Imperial College
London

Applying Information Theory to Efficient SLAM

Margarita Chli

Department of Computing, Imperial College London

Presented for the degree of Doctor of Philosophy
and the Diploma of Imperial College London.

October 2009

Acknowledgements

This thesis would not have been possible without the help and encouragement of a number of people around me during my years as a PhD student.

First and foremost, I wish to thank my supervisor Andrew Davison for his guidance and support. His generous offering of ideas and expertise throughout this work have been the most valuable source of knowledge and inspiration, while his passion for research and unreserved kindness have made me fight harder. All this put in a few words, he has been the best supervisor a student could wish for.

I am deeply grateful to José María Martínez Montiel for the intriguing discussions and the faith he has shown in this research. The numerous conversations I had with friends and collaborators have been a major stimulus for research and experimentation, and I thank them for that. My incredible office mates have been most supportive throughout the ups and downs of my PhD making my days in London sunnier than they would otherwise be.

Lastly, I wish to thank my parents and my sister whose love and patience carried me through this work.

Abstract

The problem of autonomous navigation of a mobile device is at the heart of the more general issue of spatial awareness and is now a well-studied problem in the robotics community. Following a plethora of approaches throughout the history of this research, recently, implementations have been converging towards vision-based methods. While the primary reason for this success is the enormous amount of information content encrypted in images, this is also the main obstacle in achieving faster and better solutions.

The growing demand for high-performance systems able to run on affordable hardware pushes algorithms to the limits, imposing the need for more effective approximations within the estimation process. The biggest challenge lies in achieving a balance between two competing goals: the optimisation of time complexity and the preservation of the desired precision levels. The key is in *agile manipulation of data*, which is the main idea explored in this thesis.

Exploiting the power of probabilistic priors in sequential tracking, we conduct a theoretical investigation of the *information* encoded in measurements and estimates, which provides a deep understanding of the map structure as perceived through the camera lens. Employing Information Theoretic principles to guide the decisions made throughout the estimation process we demonstrate how this methodology can boost both the efficiency and consistency of algorithms. Focusing on the most challenging processes in a state of the art system, we apply our Information Theoretic framework to local motion estimation and maintenance of large probabilistic maps. Our investigation gives rise to dynamic algorithms for quality map-partitioning and robust feature matching in the presence of significant ambiguity and variable camera dynamics. The latter, is further explored to achieve scalable performance allowing dense feature matching based on concrete probabilistic decisions.

Contents

1

Introduction

Practical spatial awareness for autonomous robots and artificial systems is gradually becoming a reality and forms the backbone of autonomous navigation. One of the most important aspects of this is Simultaneous Localisation and Mapping (SLAM) which addresses the following question:

How can a body navigate in a previously unknown environment while constantly building and updating a map of its workspace using on-board sensors only?

The capability of images to supply a wealth of data, together with the compactness and affordability of cameras, have established vision as the dominant choice of sensing in today's systems. Despite the long literature of approaches to the problem of SLAM and the plethora of implementations, robotic devices have not yet quite left the laboratory to perform everyday tasks. The work in this thesis is about investigating how the best of these methods work from a scientific perspective in order to understand how we can deal with real world data in a manageable way. As a means of exploring the theoretical aspects of existing algorithms, we employ Information Theory to provide an insight into the quality and efficiency of their performance with the prospect to guiding research towards better, even optimal algorithms.

1.1 The Progress and Vision of SLAM Research

Spatial awareness is a key requirement for autonomous robotics and a wealth of other sensor-carrying systems. In particular, the 'solution' to SLAM can provide the ability of self-controlled navigation attracting major research interest across the world. Realising the inherent uncertainties in all sources of real-world data, it is now well accepted across the robotics, vision and artificial intelligence communities that probabilistic inference provides the best way to handle them, leading to the probabilistic formulations of the navigation problem of SLAM.

Throughout the years, a diversity of implementations has emerged in the literature triggered by the applicability of systems in both specialised sectors and everyday life. Whether the question is navigation of an indoor domestic robot, an all-terrain mining vehicle or an underwater exploring device, SLAM forms the core problem that has to be solved. What differentiates implementations are the means employed to solve this problem. The nature of the environment and the application requirements are decisive factors in the choice of sensing modalities and process models to be used. When highly accurate estimates are a prerequisite for example, laser range-finders can be used, whereas if affordability is an issue then cameras are a better option. However, the growing need for generally compatible solutions has let to the establishment of cameras as the most popular sensor choice at present.

The unique ability of cameras to capture information-rich snapshots of a scene provides the potential of quality of performance in systems, however it was not until the advances in hardware that processing of visual data has become feasible. Substituting laser range-finders with camera rigs, the robotics research community has started moving towards computer vision algorithms to solve the estimation problem. In an impressive breakthrough, it has been shown that even using a single hand-held, monocular camera it is possible to estimate the trajectory followed in real-time. While Structure From Motion (SFM) has been studied extensively in photogrammetry, this was the first time that the basic idea of estimating the scene from individual images has been performed online. Monocular SLAM today has seen great improvement with state of the art systems being able to map small-scale environments, however there is still a lot to be done before truly robust and dynamic performance becomes reality.

The blend of robotics and vision algorithms through SLAM is only at the beginning, revealing new research avenues towards general and advanced systems. The prospect of importing amazing techniques from computer vision like dense matching and scene reconstruction, can enrich the 'perception' of robots and give rise to fascinating applications for embedded platforms. However, the employment of such techniques and more generally the management of visual data on top of maintaining a probabilistic map, is still a computationally intensive task. With online performance

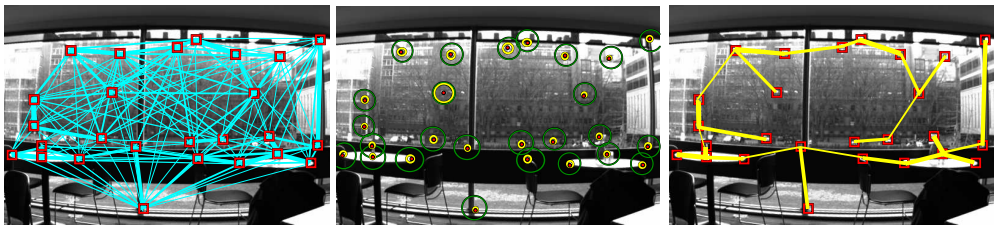
comprising a requirement in most modern systems, great challenges are imposed from the scientific perspective: while an accurate solution needs careful, extensive processing of all the available data, this is not possible within a real-time framework, which inevitably leads into a series of necessary *approximations*. Such approximations range from the assumptions made on the robot motion, the scene structure, the underlying probability distributions and the perception of the world as a set of small, measurable entities called 'features'.

In an attempt to cope with the rising demand for fast motion and bigger, denser maps, modern SLAM systems often employ ad hoc approximations to the full problem. These are usually tailored to specific tasks, lacking both generality and theoretical investigation. In fact, the performance of algorithms depends heavily on both the extend and quality of sparsifications, determining the speed, robustness and precision of the implementation. As a result, the challenge we face at this point in the history of SLAM, is to balance the benefits and losses involved in such approximations. Following this rationale, here we use Information Theory as a natural extension to Probability Theory which provides the ability to quantify uncertainty and information during the estimation process. While there has been little investigation into the value of this approach in improving the performance of systems, it has a much wider role of play in general Bayesian inference problems. Applying Information Theory in this context, we aim to explore our theoretical interest on understanding how SLAM methods really work which in turn can drive progress towards practical and advanced systems.

1.2 Aims and Goals of this Work

Across the span of existing implementations, the universal concept of the underlying 'solution' of SLAM is to establish correspondences of feature measurements made throughout the motion of the sensor-carrying body and use them to sequentially estimate the current pose. Normally, we identify features as salient aspects of the raw sensor data and use them to serve as landmarks in the constantly expanding map which is used to infer the relative trajectory of the moving body. The difficulty lies in uncertainty inherent in the body's interactions with the real world through noisy sensors and actuators. Every state estimate and every sensor measurement is uncertain. Therefore, managing computational complexity, preserving consistency in the map and coping with online data rates are issues that the SLAM community has been dealing with since the emergence of this field.

Generally, SLAM systems have now reached considerable maturity, but the exclusive use of visual data in this context is still in its infancy having great potential for improvement. In this work, we tackle challenges faced in visual SLAM and more



(a) MI graph (b) Robust matching (c) Underlying structure

Applying Information Theory to quantify uncertainty and information in SLAM. In this work we apply Information Theoretic analysis on the probabilistic estimates we maintain in SLAM and employ it to understand how algorithms work, guiding research towards more efficient and robust algorithms. The figures above give a preview of what will follow, where (a) shows a complete graph of Mutual Information links between features in measurement space, (b) shows an example of search for matching consensus, while (c) depicts the underlying tree structure in this distribution of features used to infer overall map structure.

specifically, we choose to conduct our investigations based on a monocular SLAM system as the most general and perhaps most difficult case under this category. However, nothing about the ideas and algorithms developed in this thesis precludes their application on more complex sensing arrangements. A freely moving, hand-waved camera, while providing great versatility, makes the problem far less constrained as the intentions of the carrier are hard to model. While powerful monocular implementations now exist in the literature, the need for bigger and better solutions drives research towards more effective but at the same time, quality approximations.

The richness of priors encoded in an image which is to be accredited for the success of vision-based solutions to date, is twofold: while enough data is available to infer the trajectory of the robot reliably with respect to the mapped environment, the processing load involved in converting this data into 'perception' is often overwhelming imposing a bottleneck on online performance. The desire to build more accurate maps under general tracking conditions pushes algorithms towards more conservative, careful processing. On the other hand, the strong priors available in high frame rate tracking can result to more accurate predictions in SLAM, therefore this drives investigation towards more efficient and faster algorithms able to run on such limited time budget. On top of this, the need for larger and denser maps comes to add to the challenge of intelligent manipulation of the prior data available in visual SLAM. With this in mind, this thesis aims to provide a comprehensive insight into the value of priors within the context of SLAM which will form the basis of the decisions we are making when approximating the full problem to meet the demanding requirements in a modern system.

Processing an extra piece of data is bound to refine our knowledge of the uncertain state of the camera and the scene. To assess whether or not it is indeed worth making

the effort to process it, one has to ask *how much more* information is this data is able provide. The answer is far from trivial as it is both relative to some frame of reference and dynamic with respect to a variety of influential factors. As a means of quantifying the amount of this information, we employ Information Theory to assign a value on the additional knowledge that each new piece of data is predicted to give. Through our investigation on the informational value of candidate measurements in the context of SLAM, we aim to provide a general understanding of the relationships between the members of the map and the camera state. The ultimate goal is to exploit the knowledge about these correlations at runtime to devise both efficient and robust approximations of the SLAM processes.

In order to exploit the probabilistic predictions we so carefully maintain in SLAM, here we use Information Theoretic techniques to make decisions not about the optimal motion strategy as done before, but to guide *where* to look for information and *how* to use it. Figure 1.2 gives small taster examples of the work presented in the rest of this thesis. Tackling the main components comprising a modern visual SLAM system, we apply an Information Theoretic methodology to both feature matching and map-partitioning. Good local tracking is a vital asset in a high-performance system since fusing erroneous estimates or missing matches for features that have actually been present can result to either inconsistencies in the map or tracking failure. While current solutions perform successfully in the presence of identifiable and distinctive features, here the focus is to explore the use of priors for efficient matching in the presence of outliers and generally more challenging tracking conditions.

All visual SLAM systems depend greatly on the ability to repeatedly measure visual features from a wide range of viewpoints, therefore tracking more features per frame is bound to provide more precise estimates about the camera motion. However, more data translates into more processing which accentuates the need for effective approximations to the feature matching process. Studying the structure of feature correlations through an Information Theoretic perspective, we explore their power in driving scalable matching, but also submapping which is now a heavily employed method for approximating the structure of large maps.

1.3 Organisation

The following chapter (Chapter 2) provides a general background on the methodologies used to attack the problem of SLAM. Discussing seminal works following the recognition that consistent probabilistic mapping was a fundamental problem in robotics, attention is quickly drawn to state of the art systems. The contributions of this thesis are put into context following an analysis of the challenges faced in modern

visual SLAM systems.

Chapter 3 introduces the reader to the framework of Bayesian monocular SLAM employed in the system used to demonstrate the ideas and effectiveness of algorithms presented in this thesis. This system has formed the basis of research and experimentation described in this work as a means of identifying and attacking the current challenges faced in state of the art systems.

The core theoretical concepts used throughout the rest of the thesis are described in Chapter 4. This chapter discusses the motivation behind the use of Information Theoretic principles in SLAM, developing the basic ideas of this field within the visual SLAM framework. Evaluating the knowledge encoded in the probabilistic predictions we are able to make in sequential tracking, we give taster examples of the power of Information Theoretic measures fully exploited in subsequent chapters.

Chapter 5 discusses the evolution of a fully Bayesian algorithm for frame-to-frame matching, which we call Active Matching. Driving decisions based on Shannon Information Theory while searching for global consensus, the algorithm achieves efficient and robust matching throughout a frame maintaining the multiple hypotheses naturally arising in real tracking scenarios. The capacity of this methodology is pushed to the limits exploring its strengths and weaknesses through an extensive performance analysis.

Chapter 6 tackles the issue of constantly expanding SLAM maps which imposes computation and consistency limitations on SLAM systems aiming for large maps either due to denser representations of the environment or to extended areas of tracking. Manipulating the correlations progressively built within the tracking filter, we illustrate how Information Theoretic principles can be employed to guide effective partitioning into submaps achieving quality approximations to the full SLAM map.

Building on the experience of previous chapters in manipulating information within SLAM, Chapter 7 tackles the problem of increasing complexity in dense feature matching. The biggest challenge in such scenarios is to contain processing within the real-time allowance. Dense matching strategies are therefore tailored to optimise for processing time ignoring part of the available information, in essence trading accuracy with speed. Aiming to bring the robustness of fully probabilistic techniques towards the same performance standards as randomised strategies, this chapter describes how Active Matching can be used as a prototype on top of which approximations are made based on an Information Theoretic analysis. The CLAM algorithm emerging from this research achieves online, dense matching through a series of probabilistic and information-guided decisions.

Finally, Chapter 8 closes this thesis summarising the achievements of the work presented and giving future work directions.

1.4 Publications

The biggest part of the work presented in this thesis has been peer-reviewed and presented in conferences. The journal and conference publications emerged from this research are listed below.

Chli and Davison [2008a]: Active Matching. In *Proceedings of the* 10^{th} *European Conference on Computer Vision (ECCV)*, Marseille, France, October 2008.

Chli and Davison [2008b]: Efficient Data Association in Images using Active Matching. In the *workshop 'Inside Data Association' of Robotics: Science and Systems (RSS)*, Zurich, Switzerland, June 2008.

Chli and Davison [2009a]: Automatically and Efficiently Inferring the Hierarchical Structure of Visual Maps. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009.

Chli and Davison [2009b]: Active Matching for Visual Tracking. In *Special Issue on 'Inside Data Association' of Robotics and Autonomous Systems*, 2009.

2

Related Work

This chapter intends to provide the reader with a panorama of the state of the art approaches to the problem of SLAM with particular focus on the challenging case of tracking with a single, freely moving camera. The contributions of this thesis are put into context via a discussion of the individual components comprising a modern, high-performance system as they have been developed following a series of historic advances throughout the years. Giving a background of both established and more recent methodologies we discuss their relative strengths and weaknesses, identifying the questions still open in this area.

2.1 Simultaneous Localisation And Mapping

The process of building a map of the surroundings of a mobile robot while estimating its relative pose solely on the basis of feeds coming from on-board sensors, is what we refer to as the Simultaneous Localisation And Mapping (SLAM) problem. While initial attempts to solve this problem date back more than 20 years, this area has been highly active since. Recent advances focusing on efficient implementations are able to exhibit real-time performance while demonstrating robustness and maintaining the

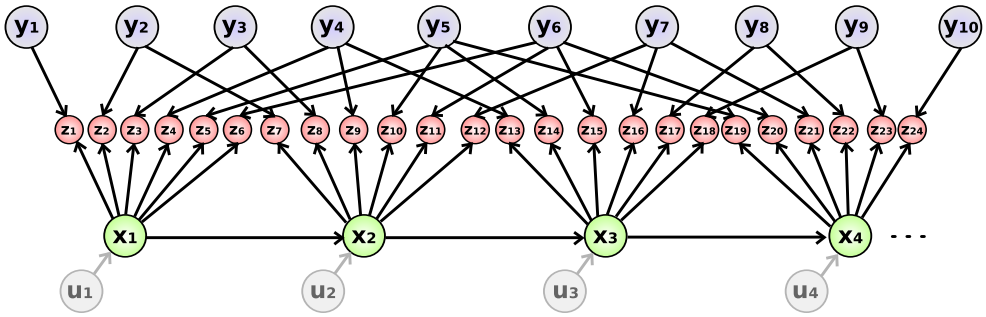


Figure 2.1: Formulation of the SLAM problem as a graph. The features in the world **y** are observed using the on-board sensor(s) via measurements **z** made from the corresponding pose **x** while the robot is moving. In some implementations the odometry input **u** which controls the robot movement is also available and taken into account. In monocular SLAM the on-board sensor is a single, hand-held camera which implies there is no odometry information. A motion model is used instead to predict camera motion and the observations of landmarks comprise of image patches, as observed from each camera viewpoint. Different approaches to SLAM attempt to optimise this graph by satisfying as many constraints between nodes as possible, often making approximations to meet real-time limits.

consistency of the map constructed.

Despite the long history of research in this field, we have only recently gained a new, general understanding of the nature of the problem. A graphical representation of SLAM as a Bayesian network is depicted in Figure 2.1, capturing the conditional dependencies formed between features of the world as perceived from different poses of the robot. It is now understood that via a full, global optimisation of this graph the best solution to SLAM can be achieved such that the consistency of the dependency constraints is maximised between the robot trajectory and the map built. This batch procedure of brute-force optimisation is often referred to as *bundle adjustment* in the visual SLAM literature, adopted from the field of photogrammetry where this technique has a long history.

The work by Thrun and Montemerlo [2006] is an example of a standard graphical formulation approach to the problem of SLAM, hence they name their algorithm GraphSLAM. Inspired by the work on globally consistent alignment of laser range scans of Lu and Milios [1997], they translate the data dependencies into a graph of nonlinear quadratic constraints. Following a nonlinear least-squares optimisation, they can resolve these constraints into a maximum likelihood map of landmarks and corresponding robot poses.

Performing bundle adjustment over the whole graph of poses and all the features ever measured is a computation-hungry process which grows constantly as more data is collected. As a result a full, batch optimisation can only be sustained online for small data sets so such methods are usually restricted to offline implementations. However,

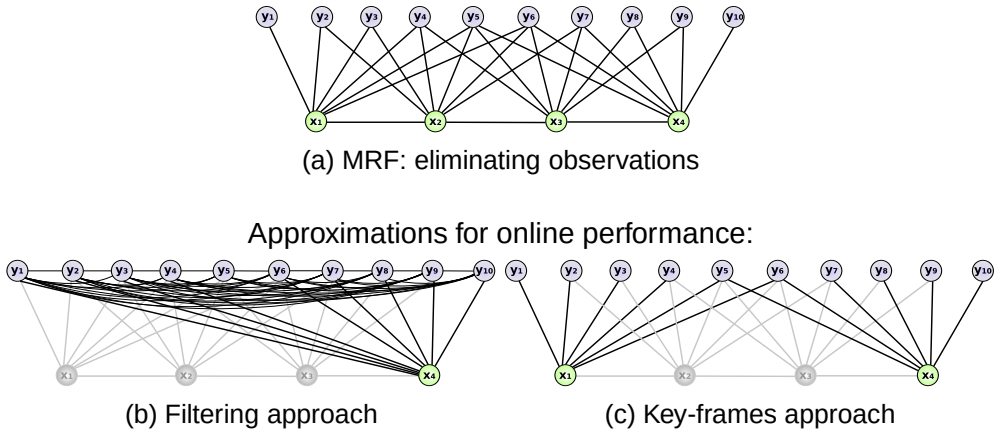


Figure 2.2: The best solution to SLAM is a full graph optimisation (bundle adjustment) of the Markov Random Field formulation depicted in (a) which arises from elimination of the observation nodes. The constantly expanding set of landmarks and poses incorporated in the graph and the costly processing of full optimisation render online performance infeasible, highlighting the need for sparsification techniques. The two most successful approaches for real-time performance are: the traditional filtering approach in (b) where the state of the map is summarised in a state vector and associated covariance with respect to the last pose, and the 'key-frames' methodology which chooses to retain the most representative poses along with their dependency links subject to optimisation, while ignoring all other measurements and poses.

if the goal is real-time localisation and mapping, often a requirement in modern systems, sparsifications and approximations to the full graph formulation of SLAM are necessary. The goal is then to estimate the current momentary pose of the robot, while a map of the environment is built incrementally. Several implementations approach real-time performance from different perspectives, but the main two axes spanning the spectrum of SLAM algorithms are the *filtering* and *key-frames* approaches.

2.1.1 Filtering vs. Keyframes for Real-Time Performance

Figure 2.2 shows a graphical representation of the two main sparsification methodologies used in online systems with respect to the Markov Random Field (MRF) graph of the SLAM problem. This is equivalent to the moralised graph of the full SLAM problem illustrated in Figure 2.1 with implicit representation of the feature measurements. It has been realised that for online, sequential positioning and mapping it is necessary to make approximations to cut down processing costs. The quality of these approximations determines the closeness of the approach to the global solution.

