to track, *in* 'Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on', pp. 593 –600. 24, 29, 36, 77

Shvarts, D. (2013), Global 3D map merging methods for robot navigation, PhD thesis, Tallinn university of technology, Ehitajate tee 5, 19086 Tallinn, Estonia. 101

Sibley, G., Mei, C., Reid, I. and Newman, P. (2009), Adaptive relative bundle adjustment, *in* 'Robotics Science and Systems (RSS)', Seattle, USA. 52, 53

Silva, B., Burlamaqui, A. and Goncalves, L. (2012), On monocular visual odometry for indoor ground vehicles, *in* 'Robotics Symposium and Latin American Robotics Symposium (SBR-LARS), 2012 Brazilian', pp. 220–225. 43

Smith, R. C. and Cheeseman, P. (1986), 'On the representation and estimation of spatial uncertainty', *The International Journal of Robotics Research* **5**(4). 15

Snavely, N., Seitz, S. M. and Szeliski, R. (2006), 'Proceedings of siggraph 2006', *ACM Transactions on Graphics* . 4, 5, 134

Snavely, N., Seitz, S. M. and Szeliski, R. (2008*a*), 'Modeling the world from internet photo collections', *International Journal of Computer Vision* **80**(2), 189–210. 7, 44

Snavely, N., Seitz, S. M. and Szeliski, R. (2008*b*), Skeletal graphs for efficient structure from motion, *in* 'IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008)', IEEE Computer Society, Anchorage, AK. 44

Solomon, C. and Breckon, T. (2011), *Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab*, 1st edn, Wiley Publishing. 22

Spang, H. A. (1962), A review of minimization techniques for nonlinear functions, Technical Report 4. 41

Strasdat, H., Montiel, J. M. M. and Davison, A. (2010), Real-time monocular slam: Why filter?, *in* 'Robotics and Automation (ICRA), 2010 IEEE International Conference on', pp. 2657–2664. 16, 21

Sturm, P. (1997), Critical motion sequences for monocular self-calibration and uncalibrated euclidean reconstruction, *in* 'Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on', pp. 1100–1105. 19, 20

Szeliski, R. (2005), Image alignment and stitching: A tutorial, Technical report, MSR-TR-2004-92, Microsoft Research, 2004. 29

Szeliski, R. (2011), *Computer Vision. Algorithms and applications*, Springer London. 28, 42, 196

Szeliski, R. and Kang, S. B. (1994), 'Recovering 3d shape and motion from image streams using nonlinear least squares', *Journal of Visual Communication and Image Representation* **5**(1), 10 – 28. 10

Tang, Q. and Eberhard, P. (2013), 'Cooperative search by combining simulated and real robots in a swarm under the view of multibody system dynamics', *Advances in Mechanical Engineering* **2013**, 1–11. 58

Thomas, J. I. and Oliensis, J. (1999), 'Dealing with noise in multiframe structure from motion', *Computer Vision and Image Understanding* **76**(2), 109–124. 21, 59, 93, 191

Tomasi, C. and Manduchi, R. (1998), Bilateral filtering for gray and color images, *in* 'In Proc. Int. Conf. Computer Vision', IEEE, pp. 839–846. 71

Tomono, M. (2005), 3-d localization and mapping using a single camera based on structure-from-motion with automatic baseline selection, *in* 'Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on', pp. 3342–3347. 16

Torr, P., Fitzgibbon, A. W. and Zisserman, A. (1998), Maintaining multiple motion model hypotheses over many views to recover matching and structure, *in* 'Computer Vision, 1998. Sixth International Conference on', pp. 485 –491. 7, 46

Torr, P. H. S. and Zisserman, A. (1997), 'Performance characterizaton of fundamental matrix estimation under image degradation', *Machine Vision and Applications* **9**, 321–333. 11, 33, 41, 147, 150

Torr, P. and Zisserman, A. (2000), 'Mlesac: A new robust estimator with application to estimating image geometry', *Computer Vision and Image Understanding* **78**(1), 138 – 156. 33

Triggs, B., McLauchlan, P., Hartley, R. I. and Fitzgibbon, A. (2000), 'Bundle adjustment - a modern synthesis', *Vision algorithms: theory and practice* **34099**, 153–177. 50, 51, 118

Tsai, R. and Huang, T. S. (1984), 'Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **PAMI-6**(1), 13–27. 196

Udengaard, M. and Iagnemma, K. (2008), Design of an omnidirectional mobile robot for rough terrain, *in* 'Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on', pp. 1666–1671. 20

Ueno, Y., Ohno, T., Terashima, K. and Kitagawa, H. (2009), The development of driving system with differential drive steering system for omni-directional mobile robot, *in* 'Mechatronics and Automation, 2009. ICMA 2009. International Conference on', pp. 1089–1094. 20

Upcroft, B., McManus, C., Churchill, W., Maddern, W. and Newman, P. (2014), Lighting invariant urban street classification, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)', Hong Kong, China. 22

Verhoeven, G. (2011), 'Taking computer vision aloft – archaeological three-dimensional reconstructions from aerial photographs with photoscan', *Archaeological Prospection* **18**(1), 67–73. 120

Verhoeven, G., Doneus, M., Briese, C. and Vermeulen, F. (2012), 'Mapping by matching: a computer vision-based approach to fast and accurate georeferencing of archaeological aerial photographs', *Journal of Archaeological Science* **39**(7), 2060 – 2070. 120

Vidal, R. and Oliensis, J. (2002), Structure from planar motions with small baselines, *in* 'Proc. in European Conference in Computer Vision', Springer, pp. 383–398. 10, 11, 59, 150, 191

Wai Yin Leung, C. (2006), Efficient methods for 3D reconnstruction from multiple images, PhD thesis, University of Queensland. 7, 44

Walber (2015), 'Precisionrecall', `http://commons.wikimedia.org/wiki/File:Precisionrecall.svg#/media/File:Precisionrecall.svg`. [Licensed under CC BY-SA 4.0 via Wikimedia Commons; Online; accessed 24-April-2015]. 105

Wan, E. A. and Merwe, R. V. D. (2001), The unscented kalman filter, *in* 'Kalman Filtering and Neural Networks', Wiley, pp. 221–280. 16, 152

Wang, Y., Velipasalar, S. and Casares, M. (2010), 'Cooperative object tracking and composite event detection with wireless embedded smart cameras', *Transactions on Image Processing, IEEE* **19**(10), 2614–2633. 58

Wei, Y.-m., Kang, L., Yang, B. and Wu, L.-d. (2013), 'Applications of structure from motion: a survey', *Journal of Zhejiang University SCIENCE C* **14**(7), 486–494. **URL:** *http://dx.doi.org/10.1631/jzus.CIDE1302* 5, 7

Wendel, A., Hoppe, C., Bischof, H. and Leberl, F. (2012), Automatic fusion of partial reconstructions, *in* 'Annals of the International Society for Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)', ISPRS. 3, 57, 97

Wenzel, F. and rainer Grigat, R. (2005), Parametrizations of the essential and the fundamental matrix based on householder transformations, *in* '10th International Workshop on Vision, Modeling and Visualization'. 40

Werfel, J., Petersen, K. and Nagpal, R. (2014), 'Designing Collective Behavior in a Termite-Inspired Robot Construction Team', *Science* **343**(6172), 754–758. 6

Wikipedia (2015*a*), 'Parallelepiped — wikipedia, the free encyclopedia', `http://en.wikipedia.org/w/index.php?title=Parallelepiped&oldid=649821039`. [Online; accessed 13-April-2015]. 192

Wikipedia (2015*b*), 'Personal computer — wikipedia, the free encyclopedia'. [Online; accessed 16-December-2015].
URL: *https://en.wikipedia.org/w/index.php?title=Personal_computer&oldid=694959103* 65

Wikipedia (2015*c*), 'Pinhole camera — wikipedia, the free encyclopedia', `http://en.wikipedia.org/w/index.php?title=Pinhole_camera&oldid=651866113`. [Online; accessed 13-April-2015]. 178

Williams, B., Klein, G. and Reid, I. (2011), 'Automatic relocalization and loop closing for real-time monocular slam', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **33**(9), 1699–1712. 100

Williams, R. L., Carter, B. E., Gallina, P. and Rosati, G. (2002), 'Dynamic model with slip for wheeled omnidirectional robots', *IEEE Transactions on Robotics and Automation* **18**(3), 285–293. 20

Ye, C., Ma, S. and Hui, L. (2011), 'An omnidirectional mobile robot', *Science China Information Sciences* **54**(12), 2631–2638. 20

Yu, F. and Gallup, D. (2014), 3d reconstruction from accidental motion, *in* 'The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)'. 6, 46

Zavlanos, M., Egerstedt, M. and Pappas, G. (2011), 'Graph-theoretic connectivity control of mobile robot networks', *Proceedings of the IEEE* **99**(9), 1525–1540. 55, 152

Zhang, G., Dong, Z., Jia, J., Wong, T.-T. and Bao, H. (2010), Efficient non-consecutive feature tracking for structure-from-motion, *in* K. Daniilidis, P. Maragos and N. Paragios, eds, 'Computer Vision - ECCV 2010', Vol. 6315 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 422–435. 11, 60, 82, 94, 101, 102, 115, 136, 148, 151

Zhang, J., Boutin, M. and Aliaga, D. (2006), Robust bundle adjustment for structure from motion, *in* 'Image Processing, 2006 IEEE International Conference on', pp. 2185–2188. 53

Zhang, Z. (1995), 'A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry', *Artificial Intelligence* **78**(1-2), 87–119. 45

Zhang, Z. (1998), 'Determining the epipolar geometry and its uncertainty: a review', *International Journal of Computer Vision* **27**(2), 161–195. 40, 191

Zhang, Z. (2000), 'A flexible new technique for camera calibration', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(11), 1330–1334. 70

Zhang, Z. (2012), 'Microsoft kinect sensor and its effect', *IEEE MultiMedia* **19**(2), 4–10. 4

Zhao, F., H. Q. and Gao, W. (2006), Image matching by normalized cross-correlation, *in* 'Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on', Vol. 2, pp. II–II. 30

Zhu, A. and Yang, S. X. (2010), 'A survey on intelligent interaction and cooperative control of multi-robot systems', *8th IEEE International Conference on Control and Automation (ICCA), 2010* pp. 1812–1817. 54, 152

Zou, J.-T., Chiang, F.-C. and Su, K. (2011), 'The study of path error for an omnidirectional home care mobile robot', *Artificial Life and Robotics* **16**(2), 127–131. 20

Zucchelli, M. (2002), Optical flow based structure from motion optical flow based structure from motion, PhD thesis, University of Stockholm. 45

Declaration

I herewith declare that I have produced this paper without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This paper has not previously been presented in identical or similar form to any examination board.

The work was conducted from 2010 to 2015 under the supervision of Dr. Toby Breckon at the department of Engineering Computing.

Cranfield University,

Appendix A

Structure from Motion - a Geometric Overview

This appendix defines a nomenclature for the Structure from Motion (SfM) method and provides a general overview of the SfM process. In addition, a background in projective geometry is given, since knowledge in this field is mandatory when working in 3D Vision. Homogeneous coordinates, necessary to operate in projective geometry, are introduced first. Secondly, the capture of the 3D world in an image is explained by developing the concepts of pinhole camera model and camera matrix. When the model of the camera is known, normalised coordinates can be used, so that the algebraic structures implemented in this work are transparent to the type of camera deployed. The core of the appendix is reached at the explanation of the epipolar geometry, crucial for a full understanding of the SfM method.