While bundle adjustment seeks to fulfil the majority of the constraints imposed between robot poses and features in the world as depicted in Figure 2.2(a), a filtering approach reduces this graph by summarising past experience in a state representation of a vector with an associated probability distribution with respect to the last estimated

robot pose. Marginalising out past camera poses in the MRF induces *correlation* links between the features in the 3D map which will be the key subject of analysis throughout the rest of this thesis, leading to a broad understanding of the scene and hence the design of efficient and robust algorithms.

From a totally different point of view, a key-frames approach does retain past robot poses and their constraints with the world, however it instead chooses to sparsify the problem by 'forgetting' intermediate poses together with their landmark dependency links. The idea here is to preserve the most representative poses along the trajectory and subject these to repeated global optimisation, simultaneously refining the scene geometry encoded in the landmark position estimates with respect to the motion of the robot. As this is basically a sparsified bundle adjustment approach it benefits from the closeness it provides to the full graph optimisation, however both quality and speed depend heavily on the approximation made. Maintaining fewer nodes in the optimisation or restricting optimisation within a sliding window of poses are both popular approaches used to improve the time complexity of this method.

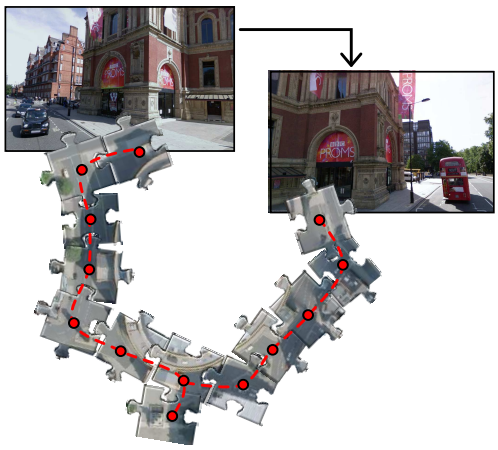
2.1.2 Components of a Modern High-Performance System

Irrespective of the sparsification methodology chosen or the application targeted, it has now become apparent that a SLAM system aiming for online and robust performance needs to be equipped with a standard set of components as presented visually in Figure 2.3. Namely, these are:

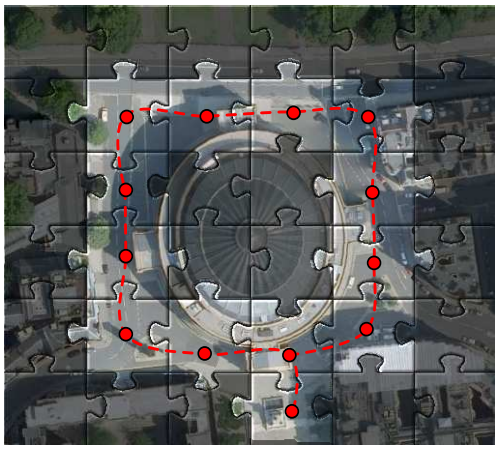
- **Good local estimate of metric motion.** Robust and accurate frame to frame motion estimation is essential in any modern system and consists of obtaining persistent correspondences and resolving mismatches. This is a whole research area on its own since the types of features suitable for tracking vary greatly depending on the sensors on board and the type of environment we are expecting to track. Data association between features in the map and acquired observations to resolve the matching consensus is key to robust performance since mismatches are inevitable when tracking with real data. Apart from providing robustness, this component should also be optimised for efficiency since it comprises a process performed on a per frame basis, therefore fast operation is a requirement.

- **Mapping and loop closure detection.** The data gathered is constantly expanding as the robot explores new areas. As a result, it is important to have an efficient way of representing this data in the map. Systems tackling large-scale tracking in particular need to sparsify into efficient data representations which allow both fast and sufficiently accurate propagation of information. In long exploration periods however, the drift due to the composition of errors in the

(a) Robust local motion estimation



(b) Mapping and loop-closure detection



(c) Global optimisation

Figure 2.3: A modern, high-performance SLAM system ought to have (a) robust local motion estimation of metric motion, (b) a way of mapping the scene with respect to the estimated trajectory and a loop-closure detection module, and finally (c) once such a loop-closure is detected, then full, global optimisation of the robot trajectory and the map constructed should follow.

Images used for this figure have been taken from Google Street View and Bing

robot and map estimates is another limiting factor. So it is usually the case that upon traversal of long loops geometry is no longer reliable enough to recognise places the robot has visited before. As a result, modern methods also use purely appearance-based methods to detect such *loop closures*.

- **Map management and optimisation.** Upon the detection of loop closures, new dependencies are introduced into the map of poses and landmarks. Therefore, optimisation is necessary then to reach a globally consistent map from both local metric and global topological constraints.

The SLAM literature has seen a variety of implementations using different sensor types selected to suit the targeted application. Localisation and mapping underwater for example, requires special attention to the tracking conditions and types of features

expected to track; Williams and Mahon [2004] use acoustic and visual sensors while Ribas et al. [2008] use solely sonar sensing arguing that the use of vision is reliable in clear waters and very close to the sea bed restricting the performance of robot navigation. Sensor fusion has been a popular choice for complex environments with the aim of exploiting the benefits of different sensing modalities. Combining GPS feeds with inertial sensing Kim et al. [2003] perform airborne navigation using Unmanned Air Vehicles (UAVs), whereas later on Kim and Sukkarieh [2007] instead incorporate data coming from a single monochrome camera and an inertial measurement unit. The precision of acceleration and rotation rate estimates promised by inertial sensing together with the high update rates it provides make it an attractive solution for such specialised environments. However the drift accumulated in the estimated inertial position is inevitable (drift rate scales cubicly with time according to Kim and Sukkarieh [2007]) which imposes the need for supplementary information like visual data. Inevitably though, when fusing data from different sensors even if they are of the same type, adds the hassle of data registration which if not handled carefully can lead to fatal inconsistencies in the acquired map.

Laser range-finders have also received major research interest from early on due to their ability to provide accurate depth estimates and form dense point clouds resembling the scene structure. Weingarten and Siegwart [2005] use laser data to achieve scene reconstruction while Bosse and Roberts [2007] perform laser-only SLAM to tackle the lack of robustness of systems in large unstructured environments. While laser sensing can provide high precision and dense correspondences, the inherent descriptiveness of data is very poor as is the case with all types of range sensing. As a result, it has been realised that using appearance information in scenarios with limited priors can provide the extra input that the system needs to resolve tracking, leading to the use of image-based loop-closure techniques even in laser-based tracking systems (e.g. [Newman et al., 2006]).

Despite the variety of sensing modalities and their combinations used in SLAM implementations appearing in the literature, for a number of years now there has been a significant trend towards vision-based approaches. Of course the choice of sensors is a question of the type of task at hand, however the need for generally applicable solutions drives research towards widely compatible implementations. Cameras can promise compactness, affordability and descriptiveness which jointly satisfy more requirements than any other type of sensor. In fact, many state of the art systems now use cameras as their primary sensor. Perceiving the world through a camera lens can be less accurate than laser range sensing, however the richness of information encoded in visual data has been proved enough to recover reliable estimates of camera motion and scene structure. On the other hand, the load of priors encrypted in an image imposes

the challenge of efficient processing to achieve online performance.

2.2 State Of The Art Vision-Based SLAM

At this point in the history of SLAM one can say that implementations have reached considerable maturity. However, the use of visual data as the primary source of information in a SLAM system has not had the time yet to converge to generally efficient and robust solutions, leaving much room for experimentation and improvement. The wide compatibility of vision-based implementations has opened up new application areas sparking growing research interest across the robotics and computer vision communities. As mentioned earlier, the high bandwidth of information provided in visual data requires careful manipulation therefore importing methodologies from computer vision and photogrammetry has been essential for successful systems. This section aims to give a brief review of modern, high-performance, vision-based systems currently considered as state of the art in this area. The skeleton of the discussion is formed around the key components comprising such a system as detailed in the previous section.

As a good example of a modern visual SLAM system, the work of Konolige and Agrawal [2008] dubbed 'FrameSLAM' is used here as a basis for describing the wider literature. In FrameSLAM the authors perform visual SLAM using a stereo rig mounted on a wheeled robot. Their results demonstrate impressive tracking performance over long trajectories ($\sim 10Km$) under very challenging conditions like traversing rough terrain in urban environments. The FrameSLAM system and the work of Mei et al. [2009] which are discussed below, comprise the most powerful robot-based SLAM systems using stereo vision at present.

2.2.1 Map Representation

Konolige and Agrawal [2008] form a 'skeleton' map representation which comprises of a graph of nonlinear constraints between selected, captured *frames* instead of the individual 3D positions of world features (hence the name of the algorithm). They essentially use a sparsified variant of the classic pose graph optimisation approach to solving SLAM. A pose graph consists of nodes representing robot poses or frames in this example, interconnected with edges describing a cost-function relationship which is defined in terms of the desired node configuration. The optimisation process therefore involves computation of the nodes' position such that the goal of this cost-function is achieved; that is the maximum likelihood (ML) map.

In this map representation they only keep relative pose information between the frames. Depending on the trajectory of the robot, they adapt this representation ac-

cordingly by explicitly selecting the frames to participate in the graph optimisation. With this sparsification formulation, they enforce the use of constant amount of space for a particular area so loopy browsing of the same area does not cause an inflated map. However, as with every approximation the quality of performance depends heavily on how close this sparsification is to the full problem.

2.2.2 Local Motion Estimation

Moving on rough terrain means that the feature tracks in images are highly jerky, as is the case with any scenario of high camera dynamics. In order to be able to track this motion, Konolige and Agrawal extract hundreds of features per pose so that there are statistically enough inliers to be able to resolve consensus later. The local motion estimation is acquired incrementally using the online visual odometry system for stereo images as presented in [Konolige et al., 2007].

Visual Odometry is the term used to refer to the process of estimating the position and orientation of a robot by analysing images taken from consecutive poses. This either means constructing a pixelwise optical flow field or matching image projections of features from one image to the next. Feature matching is a more popular approach in the robotics community, while optical flow works have been heavily explored in the vision literature. However, the latter has also been applied in robotics. The seminal work of Lucas and Kanade [1981] who assumed constant flow in local pixel neighbourhoods, has been applied by Campbell et al. [2004] for visual odometry estimation for robot exploration on different types of terrain. Notable is the work of Comport et al. [2007] who use all grey-scale information available in a stereo-pair to achieve very low drift in trajectory estimation over hundreds of meters. In the meantime, the sparser nature of correspondence-based visual odometry has led to more successful performance in terms of achieving a better balance between algorithmic accuracy and efficiency, allowing real-time operation on general hardware platforms.

Scaramuzza and Siegwart [2008] describe a system which performs visual odometry on images from an omni-directional camera mounted on top of a car. Their real-time ego-motion estimation system uses a fusion of both optical flow and feature matching approaches in an attempt to combine their strengths. The authors use SIFT features [Lowe, 2004], well-known for their capacity in descriptivess and robustness, to establish image correspondences. They then estimate the homography from one image to the next, imposing the assumption of planar motion of the camera. Examining the column shift between two consecutive unwarped frames they seed the rotational estimate between poses into the optimisation procedure for local motion estimation.

Konolige et al. [2007] use CenSurE (Centre Surround Extrema) features [Agrawal et al., 2008] which tend to pick out regions of either dark pixels surrounded by lighter

ones or vice versa. The information captured in the descriptor and the matching robustness they exhibit seems comparable to that of SIFT, while the computation process exploits the cost-effective properties of integral images and Haar wavelets [Lienhart and Maydt, 2002] which makes them a more attractive choice for use in real-time applications. In FrameSLAM, correspondences are obtained between the left and the right stereo images at a certain pose which are then matched to features obtained in the left image of the previous frame. A consensus estimate is formed using three-point pose RANSAC [Fischler and Bolles, 1981]. Since the arrangement here consists of calibrated stereo cameras, three points are enough to pin down the relative poses between frames [Hartley and Zisserman, 2004]. However, single camera implementations require a minimum of five points for visual odometry as implemented by Nistér et al. [2004]. Acquiring different pose estimates, the RANSAC hypothesis generated get scored based on the reprojection error of the rest of the features. Finally, a nonlinear least squares optimisation is performed to polish the relative pose estimates. While this procedure is used to resolve data association for local motion estimation, it is also applied to wide-baseline matching in FrameSLAM.

The recent work of Mei et al. [2009] tackles explicitly the problem of precise local mapping for stereo using the relative graph representation of Sibley et al. [2009], which together the two works form perhaps the most significant competitor of FrameSLAM. Rectifying and normalising intensities of both images at a new frame, they then extract SIFT descriptors centred on FAST corners [Rosten and Drummond, 2005] detected at different pyramid levels. Aiming to avoid the common failure mode of large inter-frame rotation they use the method of Mei et al. [2008] to estimate the 3D rotation of ego-motion so that temporal correspondences can be easily identified. Projecting fixed-size search windows for expected landmarks on both images they then establish correspondences which are cleaned from outliers using RANSAC techniques.

2.2.3 Loop-Closures: Detection and Enforcement

In FrameSLAM, the method the authors use for place recognition is fairly simple and relies on a good initial pose estimate; the search for a possible loop-closure is restricted within the vicinity of the hypothesised pose. Over large loops this means that this method becomes linear in the size of the area searched (as the skeleton grows linearly with the area explored).

The literature has seen more sophisticated methods for loop-closure detection able to exhibit robust and relatively fast performance even for the ‘kidnapped’ robot problem which is essentially the case of a complete loss of position/orientation estimate of the robot with respect to its environment. Inspired by the bag of words approach in Video Google [Sivic and Zisserman, 2003], Cummins and Newman [2007, 2008] and

more recently Cummins and Newman [2009] presented an online recognition detection over large data sets demonstrating impressively low false positive rates. Building on conclusions from the earlier work of Newman et al. [2006] on appearance-based detection of loop closures, rather than examining similarity of observations Cummins and Newman assessed the probability that these come from the same place. Overcoming the need for offline training to obtain a visual words dictionary, Angeli et al. [2008] proposed a method for online recovery of candidate place matches with an associated probability of the occurrence of a loop-closure.

Upon detection of a loop closure in [Konolige and Agrawal, 2008], a nonlinear optimisation takes place so that the two ends of the loop meet, propagating corrections throughout the whole graph. The optimal solution is the graph which minimises the reprojection error of the positions of landmarks in the images obtained at all different poses. The nonlinear nature of constraints in the graph makes convergence less trivial and more time consuming since the cost-function surface contains local troughs and peaks. This graph optimisation is usually performed using standard techniques like the Levenberg-Marquardt method or gradient descent and conjugate gradient.

Olson et al. [2006] also follow a pose graph representation and use stochastic gradient descent to optimise this, aiming to find the equilibrium state iteratively such that any antagonistic constraints are in balance. Their method solves the optimisation problem incrementally, limiting the fluctuations of nodes via a learning rate which gradually pushes the graph towards the optimal solution. The system of Konolige and Agrawal [2008] also provides an incremental solution using preconditioned conjugate gradient. In general, conjugate gradient methods are known to perform better than gradient descent alternatives, since they accumulate information on the optimisation direction from one iteration to the next, facilitating faster convergence to the optimum.

While Konolige and Agrawal [2008] adopt a generally relative representation of constraints between poses, the poses themselves and the cost function are defined with respect to a single Euclidean frame. In general, representations defined in a single coordinate frame can potentially suffer a great computational bottleneck particularly in the case of large loop closures since during optimisation the entire loop has to be visited in order to correct global errors. Instead, Sibley et al. [2009] propose an adaptive, fully relative representation which they argue is key for constant time bundle adjustment. Their method not only solves for an optimal trajectory estimate from a pose graph but they attempt to solve the full SLAM problem taking account for the landmarks structure in the optimisation. Expressing the whole graph in a relative manner means that loop-closure can be enforced using a small, local subset of the graph. Resembling what they call a ‘continuous submapping’ approach they perform a breadth-first search from the last pose to nominate nodes to enter an ‘active region’ subject to

adjustment later. Such nodes are selected according to a threshold on their reprojection error. In theory, their method guarantees arbitrarily large graph optimisation in constant time. In practice, they demonstrate achievement of a full maximum likelihood solution using stereo-image data in constant time, for more modest trajectory lengths (around 1 km).

2.3 Monocular SLAM With An Unconstrained, Perspective Camera

While an omni-directional camera or a stereo arrangement provides more information than a single perspective camera, the low cost, compact and self-contained nature of the latter makes it an appealing choice for a much wider range of applications. The complete freedom that a monocular SLAM system allows is what attracts both research and industrial interest. Overcoming the need for precise calibration of a rig of cameras and careful positioning of an omni-directional camera while allowing unconstrained dynamics means that it is no longer necessary to have a robotic platform to support any arrangement restrictions.

Assuming nothing but a freely moving camera in an unknown environment comes with obvious advantages while introducing several hurdles to overcome in a SLAM system. This section intends to provide the reader with an insight into state of the art monocular SLAM systems, tracing recent advances through time.

2.3.1 From SFM to SLAM

The estimation of camera motion from a set of images is a problem studied in depth in the vision literature, well before the appearance of SLAM. Structure from Motion (SFM) is a well-studied problem in the fields of photogrammetry and computer vision aiming for fully automated 3D scene reconstruction from a small collection of images, leading to the development of projective geometry and batch optimisation techniques. SLAM on the other hand, is a problem faced comparatively recently in the mobile robotics community, essentially addressing the hard real-time mapping and navigation problem. The main difference and the real challenge is that in SLAM we are interested in the ‘sequential’, interactive estimation of structure and motion as mentioned in [Davison and Kita, 2000] rather than post-processing of the data gathered to come to a globally consistent solution. Bridging the gap between the two fields, monocular SLAM comes to bring SFM techniques to the same basis of applications allowing similar, online performance.

According to Hartley and Zisserman [2004] who provide an excellent analysis of SFM techniques, the reconstruction problem from an image sequence is typically

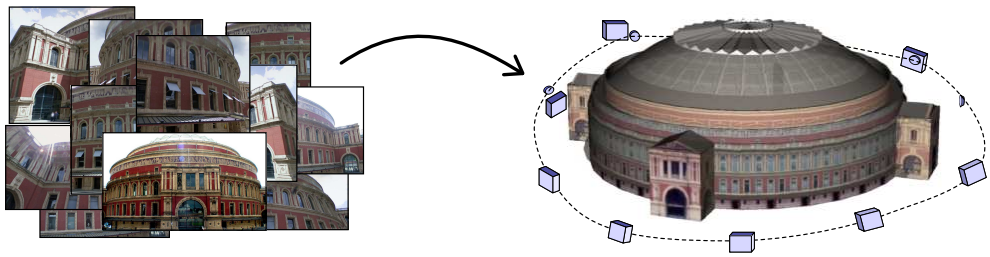


Figure 2.4: Structure from motion (SFM) is a problem studied in photogrammetry and computer vision. Given a collection of unordered images of a scene, the goal is to reconstruct the 3D geometry of both the scene and the trajectory followed by the camera. Standard procedure is to extract interest points from the input images, establish correspondences and perform bundle adjustment optimisation to recover optimal shape and movement.

Images for this figure have been taken from Google Street View, Panoramio and Google 3D Warehouse

tackled in three stages: (a) establishment of feature correspondences throughout the sequence, (b) computation of an initial reconstruction estimate and finally (c) bundle adjustment using the result of (b) as a seed. In order to recover 3D scene structure from 2D geometry, researchers have made several simplifying assumptions constraining the motion of the camera and the scene structure; a key assumption still made by modern systems is that of scene rigidity.

In the seminal work of Tomasi and Kanade [1992] feature tracks are extracted and batch processed in parallel. The reconstruction problem is formulated into a single measurement matrix which is factorised using singular value decomposition separating the effects of the camera motion from the scene structure. This Tomasi-Kanade factorisation is based on a framework only valid for orthographic projection cameras and relies on the assumption that all features are visible in every single frame throughout the sequence. However restrictive, this approach has been the basis of many SFM systems since. Overcoming partly the motion restrictions of orthographic projection cameras, Poelman and Kanade [1993] extended this methodology to the paraperspective case which is a closer approximation to perspective cameras. Szeliski and Kang [1993] generalised to simultaneous recovery of motion and shape from image sets acquired using a perspective projection camera. Following a nonlinear least squares formulation inspired by the work on two-dimensional SFM of Taylor et al. [1991], they recover 3D structure using Levenberg-Marquadt optimisation. The work of Fitzgibbon and Zisserman [1998] is now considered a typical approach to SFM building local estimates from 2-view or 3-view geometry which are then used as a starting point in the optimisation stage of bundle adjustment.

While most of the aforementioned systems require that the internal calibration parameters of the camera are known, Faugeras [1992] pioneered auto-calibration techniques using a SFM framework, recovering both external (e.g. position, rotation) and

internal (e.g. focal length, principal point, skew) camera parameters. However, determining the absolute scale of the scene or movement without additional external information is impossible; unless there is an absolute distance measure input into the system, only relative scale is recovered. Since then, researchers have tackled this problem assuming partial knowledge about the calibration parameters. Pollefeys et al. [1998] presented a self-calibration method applicable to cases with a variety of such assumptions, allowing flexibility and versatility in metric reconstruction scenarios. Azarbayejani and Pentland [1995] were the first to use the Extended Kalman Filter to estimate sequentially the focal length. Very recently, Civera et al. [2009a] presented a SLAM-based approach for online auto-calibration using a Sum of Gaussians filter [Alspach and Sorenson, 1972] to cover the multiple hypotheses arising due to the large nonlinearities in the optimisation of parameters.

The need for online solutions to the SFM problem has arisen since researchers realised their use in robot navigation and the flexibility this provides in several applications like 3D modelling. As the robot is moving from one pose to the next it needs to recover any scene or position estimates during this short period of time to feed back to the controller (this could be the human administrator in the case of guided navigation or a module in the system itself, responsible for autonomous navigation). In order to allow constant-time updates of the robot and scene state after every frame irrespective of the length of the trajectory traversed it has been realised that a constant-size state representation is crucial, leading to the use of filtering techniques to represent the robot state at every instant.

The early work of Harris [1992] used a separate Kalman Filter for every landmark obtained in the image sequence maintaining track of their 3D positions and associated uncertainties in the estimates. Broida et al. [1990] used the Extended Kalman Filter (EKF) to estimate the structure and motion of a rigid object which is a simple extension to Kalman Filtering providing the ability to cope with nonlinear state estimation, as the latter linearises about the current mean and covariance. Broida et al. [1990] used the EKF in an iterative way such that recursive estimation is performed on every update to reach convergence. Avoiding the introduction of initialisation errors upon the incorporation of new features in the system, Chiuso et al. [2002] used a separate filter to initialise each feature which under successful tracking over some probationary period gets fused into the main EKF. As this section will discuss later on, the EKF still comprises a key ingredient of some modern SLAM approaches, used as a sequential approximation to bundle adjustment to build a persistent, probabilistic representation of the state parameters.

Besides applications in auto-calibration and robot navigation, SFM techniques have expanded towards image mosaicing. The work of Szeliski and Shum [1997]

(a) View A (b) View B

Figure 2.5: A state of the art SFM system: Photosynth™ [Microsoft©, 2008]. Here are two snapshots taken while browsing the constructed mosaic of Hagia Sophia in Istanbul, Turkey. Aligned images are projected together with a point cloud to give the impression of the 3D structure even for parts of the building that have not been captured from the current viewpoint. The colour of the points is sampled from relevant views of the image database.

is a representative example where full image panoramas are constructed by aligning and stitching images together to form a large composite image, manually detecting loop-closures. Capel and Zisserman [1998] have employed RANSAC to reject outlier correspondences and achieved super-resolution mosaics following bundle adjustment optimisation, while Brown and Lowe [2003] have used SIFT [Lowe, 2004] features to tackle the problem of robust correspondences in the case of wide-baseline images. The most recent work of Civera et al. [2009b] promises drift-free, real-time mosaic building from a live camera.

Current state of the art SFM systems aim to generate a dense reconstruction of a scene or an object approaching the problem from a variety of perspectives. Vogiatzis et al. [2007] for example, tackle dense recovery of an object's 3D geometry by labelling regions as "object" or "empty" followed by a graph-cuts optimisation. The method of Habbecke and Kobbelt [2007] produces impressive 3D object models approximating the surfaces with sets of small discoidal tiles which are independently fitted and progressively expanded to imitate the true structure. In a far more costly setup, Pollefeys et al. [2008] achieve real-time 3D reconstruction of urban scenes, fusing GPS measurements with inertial and visual sensing processed on advanced hardware. Probably the most representative modern SFM system is the publicly available Photosynth™ [Microsoft©, 2008] software application. Based on the earlier published work of Snavely et al. [2006] dubbed "Photo Tourism", it allows users to upload images and generate their own "photo-synths", forming point clouds from images rather than dense models. Figure 2.5 shows snapshots from a 'photosynth' of Hagia Sophia in Istanbul demonstrating the image mosaic corresponding to the current viewpoint. The visible parts of the building which do not correspond to an image in the database

taken from the particular viewpoint are projected as three-dimensional point clouds replicating the structure of the walls.

While structure from motion and vision-based SLAM for mobile robots are generally two views of a similar problem, at this point in the history of both fields, works have substantial overlap in terms of the goals they target and this is mostly evident in monocular SLAM systems. Using the trivial and low cost setup of a single camera in SLAM can bring the best of technologies from both fields together upon the achievement of persistent, reliable and dense maps as a frame of reference for localisation. The rest of this section is dedicated to review the most advanced single camera SLAM systems at present, using the work in Eade [2008] as a basis of discussion.

2.3.2 State Of The Art in Monocular SLAM

Real-time solutions to SLAM using a single camera in the absence of any odometry information have only recently appeared in the literature. Eliminating the need for careful positioning, data fusion and the induced noise of these processes in a more complex sensing arrangement, monocular systems can provide great flexibility which is otherwise far more restricted. Their usability and scalability have been the driving force of research into this domain, leading to successful applications in wearable computing [Davison et al., 2003; Castle et al., 2007], human-computer interfaces with augmented reality for various applications like gaming [Klein and Murray, 2007] or interactive model building [Bunnun and Mayol, 2008].

The most successful, high-performance implementations of monocular SLAM are the three recent works of Eade [2008], Davison et al. [2007] and Klein and Murray [2007]. This review provides a discussion of these systems breaking them down in terms of their fundamental components as defined earlier in subsection 2.1.2. Since the work of Eade [2008] embodies the complete set of these elements, it is hereby used as a frame of reference and discussion.

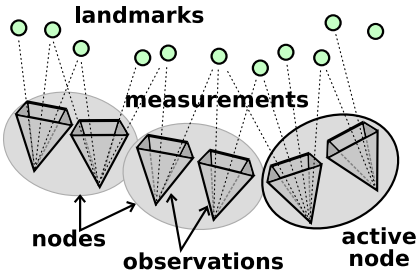
State Representation and Maintenance

Davison [2003] was the first to present a real-time monocular system named MonoSLAM, designed to track the position of an uncontrolled, hand-held camera capturing frames at rates of 30Hz and processing pose and landmark estimates on a typical laptop. A refined version of this system appears in Davison et al. [2007] which comprises the platform used to demonstrate the algorithms developed in this thesis. The authors stack all camera parameters and landmark estimates in a state vector maintained with an associated covariance, together comprising the probabilistic 3D map. This EKF-based approach propagates updates on every frame and achieves successful drift-free tracking for small, room-sized environments.

The idea of a stochastic map dates back to the seminal work of Smith et al. [1988b,a] where they proposed a probabilistic framework to describe uncertainty in geometric relationships and parametrisations. Representing these in a probability distribution with a state mean and covariance which can be built incrementally led to the conception of the stochastic map. Moutarlier and Chatila [1989] were the first to implement the idea of a stochastic map, using the EKF for sequential maintenance of their state consisting of the vehicle and landmark parameters obeying a motion model and an observation model, respectively. The EKF, which has been the most popular choice of SLAM systems to date, linearises these models representing all distributions by Gaussians.

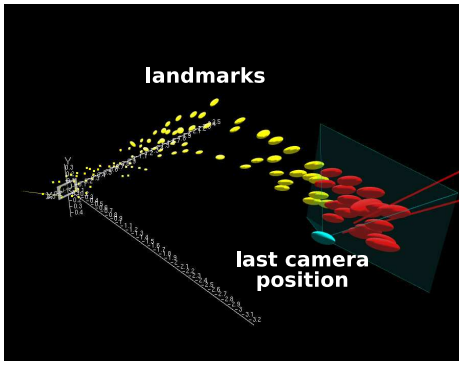
In an attempt to increase the number of landmarks maintained in the map, Eade and Drummond [2006b] employ a particle filtering approach to SLAM inspired by the FastSLAM method of Montemerlo et al. [2003]. Representing the state estimate by a particle cloud, each particle represents a camera pose and map hypothesis. Following the application of a linear dynamic model with process noise, the camera pose distribution is modified to predict the new pose at the beginning of each frame and yields a Gaussian distribution for each particle. Incorporating landmark observations the posterior distribution is computed and new sample poses are drawn. In essence, at the end of every frame the distribution is represented by pose samples with associated independent Gaussian feature estimates. Their method is able to exhibit successful real-time operation tracking 20-30 landmarks per frame which is comparable MonoSLAM's capability but as demonstrated in [Eade, 2008] with synthetic sequences, this method is potentially capable of maintaining denser maps of the order of a thousand features online.

It was soon realised that approximating the nonlinear nature of the estimation process in SLAM by linear models can cause inconsistencies in both EKF [Bailey et al., 2006a] and FastSLAM-based [Bailey et al., 2006b] approaches. On this ground and driven by the strengths and weaknesses of the aforementioned works on monocular SLAM, Eade and Drummond [2007] introduced a graph-based system dubbed Graph-SLAM (not to be confused with the GraphSLAM method of Thrun and Montemerlo [2006] mentioned in section 2.1) in which landmark estimates are 'coalesced' into graph-nodes maintaining the transformation links between these nodes as determined by any shared entries. Figure 2.6 can provide a more intuitive understanding of the different state representations used in current state of the art systems. Observations obtained in a particular frame do not generate a full update for the whole graph, instead only the *active* node is updated, selected so that the observation model is nearly linear, thus boosting consistency in the map. This framework permits the absence of a global coordinate frame which is crucial to the cheap, local update of the graph. In

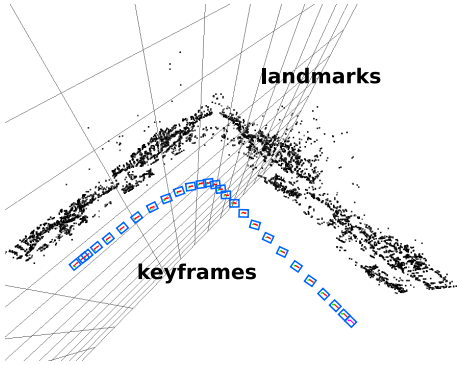


(a) Coalesced-observations graph

Image (a) is based on an illustration from [Eade and Drummond, 2007]



(b) Stochastic map



(c) Map of keyframes

Both (b) and (c) are taken from Klein and Murray [2007] only enhanced and labelled for clarity.

Figure 2.6: The state representations in the three most successful monocular SLAM systems. The diagram in (a) depicts the graph-based state representation of Eade and Drummond [2007] which groups observations into nodes, each having a local coordinate frame. In (b) is a stochastic map as used in the full EKF maintained in the system of Davison et al. [2007] summarising all information with respect to the last camera pose. Tracking the same scene using the keyframe approach of Klein and Murray [2007] instead, the map obtained is depicted in (c).

essence, with their representation the authors manage to avoid Davison's large filter which contains all the features and is destined to grow beyond online processing when mapping larger environments.

While all the above works employ incremental mapping from a filtering perspective, Klein and Murray [2007] present a very powerful monocular system using a keyframes approach in which the processes of tracking and mapping are run in separate but parallel threads. Forgetting intermediate keyframes can have a negligible effect in accuracy but a large impact on the computational efficiency. This is especially the case when the camera is stationary where consecutive frames contain redundant information but on the same basis, accuracy can be compromised at high accelerations. On the other hand, a decisive point in making the keyframe methodology so powerful is that it allows a lot more features in the system (associated locally to keyframes) providing evidence for potentially achieving better precision. In fact, this approach reveals

some of the important pitfalls in filtering methods which naively choose to incorporate data from processing every single frame irrespective of the amount information this is bound to provide; it would be much more intuitive and efficient to be able to judge if a frame is worth the effort of processing or not. Klein and Murray [2007] on the other hand do not take any particular care of which frames to drop or insert in the state representation either. Using heuristics on tracking quality, temporal distance between keyframes and metric distance with known keypoints they manage the graph. Although conditions indeed avoid insertion of identical keyframes when the camera is at rest, they can easily permit the insertion of many similar keyframes (hence containing a lot of redundancy) when the camera is moving slowly. However, running the bundle adjustment on a background thread is the key to remaining within the real-time bounds. While this representation is also bound to explode at some point restricting the size of trackable scenes, it provides fast and accurate tracking for small desktop-like environments.

Local SLAM

As a new image arrives at the camera, the filtering SLAM methods use a camera motion model to predict the motion undergone during the 'blind' interval therefore producing probabilistic estimates of the new positions of known landmarks in the new image. Davison [2003] exploits these predictions to narrow down searches for features in regions constrained by the 3σ uncertainty bounds as projected in image space. This 'active', top-down Bayesian approach to feature matching still proves more efficient than exhaustive bottom-up search over the whole image and is therefore used in many works since, including those of Eade and Drummond [2006b] and Eade and Drummond [2007].

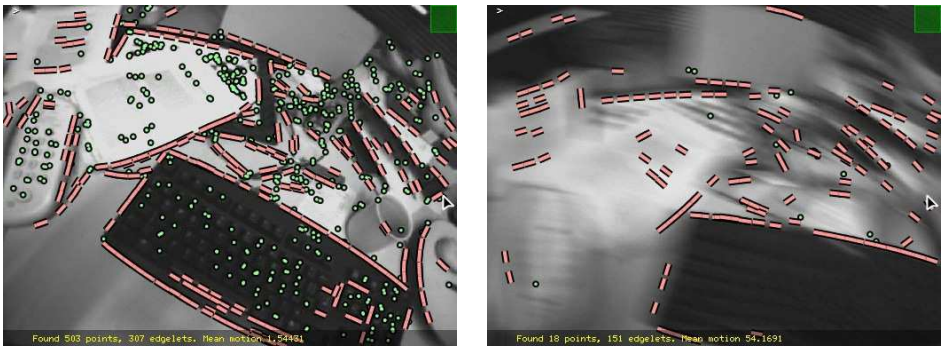
The quality of frame-to-frame feature matching determines the accuracy of a system. Provided that the camera motion model produces concrete estimates closely reflecting reality, then the problem boils down to retrieving matches from the image and resolving any ambiguity incurred. The latter is referred to as the problem of data association between predictions and observations of landmarks. Both Eade and Drummond [2007] and Davison et al. [2007] describe trackable salient image regions with image patches saved at initial detection of landmarks. When a feature measurement is acquired, the state estimates are used to predict not only the expected position of the patch, but also the appearance of the texture from the current viewpoint. In the latest MonoSLAM system, the authors employ the work of Molton et al. [2004] to further estimate a surface normal of the patch at detection assuming locally planar surfaces. This allows for full projective warping with perspective distortion and shearing to simulate rotation-invariant patches which are searched for using normalised cross-correlation.

It is important to note that the saved patch is never refined so as to avoid drift in the appearance of a landmark.

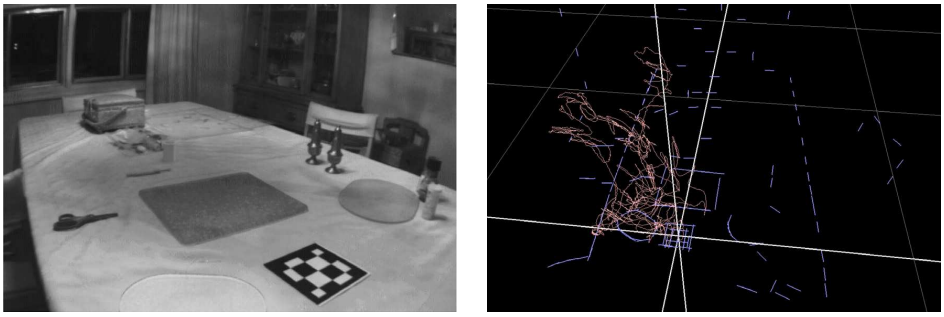
Aiming for highly distinctive and reliable features, different detectors and descriptors have been used in the context of matching. Point features are a popular choice in a variety of implementations mainly due to their simplicity. Davison et al. [2007] use the Shi-Tomasi criterion [Shi and Tomasi, 1994] while Eade and Drummond [2006b] and Eade and Drummond [2007] use the FAST detector [Rosten and Drummond, 2005] as a means of identifying well-textured regions in the image. Aiming to improve the reliability of feature matching, Chekhlov et al. [2006] go for the more invariant SIFT Lowe [2004] descriptor instead of using a template patch for each feature. Essentially, they compute a descriptor of each landmark for different scales upon detection, allowing matching at different resolutions. Eade and Drummond [2008] use SIFT-like descriptors with a sparser structure for speed, but they also use an optimised scale space extrema detector so that each interest point has an image scale. During matching, the image pyramid has to be computed so that a feature is matched at its closest scale.

Point features, however simple and well-studied, induce some problems during tracking. A monocular camera can only measure the bearing of image features. To infer the 3D position of a corner point, the camera must observe it from different viewpoints since this is the only way that depth can be estimated. As a result, distant features which exhibit very small parallax in consecutive frames take longer to initialise properly while their depth estimates are not well represented by the Gaussian distributions in the EKF. Montiel et al. [2006] suggested a method of maintaining inverse-depth estimates when initialising features which on the contrary are better modelled by Gaussians. Most modern probabilistic systems now use this parametrisation as it is widely accepted that it is enforcing consistency in map estimates.

Aiming to build maps with higher-level geometrical information, Eade and Drummond [2006a] have proposed tracking edgelets, defined to be locally linear small portions of a strong, one-dimensional intensity change (i.e. an edge) in the image. Relying only on points is indeed problematic when it comes to motion blur as depending on its severity the majority of the points, if not all, get wiped out as illustrated clearly in Figure 2.7. On the contrary, as demonstrated by Klein and Murray [2008] any edgelets parallel to the direction of blur remain intact, potentially providing all the information necessary to keep the tracker going. Using points as well as edgelets is doubtlessly enforcing robustness for rapid camera translations, but abrupt rotational motion is still a challenge as almost everything distant from the centre of rotation can be wiped out. While less descriptive than regions around points, edges have some more desirable properties like robustness to lighting and viewpoint variance resulting to their appli-



(a) Edgelets provide increased robustness against translational motion blur



(b) The camera view of the scene on the left and the constructed map on the right

Figure 2.7: Tracking edgelets as well as points provides some nice properties including robustness to translational motion blur as demonstrated in (a) which is a snapshot from the work of Klein and Murray [2008]. In (b) is an illustration from Eade and Drummond [2009] (employing the machinery of Eade and Drummond [2006a]) of the additional geometrical information that edgelets can provide in the map where some of the true structure is clearly visible in the 3D map.

cation in various SLAM implementations. The early work of Neira et al. [1997] for example used vertical edges fused with odometry information to localise a mobile robot while more recently Smith et al. [2006] demonstrated a line-based tracker built on top of the system of Davison [2003] to achieve real-time tracking using either lines alone or incorporating information from point features as well.

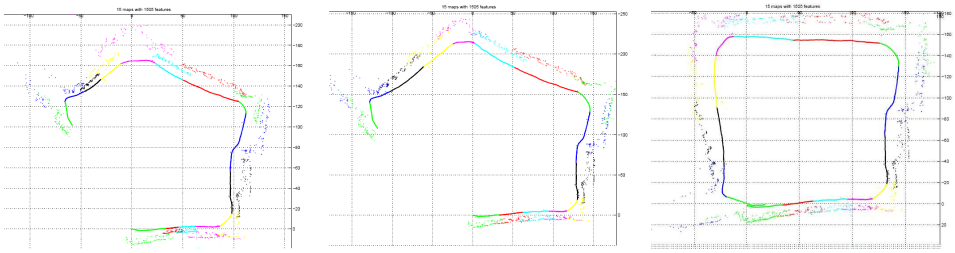
Irrespective of the type of features chosen to track, data association remains a challenge. It is true that the more descriptive a feature is, the easier it becomes to discover which observation it corresponds to. However, missed matches or false positives are inevitable when dealing with real images. Both the works of Eade and Drummond [2006b] and Davison et al. [2007] did not explicitly address this problem, relying heavily on the constrained active search region and the richness of the template patches to provide robust associations, naively accepting pairings of features with the highest scoring template match (satisfying a correlation threshold). Eade and Drummond [2006b] however, employed Nearest Neighbour (NN) to establish correspondences of edgelets followed by RANSAC [Fischler and Bolles, 1981] to reject outliers. Incorrect

pairings of predictions and observations cause jitter, inconsistencies in the map and subsequently tracking failure. While the simplicity of NN-based matching has been appreciated in many early SLAM implementations [Leonard et al., 1992; Guivant and Nebot, 2001] it was soon realised that correct data association with multiple hypothesis handling is essential for robust algorithms. RANSAC is by far the most cited technique for outlier rejection and variants like preemptive or adaptive RANSAC by Nistér [2003] and Hartley and Zisserman [2004] respectively, have also been used in an attempt to reduce the otherwise large number of hypotheses tested. The first fully probabilistic method to discover matching consensus is the Joint Compatibility Branch and Bound (JCBB) test proposed by Neira and Tardós [2001], currently considered one of the most reliable choices for data association. Following a tree-search, it looks for associations that maximise the probability of the hypothesised, jointly compatible prediction error. Variants of this method have also been proposed like the randomised JCBB [Paz et al., 2007b] to tackle the exploding computational complexity with bigger data sets by cutting down on the number of tested hypotheses.

Deviating from traditional filtering approaches, Klein and Murray [2007] employ different techniques for local SLAM estimates. Separating tracking from mapping, the authors essentially decouple the probabilistic estimates of mapping in tracking and vice versa. As a result, tracking can no longer be guided in the sense used in the aforementioned techniques but on the other hand, given that the map needs no longer to get updated at the end of every frame, it allows more time for image processing during local tracking. This method is therefore capable of tracking thousands of features per frame providing visual odometry for local estimates. Since features are detected at different scales, tracking is done in a two-stage coarse-to-fine manner: searching for 50 features at the highest pyramid levels over large search radii provides a refined pose estimate, which is in turn used to reproject predictions for the finer sets of features. No data association issues are addressed explicitly, therefore performance relies heavily on matching large numbers of features — while accepting that outliers are incorporated into the map, this method confides in bundle adjustment to get rid of inconsistencies and enforce robustness.