The epipolar geometry is produced by the specific geometric configuration created between two images viewing the same portion of a scene. This inter-image configuration will be studied in detail throughout this appendix. This particular geometry allows us to extract the fundamental matrix, which encodes the camera motion information between two images and the intrinsic characteristics of the camera. If the intrinsic configuration of the camera is known, the essential matrix can be extracted and with it the relative motion between two cameras. A linear solution for the estimation of the relative motion in ideal conditions is presented. This method will give insights as to what challenges are encountered in real robot navigation situations. This estimation of the relative motion highlights the scaling limitation of the SfM method. This is addressed as the problem

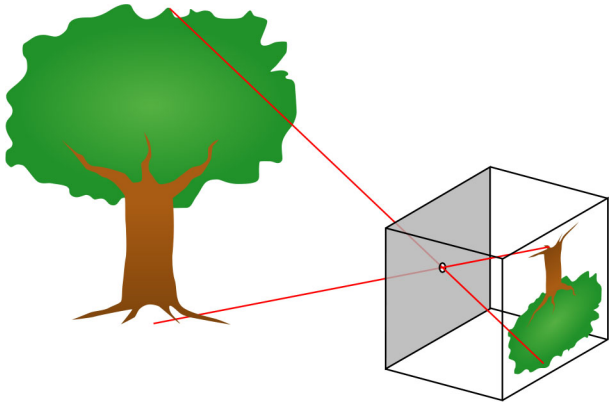


Figure A.1: The pinhole camera. Source: Wikipedia (2015*c*).

of the scale, which is solved by applying *resection*. Once the motion in a sequence of cameras has been retrieved the structure can be reconstructed by triangulation. Finally Bundle Adjustment, a non-linear method which simultaneously refines the estimation of the camera poses and structure, is introduced.
In order to fully explain the SfM process we first need to introduce some basic concepts which are applied in projective geometry, such as homogeneous coordinates.

A.1 Homogeneous Coordinates

Projective geometry makes use of *homogeneous* coordinates, which we define as follows. Given a vector $\boldsymbol{x} \in \mathbb{R}^n$ the homogeneous vector of $\boldsymbol{x}$ is a vector $\tilde{\boldsymbol{x}} \in \mathbb{R}^{n+1}$ whose first n coordinates $\tilde{x}_i$ are proportional to the coordinates x_i of $\boldsymbol{x}$, $\tilde{x}_i = kx_i,\ i = 0 \dots n,\ k \in \mathbb{R}$. The last coordinate of $\tilde{\boldsymbol{x}}$ is the scalar k. More specifically, if $n = 2$, the homogeneous point $\tilde{\boldsymbol{x}}$ of a Cartesian point $\boldsymbol{x} = (x, y)$ is defined with the coordinates $(\tilde{x}, \tilde{y}, \tilde{z}) = (xk, yk, k)$, $k \in \mathbb{R}$. This relationship can be expressed in a matrix form as:-

$$k \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{bmatrix} \tag{A.1.1}$$

Note that even though $\tilde{\boldsymbol{x}} \in \mathbb{R}^3$, it only has two degrees of freedom, since the third is fixed by k, which is arbitrary. Homogeneous coordinates have the advantage that formulae involving them are often simpler than their Cartesian (also called *inhomogeneous*) counterparts. The use of them and its implementation is convenient in computer vision,

for it allows to combine common operations such as translation, rotation, scaling and perspective projection in a concise and ordered manner.

As an example, let us consider a rigid body represented by the vector $\boldsymbol{X} = (x, y, z)^T$. If we apply an Euclidean transformation to $\boldsymbol{X}$, this transformation will be given by a rotation matrix R and a translation $\boldsymbol{t}$. In inhomogeneous coordinates, this rotation and translation on $\boldsymbol{X}$ would be expressed by a column vector $\boldsymbol{X'}$:-

$$\boldsymbol{X'} = \mathrm{R}\boldsymbol{X} + \boldsymbol{t} \tag{A.1.2}$$

However if we add the homogeneous coordinate to $\boldsymbol{X}$ so that $\tilde{\boldsymbol{X}} = (x, y, z, 1)^T$ then we have:-

$$\boldsymbol{X'} = [\mathrm{R} \mid \boldsymbol{t}]\, \tilde{\boldsymbol{X}} \tag{A.1.3}$$

where $[\mathrm{R} \mid \boldsymbol{t}]$ is a matrix created with the columns of R plus the column vector $\boldsymbol{t}$. The homogeneous representation allows to set linear systems and develop algebraic algorithms in a neat manner which otherwise would be cumbersome to address.

Additionally, and not less importantly, the coordinates of points at infinity can be represented by using homogeneous coordinates. There is no Cartesian equivalence for the homogeneous triplet $(\tilde{x}, \tilde{y}, 0)$; these points are the points at infinity. Therefore, homogeneous notation allows us to work seamlessly with points located at infinity. This is a necessary property in projective geometry, because often entities at infinity are projectively mapped to a finite point. Examples of these entities are the horizon or the vanishing points, which adopt a measurable value when projected to an image.

Likewise column vectors and following Eq. A.1.1, in matrix algebra homogeneity affects the scale of the matrix elements. Therefore two homogeneous matrices A and B are equivalent if $\mathrm{A} = k\mathrm{B}$, $k \in \mathbb{R}$. In a similar way to column vectors, this up to scale equivalence drops one degree of freedom in the elements of a matrix. Hence a homogeneous matrix $n \times n$ has $n \times n - 1$ degrees of freedom, since the scalar k is unimportant.

Homogeneous coordinates are used when modelling the projection of the 3D world on to an image. This process is called camera calibration.

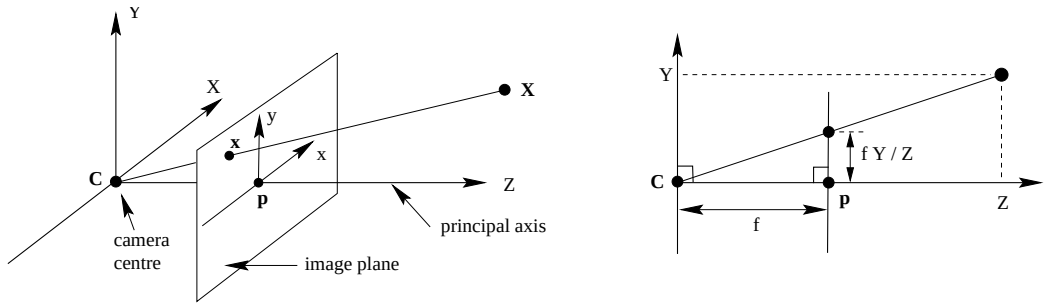


Figure A.2: The geometric camera model. The principal optical axis z intersects the image plane at the principal point $\boldsymbol{p}$. The projection $\boldsymbol{x}$ of $\boldsymbol{X}$ on the image plane is done by joining $\boldsymbol{X}$ and $\boldsymbol{C}$. Source: Hartley and Zisserman (2004).

A.2 Camera Calibration

When it comes to recovering the 3D structure of objects from images taken by a camera, the first question that emerges is: What information about the world is in an image and how is it kept? To answer this we need to define a model of the camera. Most planar projections in cameras can be modelled by the pinhole camera model.

The pinhole camera model

As shown in Fig. A.1, the pinhole camera model assumes that light arrives to the camera through a pinhole (also called focus) and is projected on the image plane. The distance between the pinhole and the image plane is the focal length, f. The pinhole camera model is the best common camera model used to represent the projection of the world into a camera image (Faugeras (1993)). For the sake of clarity, and without loss of generality, the geometric model places the image plane before the focus, as Fig. A.2 illustrates. The pinhole camera model is essentially represented by the intrinsic matrix which represents the projection from the 3D world to the 2D image plane.

A.2.1 The Intrinsic Matrix

We will derive the configuration of the intrinsic matrix with the aid of Fig. A.2. Let $\boldsymbol{O}$ in Fig. A.2 be the coordinate frame and $\boldsymbol{p}$ the intersection of $z-$axis with the image plane. The point $\boldsymbol{X}$ has coordinates (X, Y, Z) with respect to $\boldsymbol{O}$. The point $\boldsymbol{x}$ is the projection of $\boldsymbol{X}$ on the image plane, and it has (x, y) as image coordinates with respect

to $\boldsymbol{p}$. The point $\boldsymbol{C}$ is the origin of $\boldsymbol{O}$, and it is also called in the literature centre of projection (COP). From Fig. A.2 is easily deducible that:-

$$x = f\frac{X}{Z},\ y = f\frac{Y}{Z} \tag{A.2.4}$$

However, digital cameras usually present the origin of coordinates at the upper-left corner. Also, the focal length varies for each axis, yielding to f_1 for the $x-$axis and f_2 for the $y-$axis. Therefore, the correct transformation from world coordinates to image coordinates must be expressed as:-

$$x = f_1\frac{X}{Z} + x_0,\ y = f_2\frac{Y}{Z} + y_0 \tag{A.2.5}$$

where (x_0, y_0) are the coordinates of the point $\boldsymbol{p}$ with respect to a coordinate frame with origin at the upper-left corner of the image. Eq. A.2.5 can be expressed in homogeneous coordinates as:-

$$Z\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_1 & 0 & x_0 \\ 0 & f_2 & y_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{A.2.6}$$

Let $\boldsymbol{O'}$ be another world coordinate frame, with origin at some point $\boldsymbol{C'}$. The translation from $\boldsymbol{C}$ to $\boldsymbol{C'}$ is expressed by the column vector $\boldsymbol{t}$, and the rotation from the initial coordinate frame $\boldsymbol{O}$ to $\boldsymbol{O'}$ is represented by the rotation matrix R. The same point as above $\boldsymbol{X}$ has coordinates (X', Y', Z') with respect to $\boldsymbol{O'}$, and the relationship between both coordinate frames is:-

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} \mathrm{R} & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{bmatrix}\begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix} \tag{A.2.7}$$

where $\mathbf{0}$ denotes a null 3-vector. Therefore, the transformation of a point from any coordinate frame into image coordinates has the form:-

$$Z\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_1 & 0 & x_0 \\ 0 & f_2 & y_0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} \mathrm{R} & \boldsymbol{t} \\ \mathbf{0}^T & 1 \end{bmatrix}\begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix} \tag{A.2.8}$$

We denote:-

$$\mathrm{K} = \begin{bmatrix} f_1 & s & x_0 \\ 0 & f_2 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.2.9}$$

where the element $k_{12} = s$ is called skew. This parameter, s is zero in most of the cameras, although it may be significant in case there is a skewing of the pixel elements in the camera array, so that the $x-$axis and $y-$axis of the image plane are not perpendicular. This is admittedly very unlikely to happen.

Eq. A.2.8 can be written more concisely:-

$$Z \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathrm{K}\,[\mathrm{R} \mid \boldsymbol{t}]\,\boldsymbol{X'} \tag{A.2.10}$$

where $\boldsymbol{X'}$ denotes the homogeneous coordinates of the point $\boldsymbol{X}$ under $\boldsymbol{O'}$ frame coordinates. The matrix $\mathrm{K}\,[\mathrm{R} \mid \boldsymbol{t}]$ is usually called projection matrix.

The upper-triangular matrix K is called the intrinsic matrix, as it accounts for the specific optics of the camera[1], the intrinsic parameters. The matrices R and $\boldsymbol{t}$ contain the extrinsic parameters i.e. the information regarding the exterior orientation of the camera with respect to a global coordinate frame. In the literature, the process of estimating the extrinsic and intrinsic parameters of a camera is usually denominated as camera calibration (Ma et al. (2003)).

The camera model, in the form of the intrinsic matrix, allows us to define the normalised coordinates, which will simplify the equations employed.

A.2.2 Normalised Coordinates

Given the homogeneous image point $\boldsymbol{x} = (x, y, 1)$ in Fig. A.2 we define the normalised coordinates[2] $\bar{\boldsymbol{x}}$ as the homogeneous coordinates given by:-

$$\bar{\boldsymbol{x}} = \mathrm{K}^{-1}\boldsymbol{x} \tag{A.2.11}$$

[1]Except for the radial and tangential distortions introduced by the lens of the image sensor, which are addressed in Chapter 3.

[2]In the literature they are also denoted as retinal coordinates.

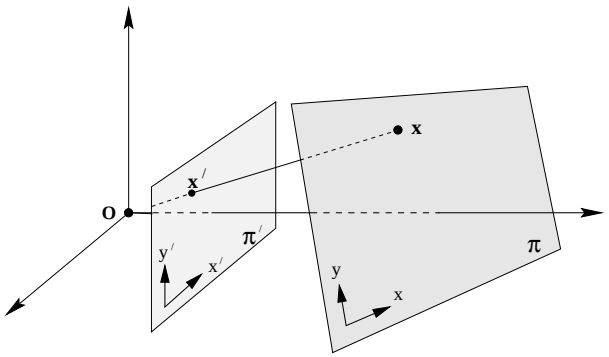


Figure A.3: The central projection between planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi'}$ is produced by projecting along rays through a common point $\boldsymbol{O}$ (the centre of projection, COP). Source: Hartley and Zisserman (2004).

The normalised coordinates and the homogeneous image coordinates would coincide if the intrinsic matrix of the camera was the identity matrix, I. The normalised coordinates is a convenient transformation that allows us to express points in the image with respect to the frame coordinate of its corresponding camera. Additionally, the algebraic constructions created with normalised coordinates are transparent to the type of camera used.

Since in this work the intrinsic parameters of the image sensor remain constant for each robot (the focal length is constant over image sequences), the normalised coordinates will be used instead of image coordinates unless stated otherwise. In fact, we can simplify further Eq. A.2.10 which leads to the definition of the camera matrix.

A.2.3 The Camera Matrix

Eq. A.2.11 combined with Eq. A.2.10 yields:-

$$\bar{\boldsymbol{x}} = [\mathrm{R} \mid \boldsymbol{t}]\, \boldsymbol{X'} \tag{A.2.12}$$

We introduce now the camera matrix $\mathrm{P} = [\mathrm{R} \mid \boldsymbol{t}]$ so that Eq. A.2.12 becomes:-

$$\bar{\boldsymbol{x}} = \mathrm{P}\boldsymbol{X'} \tag{A.2.13}$$

Throughout this work we will be using Eq. A.2.13 for denoting camera poses. Eq. A.2.13 encodes all the algebraic transformations that take place from a 3D world point to its projection on an image, as the optics of the camera have been taken into account by

the normalised coordinates $\bar{\boldsymbol{x}}$. For convenience, from now on we will refer to normalised coordinates as $\boldsymbol{x}$ unless said otherwise.
Both homogeneous and normalised coordinates, along with the intrinsic and the camera matrices K and P, are constructions employed in projective geometry and multiple view geometry, and more specifically, in the SfM problem. Homogeneous matrices are used to define the epipolar geometry between two images, and K and P play a central role in the reconstruction of a scene. In the following sections (Sections A.4 to A.6) the resolution of a general SfM is derived with the help of the concepts introduced here.

A.3 Homographies

The derivation of the epipolar geometry between two images involves the understanding of homography, an algebraic transformation often used in projective geometry. In this section the concept of homography is introduced, along with a succinct classification of homographies and a brief discussion of each type.
In projective geometry, a homography is defined as a transformation mapping usually referred to as projections. Here we will describe homographies as projections between two sets of points lying on planes, but their properties are easily translatable to sets of 3D points or other algebraic structures.
Let $\mathrm{X} = \{\boldsymbol{x_i}\}$, $\mathrm{X}' = \{\boldsymbol{x_i'}\}$, $i = 0 \dots n$ be two sets of homogeneous 2D points. We can freely assume that each set lies on a different plane in the 3D space (see Fig. A.3). A homogeneous non singular 3×3 matrix H is a homography between X and X' if and only if $\boldsymbol{x_i'} = \mathrm{H}\boldsymbol{x_i}\ \forall i \in \{0 \dots n\}$. In geometric terms, H is projecting X into X'. There are four main groups of homographies according to the geometric properties which are preserved under transformation in each case (Hartley and Zisserman (2004)):-

1. Projective transformations (or projectivities) preserve collinearity, intersections, tangency and order of contact. In a projectivity neither angles nor parallel lines are preserved. To put it in simple terms, a general quadrilateral would be a projective transformation of a square. A projectivity $\mathrm{H_P}$ has 8 degrees of freedom (dof): the 8 ratios of the elements of $\mathrm{H_P}$, since scale is unimportant in a homogeneous matrix.

$$\mathrm{H_P} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{A.3.14}$$

$$\text{dof}: \{\text{h}_{11} : \text{h}_{12} : \text{h}_{13} : \text{h}_{21} : \text{h}_{22} : \text{h}_{23} : \text{h}_{31} : \text{h}_{32} : \text{h}_{33}\}$$

2. Affine transformations preserve in addition parallel lines and ratio of areas. The affine transformation of a square would be a rhombus. The last row of an affine transformation H_A is $(0, 0, 1)$ and therefore it has 6 degrees of freedom. The last column of H_A $(t_x, t_y, 1)^T$ accounts for the translation between the two sets of points.

$$\text{H}_\text{A} = \begin{bmatrix} h_{11} & h_{12} & t_x \\ h_{21} & h_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{A.3.15}$$

$$\text{dof}: \{\text{h}_{11} : \text{h}_{12} : \text{h}_{21} : \text{h}_{22} : \text{t}_\text{x} : \text{t}_\text{y}\}$$

3. Similarities preserve additionally angles and ratio of lengths. The similarity transformation of a square would be another square in a different pose and scale. In addition to the last row $(0, 0, 1)$ fixed, in a similarity H_S the upper left 2×2 submatrix is a planar rotation matrix multiplied by a scalar k. Since in 2D a rotation is defined by an angle of rotation θ, H_S has thus 4 degrees of freedom, accounted for θ, k and the two parameters of the translation vector.

$$\text{H}_\text{S} = \begin{bmatrix} k\,cos\theta & -k\,sin\theta & t_x \\ k\,sin\theta & k\,cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{A.3.16}$$

$$\text{dof}: \{\text{k} : \theta : \text{t}_\text{x} : \text{t}_\text{y}\}$$

4. Euclidean transformations also preserve length and area. The euclidean transformation of a square would be the same square rotated. In an euclidean transformation H_E there is no scale between sets ($k = 1$) so H_E has 3 degrees of freedom: θ, t_x and t_y.

$$\text{H}_\text{E} = \begin{bmatrix} cos\theta & -sin\theta & t_x \\ sin\theta & cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{A.3.17}$$

$$\text{dof}: \{\theta : \text{t}_\text{x} : \text{t}_\text{y}\}$$

Every transformation down the list is a subset or specialisation of the previous one. Therefore each type of homography preserves all of the geometric properties from those

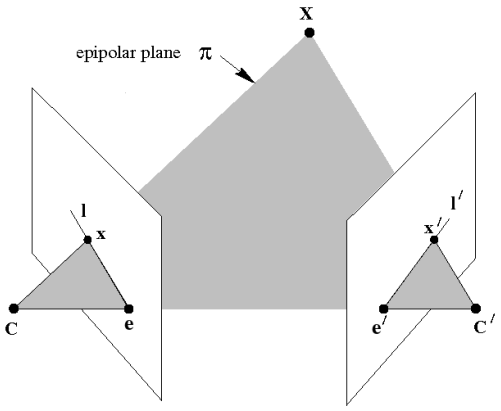


Figure A.4: The epipolar geometry. The camera centres $\boldsymbol{C}$ and $\boldsymbol{C}'$, and the point $\boldsymbol{X}$ lie on the same plane $\boldsymbol{\pi}$. Source: Hartley and Zisserman (2004).

preceding it, in addition to the characteristics specifically preserved by the transformation itself. On the other hand it is demonstrated that a given transformation can be decomposed as a product of its specialisations (Hartley and Zisserman (2004)).

A remark regarding the matrices introduced in Section A.2 is in order here, so that the geometric transformation that each matrix signifies may be understood better. Note that the camera matrix P in Eq. A.2.13 represents an Euclidean transformation in a three dimensional space, with rotation R and translation $\boldsymbol{t}$. The intrinsic matrix K in turn can be thought of as a planar affine transformation. The affine transformation performed by K would be the combination of a translation by a vector $(x_0, y_0)^T$, a 2D scaling by the scalars f_x, f_y and a shear s. It is said above that the combination of specialised homographies can explain more general geometric transformations, and Eq. A.2.10 is an example of this. Eq. A.2.10 shows that the effects of R, $\boldsymbol{t}$ and K altogether produce a projective transformation, as indeed it is the projection of an object from the 3D world onto the 2D plane of an image.

In each group of homographies there are special cases which have received particular attention from researchers (Hartley and Zisserman (2004)). For example, the projection depicted in Fig. A.3 is called central projection and it is a type of projectivity. A central projection is generated when the mapping between two planes is determined by lines concurrent to a central point $\boldsymbol{O}$. This type of projection will be specifically employed in Section A.4.1.

Homographies are extensively applied in multiple view geometry. In this work homogra-

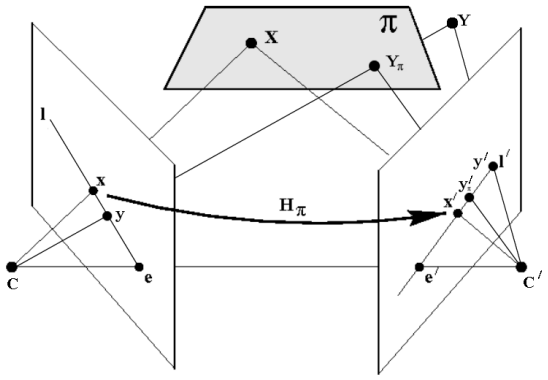


Figure A.5: The projection x is transferred to x' through the homography H_π. The projection $\boldsymbol{y'_\pi}$ of $\boldsymbol{Y_\pi}$ lies on the epipolar line $\boldsymbol{l'}$. Source: partially from Hartley and Zisserman (2004).

phies are used as an algebraic tool to express the geometric transformations undergone in the case of two images viewing the same scene, referred to as epipolar geometry.