Building Large Maps Out Of Small Parts

The $O(n^2)$ cost of maintaining the full EKF state relative to the state size as done in the work of Davison et al. [2007] mandates the construction of sparse maps, limiting the number of trackable features per frame so that online performance is sustained. While it provides accurate results for room-sized environments, the constantly expanding map in exploratory sequences is bound to degrade the real-time performance when run over larger scenes. Moreover, as mentioned earlier, the consistency problems of a large



(a) Traversing a long loop (b) Auto-scaling (c) Loop-closure is enforced

Figure 2.8: Large-scale mapping using local submaps. In (a) is the result of tracking a long exploratory sequence using the method of Clemente et al. [2007] which is using the two-level hierarchical submapping of Estrada et al. [2005] and the approach of Davison et al. [2007]. It is evident that local tracking drifts especially on the turns of the trajectory but following auto-scaling (b) and loop-closure detection, the trajectory is optimised in (c).

EKF becomes more evident with increasing the filter size. Eade and Drummond [2007] have explicitly confronted both of these issues by structuring the problem in a set of smaller, more manageable groups of features each relating a nearby node, which can be viewed as decomposing a map into submaps. Not only does this method permit denser local maps providing more consistent local estimates, but it also provides the ability of covering much larger areas than classical full EKF approaches. The method of Klein and Murray [2007], by construction exhibits some sparsification of the full SLAM problem by splitting the map using keyframes rather than submaps. The effect however is very similar, since as clearly depicted in Figure 2.2(c), a feature maintains only links with the keyframes it has been seen from. A keyframe can be viewed as similar to a node in Eade's approach while the relative positions of keyframes is not represented explicitly; the constellation is instead optimised at bundle adjustment considering the landmarks shared between keyframes.

The benefits of breaking down a large map into smaller parts have been a subject of significant interest over the last decade as SLAM methods started aiming for bigger and better solutions. The work of Tardós et al. [2002] on SLAM using sonar data, the Constrained Local Submap Filter of Williams et al. [2002] and the Divide and Conquer approach of Paz et al. [2007a] are all examples of standard sparsification strategies of the full map into statistically independent submaps each maintained by a small EKF of bounded size. As in the work of Eade and Drummond [2007], the updates are kept local providing constant update cost. However, while Eade's representation maintains relative transformations (edges) between submaps permitting real-time optimisation of the whole (still relative) graph at the end of every time step, the rest of the methods mentioned here register each new submap to the global map sequentially. In a slightly different approach, Divide and Conquer SLAM follows a binary tree of submap joinings to provide better cost than the sequential case.

Realising the power of relative representation of transformations between submaps, both Bosse et al. [2004] based on the Atlas framework [Bosse et al., 2003] and the Hierarchical SLAM of Estrada et al. [2005] attacked submapping in a two-level hierarchical manner. While the lower level corresponds to statistically independent submaps as described above, the topological configuration of these submaps is maintained using a graph of nodes in the upper hierarchy level very much resembling Eade's representation of the environment. Their common difference however, is that no constraints are imposed for the positions of landmarks shared among two or more submaps, which on one hand maintains independence and efficiency but always at the cost of mapping quality.

On a substantially different track, the Thin Junction Tree Filter of Paskin [2003] and the Treemap approach of Frese [2006] provide complex but otherwise very efficient solutions to inference by representing the joint probability distribution of the system via tree-like factorisations, which are essentially a different type of approximation of the full SLAM graph. When a map element is observed, the tree-node directly representing it gets updated issuing 'messages' to be passed to every other node, following the branch paths along the tree, essentially propagating the update in the map. While the distributions are represented with Gaussians the parametrisation is done in the information form (maintaining the inverse covariance) which allows exploitation of the sparse representation of links between variables as exploited also in Sparse Extended Information Filters [Thrun et al., 2002] to construct the tree structures. However, the information form imposes a major difficulty when it comes to data association since most algorithms are based on the covariance form as maintained in the EKF for example, rendering them inapplicable especially when this involves large data sets where matrix inversion becomes a computation bottleneck. Eustice et al. [2005b] has taken account for this issue to some extend inverting a subset of the covariance approximating the actual covariance of the variables in question.

More recently, Pinies and Tardós [2008] proposed a powerful algorithm for building a large map out of *conditionally* independent submaps of constant size. In essence, their representation enforces better management of shared information between submaps, allowing more effective propagation of updates and in linear time with respect to the number of submaps. While linear time recovery of the global map with respect to the number of submaps has been made possible in previous works, the authors here achieve it surpassing any limiting constraints in the configuration of the submaps (Paz et al. [2007a] for example restrict submaps in a tree arrangement). However, in order to maintain the conditional independence between submaps at loop closures, the features that have been recognised to belong to both the start and the end submaps need to get copied into every intermediate submap which means extensive

usage of storage in loopy sequences.

Loop-Closing, Relocalisation and Graph Optimisation

While Davison et al. [2007] address small-scale SLAM problems, closing a loop becomes increasingly difficult with growing uncertainty in the position of the camera. This translates into large active search regions for features increasing the chance of false positive matches, thus making the task of data association very challenging, even for robust techniques like JCBB [Neira and Tardós, 2001]. In the application of MonoSLAM in larger scale maps, this problem becomes ever more evident. An example is the system of Clemente et al. [2007] which maps large loops adopting the Hierarchical SLAM technique [Estrada et al., 2005], for submaps constructed in a Davison-like approach. While the bounded size of submaps allows online building of the map, loop-closures push performance past the real-time barrier. Moreover, the EKF assumptions of Gaussian representation of uncertainty in estimates break down for big maps, rendering the filter predictions erroneous.

Eade and Drummond [2008] however, using their earlier Graph-SLAM system [Eade and Drummond, 2007], explicitly tackle the problems of detection and enforcement of loop-closures achieving real-time performance for maps of 1000 features on a dual-core computer. A bag-of-words dictionary is trained online, clustering observed features based on their appearance so that image descriptors can be generated based on the occurrence of certain 'visual words'. Every node in the graph maintains a list of the observed words along with a count of their occurrence while that node has been active. The words in the dictionary are subsequently informed with a ranked list of most representative nodes they occur in (based on the term-frequency-inverse-document-frequency metric [Sivic and Zisserman, 2003] taking account of the uniqueness of words in the database). The highest-ranking nodes matching the current view that do not already share an edge with the active node, are considered as candidates for loop-closing, matching them using Nearest Neighbour based on the distance measure proposed in [Lowe, 2004]. Following the visual appearance cues, the system then seeks for a match in structure using MLESAC [Torr and Zisserman, 2000] which is a variant of RANSAC, only scoring hypotheses based on their likelihood (representing the error distribution as a mixture model) rather than the number of inliers.

Interestingly, Eade and Drummond [2008] address the problem of relocalisation as a special case of the loop-closure problem. If tracking failure occurs, the SLAM system does not need to restart; instead it creates a new graph component which is subject to unification with the rest of the graph upon revisiting of previously mapped regions. Recovery is therefore approached as the enforcement of a consistent fusion of the active graph with the rest of the known map. After the establishment of a new

edge connecting the two ends of the loop, global optimisation takes place as in the end of every normal frame to enforce any new constraints optimising the graph using preconditioned gradient descent for speedy convergence.

Klein and Murray [2007] do not really address loop-closure. While this is implicitly enforced for very local loops by projecting expected locations of known features in the current frame, for longer, exploratory sequences these are not handled meaning that their system can therefore suffer from drift. However, they address the relocalisation problem allowing recovery once the camera gets lost, adopting the method of Williams et al. [2007]. The latter work is based on Davison's system, incorporating JCBB on top of active search for robust outlier rejection. The tracking-failure flag is raised once no match is found within the predicted search regions of features, while relocalisation is tackled by searching the entire image for correspondences with known features, using the randomised-lists classifier of Lepetit and Fua [2006]. The key insight is the treatment of online feature recognition as a classification problem where classes are trained on trees of randomly generated tests of pixel intensity comparisons. As a result, at runtime, a feature can be dropped down these classification trees to provide probabilistic estimates of class memberships, with each class corresponding to a different known feature. Upon the establishment of correspondences with known parts of the map, RANSAC [Fischler and Bolles, 1981] is applied to recover the position of the camera using the three-point-pose algorithm so that tracking resumes.

Klein and Murray [2007] perform local bundle adjustment regularly optimising the pose of the most recent keyframe with its closest neighbours (5 keyframes in total) using all the measurements ever made for the landmarks seen from these viewpoints. Global bundle adjustment to refine the poses of all keyframes present in the map is also run in the background, but only whenever the size of it permits reasonable speed performance. As an example, the authors mention that maps containing more than 150 keyframes would require 'tens of seconds'. In order to mitigate the complexity cost of using all available measurements in bundle adjustment, Holmes et al. [2009] combine the benefits of the relative graph representation of Sibley et al. [2009] and the parallel tracking and mapping approach of Klein and Murray [2007] while subsampling the data input into the optimisation stage. Their method is demonstrated to permit contant time exploration and maintain conservative estimates of the map.

While the relocalisation method of Williams et al. [2007] indeed provides robust results, its processing and storage cost scale badly with increasing number of features. Allowing relocalisation for map-sizes of 1500 features Klein and Murray [2008] instead use a very simple relocalisation method. Importing most of the machinery used in their earlier system [Klein and Murray, 2007], they save a zero-mean, heavily blurred version of the sub-sampled image obtained at every keyframe. When tracking is lost,

every new frame is compared against every such keyframe 'descriptor', choosing the one with minimal difference to estimate a camera rotation by minimisation of their sum-of-squared differences after alignment.

2.4 Ongoing Challenges and Progress in this Thesis

The recent growth of monocular SLAM algorithms has deservedly established its place in the list of hot research topics at present. The engagement of smart and theoretically solid techniques has brought implementations a long way, however much remains still to be done for successful operation outside the benign conditions of a lab environment. The performance of most modern existing systems as reviewed above, is indeed limited to fairly small-range, careful camera manoeuvers. Dense mapping on one hand offers local accuracy while it proves a bottleneck when it comes to mapping slightly larger areas. In a nutshell, the focus of future monocular SLAM algorithms needs to be on:

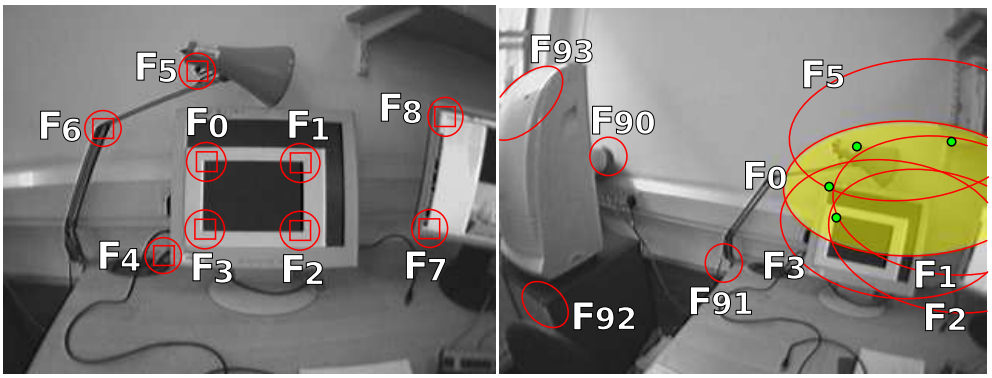
- fast camera motion
- very large scales
- rich maps towards online 3D scene reconstruction
- real robustness in unstructured environments and dynamic conditions, and
- low computation to permit embedded applications

The underlying challenge inherent in these points is to handle larger amounts of data in a more effective way. The key is in *agile manipulation of information* to exploit the value of the extra knowledge while avoiding the drawbacks of the computational burden that this is accompanied with. Sustaining truly fast motion in dynamic conditions, mandates robust matching for different levels of input priors. Large-scale and dense mapping translates into efficient data maintenance and robust outlier rejection to close larger loops. Effective approximations to the full problem can provide the solution to lower computational complexity while maintaining quality of performance. Attacking these currently open questions successfully will bring monocular systems a step closer to truly general algorithms for arbitrary environments.

2.4.1 Efficient and Robust Matching

How do observations relate to known parts of the map?

The biggest difficulty in tracking is the data association problem of correctly identifying known 3D features in the image projection of the scene from the current viewpoint. It is important that the descriptors of landmark-features are resilient to lighting and viewpoint changes, they are fast to compute and distinctive with



(a) Frame 10 (b) Frame 474

Figure 2.9: Accumulated uncertainty after traversing a loop with MonoSLAM. The projected uncertainty of features in image space as ellipses demonstrates how uncertain older features become at the loop closure frame (b). Even looking for them within their active search regions, the matcher is likely to fire at erroneous positions as evident after a search for F_0 in (b) (matches are shown in green). Similar results for the rest of the loop-closing features makes the problem of data association even harder as more outliers lie in the data set.

respect to the surroundings. Tracking in a general environment however, provides no guarantees that these conditions are met; on the contrary, it is certain that the feature matcher will fire at erroneous positions in the image while other features will not be recognised at all. As a result was the emergence of outlier rejection techniques and their application in SLAM systems like JCBB [Neira and Tardós, 2001], RANSAC [Fischler and Bolles, 1981] and variants.

Resolving correspondences of visual data (as opposed to range measurements for example) is on one hand easier due to the extra descriptiveness of appearance information, but on the other hand it becomes tricky to handle the inevitable variance in the quality of the input data. It was soon realised that not all features are equally reliable in matching, assessing them in terms of their relative uniqueness and repeatability of recognition. Newman et al. [2006] propose discarding information coming from repeatable appearance, while the more advanced bag-of-words approach [Sivic and Zisserman, 2003; Cummins and Newman, 2007] is becoming increasingly popular in detecting loop-closures.

In fact, loop-closure detection, relocalisation and frame-to-frame matching are sibling problems all aiming to solve the puzzle of data association. Their only difference is the strength of input prior information. While from one frame to the next the motion of the camera can be predicted with some accuracy (depending on the frame rate and camera dynamics), when the robot is at the end of a long loop, the uncertainty in the camera position is much bigger. This is clearly illustrated in Figure 2.9. Even further, when the camera is lost, by definition the pose uncertainty can be arbitrarily large (depending on the time elapsed since tracking failed). The goal always being to relate

current observations to known parts of the map, the feature matcher can be employed to establishing correspondences are reject outliers.

While RANSAC is the dominant approach to matching, it is far from a gold standard method. The method itself is indeed very simple to implement but due to the fact that it relies on arbitrary thresholds and most importantly randomness, it is prone to extensive testing of erroneous hypotheses using up precious processing time and increasing the likelihood of a spurious result. More recent variants mentioned earlier have explicitly tackled this problem but they still rely on heuristics and random numbers.

The increasing popularity of JCBB proves there is room for improvement in robust matching in the presence of priors, but its exponential degradation in performance in the presence of many outliers is limiting its use to highly uncertain scenarios. In the Active Matching algorithm discussed later as a contribution in this thesis, individual feature characteristics can be taken into account together with the degree of correlation between features to guide the course of matching towards a probabilistically backed-up result. As opposed to traditional approaches of getting correspondences first and resolving them later, the methodology follows a top-down approach to search for candidates resulting in fewer image processing operations and less contamination of data.

2.4.2 Scaling and Map Management

When is it worth splitting a map into two submaps?

As the need for bigger and denser maps is increasing, so is the need for efficient data manipulation. Earlier, this chapter discussed how researchers have realised this need leading to the emergence of sparsification techniques. Tree approximations, frames, nodes, keyframes or submaps have been vital in sustaining online SLAM performance for large-scale mapping irrespective of the sensing modalities used. By bounding accumulated uncertainty, number of landmarks or distance since last partition, systems have managed real-time performance at the expense of accuracy of approximation. However, there has been little theoretical investigation of efficient map management in terms of the quality of sparsifications performed, limiting the applicability of existing techniques to more challenging scenarios.

Taking the popular choice of representing the state with a stochastic map as an example, splitting this map into two submaps means cutting the correlation links between features lying in separate submaps. In the most common case of preserving full EKF filters for each submap, this successfully bounds the $O(n^2)$ maintenance cost for small numbers of n while updates across the whole map become linear to the number of submaps. While besides the computational benefit, existing works report improve-

ments in accuracy in terms of limiting the linearisation errors inflicted in the case of keeping a long EKF. However, no particular care is taken about *where* to place submap partitions.

The stochastic map by definition models explicitly the relationships between different parts of the state as correlations. A very useful property of these correlations is that measuring one part of the state tells a lot about the others as the updates can be propagated through the correlation links between the variables — and indeed this is one of the key ideas used in the Active Matching algorithm. It is the strength of the correlation links that we cut during an approximation that determines the quality preserved in the map. Moreover, the infinite range of a camera makes correlation structure in a visual-SLAM map more tricky since distant features in the scene can appear very close together in image space strengthening their correlation. Quantifying the level of correlation shared between different landmarks in terms of Information Theoretic measures, Chapter 6 demonstrates that covisibility is not the only factor that provides strong correlations as previously believed, but coherency of motion is even more important. Understanding the tracker's impression of the scene, we present a simple and fully automatic method for partitioning general visual maps which optimises the quality of sparsification.

3

A Top-Down, Filtering Approach to Monocular SLAM

While the literature has seen a plethora of approaches to SLAM using expensive and highly accurate range sensors, a trend towards the cheaper option of cameras has recently become established. Allowing greater freedom and versatility, are single, hand-held camera implementations relaxing problem specific constraints. On the other hand, assuming nothing but a freely moving camera in an unknown environment imposes several additional obstacles to the estimation process. Despite the inherent difficulty and challenges faced in monocular SLAM, recent implementations have shown that map and trajectory estimation is possible within small-range environments, as discussed in the previous chapter. This chapter is dedicated to familiarise the reader with the key concepts and notation used in sequential, Bayesian monocular SLAM through a description of the MonoSLAM system proposed by Davison [2003] and later refined in [Davison et al., 2007]. This system comprises the platform used throughout the rest of this thesis to demonstrate the methodologies and algorithms developed.

The online recovery of the 3D trajectory of the camera in MonoSLAM is achieved by incrementally building a map of natural landmarks (features) on the fly, as detected

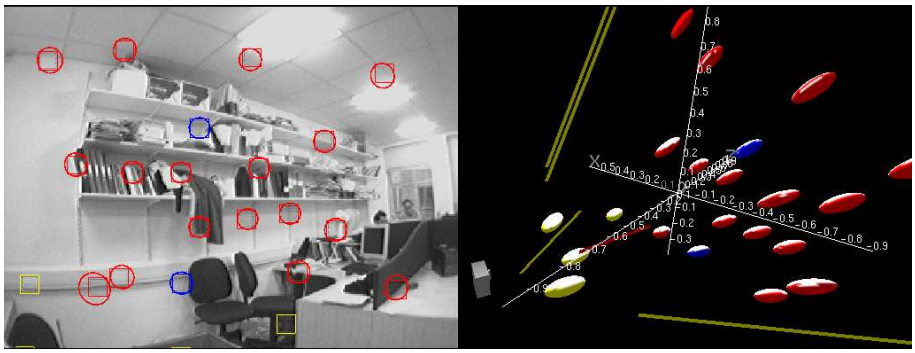


Figure 3.1: Tracking the camera movement relative to the observed landmarks. On the left is the current camera view and on the right is a representation of the perceived map of the scene. The natural landmarks selected for tracking (the image patches on the left image) correspond to ellipsoids on the right image, the shape of which encodes the nature of uncertainty in the position of the relevant feature.

and tracked throughout the movement of the camera. The map constructed is then used to guide the search for features in subsequent frames. Using an Extended Kalman Filter (EKF), the system produces estimates of the joint distribution over the 3D location of the camera and the set of known features. On the arrival of a new image, a probabilistic motion model is applied to the accurate posterior estimate of the previous frame, adding uncertainty to the estimated new camera position. In standard configuration the system then makes independent probabilistic predictions of the image location of each of the features of interest, and each feature is independently searched for within the ellipse defined by a three standard deviation gate. Constantly measuring and refining the constructed map of the surroundings, MonoSLAM achieves real-time tracking of the path followed by the camera. Following, is a more detailed description of the key components in the top-down filtering approach of MonoSLAM.

3.1 Representation of the World

Based on the probabilistic framework introduced by Smith et al. [1988b], the belief about the state of the world at any time instant can approximated with a single, multivariate Gaussian distribution. Therefore, the probability density describing the state vector $\mathbf{x}$ is defined as below, in terms of its estimated mean $\hat{\mathbf{x}}$ and covariance matrix $\mathtt{P}$:

$$p(\mathbf{x}) = (2\pi)^{-\frac{d}{2}}|\mathtt{P}|^{-\frac{1}{2}}\, exp\{-\frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^{\top}\mathtt{P}^{-1}(\mathbf{x}-\hat{\mathbf{x}})\}\,. \tag{3.1}$$

Here, d denotes the dimension of $\hat{\mathbf{x}}$ and $\mathtt{P}$ is a square $(d \times d)$ matrix, partitioned as follows:

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{x}}_c \\ \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \\ \vdots \end{pmatrix} \quad , \quad \mathtt{P} = \begin{bmatrix} \mathtt{P}_{xx} & \mathtt{P}_{xy_1} & \mathtt{P}_{xy_2} & \dots \\ \mathtt{P}_{y_1x} & \mathtt{P}_{y_1y_1} & \mathtt{P}_{y_1y_2} & \dots \\ \mathtt{P}_{y_2x} & \mathtt{P}_{y_2y_1} & \mathtt{P}_{y_2y_2} & \dots \\ \vdots & \vdots & \vdots & \end{bmatrix} . \tag{3.2}$$

The state vector $\mathbf{x}$ stacks the camera state $\mathbf{x}_c$ and the 3D position $\mathbf{y}_i$ of each feature i in the map. The uncertainty in the individual entries stated in the mean vector together with their relationships are described in the covariance matrix. More specifically, the blocks along the diagonal express the uncertainty in each estimate of the mean vector while the off-diagonal blocks encode the correlation between these estimates. Essentially, $\hat{\mathbf{x}}$ and $\mathtt{P}$ comprise a snapshot of the current state estimates of the camera and the features in the map. Figure 3.1 is an example of the 3D map constructed after a few frames of tracking using MonoSLAM.