A.4 The Epipolar Geometry

As its name indicates, the SfM method retrieves 3D structure out of the motion of a camera transiting a given scene. We focus first on the simple case of two images, which can be seen as two pictures of the same scene taken by a camera as it is displaced (see Fig. A.4). The first step of the method is to extract the camera pose of each image of the pair. This is achieved by applying the algebraic equations and properties yielded by the particular configuration between these two images, called epipolar geometry. In this section these equations are derived and their implications discussed.

Fig A.4 illustrates the layout of this problem with the point $\boldsymbol{X}$ being projected into two images at the image coordinates $\boldsymbol{x}$ and $\boldsymbol{x'}$. Since $\boldsymbol{x}$ and $\boldsymbol{x'}$ are corresponding views on different images of the same 3D world point $\boldsymbol{X}$ it is said that they form a point correspondence. Section 3.4 elaborates the concept of correspondence in detail.

The specific characteristics of this configuration define the epipolar geometry between two images. To describe the problem in algebraic terms a camera frame is placed for each image. In Fig. A.4 the origins of each camera frame are placed on $\boldsymbol{C}$ and $\boldsymbol{C'}$. The $z-$axis of each coordinate system is orthogonal to its image plane.

The epipolar geometry consists of the relationships which arise from the COPs $\boldsymbol{C}$ and $\boldsymbol{C'}$, the point $\boldsymbol{X}$ and its projections $\boldsymbol{x}$ and $\boldsymbol{x'}$, being all these entities coplanar: Fig. A.4

shows that $\boldsymbol{X}$, $\boldsymbol{C}$ and $\boldsymbol{C'}$ create a triangle. $\boldsymbol{x'}$ and $\boldsymbol{x}$ are located on two sides of this triangle, and the remaining side is the baseline i.e. the line $\overline{\boldsymbol{C'C}}$. The intersection of the baseline with each image is called epipole. In Fig. A.4 the epipoles are referred to as $\boldsymbol{e}$ and $\boldsymbol{e'}$. The lines $\boldsymbol{l}$ and $\boldsymbol{l'}$ that connect the epipoles with the projections of $\boldsymbol{X}$ are called epipolar lines, and the plane $\boldsymbol{\pi}$ that contains the triangle $\widehat{\boldsymbol{CXC'}}$ is named the epipolar plane. Note that the projection on the left image of every point contained by the ray $\overline{\boldsymbol{XC'}}$ falls on the epipolar line $\boldsymbol{l} = \overline{\boldsymbol{ex}}$, and vice versa. In fact, any point belonging to the plane $\boldsymbol{\pi}$ projected into the images will fall on the epipolar lines $\boldsymbol{l}$ and $\boldsymbol{l'}$.

This particular geometric configuration allows us to determine an algebraic relationship between the two images, and to express it in the form of a homogeneous matrix. This matrix is called the fundamental matrix.

A.4.1 The Fundamental Matrix

We will define the fundamental matrix with the aid of an auxiliary plane, as shown in Fig. A.5. The only condition imposed upon the auxiliary plane $\boldsymbol{\pi}$ is that none of the COPs $\boldsymbol{C}$ or $\boldsymbol{C'}$ can belong to $\boldsymbol{\pi}$. Let $\mathbf{w} = \{\boldsymbol{x_i}, \boldsymbol{x'_i}\},\ i = 0 \ldots n$ be a set of 2D point pairs in both images that correspond to the projections of the set of world points $\boldsymbol{W} = \{\boldsymbol{X_i}\},\ i = 0 \ldots n$. The set $\mathbf{w}$ is a group of corresponding features. An element $\{\boldsymbol{x_i}, \boldsymbol{x'_i}\}$ of $\mathbf{w}$ is called point correspondence. $\boldsymbol{W}$ is the set of 3D points simultaneously seen by the two images.

Let $\{\boldsymbol{x}, \boldsymbol{x'}\}$ any pair of corresponding homogeneous 2D points of the set $\boldsymbol{w}$ and $\boldsymbol{X} \in \boldsymbol{W}$ the inhomogeneous 3D point seen by this pair. Since $\boldsymbol{x}$ and $\boldsymbol{x'}$ are connected through $\boldsymbol{X}$, we can say that $\boldsymbol{x}$ is transferred to $\boldsymbol{x'}$ via the plane $\boldsymbol{\pi}$ (Fig. A.5). Algebraically this transfer is justified as follows. Since $\boldsymbol{X}$ belongs to the plane $\boldsymbol{\pi}$ it is possible to see the coordinates of $\boldsymbol{X}$ as the homogeneous 2D coordinates with respect to some 2D coordinate frame of $\boldsymbol{\pi}$. We can therefore treat $\boldsymbol{X}$ as a homogeneous 2D point which belongs to $\boldsymbol{\pi}$. The projection from plane $\boldsymbol{\pi}$ to the left image is a central projection (see Fig. A.3), and therefore there is a 2D homography H which relates $\boldsymbol{X}$ and $\boldsymbol{x}$, such that $\boldsymbol{x} = \mathrm{H}\boldsymbol{X}$. Similarly, there is a 2D homography H' which relates $\boldsymbol{X}$ and $\boldsymbol{x'}$, such that $\boldsymbol{x'} = \mathrm{H}'\boldsymbol{X}$. Hence there is a 2D homography H_π mapping $\boldsymbol{x}$ to $\boldsymbol{x'}$ since

$\boldsymbol{x'} = \mathrm{H}'\boldsymbol{X} = \mathrm{H}'\mathrm{H}^{-1}\boldsymbol{x}$:-

$$\boldsymbol{x'} = \mathrm{H}_\pi \boldsymbol{x} \tag{A.4.18}$$

where $\mathrm{H}_\pi = \mathrm{H}'\mathrm{H}^{-1}$.

Following a vectorial representation of a line (see Appendix B, Eq. B.2.5) the epipolar line $\boldsymbol{l'}$ in Fig. A.5 can be expressed as the cross product of the two points $\boldsymbol{e'}$ and $\boldsymbol{x'}$, for both belong to $\boldsymbol{l'}$. Therefore:-

$$\boldsymbol{l'} = \boldsymbol{e'} \times \boldsymbol{x'} = \left[\boldsymbol{e'}\right]_x \boldsymbol{x'} \tag{A.4.19}$$

where $[\boldsymbol{e'}]_x$ denotes the skew-symmetric matrix derived from the column vector $\boldsymbol{e'} = (e'_1, e'_2, e'_3)^T$ (Appendix B.1). The application of the Eq. A.4.18 to this result yields:-

$$\boldsymbol{l'} = \left[\boldsymbol{e'}\right]_x \mathrm{H}_\pi \boldsymbol{x} = \mathrm{F}\boldsymbol{x} \tag{A.4.20}$$

where $\mathrm{F} = [\boldsymbol{e'}]_x \mathrm{H}_\pi$ is called the fundamental matrix. Proceeding analogously with the left epipolar line $\boldsymbol{l} = \boldsymbol{e} \times \boldsymbol{x} = [\boldsymbol{e}]_x \boldsymbol{x}$, and $\boldsymbol{x} = \mathrm{H}_\pi^{-1}\boldsymbol{x'}$ yields $\boldsymbol{l} = [\boldsymbol{e}]_x \mathrm{H}_\pi^{-1}\boldsymbol{x'}$. One may verify that $[\boldsymbol{e}]_x \mathrm{H}_\pi^{-1} = \mathrm{F}^{\mathrm{T}}$ and therefore $\boldsymbol{l} = \mathrm{F}^{\mathrm{T}}\boldsymbol{x'}$.

Eq. A.4.20 holds for the whole set $\boldsymbol{W}$. To demonstrate this, let $\boldsymbol{Y} \in \boldsymbol{W}$ be a inhomogeneous 3D point which does not lay on the plane $\boldsymbol{\pi}$. For the sake of clarity in Fig. A.5 the projection of $\boldsymbol{Y}$ lies on the same epipolar lines as $\boldsymbol{X}$, but this argument also applies if points projecting to different epipolar lines are picked.

Even though $\boldsymbol{Y}$ is not on the plane $\boldsymbol{\pi}$ the line joining $\boldsymbol{C}$ and $\boldsymbol{Y}$ will intersect the plane $\boldsymbol{\pi}$ at the point $\boldsymbol{Y_\pi}$ (see Fig. A.5). Note that $\boldsymbol{y'_\pi}$ lies on the epipolar line $\boldsymbol{l'}$ since $\boldsymbol{Y_\pi}$ also belongs to the epipolar plane defined by $\widehat{\boldsymbol{CXC'}}$. Eq. A.4.18 becomes $\boldsymbol{y'_\pi} = \mathrm{H}_\pi \boldsymbol{y}$, where $\boldsymbol{y}$ and $\boldsymbol{y'_\pi}$ are the projections of $\boldsymbol{Y_\pi}$ on the left and right images respectively. Since $\boldsymbol{y'_\pi}$ lies on $\boldsymbol{l'}$ we can write $\boldsymbol{l'} = \boldsymbol{e'} \times \boldsymbol{y'_\pi} = [\boldsymbol{e'}]_x \boldsymbol{y'_\pi}$ and subsequently $\boldsymbol{l'} = [\boldsymbol{e'}]_x \mathrm{H}_\pi \boldsymbol{y} = \mathrm{F}\boldsymbol{y}$.

Note that in Eq. A.4.20 F maps points from one image to epipolar lines in the another. Since $[\boldsymbol{e'}]_x$ has rank 2 and H_π has rank 3, F has rank 2. Therefore $\det(\mathrm{F}) = 0$. Also, since F is derived from a projectivity, F is a homogeneous matrix, and it has an overall scaling. Hence F has 7 degrees of freedom, accounted for by the 8 ratios between the elements of F minus one degree of freedom fixed by the equation $\det(\mathrm{F}) = 0$.

The most important feature of F is that it correlates point correspondences $\{\boldsymbol{x}, \boldsymbol{x'}\}$ via the epipolar equation.

The Epipolar Equation

Since $\boldsymbol{x'}$ is on $\boldsymbol{l'}$, we can write $\boldsymbol{x'}^T\boldsymbol{l'} = 0$ (see Appendix B, Eq. B.2.5), which with Eq. A.4.20 leads to:-

$$\boldsymbol{x'}^T \mathrm{F} \boldsymbol{x} = 0 \tag{A.4.21}$$

Eq. A.4.21 is the epipolar equation, and it allows us to relate two corresponding images without reference to their camera matrices. F is used when working with uncalibrated images (intrinsic matrix K unknown), and its application on SfM leads to a projective reconstruction i.e. a 3D reconstruction where neither angles nor parallel lines are preserved and the scene appears distorted.
Since $\boldsymbol{e'}$ belongs to all the epipolar lines $\boldsymbol{l'_i},\ i = 0 \dots n$ we can write $\boldsymbol{e'}^T\boldsymbol{l'_i} = 0,\ i = 0 \dots n$. According to Eq. A.4.21, we thus have $\boldsymbol{e'}^T \mathrm{F} \boldsymbol{x_i} = 0$ for all $\boldsymbol{x_i},\ i = 0 \dots n$. It follows that $\boldsymbol{e'}^T \mathrm{F} = 0$, i.e. $\boldsymbol{e'}$ is the left null-vector of F. Likewise $\mathrm{F}\boldsymbol{e} = 0$ and $\boldsymbol{e}$ is the right-null vector of F.
Eq. A.4.21 is mostly used to estimate F, by means of known point correspondences (this process is described in next section), and to estimate the accuracy of a given F' over a set of point correspondences, by calculating how close to 0 the product $\boldsymbol{x'}^T \mathrm{F}' \boldsymbol{x}$ is. This product is usually called epipolar error.