In order to describe the camera state vector $\mathbf{x}_c$ we need to define a fixed 'world' coordinate frame and a camera coordinate frame (defined with respect to the current position and orientation of the camera) denoted by superscripts 'w' and 'c', respectively. Hence, adopting the notation of Davison et al. [2007], $\mathbf{x}_c$ comprises of the following set of parameters: the 3D position of the camera $\mathbf{r}^w$, the quaternion $\mathbf{q}^{wc}$ describing the rotation transformation between the two coordinate frames, the linear velocity of the camera $\mathbf{v}^w$ and its angular velocity ω^c. In total, $\mathbf{x}_c$ is composed by 13 parameters and explicitly is expressed as:

$$\mathbf{x}_c = \begin{pmatrix} \mathbf{r}^w \\ \mathbf{q}^{wc} \\ \mathbf{v}^w \\ \omega^c \end{pmatrix} . \tag{3.3}$$

A feature state vector $\mathbf{y}_i$ is generally comprised by a three-element vector describing the 3D coordinates of the feature with respect to the world coordinate frame (w). However, when a feature is first initialised in a monocular system, its depth estimate is infinitely uncertain and is only bound to become more precise if viewed from different camera poses. Demonstrating that uncertainties of such extend are not well-modelled by a standard Gaussian distribution, Montiel et al. [2006] proposed a reparametrisation of the representation of a feature state vector in terms of its inverse depth. Illustrating low linearisation errors at low parallax, this modification has been shown to model the estimation uncertainty at feature initialisation much more accurately by a Gaussian. Hence, this parametrisation has been adopted in MonoSLAM, encoding newly initialised features by a six-dimensional vector $\mathbf{y}_i$ comprising the camera position from which the feature was first observed ($\mathbf{r}^w$ using the above notation), the azimuth $\phi_i{}^w$

and elevation ${\theta_i}^w$ angles defining the direction of the ray of observation, and finally the point's inverse depth $1/d_i$ along this ray:

$$\mathbf{y}_i = \begin{pmatrix} \mathbf{r}^w \\ {\phi_i}^w \\ {\theta_i}^w \\ 1/d_i \end{pmatrix} . \tag{3.4}$$

Upon observation of this feature from a sufficiently wide baseline, the uncertainty in its depth estimate reduces enough to allow conversion into a 'fully-initialised' point representation, requiring solely the 3D coordinates of that feature.

3.2 Motion and Probabilistic Prediction

In probabilistic form, SLAM requires computation of the posterior given all state estimates and observations up to the current frame (if odometry was available, all the control inputs would be included here also). However, this can be formed in a recursive estimation of posterior of the new state parameters $\mathbf{x}^{(k)}$ at time instant k given the last estimated state $\mathbf{x}^{(k-1)}$ and the observations $\mathbf{z}_T^{*(k)}$ made since then:

$$p(\mathbf{x}^{(k)}|\mathbf{x}^{(k-1)}, \mathbf{z}_T^{*(k)}) \quad \sim \mathcal{N}\left(\hat{\mathbf{x}}^{(k|k)}, \mathtt{P}^{(k|k)}\right) , \tag{3.5}$$

where $\mathcal{N}$ denotes a Normal distribution characterised by its first two moments. However, before processing the input image to make observations from the new camera pose, the system makes a series of predictions based on past experience and implicit assumptions to enforce consistency and efficiency of processing:

1. **Camera state**: the system makes a guess of the motion undergone by the camera during the 'blind' interval between the last estimated pose and the new one.
2. **Candidate measurements**: based on the predicted new viewpoint, any known map-features that should be visible/measurable are identified.
3. **Search regions**: for every feature selected for measurement in the new frame, a corresponding likelihood region in the image is estimated so that the search for each feature is localised.

Below, we elaborate on the way these predictions are made and discuss their importance in the performance of the system.

3.2.1 Camera State

As the camera captures frames processing them sequentially, unless there is some motion model to describe how the camera moves in between frames, there is no way to

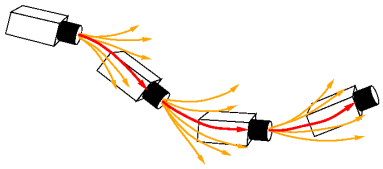


Figure 3.2: MonoSLAM uses a smooth camera motion model of constant angular and linear velocity, to imply that on average, we expect undetermined accelerations to occur with a Gaussian profile.

predict the new pose of the camera before analysing the input image. In contrast to robot-based implementations, monocular SLAM has no access to odometry information (a term used to describe the series of commands to control the motion of a robot platform). The movement of a hand-waved camera in fact, is particularly difficult to model as it does not allow for precise assumptions to be made about the dynamics of the camera or the intentions of the carrier. Acknowledging this issue, MonoSLAM uses a *constant velocity motion model*. This model essentially asserts that between one frame and the next, the camera is expected to experience linear and angular changes in velocity which are unknown in detail but can be characterised probabilistically by a zero-mean Gaussian distribution. The variance of the Gaussian distribution used depends on both the level of dynamic motion expected of the camera and the inter-frame time interval. Large frame-to-frame motion uncertainty occurs when vigorous, jerky movements are expected for when the frame-rate is low. Smooth motions or high frame-rates allow more precise motion predictions and with lower uncertainty.

The motion model can be described in terms of a probability distribution on state transitions:

$$p(\mathbf{x}^{(k|k-1)}) = p(\mathbf{x}^{(k)}|\mathbf{x}^{(k-1)}) \quad \sim \mathcal{N}\left(\hat{\mathbf{x}}^{(k|k-1)}, \mathtt{P}^{(k|k-1)}\right). \tag{3.6}$$

Denoting the motion model $\mathbf{f}$ we can predict the new camera state $\mathbf{x}_c^{(k|k-1)}$ in terms of the camera parameters from the last known state $\mathbf{x}^{(k-1)}$. Therefore, if the uncertainty introduced through this process is denoted by $\mathtt{Q}$ (process noise), the predicted state $\mathbf{x}^{(k|k-1)}$ after the motion of the camera (EKF prediction step) is described by:

$$\hat{\mathbf{x}}^{(k|k-1)} = \begin{pmatrix} \mathbf{f}(\hat{\mathbf{x}}_c^{(k-1)}) \\ \hat{\mathbf{y}}_1^{(k-1)} \\ \hat{\mathbf{y}}_2^{(k-1)} \\ \vdots \end{pmatrix}, \tag{3.7}$$

and dropping the superscript in $\hat{\mathbf{x}}_c^{(k-1)}$ for clarity,

$$\mathtt{P}^{(k|k-1)} = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial \mathbf{x}_c}\mathtt{P}_{xx}^{(k-1)}\frac{\partial \mathbf{f}}{\partial \mathbf{x}_c}^{\top} + \mathtt{Q} & \frac{\partial \mathbf{f}}{\partial \mathbf{x}_c}\mathtt{P}_{xy_1}^{(k-1)} & \frac{\partial \mathbf{f}}{\partial \mathbf{x}_c}\mathtt{P}_{xy_2}^{(k-1)} & \dots \\ \mathtt{P}_{y_1x}^{(k-1)}\frac{\partial \mathbf{f}}{\partial \mathbf{x}_c}^{\top} & \mathtt{P}_{y_1y_1}^{(k-1)} & \mathtt{P}_{y_1y_2}^{(k-1)} & \dots \\ \mathtt{P}_{y_2x}^{(k-1)}\frac{\partial \mathbf{f}}{\partial \mathbf{x}_c}^{\top} & \mathtt{P}_{y_2y_1}^{(k-1)} & \mathtt{P}_{y_2y_2}^{(k-1)} & \dots \\ \vdots & \vdots & \vdots & \end{bmatrix} . \tag{3.8}$$

3.2.2 Candidate Measurements and Search-Regions of Selected Measurements

When a new image arrives, we can project the current probability distribution over state parameters into measurement space to *predict* the image locations of all the features candidate for measurement. Defining stacked vector $\mathbf{z}_T = \begin{pmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \dots \end{pmatrix}^{\top}$ containing all predicted feature measurements and stacked likelihood function $p(\mathbf{z}_T|\mathbf{x})$, the density:

$$p(\mathbf{z}_T) = \int p(\mathbf{z}_T|\mathbf{x})p(\mathbf{x})d\mathbf{x} , \tag{3.9}$$

is a probabilistic prediction not just of the most likely image position of each feature, but a joint distribution over the expected locations of all of them. This joint distribution, if formulated correctly, takes full account of both individual feature motion assumptions and global inter-feature constraints.

Given the prediction of the new camera viewpoint at the new pose following the application of the motion model, we can predict which of the known landmarks, if any, should be visible using the measurement model **h** (also referred to as 'observation model'). If $\mathbf{x}_m$ denotes the stack of state parameters in measurement space (i.e. the camera state and the vector $\mathbf{z}_T$), then the application of this model gives the probability of distribution of $\mathbf{x}_m$ given the predicted new pose:

$$p(\mathbf{x}_m|\mathbf{x}^{(k|k-1)}) \quad \sim \mathcal{N}(\hat{\mathbf{x}}_m, \mathtt{P}_m) . \tag{3.10}$$

The measurement model acts on the predicted state $\mathbf{x}^{(k|k-1)}$ at the new frame to produce expectations on the individual feature measurements $\mathbf{z}_i^{(k|k-1)}$ comprising the vector $\mathbf{z}_T^{(k|k-1)}$. Here, a measurement for a feature yields its 2D image coordinates. As a result, the state parameters in image space can be described via:

$$\hat{\mathbf{x}}_m = \begin{pmatrix} \hat{\mathbf{x}}_c^{(k|k-1)} \\ \hat{\mathbf{z}}_1^{(k|k-1)} \\ \hat{\mathbf{z}}_2^{(k|k-1)} \\ \vdots \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{x}}_c^{(k|k-1)} \\ \mathbf{h}_1(\hat{\mathbf{x}}^{(k|k-1)}) \\ \mathbf{h}_2(\hat{\mathbf{x}}^{(k|k-1)}) \\ \vdots \end{pmatrix} , \tag{3.11}$$

and dropping the superscripts in $\hat{\mathbf{x}}^{(k|k-1)}$ and $\mathtt{P}^{(k|k-1)}$ for clarity,

$$\mathtt{P}_m = \begin{bmatrix} \mathtt{P}_{xx}^{(k|k-1)} & \mathtt{P}\frac{\partial \mathbf{h}_1}{\partial \hat{\mathbf{x}}}^\top & \mathtt{P}\frac{\partial \mathbf{h}_2}{\partial \hat{\mathbf{x}}}^\top & \dots \\ \frac{\partial \mathbf{h}_1}{\partial \hat{\mathbf{x}}}\mathtt{P} & \frac{\partial \mathbf{h}_1}{\partial \hat{\mathbf{x}}}\mathtt{P}\frac{\partial \mathbf{h}_1}{\partial \hat{\mathbf{x}}}^\top + \mathtt{R}_1 & \frac{\partial \mathbf{h}_1}{\partial \hat{\mathbf{x}}}\mathtt{P}\frac{\partial \mathbf{h}_2}{\partial \hat{\mathbf{x}}}^\top & \dots \\ \frac{\partial \mathbf{h}_2}{\partial \hat{\mathbf{x}}}\mathtt{P} & \frac{\partial \mathbf{h}_2}{\partial \hat{\mathbf{x}}}\mathtt{P}\frac{\partial \mathbf{h}_1}{\partial \hat{\mathbf{x}}}^\top & \frac{\partial \mathbf{h}_2}{\partial \hat{\mathbf{x}}}\mathtt{P}\frac{\partial \mathbf{h}_2}{\partial \hat{\mathbf{x}}}^\top + \mathtt{R}_2 & \dots \\ \vdots & \vdots & \vdots & \end{bmatrix}, \tag{3.12}$$

where $\mathtt{R}_i$ denotes the measurement noise in each prediction. The lower-right portion of $\mathtt{P}_m$ encoding the covariance values of the elements $\mathbf{z}_T^{(k|k-1)}$, is known as the *innovation covariance matrix* $\mathtt{S}$ in Kalman filter tracking. Generally, during the process of tracking, the matrix $\mathtt{S}$ becomes dense as is the matrix $\mathtt{P}$ progressively building correlations between different parts of the map. Practically, high correlation between two feature position estimates means that while we are uncertain about their absolute locations, their relative locations may be known with high accuracy. These correlations are of great importance to a convergent solution in SLAM and as stated by Durrant-Whyte and Bailey [2006]: the more these correlations grow, the better the solution. In fact, the analysis performed in this thesis to enforce robustness and efficiency of algorithms aims to exploit these correlation links as much as possible.

Predicting the locations of map features in the new image, the system can decide whether a feature lies in the predicted field of view and therefore enlist it as a measurement candidate. Given individual $p(\mathbf{z}_i)$ parts of this prediction, the image search for each feature can sensibly be limited to high-probability regions (what we call *active search*), which will practically often be small in situations such as tracking. Therefore, in MonoSLAM we can avoid costly exhaustive search of each predicted-to-be-visible feature in the whole image by restricting search to 'gated' elliptical regions around the predicted feature locations, of size determined by the innovation covariance $\mathtt{S}_i$ and a chosen number of standard deviations. Innovation is defined to be the discrepancy between the predicted and the observed feature locations. Hence, the innovation covariance is the expected deviation from this prediction. Expanding the sub-matrix on the diagonal of $\mathtt{P}_m$ corresponding to feature *i*, its innovation covariance is defined by:

$$\mathtt{S}_{ii} = \frac{\partial \mathbf{h}_i}{\partial \hat{\mathbf{x}}_c}\mathtt{P}_{xx}\frac{\partial \mathbf{h}_i}{\partial \hat{\mathbf{x}}_c}^\top + \frac{\partial \mathbf{h}_i}{\partial \hat{\mathbf{x}}_c}\mathtt{P}_{xy_i}\frac{\partial \mathbf{h}_i}{\partial \hat{\mathbf{y}}_i}^\top + \frac{\partial \mathbf{h}_i}{\partial \hat{\mathbf{y}}_i}\mathtt{P}_{y_i x}\frac{\partial \mathbf{h}_i}{\partial \hat{\mathbf{x}}_c}^\top + \frac{\partial \mathbf{h}_i}{\partial \hat{\mathbf{y}}_i}\mathtt{P}_{y_i y_i}\frac{\partial \mathbf{h}_i}{\partial \hat{\mathbf{y}}_i}^\top + \mathtt{R}_i \tag{3.13}$$

Uncertainty in the probabilistic prediction of feature image locations in MonoSLAM is dominated by the first term in the above expression which takes account for the uncertainty in camera pose introduced by the frame-to-frame motion model. Therefore, the size of these ellipses reflects the variance of the Gaussian distribution used for the motion model.

3.3 Active Feature Measurement and Map Update

MonoSLAM allows 'active' feature measurements in the sense that search for a feature can be confined within the bounds of the innovation covariance as defined in equation 3.13, rather than across the whole image which can be very time consuming. Performing exhaustive template match search within the three standard deviation gated ellipse for each feature, the top-scoring template match is taken as correct if its normalised sum-of-squared-difference score passes a threshold. The position of the match is further refined to subpixel accuracy by fitting a paraboloid in the local neighbourhood of the match and estimating its peak. Generally, subpixel refinement is considered to result to smoother estimated trajectories.

At low levels of motion model uncertainty, mismatches via this method are relatively rare, but in advanced applications of the algorithm or when the motion modelling is poor [Chli and Davison, 2008a; Clemente et al., 2007; Williams et al., 2007] it has been observed that compatibility tests find a significant number of matching errors and greatly improves performance. Chapter 5 addresses this issue explicitly and offers a way to handle multiple occurring matches per feature.

The saved patch of each feature to be measured is warped to produce a template of its expected appearance from the new camera position. The system maintains a record for both the successful and attempted measurements of each landmark. If a feature fails to match in a certain number of successive frames, it is classified as an unreliable landmark for tracking and thus gets automatically deleted from the map.

The vector of measurements $\mathbf{z}_T^{*(k)}$ *obtained* from feature matching in the new frame is then fed back into the system to update the map (EKF update step). An estimate of the posterior distribution over $\mathbf{x}^{(k)}$ can then be acquired as stated in equation 3.5. The mean and covariance of this estimate can be obtain as follows:

$$\hat{\mathbf{x}}^{(k|k)} = \hat{\mathbf{x}}^{(k|k-1)} + \mathtt{K}(\mathbf{z}_T^{*(k)} - \hat{\mathbf{z}}_T^{(k|k-1)}) \tag{3.14}$$

$$\mathtt{P}^{(k|k)} = \mathtt{P}^{(k|k-1)} - \mathtt{KSK}^{\top}, \tag{3.15}$$

where the Kalman gain $\mathtt{K}$ is defined to as $\mathtt{K} = \mathtt{P}^{(k|k-1)} \frac{\partial \mathbf{h}}{\partial \hat{\mathbf{x}}^{(k|k-1)}}^{\top} \mathtt{S}^{-1}$. From this point onward, the superscripts referring to the different time-stamps (i.e. k, $k-1$, etc.) are omitted for the sake of clarity and hereafter, the notation '$\mathbf{z}$' will generally refer to *predicted* measurements while the asterisk in $\mathbf{z}^*$ will denote *obtained* measurements.

3.4 System Initialisation and Map Maintenance

Since tracking with a single camera makes the task of estimating the depth of features harder and there is no way we can estimate the absolute scale of the scene, the system

can be initialised with a set of features with known relative positions to overcome that. This is not a necessary condition though since scale can be considered as a completely unknown degree of freedom without affecting performance, and it has been shown that reparametrisation of point features in terms of inverse depth as described earlier, allows efficient initialisation of the system.

At any given frame where the number of 'trackable' features drops below the desired threshold, the system looks to initialise new, distinctive visual landmarks by looking into randomly selected image regions where other features are not already present. In MonoSLAM we rely on the Shi and Tomasi criterion [Shi and Tomasi, 1994] to extract the visually salient regions in the form of 2D patches which are saved at the time the feature is initialised. The system however, is not specifically tied to this feature selector so any other feature detector/descriptor can also be used.

The maintenance of the SLAM map using the EKF requires quadratic computational complexity in the total number of features in the map. Hence, the key to achieving real-time performance lies in the assumptions made on the information retrieved during motion. Maintaining a sparse map of high-quality features, modelling the camera motion and actively searching for features guided by uncertainty, are the main techniques used to optimise the use of processing resources. However, if there is the need for a more detailed map or the environment mapped is large, the real-time performance of MonoSLAM is compromised since the time to update the filter begins to grow rapidly with the number of features. Also, as mentioned in section 3.3, during frame-to-frame matching the association of the observations with the map features is crucial for the accuracy and consistency of the tracker since once accepted they cannot be undone. In a nutshell, real-time performance of a visual SLAM system able cope with general camera motion and large map sizes is still a challenge. The rest of this thesis tackles this issue by consulting Information Theoretic principles introduced in the following chapter, to carefully guide the allocation of resources and make the least wasteful approximations to the complete problem.

4

Information Theory and Probabilistic Predictions

Our perception of the uncertain state of an event has the potential to advance upon the acquisition of additional relevant knowledge. In order to translate raw input data into useful cues however, we employ probabilistic reasoning essentially redistributing perceptual uncertainty based on the new evidence. On the whole, while an extra piece of information can prove of great importance in resolving uncertainty, this can only be accomplished at the expense of processing resources. When bounds are imposed on the resources available, as is the case in real-time applications, then achieving a compromise evolves into a real challenge: staying within the limited time budget of online performance requires balancing the potential gains and costs involved in processing incoming data.

In sequential tracking, any information extracted from the current scene in conjunction with any models available describing the processes involved, can serve as prior knowledge for the next frame. The use of such priors is of great benefit to SLAM as they tell us where to look for cues in the image, alleviating the burden of frame-to-frame processing. Surely though, the amount of information available in an incoming

image can be enormous. Thus, extracting and processing all of it can be an extensive and costly process which brings us to the conclusion that some selection is necessary. But on what grounds should we base our decisions on what is *worthy* of our time and effort from what is *dispensable*? Also, are we really making the most of the prior knowledge we choose to carry forward in the end?

This chapter looks deep into the value of the probabilistic predictions we can make in visual SLAM and ways we can use these efficiently at run-time. As will become evident in this and subsequent chapters, investing in 'smart' manipulation of the information available not only can help boost the computational efficiency and consistency of algorithms, but more importantly it provides a comprehensive insight into the task at hand. Here, we discuss the concepts and measures forming the Information Theoretic basis used throughout the rest of this thesis to guide efficient image processing and map-management in SLAM.

4.1 Principles of Information Theory

Information Theory is generally considered to have been founded by Claude Shannon in 1948. Initially aiming at tackling the engineering problem of reliable data transmission over a noisy channel, it has since been established as a means of *quantifying information content*. This section devotes some time to notation and description of the quantities of information later used to analyse the expected 'value' of features in the SLAM map and their relations.

4.1.1 Entropy

Using the notation introduced by Mackay [2003], a discrete variable x can take values in the set $A_X = \{a_1, a_2, \ldots, a_n\}$ with associated probabilities $P_X = \{p_1, p_2, \ldots, p_n\}$ such that $p_i = p(x = a_i)$. Therefore, the ensemble X describing the triplet $\{x, A_X, P_X\}$ has information entropy $H(X)$ defined as:

$$H(X) = E\left[\log \frac{1}{p(x)}\right] = \sum_{x \in A_X} p(x) \log \frac{1}{p(x)} \; . \tag{4.1}$$

In words, this describes the expected information content of a possible outcome on the value of x given the set of different possibilities '$x = a_i$' and their associated probabilities of occurrence. More intuitively, the entropy of a variable represents the uncertainty in its current state, often referred to as the *expected surprise* of the distribution. Thinking in terms of simple examples, there is less surprise expected when tossing a fair coin (where both heads and tails are equally likely) than when rolling a die (there are six equally likely outcomes). There is more uncertainty (and thus entropy) involved in

predicting the outcome of the die than that of the coin.

Taking this rationale a step further, the entropy of a particular variable reaches its maximum value in the most unpredictable (hence uncertain) case: when all the events in the event space are equiprobable. Hence, in the case of variable x above this happens when each possible outcome a_i has a probability $p(x = a_i) = 1/n$ of occurrence, resulting to the entropy of $H(X) = \log n$.

The choice of the base of the logarithm determines the unit of measurement of entropy. In this thesis, we use the binary logarithmic scale (with 'log' used as short for '$\log_2$') measuring entropy and information in absolute numbers of **bits**.

4.1.2 Joint Entropy

Introducing another discrete variable y taking values in the finite set B_Y with associated probabilities P_Y, we can deduce useful measures like the joint entropy of both x and y. This is defined as the entropy of the probability distribution of the pairings (x,y), expressed as:

$$H(X,Y) = \sum_{x \in A_X, y \in B_Y} p(x,y) \log \frac{1}{p(x,y)} \,. \tag{4.2}$$

In the special case that x and y are independent i.e. $p(x,y) = p(x)p(y)$ then it can easily be shown that their joint entropy is equivalent to the sum of their individual marginal entropies $H(X,Y) = H(X) + H(Y)$. If however the outcome of x is in some way affected by the outcome of y and vice versa, their joint entropy is also affected depending on the amount and the nature of their correlation (positive or negative).

4.1.3 Conditional Entropy

In the case that two variables are indeed correlated, then learning the exact value of one can affect the uncertainty (or the entropy) of the other. For example, if y is found to be equal to b_k (such that $b_k \in B_Y$), the entropy of the conditional distribution of x for this value of y then becomes:

$$H(X|y = b_k) = \sum_{x \in A_X} p(x|y = b_k) \log \frac{1}{p(x|y = b_k)} \,. \tag{4.3}$$

More generally, if y is to become known then we can predict the conditional entropy of the distribution of X given Y averaging over all possible outcomes of y:

$$H(X|Y) = \sum_{y \in B_Y} p(y) \left[\sum_{x \in A_X} p(x|y) \log \frac{1}{p(x|y)} \right] \tag{4.4}$$

$$= \sum_{x \in A_X, y \in B_Y} p(x,y) \log \frac{1}{p(x|y)} \,. \tag{4.5}$$

Observing Equations 4.1, 4.2 and 4.5 one can point out that

$$H(X|Y) = H(X,Y) - H(Y) \tag{4.6}$$

which indeed captures the true meaning of conditional entropy: it reflects the uncertainty expected to remain in the outcome of one variable, upon a supposed observation of the other.

4.1.4 Mutual Information

Entropy, which encodes the uncertainty in the state of a variable and Mutual Information, which expresses the amount of information *shared* between two variables, are the two key measures in Shannon Information Theory. Thinking of the common information between variables x and y in terms of the reduction in uncertainty we expect in the distribution of one upon a supposed observation of the other, below we derive the expression from first principles:

$$\begin{aligned} I(X;Y) &= E\left[\log \frac{p(x|y)}{p(x)}\right] && (4.7)\\ &= H(X) - H(X|Y) && (4.8)\\ &= \sum_x p(x) \log \frac{1}{p(x)} - \sum_{xy} p(x,y) \log \frac{1}{p(x|y)} \\ &= \sum_{xy} p(x,y) \log \frac{p(x|y)}{p(x)} \\ &= \sum_{xy} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} && (4.9) \end{aligned}$$

This measure which is termed as the Mutual Information between X and Y, is symmetric so $I(X;Y) = I(Y;X)$ holds. In the special case where the two variables are independent, then intuitively they share no information at all, which is confirmed when expanding $p(x,y)$ resulting to the cancellation of the numerator and denominator inside the logarithm of equation 4.9. If we now substituting Equation 4.6 into 4.8 we get,

$$I(X;Y) = H(X) + H(Y) - H(X,Y) \tag{4.10}$$

which can be visualised in Figure 4.1 together with all implicit relationships between the rest of the measures introduced in this section.

4.1.5 Continuous Variables

On attempting to extend these quantities for continuously distributed variables, one has to be cautious of preserving consistency. Standard procedure involves splitting

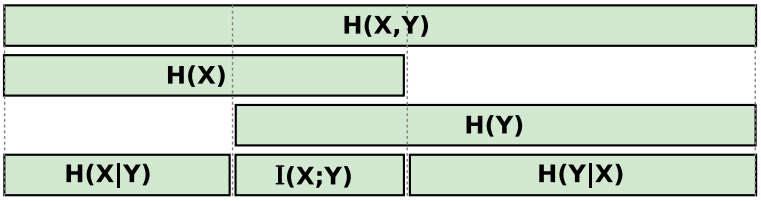


Figure 4.1: The relationship between marginal, joint and conditional entropies with mutual information.

This figure is based on an illustration from Mackay [2003]

the now continuous range of X into discrete interval bins each Δx wide, such that the probability of x falling within a particular bin is equal to $p(x)\Delta x$. Therefore entropy is described as:

$$H(X) = \sum_x p(x)\Delta x \, \log \frac{1}{p(x)\Delta x} \, .$$

Taking now the limit of $\Delta x \to 0$ it is evident that on every halving of Δx the entropy content increases by 1 bit, rendering the expression ill-behaved as indicated by Mackay [2003]. It is permissible however to take this limit for both continuous ranges of X and Y on the analogous expression for mutual information. Therefore following from the expression for discrete variables in Equation 4.9, the mutual information of the two continuous variables is derived by:

$$\begin{aligned} I(X;Y) &= \lim_{\Delta x,\Delta y \to 0} \left[\sum_{xy} p(x,y)\Delta x \Delta y \, \log \frac{p(x,y)\Delta x \Delta y}{p(x)p(y)\Delta x \Delta y} \right] && (4.11) \\ &= \int_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \, dxdy && (4.12) \\ &= \int_{x,y} p(x,y) \log \frac{p(x|y)}{p(x)} \, dxdy \, . && (4.13) \end{aligned}$$

4.1.6 Mutual Information in a Multivariate Gaussian

Given the brief introduction into the most important Shannon information measures, we now look at the special case of multivariate Gaussians since the goal is to use Information Theory in the context of SLAM. Considering a single, multi-variate Gaussian probability density describing the state vector $\mathbf{x}$ by a mean vector $\hat{\mathbf{x}}$ and a covariance matrix $\mathtt{P}$, we consider the mutual information shared between disjoint subsets of variables included in the state vector. Denoting the two subset-vectors by $\mathbf{a}$ and $\mathbf{b}$ of lengths N_a and N_b respectively, then we consider the partition of $\hat{\mathbf{x}}$ and $\mathtt{P}$ as follows:

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{a}} \\ \hat{\mathbf{b}} \end{pmatrix} \; ; \; \mathtt{P} = \begin{bmatrix} \mathtt{P}_{aa} & \mathtt{P}_{ab} \\ \mathtt{P}_{ba} & \mathtt{P}_{bb} \end{bmatrix} \, . \qquad (4.14)$$

Considering the effect that a supposed observation of vector **b** will have on the state of partition **a**, Davison [2005] has shown that the mutual information shared between two such partitions of **x** can be expressed as

$$I(\mathbf{a};\mathbf{b}) = E\left[\log\frac{p(\mathbf{a}|\mathbf{b})}{p(\mathbf{a})}\right] \tag{4.15}$$

$$= \frac{1}{2}\log\frac{|\mathrm{P}_{aa}|}{|\mathrm{P}_{aa}-\mathrm{P}_{ab}\mathrm{P}_{bb}^{-1}\mathrm{P}_{ba}|}\ . \tag{4.16}$$

Essentially, equation 4.16 suggests that the information content shared between **a** and **b** can be evaluated considering the reduction in the uncertainty of **a** incurred by the observation of **b**. While this expression seems very intuitive in terms of dividing the magnitude in the variance of **a** before and after measuring **b**, computing the actual value can be very expensive. Consider for example evaluating the shared information content between different parts of a 1000-long vector; handling matrix inversions and multiplications in the denominator are bound to take up vast amounts of processing time, consequently rendering such computations impossible for real-time processing. However, viewing this expression from a slightly different angle, Chli and Davison [2008a] considered the entropy in vector **x** as the joint entropy of the $(\mathbf{a},\mathbf{b})$ pair. This minor change in perspective proved key to reaching a much more cost-effective representation of the mutual information of **a** and **b**, as derived below. Starting from first principles and applying Bayes' rule in Equation 4.7:

$$I(\mathbf{a};\mathbf{b}) = E\left[\log\frac{p(\mathbf{a}|\mathbf{b})}{p(\mathbf{a})}\right] = E\left[\log\frac{p(\mathbf{a},\mathbf{b})}{p(\mathbf{a})p(\mathbf{b})}\right]\ . \tag{4.17}$$

However the joint probability density of $p(\mathbf{a},\mathbf{b})$ is by definition, equal to $p(\mathbf{x})$. Moreover, if N refers to the length of **x**, the probability distribution $p(x)$ can be expanded as

$$p(\mathbf{x}) = (2\pi)^{-\frac{N}{2}}|\mathrm{P}|^{-\frac{1}{2}}\ e^{-\frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^\top \mathrm{P}^{-1}(\mathbf{x}-\hat{\mathbf{x}})}\ . \tag{4.18}$$

A similar expression is applicable for the PDF of each partition, adapting the symbols accordingly. Therefore, expanding out these PDFs in Equation 4.17:

$$I(\mathbf{a};\mathbf{b}) = E\left[\log\frac{p(\mathbf{x})}{p(\mathbf{a})p(\mathbf{b})}\right] \tag{4.19}$$

$$= E\left[\log\frac{|\mathrm{P}|^{-\frac{1}{2}}\ e^{-\frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^\top \mathrm{P}^{-1}(\mathbf{x}-\hat{\mathbf{x}})}}{|\mathrm{P}_{aa}|^{-\frac{1}{2}}\ e^{-\frac{1}{2}(\mathbf{a}-\hat{\mathbf{a}})^\top \mathrm{P}_{aa}{}^{-1}(\mathbf{a}-\hat{\mathbf{a}})}\ |\mathrm{P}_{bb}|^{-\frac{1}{2}}\ e^{-\frac{1}{2}(\mathbf{b}-\hat{\mathbf{b}})^\top \mathrm{P}_{bb}{}^{-1}(\mathbf{b}-\hat{\mathbf{b}})}}\right]$$

$$= \frac{1}{2}\log\frac{|\mathrm{P}_{aa}||\mathrm{P}_{bb}|}{|\mathrm{P}|} - \frac{1}{2\ln 2}E\left[(\mathbf{x}-\hat{\mathbf{x}})^\top \mathrm{P}^{-1}(\mathbf{x}-\hat{\mathbf{x}})\right]$$

$$+ \frac{1}{2\ln 2}\left(E\left[(\mathbf{a}-\hat{\mathbf{a}})^\top \mathrm{P}_{aa}{}^{-1}(\mathbf{a}-\hat{\mathbf{a}})\right] + E\left[(\mathbf{b}-\hat{\mathbf{b}})^\top \mathrm{P}_{bb}{}^{-1}(\mathbf{b}-\hat{\mathbf{b}})\right]\right) \tag{4.20}$$

Using arguments of Cover and Thomas [2006]:

$$E\left[(\mathbf{a}-\hat{\mathbf{a}})^\top \mathtt{P}_{aa}{}^{-1}(\mathbf{a}-\hat{\mathbf{a}})\right] = E\left[(\mathbf{a}-\hat{\mathbf{a}})(\mathbf{a}-\hat{\mathbf{a}})^\top \cdot \mathtt{P}_{aa}{}^{-1}\right] \quad (4.21)$$

$$= \left(E\left[\mathbf{a}\mathbf{a}^\top\right] - \hat{\mathbf{a}}\hat{\mathbf{a}}^\top\right) \cdot \mathtt{P}_{aa}{}^{-1} \quad (4.22)$$

$$= \mathtt{P}_{aa} \cdot \mathtt{P}_{aa}{}^{-1} = N_a \,. \quad (4.23)$$

Therefore, applying this result into Equation 4.20 and using $N = N_a + N_b$:

$$I(\mathbf{a};\mathbf{b}) = \frac{1}{2}\log\frac{|\mathtt{P}_{aa}||\mathtt{P}_{bb}|}{|\mathtt{P}|} + \frac{1}{2ln2}\left(-N + N_a + N_b\right) \quad (4.24)$$

$$= \frac{1}{2}\log\frac{|\mathtt{P}_{aa}||\mathtt{P}_{bb}|}{|\mathtt{P}|} \,. \quad (4.25)$$

Consequently, this last expression is a much more efficient evaluation to perform than that of Equation 4.16. In fact, this formulation is fundamental to making our Active Matching algorithm (discussed in Chapter 5) perform in real-time, since the computation of the shared information content between two partitions of the state vector is performed many times throughout the processing of a single frame.

4.2 Information Theory in Probabilistic Robotics

The notion of information has long been used in SLAM and probabilistic robotics, albeit in a quite different sense than discussed in this thesis. For the sake of completeness, before introducing the contributions of this work, below we summarise the most important uses of information in the field.

4.2.1 Active Control for Exploration

Mutual information has primarily been used in the sense of actively controlling the behaviour of the robot or camera during exploration. Such examples are the works of Bryson and Sukkarieh [2005] and Vidal-Calleja et al. [2006] who evaluate the information gain for all possible actions (e.g. turn left, keep on straight) essentially guiding the robot movements to maximise the quality of estimates. Depending on the question posed, mutual information can be formulated accordingly to achieve the objective: whether this is active localisation of the robot with respect to a known map, or the converse problem of optimising mapping for a known trajectory, or even a combination of both localisation and mapping as is the case in SLAM, the key for efficient exploration is in the evaluation of input information.

However, while all these methods study the enhancement of the navigation strategy assuming the ability to control the motion of the robot, this thesis looks at the more

fundamental problem of making the most of the data that is already available during general tracking. *Active* behaviour in this work can only refer to dynamic decisions that we are able to make only within the context of manipulating the input material at every frame – these decisions cannot have an effect on the trajectory followed or the camera viewpoint as this is left entirely up to the intentions of the carrier of the camera (human or robot). Dealing with a less constrained problem, the questions tackled here lead to a methodology applicable on a wider range of problems.

4.2.2 Information Filters

A special family of filters called 'information filters' appear in the literature to characterise probability distributions. Like the Kalman filter and the Extended Kalman filter, an information filter represents the belief about the state of a variable by a Gaussian. However, rather than parametrising a multivariate normal distribution $\mathcal{N}(\hat{\mathbf{x}}, \mathtt{P})$ by its mean and covariance matrix, it is instead parametrised by $\mathcal{N}^{-1}(\eta, \Lambda)$ where η is called the 'information vector' and Λ is the 'information matrix' such that $\eta = \mathtt{P}^{-1}\hat{\mathbf{x}}$, and $\Lambda = \mathtt{P}^{-1}$. Consequently, Λ is also referred to as the inverse covariance matrix. Transforming the standard expression describing the PDF of vector $\mathbf{x}$ by its mean and covariance, one can easily confirm that the dual expression in the information form corresponds to:

$$p(\mathbf{x}) = const.\ e^{\left(-\frac{1}{2}\mathbf{x}^\top \Lambda \mathbf{x} + \mathbf{x}^\top \eta\right)}\ , \tag{4.26}$$

where '*const.*' here is a constant. This formulation, while having very similar properties to the Kalman-based representations, it has the advantage that conditioning on a subset of variables is a very cheap operation meaning that integration of new input data into the state can be fast. However, marginalising out a subset of variables requires matrix inversion to resolve estimates back into a probability distribution which is much more expensive. On the other hand, marginalisation corresponds to a straightforward deletion of entries (corresponding to the variables undergoing marginalisation) in the equivalent covariance form. In essence, as Eustice et al. [2005a] has shown very clearly, the conditioning and marginalisation processes in the two representations are inversely analogous and so are their costs.

Several successful applications of the information filters exist in the literature. Multi-sensor systems like the work of Manyika [1993] and more recently Eustice et al. [2006] benefit from the fast information fusion implied in this formulation, while interestingly other works reviewed in detail by Thrun et al. [2005] aim at exploiting the sparse structure of the inverse covariance matrix and the fact that information filters can be thought of as graphs, otherwise referred to as Gaussian Markov random fields. Indeed, any two conditionally independent variables (or 'd-separated' in graph theoretic terminology) while having a non-zero cross-covariance entry, their corresponding

inverse-covariance entry is zero. Thinking in terms of the Shannon measures defined earlier, these two variables do share a non-zero mutual information – it only becomes zero when the variables are entirely independent. Following the intuition of this example, it is evident that the entries of the information matrix Λ are subtly different from Shannon's mutual information. In fact, the term 'information matrix' comes from a different measure of information, namely the Fisher information as pointed out by Mutambara [1998]. In the case of multivariate distributions, this corresponds to the matrix $J(\theta)$ defined for distributions parametrised by vector θ. Following the notation of Cover and Thomas [2006], given the density function $f(\mathbf{x};\theta)$ relating vector $\mathbf{x}$ with the parameters vector θ, the elements of matrix the Fisher Information matrix J are defined as follows:

$$J_{ij}(\theta) = \int f(\mathbf{x};\theta)\frac{\partial}{\partial\theta_i}\ln f(\mathbf{x};\theta)\frac{\partial}{\partial\theta_j}\ln f(\mathbf{x};\theta)d\mathbf{x}\,. \tag{4.27}$$

Expressing this in terms of the likelihood of measurements with respect to the state vector we aim to estimate in SLAM, it has been shown that J becomes equivalent to the inverse covariance matrix Λ (assuming Gaussian noise and minimum mean-squared-error predictions). While this information matrix exhibits attractive properties like sparseness, the true meaning of individual entries is not at all obvious. This thesis studies Shannon-based information measures and their application to different parts of the processes involved in SLAM with the aim to provide a more comprehensive insight into the relationship of the quantities discussed.

4.3 Information Value of a SLAM Measurement

Probabilistic filtering for SLAM involves maintaining a state representation of the map and the camera parameters, which are constantly refined based on the measurements made in the course of tracking. Using probabilistic inference we are able to both make predictions regarding the new state since the last estimated pose but also update the state parameters following a collection of observations made on a subset of these parameters. In this thesis we use the term 'observation' or 'measurement' to refer to the process of acquiring matches of feature-patches predicted to be visible in the given image.

As explained in Chapter 3, in a visual SLAM system like MonoSLAM the system bases any state updates upon the successful or failed matches of the features it has attempted to measure. Since this is the only observable part of the state vector (the new camera position or viewpoint can only be inferred from the observations), these measurements are vital to successful tracking. However, as hinted earlier not all measurements can be equally informative when inferring the state of the non-observable

parts of the state. The amount of information each measurement can provide though depends on a variety of factors like the uncertainty in either the position of the feature prior to measurement or the camera state, but more importantly the correlation between the two. Acknowledging this fact, one can make predictions about the value of a measurement prior to making it effectively guiding the consumption of processing resources towards more promising actions.

Selective processing is key to online processing since handling the incoming bulk of information at frame-rate becomes a challenge. Aiming to achieve a balance between the knowledge gained and the time spent on processing, here we investigate the information value of possible feature measurements in a SLAM map. While work has previously been done in cutting down unnecessary processing upon the reception of a new image (e.g. active feature search during frame-to-frame matching), the cues that are available in probabilistic filtering and have usually been overlooked are the correlations of predicted, candidate feature measurements. These correlations are often very strong, since all predictions about feature locations depend on common parts of the scene state. In a nutshell, the presence of strong correlation between two candidate measurements means that measuring one feature tells us a lot about where to look for the other. However, the level of correlation of either features with each other or with the camera state can vary substantially, confirming that not all image cues are equally valuable in resolving the current uncertain state.

The value of a measurement primarily depends on the reference question we aim to answer (e.g. 'where is the camera?' vs. 'where is feature f?') but also a long list of influential factors including the the type of environment where tracking takes place, the camera dynamics or the density of features being tracked. Each and every factor has a substantial effect on the estimation process and therefore on the worth of each measurement, however it is practically impossible to examine them individually at runtime. This is where Information Theory can be employed to provide dynamic measures of how valuable a feature measurement really is, exploiting the power of the fully probabilistic framework maintained in filtering.