F as correlation between points and lines

The relationship created by F is weak and unstable. As it has already been highlighted, F projects a point from one image to a epipolar line that contains its correspondence in the other image, i.e $\boldsymbol{l'} = \mathrm{F}\boldsymbol{x}$ and $\boldsymbol{l} = \mathrm{F}^{\mathrm{T}}\boldsymbol{x'}$. In order for these equivalences to hold it is only necessary that $\boldsymbol{x}$ and $\boldsymbol{x'}$ belong to $\boldsymbol{l}$ and $\boldsymbol{l'}$ respectively. This condition was used in the discussion of the epipolar relationships induced by $\boldsymbol{y'_\pi}$.
To illustrate this idea, let us consider two different pairs of correspondences $\{\boldsymbol{x_1}, \boldsymbol{x'_1}\}$ and $\{\boldsymbol{x_2}, \boldsymbol{x'_2}\}$ which lie on the same epipolar lines i.e. $\boldsymbol{x_i} \in \boldsymbol{l},\ i = 1, 2$ and $\boldsymbol{x'_i} \in \boldsymbol{l'},\ i = 1, 2$. Therefore we have $(\boldsymbol{x'_1})^T\boldsymbol{l'} = (\boldsymbol{x'_1})^T \mathrm{F} \boldsymbol{x_2} = 0$, since $\boldsymbol{x_2}$ belongs to $\boldsymbol{l}$ and thus $\boldsymbol{l'} = \mathrm{F}\boldsymbol{x_2}$. Similarly, $(\boldsymbol{x'_2})^T \mathrm{F} \boldsymbol{x_1} = 0$.
These results show that F will not "notice" one point from another so long as the points lie on the same epipolar line. This means that the operation $\boldsymbol{l} = \mathrm{F}\boldsymbol{x}$ can not be inverted and it is a consequence of F not being of full rank (F it is not invertible). This has

serious implications when estimating F from corresponding points, for every feature corresponding error along the epipolar line is not penalised. Therefore, the estimation of F is an ill-conditioned problem. This is a major issue when estimating the epipolar geometry and robust filters and non-linear minimisation algorithms are required as soon as noise appears in the images. Specifically, noise has been one of the main obstacles to achieve SfM in this work. Section 2.2 explains how the type of noise that this work has addressed is produced, and shows that significant research has been devoted to minimise the effect of noise in the estimation of F. In Chapter 3 the algorithms devised in this work to overcome the problem of noise are described, so that the state of the art works (Chang and Hebert (2002); Thomas and Oliensis (1999); Vidal and Oliensis (2002)) are extended to cope with the levels of noise encountered in this case.
The irreversibility in the estimation of F makes its recovery from image correspondences a difficult task. At the same time, it is the very kernel of the whole SfM process, so it has brought about many studies about its characteristics and ways of estimation (Zhang (1998)). Many of the methods reviewed in Section 2.6.2 try to propose rapid and robust solutions to work around the weak relationship created by F.
The weak relationship established by F can be narrowed down if the intrinsic parameters of the cameras are known. In this case (where the two images of study are calibrated) the matrix which governs the epipolar geometry is the essential matrix.

A.4.2 The Essential Matrix

We have established an algebraic relationship between two uncalibrated images viewing the same scene, by means of the existent projective relationship between them. However it is still possible to establish a more narrow relationship between frames, provided that we know the mathematical projection model of the camera. This relationship will be expressed by a matrix similar to the fundamental matrix, called essential matrix.
Let us redefine the points $\boldsymbol{x}$ and $\boldsymbol{x'}$ as the normalised coordinates of $\boldsymbol{X}$ with respect to the camera frames $\boldsymbol{C}$ and $\boldsymbol{C}'$ respectively. Let the respective camera matrices of the cameras in Fig. A.5 be $\mathrm{P} = [\mathrm{I} \mid 0]$ and $\mathrm{P}' = [\mathrm{R} \mid \boldsymbol{t}]$. The rotation matrix R is the rotation from the right camera to the left camera, and the translation column vector $\boldsymbol{t}$ is the translation from the right camera to the left camera, coincident with the baseline $\overline{\boldsymbol{C'C}}$. Since $\boldsymbol{x}$ and $\boldsymbol{x'}$ are the views corresponding to $\boldsymbol{X}$ and are expressed in terms of their camera frames, according to Eq. A.2.13 it follows $\boldsymbol{x} = \mathrm{P}\boldsymbol{X} = \boldsymbol{X}$ and

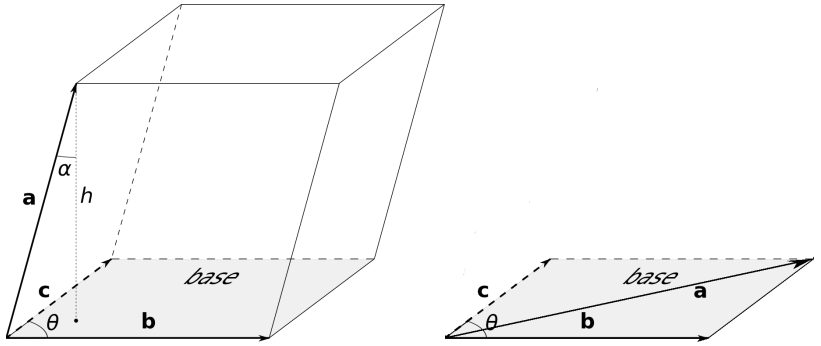


Figure A.6: Geometrically the triple scalar product can be expressed as $\boldsymbol{a} \cdot (\boldsymbol{b} \times \boldsymbol{c})$. Consequently the scalar product is the signed volume of the parallelepiped created by the vectors $\boldsymbol{a}, \boldsymbol{b}$ and $\boldsymbol{c}$. The cross-product $(\boldsymbol{b} \times \boldsymbol{c})$ represents the base of the parallelepiped and $\boldsymbol{a}$ represents the height of the parallelepiped. If the three vectors happen to be coplanar the height of the parallelepiped is zero and so is the triple scalar product. Source: partially from Wikipedia (2015*a*).

$\boldsymbol{x'} = \mathrm{P}'\boldsymbol{X} = [\mathrm{R} \mid \boldsymbol{t}]\,\boldsymbol{X} = [\mathrm{R} \mid \boldsymbol{t}]\,\boldsymbol{x}$. Therefore $\boldsymbol{x}$ and $\boldsymbol{x'}$ are related by a rigid-body transformation in the following way (as illustrated in Fig. A.5):

$$\boldsymbol{x'} = \mathrm{R}\boldsymbol{x} + \boldsymbol{t} \tag{A.4.22}$$

Taking the cross product of both sides with $\boldsymbol{t}$ (see Appendix B.1, Eq. B.1.3) in order to eliminate it on the right hand side yields:-

$$[\boldsymbol{t}]_{\boldsymbol{x}}\boldsymbol{x'} = [\boldsymbol{t}]_{\boldsymbol{x}}\mathrm{R}\boldsymbol{x} \tag{A.4.23}$$

The dot product of both sides with $\boldsymbol{x'}$ leaves:-

$$\boldsymbol{x'}^{T}[\boldsymbol{t}]_{\boldsymbol{x}}\boldsymbol{x'} = \boldsymbol{x'}^{T}[\boldsymbol{t}]_{\boldsymbol{x}}\mathrm{R}\boldsymbol{x} = 0 \tag{A.4.24}$$

The expression $\boldsymbol{x'}^{T}[\boldsymbol{t}]_{\boldsymbol{x}}\mathrm{R}\boldsymbol{x}$ can be seen as a scalar triple product between vectors $\boldsymbol{x'}$, $\boldsymbol{t}$ and $\mathrm{R}\boldsymbol{x}$ (see Appendix B, Eq. B.1.3). The triple scalar product represents the signed volume of the parallelepiped created by three vectors, as shown in Fig. A.6. In this case, the vectors considered $\boldsymbol{x'}$, $\boldsymbol{t}$ and $\mathrm{R}\boldsymbol{x}$ are coplanar, as it is easily deducible from Eq. A.4.22. The co-planarity of $\boldsymbol{x'}$, $\boldsymbol{t}$ and $\mathrm{R}\boldsymbol{x}$ leads Eq. A.4.24 to equal zero, as the parallelogram defined by them would be flat and have no volume (see Fig. A.6). Another way to justify this equality is that as $[\boldsymbol{t}]_{\boldsymbol{x}}$ is skew-symmetric, it returns 0 when pre- and

post- multiplied by a column vector (see Appendix B, Eq. B.1.4), in this case $\boldsymbol{x'}$, as it happens on the left side of Eq. A.4.24.

Therefore the epipolar geometry can be expressed again in normalised coordinates:-

$$\boldsymbol{x'}^T \mathrm{E} \boldsymbol{x} = 0 \tag{A.4.25}$$

where the matrix

$$\mathrm{E} = [\boldsymbol{t}]_{\boldsymbol{x}} \mathrm{R} \tag{A.4.26}$$

is the essential matrix. Note the similarity between Eq. A.4.25 and Eq. A.4.21. It is possible to relate the fundamental matrix F and the essential matrix E by means of Eq. A.2.11. Applying the relationship between normalised coordinates and image coordinates to Eq. A.4.21 yields:-

$$\boldsymbol{x'}^T \mathrm{K}^{\mathrm{T}} \mathrm{F} \mathrm{K} \boldsymbol{x} = 0 \tag{A.4.27}$$

which gives:-

$$\mathrm{E} = \mathrm{K}^{\mathrm{T}} \mathrm{F} \mathrm{K} \tag{A.4.28}$$

In fact, E can be thought of as a fundamental matrix of a camera whose intrinsic matrix is the identity matrix, I.

The essential matrix has the same properties as F, and it also satisfies the additional condition that its two singular values [3] are equal, (the third one being 0 as E is singular). E has 5 degrees of freedom (3 for R plus 3 for $\boldsymbol{t}$ minus the scaling factor).

The essential matrix can be computed when the intrinsic parameters of the camera are known. Its computation enables the Euclidean reconstruction of a scene, i.e. a reconstruction where right angles and parallel lines are preserved. The estimation of the essential matrix, which solves the epipolar geometry of two overlapping images, can be performed via the epipolar equation if enough number of point correspondences between the two images are known.

[3] The singular values of a real matrix A are the square roots of the eigenvalues of the product $\mathrm{A}^{\mathrm{T}}\mathrm{A}$.

A.4.3 Estimation of the Epipolar Geometry

A linear method for the extraction of the epipolar geometry is presented here. This method is explained for the estimation of E, but it can be equally applied for obtaining F.

Eq. A.4.25 is the starting point of the estimation of E. In order to estimate E we make use of the feature correspondences between two images: given a set of n homogeneous image correspondences $\{\boldsymbol{x_i}, \boldsymbol{x'_i}\}$, $i = 0 \dots n$, where $\boldsymbol{x_i} = (x_i, y_i, 1)^T$ and $\boldsymbol{x'_i} = (x'_i, y'_i, 1)^T$, the element-wise version of Eq. A.4.25 is expressed as:-

$$\begin{array}{ccccccc} x_i x'_i e_{11} & + & y_i x'_i e_{12} & + & x'_i e_{13} & + & \\ x_i y'_i e_{21} & + & y_i y'_i e_{22} & + & y'_i e_{23} & + & \\ x_i e_{31} & + & y_i e_{32} & + & e_{33} & = & 0 \end{array} \tag{A.4.29}$$

for $i = 0 \dots n$. The unknowns e_{ij} are the elements of E. Therefore, we set out a linear system $\mathrm{A}\bar{\mathrm{E}} = 0$ with a $n \times 9$ coefficient matrix A and a 9×1 column vector of unknowns $\bar{\mathrm{E}}$. A is made up of the coefficients of e_{ij} in Eq. A.4.29, and $\bar{\mathrm{E}}$ is the stacked column vector of E. Since E is homogeneous (the scale is not significant) and we need to avoid the trivial solution given by the null 9×1 column vector, we can impose on E to be unitary, $\|\bar{\mathrm{E}}\| = 1$. Therefore we are interested in the unitary subset of the null space of A. It is proven that the solution for $\mathrm{A}\bar{\mathrm{E}} = 0$ subject to $\|\bar{\mathrm{E}}\| = 1$ is obtained by applying Singular Value Decomposition (SVD) to A: if $\mathrm{A} = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$ then the solution $\bar{\mathrm{E}}$ is the last column of V (Hartley and Zisserman (2004)).

It is noticeable in Eq. A.4.29 that some coefficients of e_{ij} are product of two coordinate values whereas others are just one coordinate (or even the identity). For comparable noise in correspondence measurements the terms that are products of two measurements have their noise amplified. This causes that points with large coordinates have greater influence that others that are closer to the image centre.

To avoid this behaviour Hartley (1997) proposes a normalisation of the set of correspondences by scaling and translating the sample so that the mean of the distribution of correspondences becomes 0 and the variance is unity. Hence the new distributions of the coordinates of $\{\boldsymbol{x_i}\}$, $i = 0 \dots n$ are:-

$$\tilde{x}_i = s\,(x_i - \mu_x)\,, \quad \tilde{y}_i = s\,(y_i - \mu_y) \tag{A.4.30}$$

where $\sum_i \tilde{x}_i = \sum_i \tilde{y}_i = 0$ and $\sum_i \tilde{x}_i^2 + \sum_i \tilde{y}_i^2 = 2n$. The global variance s is estimated as:-

$$s = \frac{\sqrt{2} s_x s_y}{\sqrt{s_x^2 + s_y^2}} \tag{A.4.31}$$

These transformations also apply to the coordinates of $\{\boldsymbol{x'_i}\}$ $i = 0 \dots n$.
Once the normalised essential matrix $\tilde{\mathrm{E}}$ is computed, E can be recovered as $\mathrm{E} = \mathrm{T'}^T \tilde{\mathrm{E}} \mathrm{T}$, where $\tilde{\boldsymbol{x}}_{\boldsymbol{i}} = \mathrm{T}\boldsymbol{x_i}$, $\tilde{\boldsymbol{x}}'_{\boldsymbol{i}} = \mathrm{T'}\boldsymbol{x'_i}$.
Since E has 5 degrees of freedom, one may think that it is only necessary to have 5 good pairs of correspondences in order to set out a linear system and recover E. There exist algorithms that manage to estimate E out of only 5 pairs of correspondences (Nister (2004)). In fact, the five-point solver can extract the epipolar information also from planar scenes, although it is common in the literature to apply the same algorithms for estimating E as for F, which requires at least 7 corresponding points between the images, like in Philip (1998). However, the most popular linear method for extracting F out of correspondences is the 8-point algorithm developed by Hartley (1997). The 8-point algorithm has become the linear method of reference for computing the epipolar geometry. Indeed, its computational simplicity and the stability of its results make of the 8-point algorithm the usual choice for the initialisation of non-linear minimisation algorithms. The 8-point algorithm follows the steps described here for the estimation of F.
The minimum number of points required by an algorithm depends on the parametrisation of the problem, i.e how many parameters should be used to model the epipolar geometry. An over-parametrization is customarily advisable, mainly when non-linear methods are further applied (see Section A.7). There are two reasons for this. Firstly, a non-linear minimisation algorithm will "realise" that there is no need to move along redundant directions, thus it is not necessary to use a minimal parametrisation. Secondly, it is found experimentally that the cost function surface of an optimisation algorithm will be more complicated if a minimal parametrisation is employed, making the minimal parametrisation not recommendable.
In this work the 8-point method has been used as first estimation of the essential matrix, given its algorithmic and computational advantages, and provided the convenience of over-parametrisation over minimal parametrisation.

A.4.4 The Essential Space

The 8-point algorithm was devised for the estimation of F. If we apply it for recovering E we need to ensure that the third property of the essential matrix is fulfilled, i.e. its two non-zero singular values are equal (see Section A.4.2). Every 3×3 singular matrix that fulfils this property belongs to the space of essential matrices (called essential space, $\boldsymbol{\Theta}$). The 8-point algorithm provides a singular and homogeneous matrix E', whose singular values $\{\alpha_1, \alpha_2, 0\}$ may be such that $\alpha_1 \neq \alpha_2$. Equally, if we apply SVD to E' we have that in general $\mathrm{E}' = \mathrm{U}'\Sigma'\mathrm{V}'^T$, with U' and V' being orthogonal, but not orthonormal. We must find a matrix $\mathrm{E} \in \boldsymbol{\Theta}$ which minimises $\|\mathrm{E} - \mathrm{E}'\|$. This is achieved by defining:-

$$\Sigma = \mathrm{diag}\,\{1, 1, 0\} \tag{A.4.32}$$

and

$$\mathrm{U} = (-1) \cdot \mathrm{U}',\ \mathrm{V} = (-1) \cdot \mathrm{V}' \tag{A.4.33}$$

in case that $\det(\mathrm{U}') = -1$ or $\det(\mathrm{V}') = -1$, respectively. Tsai and Huang (1984) proved that the matrix $\mathrm{E} = \mathrm{U}\Sigma\mathrm{V}^T$ minimises the Frobenius norm of $\mathrm{E} - \mathrm{E}'$, as required.
Eqs. A.4.32 and A.4.33 are called altogether projection on to the essential space $\boldsymbol{\Theta}$.

A.4.5 Extraction of R and $\boldsymbol{t}$

Given Eq. A.4.26 and the specific characteristics of skew-symmetric and rotation matrices it is proven that there are two rotation matrices and two translation matrices whose four possible combinations fulfil Eq. A.4.26 (Hartley and Zisserman (2004); Ma et al. (2003); Szeliski (2011)). Although there are 4 possible mathematical solutions only one of them is physically plausible. This solution corresponds to the pair (R, t) that place the camera in front of the 3D point cloud.
R and $\boldsymbol{t}$ are extracted out of E as follows. In Section A.4.4 we obtained an essential matrix E which satisfies:-

$$\mathrm{E} = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}} \tag{A.4.34}$$

The matrix $\boldsymbol{t}$ is the left-null space of E and it is expressed, up to scale, as:-

$$\boldsymbol{t} = \pm \begin{bmatrix} U_{13} \\ U_{23} \\ U_{33} \end{bmatrix} \tag{A.4.35}$$

R is derived from A.4.34 as follows:-

$$\mathrm{R} = \mathrm{U}\mathrm{W}^{\mathrm{T}}\mathrm{V}^{\mathrm{T}} \text{ or } \mathrm{R} = \mathrm{U}\mathrm{W}\mathrm{V}^{\mathrm{T}} \tag{A.4.36}$$

where W represents the 90° degrees rotation matrix:-

$$\mathrm{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.4.37}$$

Considerations on the estimation of the essential matrix

Finding a solution of Eq. A.4.25 for E is difficult because E is a *correlation*, that is, it projectively maps points to lines, and as such it has no inverse mapping. As it is mentioned in Section A.4.1, small errors in the corresponding points produce big errors when extracting E that diverge when projecting E on to $\boldsymbol{\Theta}$ and computing R and $\boldsymbol{t}$. In the reconstruction step R and $\boldsymbol{t}$ will be applied to the former correspondences, subsequently increasing the displacements. In addition, since the estimation of the epipolar geometry relies on Eq. A.4.29, it is necessary a minimum number of inter-image point correspondences to ensure the stability of the method. These and other problems (there are special inter-image configurations where it is not possible to retrieve the motion between images; these cases are called degeneracies, and are tackled in Section 2.6.4) provoke that in presence of noisy and sparse sets of inter-image correspondences and with narrow baselines this problem becomes a challenge. Significant part of the present work has been devoted to find robust algorithms which overcome the instability of Eq. A.4.25 under the conditions just described.

With the retrieval of the relative motion (R and $\boldsymbol{t}$) between two images the core of the SfM process is attained. The next step is to extend the pair case to a sequence of images. The main obstacle of this step is provided by the homogeneous character of the essential matrix. This issue is addressed in this work as the problem of the scale.

A.5 Sequential SfM: the Problem of the Scale.

In Section A.4 the epipolar geometry between two images is studied and the motion from one image to another is attained, up to scale. Note that this scaling factor affects the translation $\boldsymbol{t}$. This makes sense, as it is not possible to establish from the view of an object in an image whether the object is r metres high and it is s metres away from the camera, or whether the object is kr metres high and it is ks metres away from the camera for a given scalar factor k. This is not really a problem when applying SfM over a pair of images, but it becomes a notable challenge as soon as we intend to deploy SfM on multiple views.

Let us consider a global frame of reference $\boldsymbol{O}$ and a set of images $S_m = \{I_i\}\,,\ i = 0 \dots m$ with their corresponding frames of references $\boldsymbol{O_i}$, where $\boldsymbol{O_i}$ is the frame of reference of I_i. A homogeneous 3D point $\boldsymbol{X}$ of $\boldsymbol{O}$ will thus be expressed as $\boldsymbol{X_i}$ with respect to $\boldsymbol{O_i}$. Certainly, if we estimate the epipolar geometry between $\boldsymbol{O_i}$ and $\boldsymbol{O_j}$ then Eq. A.4.22 relates $\boldsymbol{O_i}$ with $\boldsymbol{O_j}$ and we can write:-

$$\boldsymbol{X_i} = \mathrm{R_{ij}}\boldsymbol{X_j} + \boldsymbol{t_{ij}} \text{ or } \boldsymbol{x_i} = \mathrm{R_{ij}}\boldsymbol{x_j} + \boldsymbol{t_{ij}} \tag{A.5.38}$$

where $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ are the normalised coordinates associated to $\boldsymbol{X_i}$ and $\boldsymbol{X_j}$, respectively (see Section A.4.2). $\mathrm{R_{ij}}$ is the rotation from the frame of reference $\boldsymbol{O_i}$ to the frame of reference $\boldsymbol{O_j}$. Likewise, $\boldsymbol{t_{ij}}$ is the translation from the frame of reference $\boldsymbol{O_i}$ to the frame of reference $\boldsymbol{O_j}$. In Eq. A.5.38 the translation $\boldsymbol{t_{ij}}$ is up to scale, and for convenience we set $\|\boldsymbol{t_{ij}}\| = 1$. Since we are performing sequential SfM, in our work $j = i - 1$, and Eq. A.5.38 becomes:-

$$\boldsymbol{x_i} = \mathrm{R_{i(i-1)}}\boldsymbol{x_{(i-1)}} + \boldsymbol{t_{i(i-1)}} \tag{A.5.39}$$