This section studies the application of mutual information in the context of SLAM providing an understanding of the relationships between features and the inherent redundancy in the map. Previously, Manyika [1993] has suggested an information-based framework in robot localisation and mapping and more recently Davison [2005] suggested the application of Information Theory principles in SLAM techniques to guide efficient image processing. Following this direction, here we explore further the insights provided by mutual information which leads to effective and efficient algorithms for guided processing of incoming data discussed in subsequent chapters. Since the work of Davison [2005] has been the main inspiration for the contributions of this the-

sis, below we begin with a brief discussion of the ideas and conclusions drawn there before presenting our investigation on mutual information.

4.3.1 Feature MI in Measurement Space

Upon the reception of a new frame during sequential SLAM, the system makes a series of predictions including the new camera pose and therefore the subset of known features predicted to be visible from the new viewpoint. These features consequently comprise the set of candidate measurements. Such candidate measurements vary in two significant ways: the amount of information which they are expected to offer, and the amount of image processing likely to be required to extract a match; both of these quantities can be computed directly from the current search prior. There are ad-hoc ways to score the value of a measurement such as search ellipse size, used for simple active search for instance in Davison and Murray [1998]. However, Davison [2005], building on early work by others such as Manyika [1993], explained clearly that the Mutual Information (MI) between a candidate and the scene state is the essential probabilistic measure of measurement value.

Following the notation introduced in Chapter 3 and the expression for the MI of continuously distributed variables in Equation 4.13, below we evaluate the MI of the camera state $\mathbf{x}_c$ and a candidate predicted measurement $\mathbf{z}_i$ as:

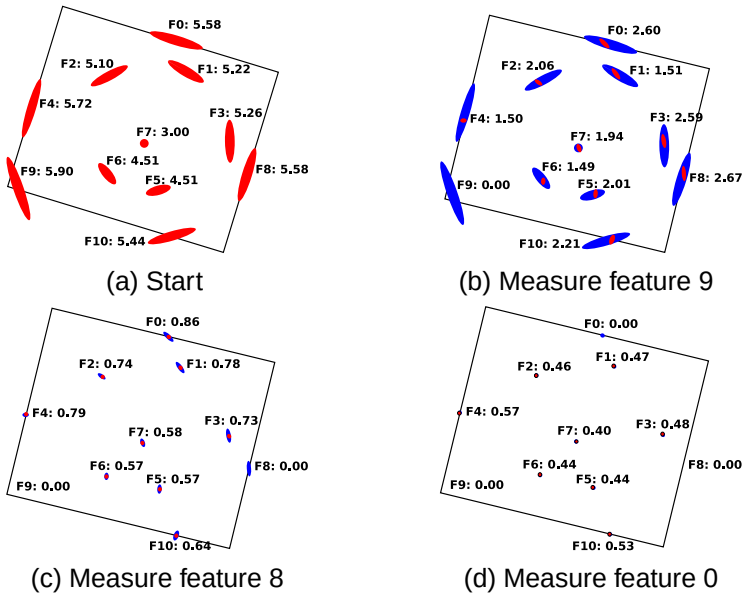
$$I(\mathbf{x}_c;\mathbf{z}_i) = E\left[\log \frac{p(\mathbf{x}_c|\mathbf{z}_i)}{p(\mathbf{x}_c)}\right] \tag{4.28}$$

$$= \int_{\mathbf{x}_c,\mathbf{z}_i} p(\mathbf{x}_c,\mathbf{z}_i)\log \frac{p(\mathbf{x}_c|\mathbf{z}_i)}{p(\mathbf{x}_c)} d\mathbf{x}_c d\mathbf{z}_i \ . \tag{4.29}$$

Therefore $I(\mathbf{x}_c;\mathbf{z}_i)$ describes the number of **bits** of information we expect to learn about the uncertain vector $\mathbf{x}_c$ by determining the exact value of $\mathbf{z}_i$. Using this measure to evaluate the MI scores of each candidate predicted measurement $\mathbf{z}_i$, we can fairly compare them to determine which one has most utility in reducing uncertainty in the state $\mathbf{x}_c$, even if the features themselves have different types (e.g. point feature vs. edge feature). Furthermore, aiming to quantify the effort incurred in a supposed measurement of a given candidate Davison [2005] proposed the 'information efficiency' score obtained by dividing the MI value of a candidate by a measure of the associated measurement cost, essentially describing the number of bits to be gained per unit of computation.

Mutual Information vs. Information Efficiency

Suppose that a rectangular rigid object is free to move in the $2D$ space as shown in Figure 4.2 and the aim is to estimate its position and orientation given that a set of



(a) Start (b) Measure feature 9

(c) Measure feature 8 (d) Measure feature 0

Figure 4.2: Pinning down the object by selecting measurements according to the Mutual Information they are predicted to provide to the state of the object. Three standard deviation uncertainty bounds are shown before and after measurement in blue and red respectively. Alongside the integer label of each measurement candidate is its MI with the object state.

This figure is based on an illustration from Davison [2005]

candidate feature measurements is available to us. These features have a predicted location and an associated uncertainty each and they all lie on the object implying that their position estimates are tightly correlated. The features with the biggest uncertainty are predicted to provide the most information to the state of the object as illustrated in Figure 4.2(a). Measuring the feature with the highest MI value with the object state first, has indeed a big impact on the object state which is reflected in the dramatically reduced uncertainty regions of the rest of the candidates in Figure 4.2(b). Proceeding with measuring the candidate with the biggest predicted MI with the object state at each step, it is evident that the object is soon localised with sufficient accuracy (this is assumed to be when further candidate measurements are predicted to provide less than 1 *bit* of information).

Of all the possible sequences of measurement, the one used in Figure 4.2 is the optimal one with respect to localising the object in the fewest measurements possible. However, each search for a feature match requires a number of image processing operations proportional to the 3σ uncertainty region of that feature depicted with ellipses in both figures 4.2 and 4.3. Indeed a feature measurement involves exhaustive search for a match across all possible locations within that region. As a result, when interested in the speed of processing as well as accuracy of the result one has to take into account that when choosing to measure the candidate with the biggest uncertainty first

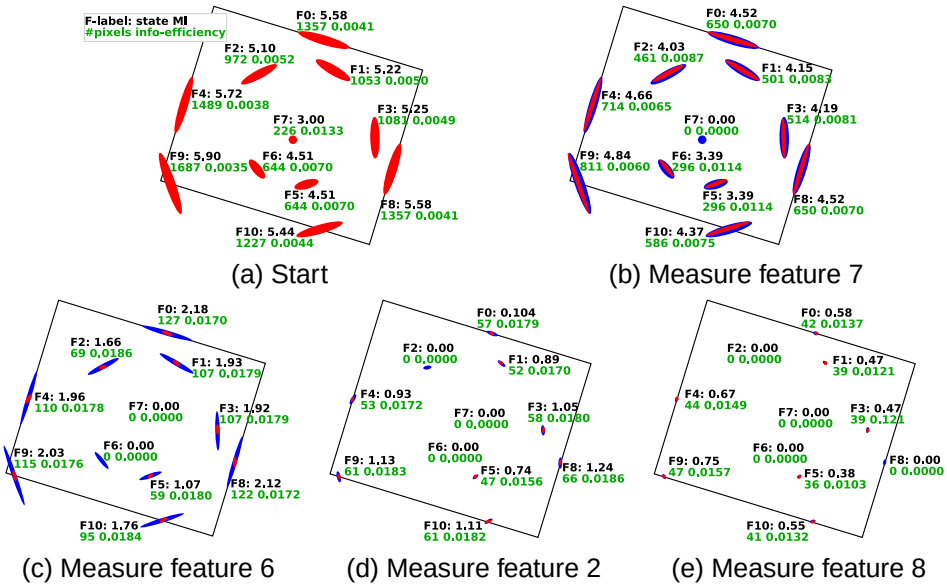


Figure 4.3: Selecting measurements by information efficiency value. Here, aside from the label of each feature and its MI with the object state, we also display the area of its corresponding search ellipse in pixels and the relevant information efficiency score (in bits per pixel).

This figure is based on an illustration from Davison [2005]

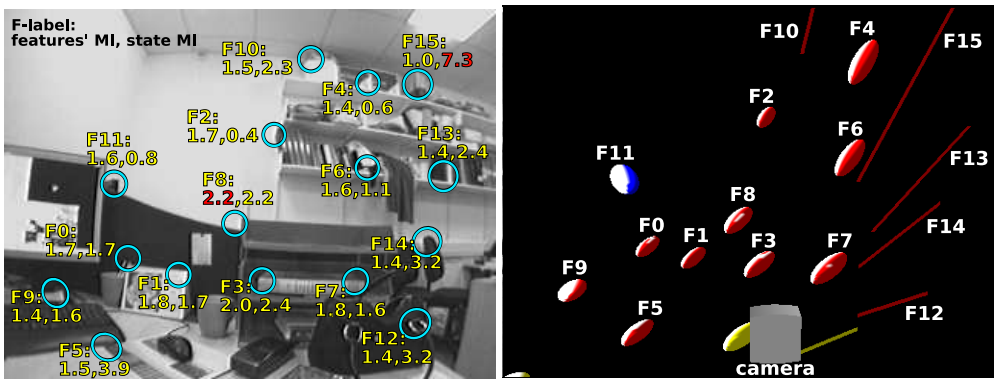
it will inevitably yield a large number of image processing operations. In an attempt to optimise then with respect to both accuracy and speed, Davison [2005] proposed an alternative scheme of ordering measurements which takes both mutual information and computational cost into account making decisions based on the **information efficiency** ratio of each feature:

$$\textbf{Information Efficiency} = \frac{\textbf{MI with the object state}}{\textbf{Area of search region}} \tag{4.30}$$

Figure 4.3 presents the matching steps involved for the same positioning scenario, but making decisions on a highest-information-efficiency-first basis. In this case, one additional measurement is necessary achieve the desired level of accuracy but it is evident that the overall area searched is reduced (and therefore the computational cost). In Chapter 5 where we discuss our Active Matching algorithm, we make use of these findings extending the methodology to cope with the hurdles imposed when dealing with real images as opposed to simulation scenarios.

Feature's MI with All Other Features

While Davison [2005] has proposed using the measure of the MI a candidate measurement has with the object state, evaluating this in practice while tracking real scenes with SLAM has not proved very stable: the most recently initialised features have a

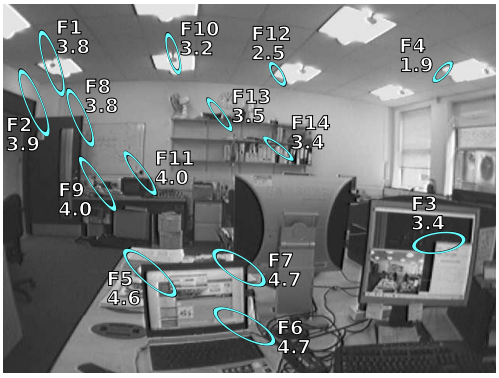


(a) Camera view (b) 3D map view

Figure 4.4: Comparing MI scores of each candidate with the rest (features' MI) and the MI of each candidate with the camera state (state MI). Figure (a) illustrates both MI measures in bits, where for each feature we display its label along with its features' MI score on the left and its state MI score on the right, as computed based on filter data for this frame. (b) is a visualisation of the global 3D uncertainty in each feature. Typically, newly initialised features have large depth uncertainty and inherit most of the camera's uncertainty boosting their state MI scores like $F15$ here (note that in this example feature labels are chosen to reflect the order of initialisation). However, $F15$ being the 'youngest' feature in the system has not built strong correlations with the rest of the features achieving the smallest features' MI. Features' MI scores exhibit more stable behaviour since they only take local frame data into account. $F8$ being a moderately old feature with a good position estimate, has the highest features' MI score promising large uncertainty reductions in the rest of the visible features which is more desirable for frame-to-frame matching.

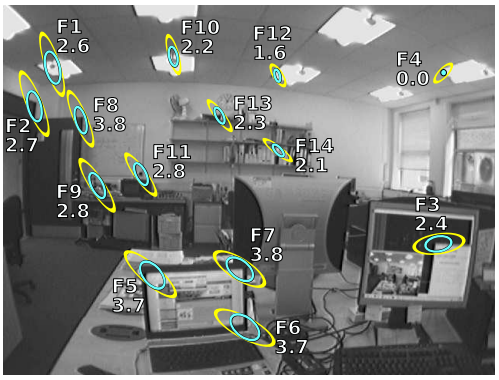
large uncertainty in their depth estimate and also inherit the increasing uncertainty in the camera pose, which immediately makes their MI values rise substantially with respect to older features visible in the frame as demonstrated in the example of Figure 4.4. As a result, these features lie at the top of the MI-scores list influencing the order of measurement until they get initialised properly. Moreover, when the global uncertainty in the map is large after continued exploration, 'younger' features again promise larger reductions in state uncertainty while this is not necessarily desirable when the objective is frame-to-frame matching. On the other hand, transforming all calculations into measurement space (i.e. image space) and computing the MI scores of features with respect to the rest of the candidates for measurement not only can we capture the true worth of a measurement action in local matching but also the MI values obtained exhibit more constant behaviour due to the very fact that relative data is taken into account (as opposed to the global camera location as before).

This measure has the very satisfying property that active search for features can proceed purely in measurement space while it is also appealing in problems where it is not desirable to make manipulations of the full state distribution during active search, such as SLAM or SFM applications where the state vector is potentially very

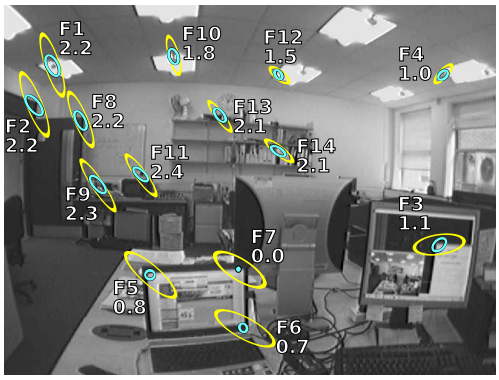


(a) 14 candidate measurements

Select a candidate, measure it and update the search-state



(b) $\mathbf{F}_4$ is measured



(c) $\mathbf{F}_7$ is measured

Figure 4.5: The information value of a feature with respect to all the candidates for measurement is the total reduction expected in uncertainty. This is an example illustrating the MI scores of candidates with respect to the rest of the features in the scene, as computed using Equation 4.31 and displayed in absolute number of bits. Before the matching begins in (a) every feature has some uncertainty in its position. Measuring one of the candidates and updating using the EKF update rules, at the bottom row of images is a superposition of the outcomes depending on the choice made: measuring a feature with low information value as in (b) or a feature with high MI as in (c). The effect of each of these measurements is evident in the relative reduction in the ellipse-areas and is also reflected in the updated MI value of each feature. In essence, the resulting overall uncertainty in the unmeasured features is reduced much more when measuring $\mathbf{F}_7$ predicted to provide 4.7 *bits* than measuring $\mathbf{F}_4$ with predicted MI of 1.9 *bits*.

large. Following our derivation in Equation 4.25 of the MI between two partitions of a means vector, we consider the MI of a predicted measurement $\mathbf{z}_i$ with respect to the rest of the feature-candidates $\mathbf{z}_{T\neq i}$:

$$I(\mathbf{z}_i;\mathbf{z}_{T\neq i}) = \frac{1}{2}\log\frac{|\mathtt{S}_{ii}||\mathtt{S}_{T\neq i\ T\neq i}|}{|\mathtt{S}|}\ , \tag{4.31}$$

where the vector of predicted measurements $\mathbf{z}_T$ and the innovation covariance matrix $\mathtt{S}$ are partitioned as follows:

$$\mathbf{z}_T = \begin{pmatrix} \mathbf{z}_i \\ \hline \\ \mathbf{z}_{T\neq i} \end{pmatrix}, \quad \mathtt{S} = \left[\begin{array}{c|c} \mathtt{S}_{ii} & \mathtt{S}_{i\ T\neq i} \\ \hline \\ \mathtt{S}_{T\neq i\ i} & \mathtt{S}_{T\neq i\ T\neq i} \end{array}\right]. \tag{4.32}$$

This expression is used in Chapter 5 to suggest the order in which to carry out measurements for efficient and robust frame-to-frame feature searching and matching. Figure 4.5 illustrates through a real tracking example the application the features' MI scores and the effect different measurements have to the uncertainty left in matching that frame. Projecting the 3σ uncertainty regions of visible features feature in image space as computed from $\mathtt{S}$ Figures 4.5(b) and (c) superimpose the individual uncertainty reductions of unmeasured features when measuring a candidate with a low and a high features' MI score respectively.

4.3.2 Pairwise MIs Between Features

Aiming at isolating the effect that a candidate measurement has on individual features, here we also introduce the notion of pairwise MI as the mutual information between two different features in the SLAM map, in measurement space. Namely, the MI shared between candidates $\mathbf{z}_i$ and $\mathbf{z}_k$ is:

$$\begin{aligned} I(\mathbf{z}_i;\mathbf{z}_k) &= E\left[\log\frac{p(\mathbf{z}_i|\mathbf{z}_k)}{p(\mathbf{z}_i)}\right] && (4.33)\\ &= \int_{\mathbf{z}_i,\mathbf{z}_k} p(\mathbf{z}_i,\mathbf{z}_k)\log\frac{p(\mathbf{z}_i|\mathbf{z}_k)}{p(\mathbf{z}_i)}d\mathbf{z}_i d\mathbf{z}_k && (4.34)\\ &= \frac{1}{2}\log\frac{|\mathtt{S}_{ii}||\mathtt{S}_{kk}|}{|\mathtt{S}_{i,k}|}\ , && (4.35) \end{aligned}$$

where $\mathtt{S}_{i,k}$ is the joint innovation covariance of both candidates. The mutual information between predicted measurements of features captures their common information content, therefore providing an absolute, normalised measure of their correlation.

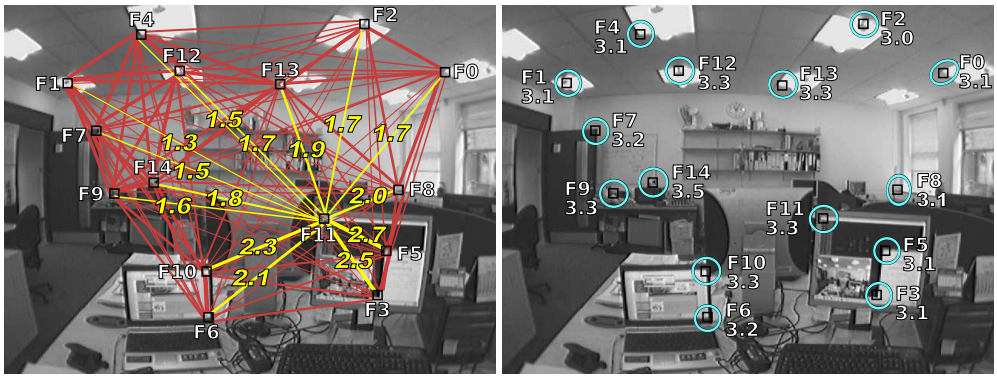
The MI Matrix

We can now define the Mutual Information matrix as below, so that every off-diagonal entry is calculated based on sub-blocks of $\mathtt{S}$ and represents the expected information gain of a candidate measurement given the exact state of another. If N is the total number of candidates then:

$$\mathtt{I}(\mathbf{z}_T) = \begin{bmatrix} * & I(\mathbf{z}_1;\mathbf{z}_2) & \dots & I(\mathbf{z}_1;\mathbf{z}_N) \\ I(\mathbf{z}_2;\mathbf{z}_1) & * & \dots & I(\mathbf{z}_2;\mathbf{z}_N) \\ \vdots & \vdots & \vdots & \vdots \\ I(\mathbf{z}_N;\mathbf{z}_1) & I(\mathbf{z}_N;\mathbf{z}_2) & \dots & * \end{bmatrix}. \tag{4.36}$$

This matrix is symmetric and the elements on the diagonal are not defined here and therefore filled with $*$'s (while mutual information of a variable with itself can be shown to be equal to the entropy of that variable [Cover and Thomas, 2006], it is a meaningless entity in this study of relationships between variables). The matrix has a value for every pair of features predicted to be observed on each frame and we can use it to analyse feature correlations on a frame by frame basis. While two features that have never been predicted to be observed together will have an MI value of zero, any features being covisible throughout a substantial number of frames *and* moving consistently will share strong mutual information links. On the other hand, if two features despite being co-observed, have significant depth difference in the scene they are bound to share a weaker MI link since this translates into parallax difference in image space meaning that they won't really move consistently from the viewpoint of the camera. In Figure 4.6(a) is a visual projection of how the MI matrix looks in a real tracking example frame. This MI matrix forms the basis for all of the analysis we conduct in Chapter 6 to discover the map structure in the context of SLAM and use it to suggest meaningful approximations for large-scale mapping.