Note that Eq. A.5.39 transforms a point $\boldsymbol{x_{(i-1)}}$ of the frame of reference $\boldsymbol{O_{(i-1)}}$ to a point $\boldsymbol{x_i}$ of the frame of reference $\boldsymbol{O_i}$, with a translation $\boldsymbol{t_{i(i-1)}}$ subject to $\|\boldsymbol{t_{i(i-1)}}\| = 1$. However the 3D reconstruction of the scene covered by S_m requires that all the camera matrices $\mathrm{P_i} = [\mathrm{R_i} \mid \boldsymbol{t_i}]\,,\ \mathrm{i} = 0 \dots \mathrm{m}$ must be defined with respect to the global frame of reference $\boldsymbol{O}$. Therefore we need to express $\boldsymbol{O_i}$ with respect to $\boldsymbol{O}$. Our problem resides in that there is not straightforward conversion from $\boldsymbol{O_i}$ to $\boldsymbol{O}$ due to the loss of scaling

information in the pair-wise extraction of the epipolar geometry, as described in Section A.4. A clear description of the problem can be found in Mouragnon et al. (2006*a*)
To show this we set without loss of generality the motion of the first image I_0 of the set S_m with respect to the global frame of reference $\boldsymbol{O}$ as $\mathrm{R}_0 = \mathrm{I}$, $\boldsymbol{t_0} = 0$ and by the definition of a camera matrix (Eq. A.2.13) we have $\boldsymbol{x_0} = \mathrm{P}_0\boldsymbol{X} = \boldsymbol{X}$, and therefore:-

$$\boldsymbol{x_0} = \boldsymbol{X} \tag{A.5.40}$$

Eq. A.5.39 establishes for $i = 1$ that $\boldsymbol{x_1} = \mathrm{R}_{10}\boldsymbol{x_0} + \boldsymbol{t_{10}}$, $\|\boldsymbol{t_{10}}\| = 1$ and we can write:-

$$\boldsymbol{x_1} = \mathrm{R}_1\boldsymbol{X} + \boldsymbol{t_1} \tag{A.5.41}$$

where $\mathrm{R}_1 = \mathrm{R}_{10}$ and $\boldsymbol{t_1} = \boldsymbol{t_{10}}$, $\|\boldsymbol{t_1}\| = 1$, is the motion of image I_1 with respect to the global frame of reference $\boldsymbol{O}$. Following this reasoning, if we set $i = 2$ in Eq. A.5.39 it yields $\boldsymbol{x_2} = \mathrm{R}_{21}\boldsymbol{x_1} + \boldsymbol{t_{21}}$, $\|\boldsymbol{t_{21}}\| = 1$ and one is inclined to write:-

$$\boldsymbol{x_2} = \mathrm{R}_{21}\mathrm{R}_1\boldsymbol{X} + \mathrm{R}_{21}\boldsymbol{t_1} + \boldsymbol{t_{21}} \tag{A.5.42}$$

which would lead to a definition of the motion of image I_2 with respect to the global frame of reference $\boldsymbol{O}$.:-

$$\boldsymbol{x_2} = \mathrm{R}_2\boldsymbol{X} + \boldsymbol{t_2} \tag{A.5.43}$$

where $\mathrm{R}_2 = \mathrm{R}_{21}\mathrm{R}_1$, $\boldsymbol{t_2} = \mathrm{R}_{21}\boldsymbol{t_1} + \boldsymbol{t_{21}}$. Whereas this transformation holds for R_2, it does not for $\boldsymbol{t_2}$, as we have forced $\|\boldsymbol{t_1}\| = \|\boldsymbol{t_{10}}\| = 1$ and $\|\boldsymbol{t_{21}}\| = 1$ but in general $\|\boldsymbol{t_{10}}\| \neq \|\boldsymbol{t_{21}}\|$ (the distance between the first and the second camera is in general not the same as the distance between the second and the third camera). We therefore need to find a mechanism that allows us to define $\boldsymbol{t_2}$ with respect to $\boldsymbol{O}$ so that we can locate a point $\boldsymbol{x_2}$ in the global frame of reference $\boldsymbol{O}$. This is achieved by taking as reference the 3D reconstructed points seen by the image I_2.
Let $q_1 = \{\boldsymbol{x_i}\}$, $i = 0 \dots n_1$ be the set of points seen simultaneously by images I_0 and I_1. It has been demonstrated that up to an overall scale it is possible to define the camera matrices of the first two images, $\mathrm{P}_0 = [\mathrm{R}_0 \mid \boldsymbol{t_0}]$, $\mathrm{P}_1 = [\mathrm{R}_1 \mid \boldsymbol{t_1}]$. Therefore q_1 can be reconstructed, since the camera matrices related to the set q_1 are known (the reconstruction step is explained in Section A.6.1). Let $q_2 = \{\boldsymbol{x_j}\}$, $j = 0 \dots n_2$ be the set of points seen simultaneously by images I_1 and I_2 and $q_{12} = \{\boldsymbol{x_k}\}$, $k = 0 \dots n_{12}$

the intersection between q_1 and q_2. The 3D points $\boldsymbol{X_k}$, $k = 0 \dots n_{12}$ associated to the normalised image points $\boldsymbol{x_k}$, $k = 0 \dots n_{12}$ have been reconstructed and are expressed in terms of the global frame of reference $\boldsymbol{O}$. Hence $\boldsymbol{X_k}$, $k = 0 \dots n_{12}$ carry the information regarding the global scale. Since the points $\boldsymbol{X_k}$, $k = 0 \dots n_{12}$ are seen by the image I_2 we can apply on them the Eq. A.2.13 to fix the translation of the camera matrix P_2 up to a global scale.

Here a crucial note should be done: in order to fix a given image I_i in a global coordinate frame $\boldsymbol{O}$, I_i must have enough common corresponding points with the image I_{i-2}. This constraint has been difficult to meet in this work, as the filters devised to rule out noise would significantly trim the sets of corresponding features between images. Chapter 3 tackles this problem, and describes the novel feature tracking system developed to work around the lack of corresponding features.

The method outlined here, which in short estimates the camera matrix of an image I_i out of 3D reconstructed points seen by I_i is called *resection* (Hartley and Zisserman (2004)).

A.5.1 Resection

Resection is the method by which the camera matrix of an image is fixed within a global frame of reference. Many algorithms are proposed for resectioning cameras (see Section 2.8), and most of them involve iterative minimisation methods. Here the linear method, also named *Direct Linear Transformation* (DLT, introduced by Abdel-Aziz and Karara (1971)) is described to set out the approach of the problem. This method is usually taken as initialisation of further iterative methods.

Given a number of point correspondences $\boldsymbol{X_i} \longleftrightarrow \boldsymbol{x_i}$, $i = 0 \dots n$ between homogeneous 3D points $\boldsymbol{X_i}$ and their correlative 2D normalised image points $\boldsymbol{x_i}$, Eq. A.2.13 can be expressed in terms of the vector cross product as $\boldsymbol{x_i} \times \mathrm{P}\boldsymbol{X_i} = 0$. We can set up a linear system by breaking down this cross product. If $\boldsymbol{p_j}$ is the row j of P, we can write:-

$$\mathrm{P}\boldsymbol{X_i} = \begin{bmatrix} \boldsymbol{p_1}^T \boldsymbol{X_i} \\ \boldsymbol{p_2}^T \boldsymbol{X_i} \\ \boldsymbol{p_3}^T \boldsymbol{X_i} \end{bmatrix} \tag{A.5.44}$$

and:-

$$\boldsymbol{x_i} \times \mathrm{P}\boldsymbol{X_i} = \begin{bmatrix} y_i \boldsymbol{p_3}^T \boldsymbol{X_i} - z_i \boldsymbol{p_2}^T \boldsymbol{X_i} \\ z_i \boldsymbol{p_1}^T \boldsymbol{X_i} - x_i \boldsymbol{p_3}^T \boldsymbol{X_i} \\ x_i \boldsymbol{p_2}^T \boldsymbol{X_i} - y_i \boldsymbol{p_1}^T \boldsymbol{X_i} \end{bmatrix} = 0 \tag{A.5.45}$$

Since $\boldsymbol{p_j}^T \boldsymbol{X_i} = \boldsymbol{X_i}^T \boldsymbol{p_j}$, this can be rewritten as a linear system:-

$$\begin{bmatrix} 0^T & -z_i \boldsymbol{X_i}^T & y_i \boldsymbol{X_i}^T \\ z_i \boldsymbol{X_i}^T & 0^T & -x_i \boldsymbol{X_i}^T \\ -y_i \boldsymbol{X_i}^T & x_i \boldsymbol{X_i}^T & 0^T \end{bmatrix} \begin{bmatrix} \boldsymbol{p_1} \\ \boldsymbol{p_2} \\ \boldsymbol{p_3} \end{bmatrix} = 0 \tag{A.5.46}$$

The 3 rows of the coefficient matrix are linearly dependent, so we may reduce the system to:-

$$\begin{bmatrix} 0^T & -z_i \boldsymbol{X_i}^T & y_i \boldsymbol{X_i}^T \\ z_i \boldsymbol{X_i}^T & 0^T & -x_i \boldsymbol{X_i}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{p_1} \\ \boldsymbol{p_2} \\ \boldsymbol{p_3} \end{bmatrix} = 0 \tag{A.5.47}$$

This system is of the form $\mathrm{A}\bar{\mathrm{P}} = 0$ where A is a $2n \times 12$ matrix and each $\boldsymbol{p_i}$ a 4-vector. The vectors $\boldsymbol{p_i},\ i = 0 \ldots 3$ make up $\bar{\mathrm{P}}$, the stacked column vector of P. Like in Section A.4.3, it is necessary to normalise the data to obtain a robust estimation of $\bar{\mathrm{P}}$. The solution for Eq. A.5.47 is achieved in a similar manner as Eq. A.4.29.

Resection enables the definition of all the camera poses of a given sequence with respect to a single coordinate frame. Now it is possible to use the knowledge of the motion between cameras to obtain a 3D mapping of a scene covered by a sequence of cameras.

A.6 Reconstruction

Once the camera poses have been extracted, the second phase of SfM takes place and the structure is recovered. With the knowledge of $\mathrm{P_i} = [\mathrm{R_i} \mid \boldsymbol{t_i}]\,,\ \mathrm{i} = 0 \ldots \mathrm{m}$ of a sequence of images $S_m = \{I_i\}\,,\ i = 0 \ldots m$ and with the projections $\boldsymbol{x_i},\ i = 0 \ldots n$ of the 3D points $\boldsymbol{X_i},\ i = 0 \ldots n$ over such sequence we can establish a simple linear system of equations and extract the depth Z of each 3D point. This problem is called *triangulation*. The image points $\boldsymbol{x_i},\ i = 0 \ldots n$ represent the projection rays from $\boldsymbol{X_i},\ i = 0 \ldots n$ to the cameras $\boldsymbol{I_i},\ i = 0 \ldots m$. As shown in Fig. A.7, the lines from the COPs through the image points $\boldsymbol{x_i},\ i = 0 \ldots n$ to the 3D points $\boldsymbol{X_i},\ i = 0 \ldots n$ create a set of triangles

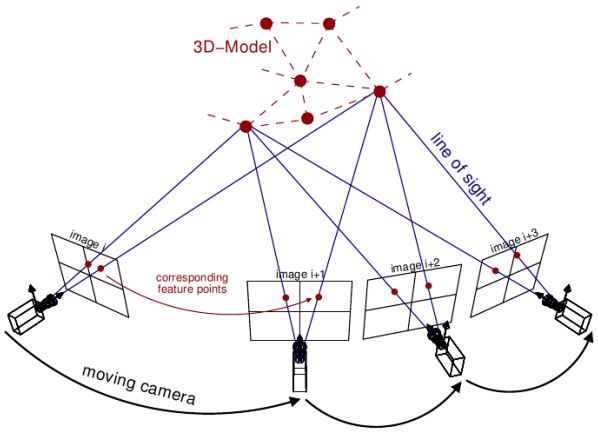


Figure A.7: The triangulation problem. Points in the 3D world are simultaneously seen in various positions of a moving camera, generating 2D views on the image sequence, which can be triangulated.

throughout the sequence. If two or more cameras view a 3D point $\boldsymbol{X}$ then it is possible to *triangulate* $\boldsymbol{X}$, obtaining as a result its depth Z.

As with *resection* method, much research has been done on different approaches for the *triangulation* method (Section 2.7). Here the linear method (DLT) in a noise-free configuration is derived, so as to provide a framework for Chapters 3 and 4.

A.6.1 Triangulation

Fig. A.7 depicts the problem graphically between multiple camera positions. As explained in Section A.2, P relates the 3D world coordinates of a point with the coordinates of its projection in an image.

Let $\boldsymbol{x} = (\tilde{x}, \tilde{y}, \tilde{z})$ be a homogeneous point in an image and $\boldsymbol{X} = \left(\tilde{X}, \tilde{Y}, \tilde{Z}, \tilde{W}\right)$ its corresponding homogeneous world point such that:-

$$\boldsymbol{x} = \mathrm{P}\boldsymbol{X} \tag{A.6.48}$$

Let $\boldsymbol{p_i}$ be the row i of P. Taking into account the relationship given by Eq. A.1.1 for both $\boldsymbol{x}$ and $\boldsymbol{X}$, we can rewrite Eq. A.6.48 and break it down per image coordinates:-

$$\begin{aligned} x &= \frac{\tilde{x}}{\tilde{z}} = \frac{\boldsymbol{p_1}\cdot\boldsymbol{X}}{\boldsymbol{p_3}\cdot\boldsymbol{X}} \\ y &= \frac{\tilde{y}}{\tilde{z}} = \frac{\boldsymbol{p_2}\cdot\boldsymbol{X}}{\boldsymbol{p_3}\cdot\boldsymbol{X}} \end{aligned} \tag{A.6.49}$$

This can be reorganised as:-

$$\begin{aligned}(x\boldsymbol{p_3} - \boldsymbol{p_1})\,\boldsymbol{X} = 0 \\ (y\boldsymbol{p_3} - \boldsymbol{p_2})\,\boldsymbol{X} = 0\end{aligned} \tag{A.6.50}$$

Therefore each view of a 3D point provides two equations for the estimation of $\boldsymbol{X}$. Given $n \geqslant 2$ views of a 3D point over a sequence of images we can then set up a linear system for each 3D point $\boldsymbol{X}$:-

$$\mathrm{A}\boldsymbol{X} = 0 \tag{A.6.51}$$

where A is a $2n \times 4$ matrix and $\boldsymbol{X}$ is a 4×1 column vector. Since $\boldsymbol{X}$ is homogeneous we are interested in the unitary subset of the null space of A, and it is found by applying SVD to A.

Although the SfM method as such finishes with the reconstruction of the scene, the results obtained are usually suboptimal and it is common to carry out one last step to jointly refine the estimations of the 3D points $\boldsymbol{X_i},\ i = 0 \ldots n$ and the camera poses $\mathrm{P_i}$. This step is performed by applying the *bundle adjustment* (BA) method.

A.7 Bundle Adjustment

Bundle adjustment (BA) reduces to the minimisation of the total reprojection error between the image locations of observed and predicted image points (Madsen et al. (2004)). The name of this method refers to the bundles of light rays originating from each 3D feature and converging on each camera's optical centre (see Fig. A.7). If the image error is zero-mean Gaussian, it is proven that BA is the Maximum Likelihood Estimator of the 3D reconstruction and camera poses sought.

Before introducing BA we should first define the variable that will be minimised, the reprojection error.

A.7.1 Reprojection Error

BA needs to measure how different a estimated set of camera poses and structure is from the ideal solution. The metrics for this is given by the reprojection error.

Given a cloud of 3D points $T_n = \{\boldsymbol{X_j}\},\ j = 0 \ldots n$, a set of camera matrices $S_m = \{\mathrm{P_i}\},\ i = 0 \ldots m$ where $\mathrm{P_i}$ is the camera matrix of image i and a set of views $B_{mn} =$

$\{\boldsymbol{x_{ij}}\}$, $i = 0 \dots m$, $j = 0 \dots n$, we define the reprojection error shed by $\boldsymbol{X_j}$ over $\boldsymbol{x_{ij}}$ as the Euclidean distance between the projection $\overline{\boldsymbol{x}}_{\boldsymbol{ij}} = \mathrm{P_i}\boldsymbol{X_j}$ and the measurement $\boldsymbol{x_{ij}}$:-

$$\epsilon_{ij} = d\left(\overline{\boldsymbol{x}}_{\boldsymbol{ij}},\, \boldsymbol{x_{ij}}\right) = d\left(\mathrm{P_i}\boldsymbol{X_j},\, \boldsymbol{x_{ij}}\right) \tag{A.7.52}$$

where $d(\boldsymbol{a}, \boldsymbol{b})$ denotes the Euclidean distance between two points $\boldsymbol{a}$ and $\boldsymbol{b}$.
The total reprojection error is the sum of the reprojection errors of the cloud of 3D points T_n over the set of views B_{mn} throughout the sequence of cameras S_m:-

$$\sum_{j=0}^{j=ni=m}\sum_{i=0} v_{ij} d\left(\mathrm{P_i}\boldsymbol{X_j},\, \boldsymbol{x_{ij}}\right) \tag{A.7.53}$$

where v_{ij} denotes the binary variables that equal 1 if point $\boldsymbol{X_j}$ is visible in image i and 0 otherwise.
The definition of the reprojection error allows us to specify which minimisation algorithm should be employed when performing BA. The most appropiate algorithm is the Levenberg–Marquardt algorithm (L-M).

A.7.2 Levenberg–Marquardt Algorithm

The BA method must simultaneously optimise T_n and S_m so that the total reprojection error is minimised. The problem therefore consists of minimising the cost function derived from Eq. A.7.53, specifically:-

$$\min_{\mathrm{P_i}, \boldsymbol{X_j}} \sum_{j=0}^{j=ni=m}\sum_{i=0} v_{ij} \|\mathrm{P_i}\boldsymbol{X_j} - \boldsymbol{x_{ij}}\|^2 \tag{A.7.54}$$

The cost function A.7.54 represents a least square fitting problem, where the parameters of T_n and S_m are adjusted to fit best Eq. A.2.13. Amongst the numerical methods that apply to non-linear problems, the Levenberg-Marquardt algorithm (L-M) has been found to be the most convenient for this case (Madsen et al. (2004)). L-M is an iterative parameter minimisation method for non-linear problems, most commonly least-square curve fitting. Conceived as an interpolation between Gauss-Newton (GN) method and Gradient Descent (GD) algorithm, L-M takes the best features of both and avoids their shortcomings. The damped version of the normal equations allows L-M to have the robustness of GD while being nearly as fast as GN. The over-parametrisation issue

is smoothly handled thanks to the specific configuration of L-M. Additionally, thanks to its normal equations zeros pattern L-M can be employed in a sparse fashion, from which software implementations gain tremendous computational benefits. All these characteristics make of L-M the most suitable algorithm for applying BA to a 3D point cloud over a sequence of cameras.

As numerical algorithm L-M may end up in a local minimum if the initial estimations T_n and S_m are not close enough to the solution. It is therefore crucial to arrive at this stage with a good initial guess. This is the reason why an appropriate method for the computation of the epipolar geometry in real situations is decisive in SfM. Chapter 3 describes in detail the methods and devices utilised in order to attain SfM from noisy images released by a low-budget mobile platform. These methods extend prior work of Chang and Hebert (2002) on noise and Rohith et al. (2013) on feature tracking, as they enable processing highly noisy images in an unexplored context of image sequences taken by an omnidirectional platform (Oliveira et al. (2009)).

Appendix B

Algebraic Definitions

B.1 Hat operator and cross product

The hat operator is widely used in 3D vision. It is defined as follows: given a 3-vector $\boldsymbol{a} = (a_1, a_2, a_3)^T$, the hat operator $[\boldsymbol{a}]_x$ is:-

$$[\boldsymbol{a}]_x = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{B.1.1}$$

$[\boldsymbol{a}]_x$ (sometimes denoted as $\hat{\boldsymbol{a}}$) is a skew-symmetric matrix ($[\boldsymbol{a}]_x^T = -[\boldsymbol{a}]_x$) and as such has special properties. $[\boldsymbol{a}]_x$ is singular and $\boldsymbol{a}$ is its null-vector (right or left). Hence, a 3×3 skew-symmetric matrix is defined up to scale by its null-vector, and $[\boldsymbol{a}]_x \boldsymbol{a} = 0$. It is easy to demonstrate that $rank([\boldsymbol{a}]_x) = 2$.

Along with other applications, the hat operator is used in linear algebra to represent the cross product as a matricial product. The cross product of two 3-vectors $\boldsymbol{a} \times \boldsymbol{b}$ is:-

$$\boldsymbol{a} \times \boldsymbol{b} = \begin{bmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{bmatrix} \tag{B.1.2}$$

Therefore, the cross product can be expressed as a product between $[\boldsymbol{a}]_x$ and $\boldsymbol{b}$ as:-

$$\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_x \boldsymbol{b} = \left(\boldsymbol{a}^T [\boldsymbol{b}]_x\right)^T \tag{B.1.3}$$

A corollary to this result is that when a skew-symmetric matrix $[\boldsymbol{a}]_x$ is pre- and post-

multiplied by a vector $\boldsymbol{b}$ the result is zero, since $\boldsymbol{b}$ is orthogonal to $\boldsymbol{a} \times \boldsymbol{b}$:

$$\boldsymbol{b}^T \left[\boldsymbol{a}\right]_x \boldsymbol{b} = \boldsymbol{b}^T(\boldsymbol{a} \times \boldsymbol{b}) = 0 \tag{B.1.4}$$

B.2 Line between two points

In homogeneous coordinates, the line $\boldsymbol{l}$ that passes through two given points $\boldsymbol{a}$ and $\boldsymbol{b}$ can be represented by:-

$$\boldsymbol{l} = \boldsymbol{a} \times \boldsymbol{b} \tag{B.2.5}$$

Since $\boldsymbol{a} \cdot (\boldsymbol{a} \times \boldsymbol{b}) = \boldsymbol{b} \cdot (\boldsymbol{a} \times \boldsymbol{b}) = 0$ it is clear that the line $\boldsymbol{l} = \boldsymbol{a} \times \boldsymbol{b}$ accomplishes $\boldsymbol{a}^T\boldsymbol{l} = \boldsymbol{b}^T\boldsymbol{l} = 0$