Both measures of MI introduced in this section prove useful in different problems as will become apparent in the chapters to follow due to the fact that they provide the answer to different, equally important questions: features' MI gives a measure of the *joint* expected reduction in uncertainty upon a measurement over the rest of the visible features in the scene, while pairwise MI describes the information content shared between individual combinations of visual features. The two measures are indeed related since they describe information based on the same probabilistic data, however their relationship is not straightforward. One might naively say that the features' MI score is equivalent to the sum of pairwise MIs it shares with the rest of the features. However, this does not hold since the *type* of information shared between features is crucial in the relationship of the two measures: while a candidate measurement A is predicted to provide some n bits of pairwise MI to either of B and C, part of the information that this



(a) Pairwise MIs (b) Features' MIs

Figure 4.6: Comparison of pairwise vs. features' MI. While features' MIs in (b) provide a measure of the MI a candidate measurement is predicted to provide to the rest of the features in the scene, pairwise MIs in (a) reflect the individual information gain in each feature with respect to the candidate in question. In (a) is a visual projection of the MI matrix for this frame highlighting the MI links spanning out of $\mathbf{F}_{11}$ (MI values are shown in absolute number of bits). Depending on the strength of correlation between two features, their pairwise MI reflects how much information they share in common. For example, measuring $\mathbf{F}_{11}$ is only predicted to provide 1.3 *bits* to $\mathbf{F}_1$ but 1.4 *bits* more to its strongly correlated neighbour $\mathbf{F}_5$. To contrast the two different MI scores introduced here, (b) shows the MI score of each feature with respect to *all* other unmeasured features for the same frame. In this example, it is evident that while the scores in (b) do not vary much, the pairwise MIs in (a) capture more subtle differences between combinations of features. Both measures however, prove useful in different problems as discussed in subsequent chapters.

measurement will pass on to B is the same as the information it will pass on to C as the uncertainty of B and C will be reduced along the same direction and magnitude as the uncertainty of A. The amount of information overlap passed on from a measurement to the rest of the features is of course a function of the feature correlations and initial uncertainties. The subsequent chapters discuss how these MI measures can be applied in SLAM to guide efficient processing while enforcing consistency of the algorithms involved.

5

Active Matching

In the feature matching tasks which form an integral part of visual tracking or SLAM, there are invariably priors available on the absolute and/or relative image locations of features of interest. Usually, these priors are used post-hoc in the process of resolving feature matches and obtaining final scene estimates, via 'first get candidate matches, then resolve' consensus algorithms such as RANSAC or JCBB. In this chapter we show that the dramatically different approach of using priors dynamically to guide a feature by feature matching search can achieve global matching with far fewer image processing operations and lower overall computational cost. Essentially, we put image processing *into the loop* of the search for global consensus. In particular, our approach is able to cope with significant image ambiguity thanks to a dynamic mixture of Gaussians treatment. In our fully Bayesian algorithm denoted Active Matching, the choice of the most efficient search action at each step is guided intuitively and rigorously by expected Shannon information gain as discussed in Chapter 4. We demonstrate the algorithm in feature matching as part of the sequential MonoSLAM system for 3D camera tracking with a range of settings, and give a detailed analysis of performance which leads to performance-enhancing approximations to the full algorithm.

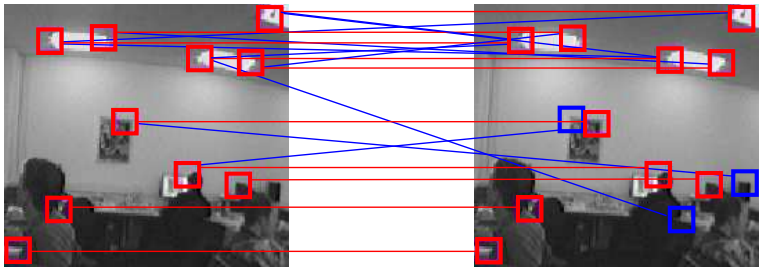


Figure 5.1: Bottom-up matching: get candidate matches, then resolve. The first cue for matching is similar appearance of features. The occurrence of mismatches in inevitable for matching in a real scene, as demonstrated in this example. This problem of data association is tackled by searching for consensus. RANSAC is an example of a standard method which resolves mismatches by choosing a random set of correspondences, hypothesising a solution and checking the number of matches in agreement with the proposed model.

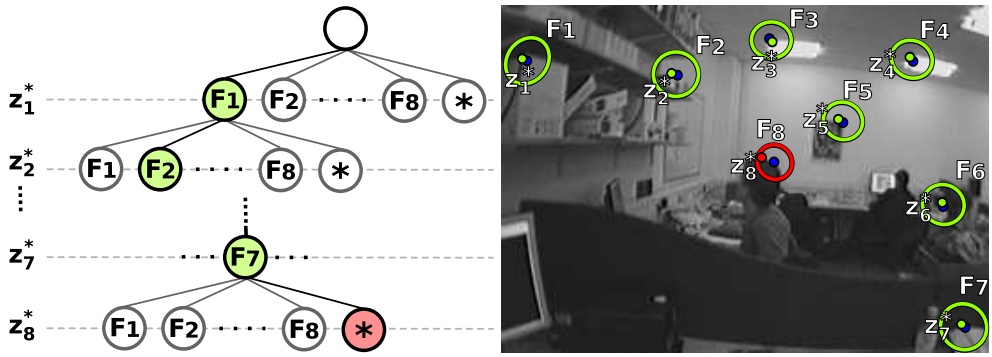
5.1 Introduction

It is well known that the key to obtaining correct feature associations in potentially ambiguous matching (data association) tasks using computer vision or other sensors is to search for a set of correspondences which are in *consensus*: they are all consistent with a believable global hypothesis. The usual approach taken to search for matching consensus is as follows: first candidate matches are generated, for instance by detecting all of a certain type of salient features in a pair of images and pairing up features which have similar appearance descriptors. Then, incorrect 'outlier' matches are pruned by proposing and testing hypotheses of global parameters which describe the world state of interest — the 3D position of an object or the camera itself, for instance. The random sampling and voting algorithm RANSAC proposed by Fischler and Bolles [1981] has been widely used to achieve this in geometrical vision problems.

Outliers are match candidates which lie outside of bounds determined by global consensus constraints. The idea that inevitable outlier matches must be 'rejected' from a large number of candidates achieved by some blanket initial image processing is deeply entrenched in computer vision and robotics.

The approach of our *Active Matching* paradigm is very different — to cut outliers out at source wherever possible by searching only the parts of the image where true positive matches are most probable. Both individual feature motion assumptions (such as that the image displacement of a feature between consecutive video frames will be bounded) *and* global consensus constraints can be expressed as priors on the true absolute and relative locations of features within a rigorous Bayesian framework.

In Active Matching, instead of searching for all features and then resolving, feature searches occur one by one within tightly targeted regions. The results of each search affect the regions within which it is likely that each of the other features will



(a) Interpretation tree (b) Image space

Figure 5.2: JCBB [Neira and Tardós, 2001] applied to visual tracking. This is a probabilistic algorithm for resolving global consistency between candidate matches. An interpretation tree and branch and bound search are used to evaluate the joint probability of proposed matches given a probabilistic prior on their joint location. The matches need to be obtained before resolving consensus, hence this is still a bottom-up method. Here is an example where the tree in (a) is used to pair each observation $\mathbf{z}_i^*$ with a known feature such that the all such pairings are jointly compatible. The match found for $\mathbf{F}_8$ gets rejected (considered as spurious) since the implied prediction error does not comply with the rest of the pairings (note: the blue blobs denote the predicted locations of features before measurement).

lie. This is thanks to the same inter-feature correlations of which standard consensus algorithms take advantage — but our algorithm's dynamic updating of these regions within the matching search itself means that low probability parts of the image are *never examined at all*. The result is that the number of image processing operations required to achieve global matching is reduced by a large factor.

Based in the information theoretic framework analysed in Chapter 4, we demonstrate the ability of information theory to intelligently guide the step by step search process and answer the question "where to look next?". The expected information content of each candidate measurement is computed and compared, and can also be traded off against the expected computational cost of the image processing required. The absolute bit units of information scores mean that heterogeneous feature types can be rigorously and intuitively combined within the same matching process. Information theory can also indicate when matching should be terminated at a point of diminishing returns.

While matching is often formulated as a search for correspondence between one image and another (for example in the literature on 3D multi-view constraints with concepts such as the multi-view tensors), stronger constraints are available when we consider matching an image to a *state* — an estimate of world properties perhaps accumulated over many images. Uncertainty in a state is represented with a probability distribution. Matching constraints are obtained by projecting the uncertain world state into a new image, the general result being a joint prior probability distribution over the

image locations of features. These uncertain feature *predictions* will often be highly correlated. When probabilistic priors are available, the random sampling and preset thresholds of RANSAC are unsatisfying. In more recent variants of the algorithm it has been realised that an unnecessarily large number of association hypotheses gets tested, therefore speedups have been proposed either by a two-step randomised selection of hypotheses as done in [Chum and Matas, 2008] or taking some motion priors into account proposed by Tordoff and Murray [2005]. However, the true value of the probabilistic priors available has not yet fully been appreciated and exploited in these methods which rely heavily on randomness and arbitrary thresholds. This has been improved by probabilistic methods such as the Joint Compatibility Branch and Bound (JCBB) algorithm proposed by Neira and Tardós [2001] which matches features via a deterministic interpretation tree [Grimson, 1990] and has been applied to geometric image matching in [Clemente et al., 2007]. JCBB which is demonstrated with an example in Figure 5.2, takes account of a joint Gaussian prior on feature positions and calculates the joint probability that any particular hypothesised set of correspondences is correct.

Our algorithm aims to perform at least as well as JCBB in determining global consensus while searching much smaller regions of an image. It goes much further than previously published 'guided matching' algorithms such as the Guided-MLESAC of Tordoff and Murray [2005] in guiding not just a search for consensus but the image processing to determine candidate matches themselves.

Davison [2005] presented a theoretical analysis of information gain in sequential image search. However, this work had the serious limitation of representing the current estimate of the state of the search at all times with a single multi-variate Gaussian distribution. This meant that while theoretically and intuitively satisfying active search procedures were demonstrated in simulated problems, the technique was not applicable to real image search because of the lack of ability to deal with discrete multiple hypotheses which arise due to matching ambiguity — only simulation results were given. Here we use a dynamic mixture of Gaussians (MoG) representation which grows as necessary to represent the discrete multiple hypotheses arising during active search. We show that this representation can now be applied to achieve highly efficient image search in real, ambiguous tracking problems.

This chapter presents in full detail the Active Matching algorithm which was first introduced in [Chli and Davison, 2008a,b] and analysed further in terms of performance in [Chli and Davison, 2009b]. We start with an in-depth explanation of the motivation for the mixture representation via a histogram-based analysis of the underlying probability distributions. Applying an Information Theoretic methodology on the probabilistic estimates maintained throughout matching, we demonstrate the in-

fluence on the decisions the matcher makes and we detail the way that the mixture is maintained depending on the outcome of each individual feature-search. Lastly, we discuss the results of a comprehensive set of experiments pushing the capabilities of the algorithm to the limits, with the aim of assessing its strengths and weaknesses. Our study on the evolution of the informational value of measurements throughout the matching process indicates the route towards effective approximations which can further increase the efficiency of Active Matching.

5.2 Active Search and Beyond

In our general matching formulation, we consider making image measurements of an object or scene of which the current state of knowledge is modelled by a probability distribution over a finite vector of parameters $\mathbf{x}$. These parameters may represent the position of a moving object or camera as is the case in MonoSLAM for instance. The probability distribution $p(\mathbf{x})$ which describes our uncertain knowledge of the parameters at the moment an image arrives will be determined by general prior knowledge and what has happened previously to the system. For instance, in the common case of sequential tracking of motion through an image sequence, $p(\mathbf{x})$ at each time step will be the result of projecting the distribution determined at the previous frame forward through a motion model.

In an image, we are able to observe *features*: measurable projections of the state. A measurement of feature i yields the vector of parameters $\mathbf{z}_i^*$. In MonoSLAM for example, $\mathbf{z}_i^*$ holds the 2D image coordinates of a keypoint of known appearance, the position of an edge or a higher-dimensional parameterisation of a more complex image entity. In each case, a likelihood function $p(\mathbf{z}_i|\mathbf{x})$ models the measurement process, yielding the predicted parameters $\mathbf{z}_i$.

Projecting the current probability distribution over state parameters $\mathbf{x}$ into feature space, we can predict the image locations of all the features which are predicted to be visible from the current viewpoint, as explained in Section 3.2.2. Our goal is to use the joint distribution over all such measurement candidates $p(\mathbf{z}_T)$, to guide intelligent active search and matching. The first possibility one might consider is to marginalise elements $p(\mathbf{z}_i)$ to give individual predictions of the image location of each feature under consideration, thus allowing active search for features within their corresponding high-probability regions. This procedure is relatively common in visual tracking, where strong motion models mean that these search regions are often small and efficiently searched. Several Kalman Filter-based trackers such as MonoSLAM implement the same scheme by using gates at a certain number of standard deviations to restrict the search. In the Condensation algorithm of Isard and Blake [1996] feature

searches take place in fixed-size windows around predetermined measurement sites centred at a projection into measurement space of each of the particles representing the state probability distribution.

The fact that has usually been neglected in feature search, however, is that the predictions of the values of different measurements $\mathbf{z}_i$ are very often correlated since they all depend on common parts of the scene state $\mathbf{x}$. As discussed in Chapter 4, these correlations are the key to efficient coupled active search, we thus exploit them in Active Matching to guide a step by step approach to search rather than blanket examination of all feature regions.

5.2.1 Single Gaussian Model

To attack the coupled search problem, Davison [2005] made the simplifying assumption that the PDFs describing the knowledge of the camera state $\mathbf{x}_c$ and the features' image coordinates $\mathbf{z}_T$ can be approximated always by single multi-variate Gaussian distributions, as defined by measurement state vector $\mathbf{x}_m$ and associated covariance matrix $\mathtt{P}_m$ in Equations 3.11 and 3.12 respectively. Using this single Gaussian formulation and as explained in detail in Section 4.3.1, Davison showed via simulations how Information Theory can guide active search reducing the search-space by pinning down an object given some candidate measurements with associated uncertainty.

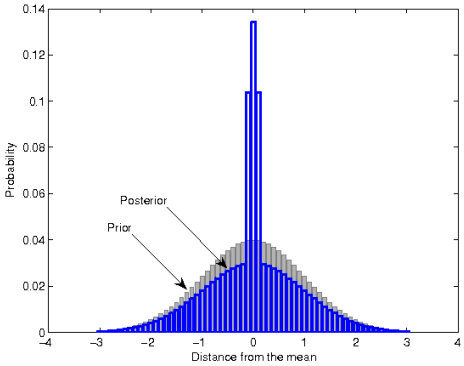
The simulation examples presented in [Davison, 2005] (also shown in Figures 4.2 and 4.3) are based on the assumption that the matching is perfect: every search for a candidate yields a single match occurring at the *true* feature position. This is a very optimistic assumption to make when dealing with real images where ambiguity and more generally, perceptual aliasing is inevitable. Davison's technique is therefore inapplicable outside the benign conditions of a simulation environment. However, we believe that Information Theory is the key to maintaining the optimal balance between processing costs and information gains, therefore this work has been very much inspired by Davison's approach.

5.2.2 Full Histograms and Multiple Hypotheses

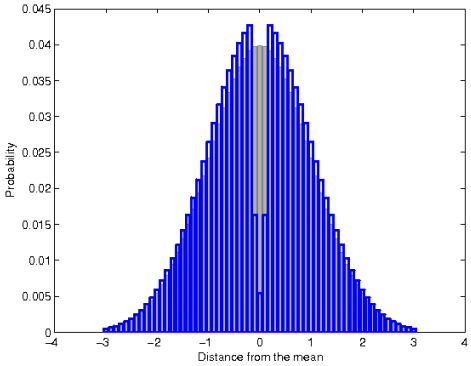
The weakness of the single Gaussian approach to matching is that, as ever, a Gaussian is uni-modal and can only represent a PDF with one peak. In real image search problems, no match (also referred to here as a failed match) can be fully trusted: true matches are sometimes missed (false negatives), and clutter similar in appearance to the feature of interest can lead to false positives.

To investigate the theoretical performance of active search in such ambiguous cases, a simulation of 1D Bayesian active search for a single feature has been developed which uses a simple but exhaustive histogram representation of probability.

Test the central pixel of the array for a match:

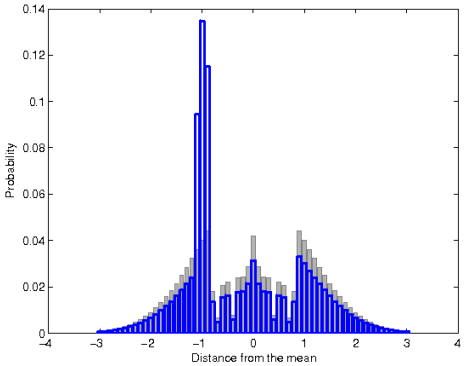


(a) Successful match at central pixel

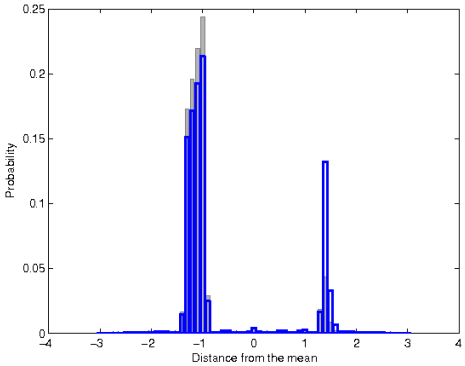


(b) No match at central pixel

The histogram distribution at later stages:



(c) 18 positions measured



(d) All positions measured

Figure 5.3: Pixel-by-pixel search using a histogram representation. Suppose that a feature is predicted to lie within an 1D array of pixels and that a full, normalised histogram represents the probability distribution that this feature truly lies at each such image location. The distribution is refined sequentially starting off with a Gaussian prior and updated accordingly as each pixel location is tested for a template match. Figures (a) and (b) show the outcome of either a successful or failed match at the pixel in the centre of the prior which is checked first: a success causes a spike in the distribution while a failure results to a trough. In (c), measurements at a number of central sites have led to an intermediate distribution, and (d) shows the final posterior distribution in a situation where all positions have been checked to reveal two significant candidate locations for the true feature.

As demonstrated with an example in Figure 5.3, the goal is to locate a feature in a one-dimensional search region by making pixel-by-pixel attempts at template matching. Each pixel is represented by a discrete histogram bin storing the current probability that the true feature is in that location. The true feature must lie in exactly one true position, so at all times the discrete histogram is normalised to total probability one. At the start of search, we initialise a Gaussian prior across the region.

Active search proceeds by a selecting pixel location i as a candidate, measuring it and updating the whole histogram via Bayes rule accordingly. The update uses the following likelihood expression:

$$P(M_i|B_k) = C_{FP} + C_{TP} e^{-\frac{1}{2}\frac{(i-k)^2}{\sigma^2}}. \tag{5.1}$$

Hence $P(F_i|B_k) = 1 - P(M_i|B_k)$ holds, for the probabilities of making a template match M_i or a failed match F_i at position i given B_k, that the feature is truly at position k. Here C_{FP} is a constant representing the per-pixel false-positive probability of finding a template match to clutter, and C_{TP} is a constant proportional to the true-positive probability of matching to the feature in its true position. This likelihood function says that if the feature is at k then there is a raised, Gaussian-profile probability of making a match at nearby locations, the parameter σ specifying the standard deviation of the feature's 'measurement uncertainty' (here set to 1 pixel).

The final distribution after all positions have been measured in Figure 5.3(d) is the motivation for the mixture of Gaussians formulation used in the rest of the paper. The single Gaussian method of Section 5.2.1 cannot represent the clear multiple hypotheses present here. This histogram representation really gets to the truth of active search, but is impractical in reality because of the computational cost of maintaining a histogram — rising exponentially with the number of dimensions of the total measurement vector. Practical real-time searches happen not by one-by-one pixel checks followed by probabilistic updates, but by examining a whole region at once and obtaining zero, one or more candidate matches. Figure 5.3(d) suggests that a mixture of Gaussians represents the posterior in this case well.

5.3 Active Matching Algorithm

Ideally, any features selected for measurement would be absolutely unique and always recognisable, meaning that they produce a match only when present and at the true feature location. Since this is not the case in real image search problems, we can never fully trust the matching outcome of a feature search. Modelling the probabilistic 'search state' as a mixture of Gaussians, we wish to retain the feature-by-feature quality of active search. Our new MoG representation allows dynamic, online updating of

the multi-peaked PDF over feature locations which represents the multiple hypotheses arising as features are matched ambiguously.

Our Active Matching algorithm searches for global correspondence in a series of steps which gradually refine the probabilistic search state initially set as the prior on feature positions. Each step consists of a search for a template match to one feature within a certain bounded image region, followed by an update of the search state which depends on the search outcome. After many well-chosen steps, the search state collapses to a highly peaked posterior estimate of image feature locations — and matching is finished. Figure 5.4 illustrates the a step-by-step example of Active Matching (AM), operating on a typical MonoSLAM frame where some ambiguity is encountered but consensus is successfully resolved following a series of selective measurements.

5.3.1 Search State Mixture of Gaussians Model

A single multi-variate Gaussian probability distribution over the vector $\mathbf{x}_m$ which stacks the object state and candidate measurements, is parameterised by a 'mean vector' $\hat{\mathbf{x}}_m$ and its full covariance matrix $\mathtt{P}_m$. We use the shorthand $\mathbf{G}(\hat{\mathbf{x}}_m, \mathtt{P}_m)$ to represent the explicit normalised PDF:

$$p(\mathbf{x}_m) = \mathbf{G}(\hat{\mathbf{x}}_m, \mathtt{P}_m) \tag{5.2}$$

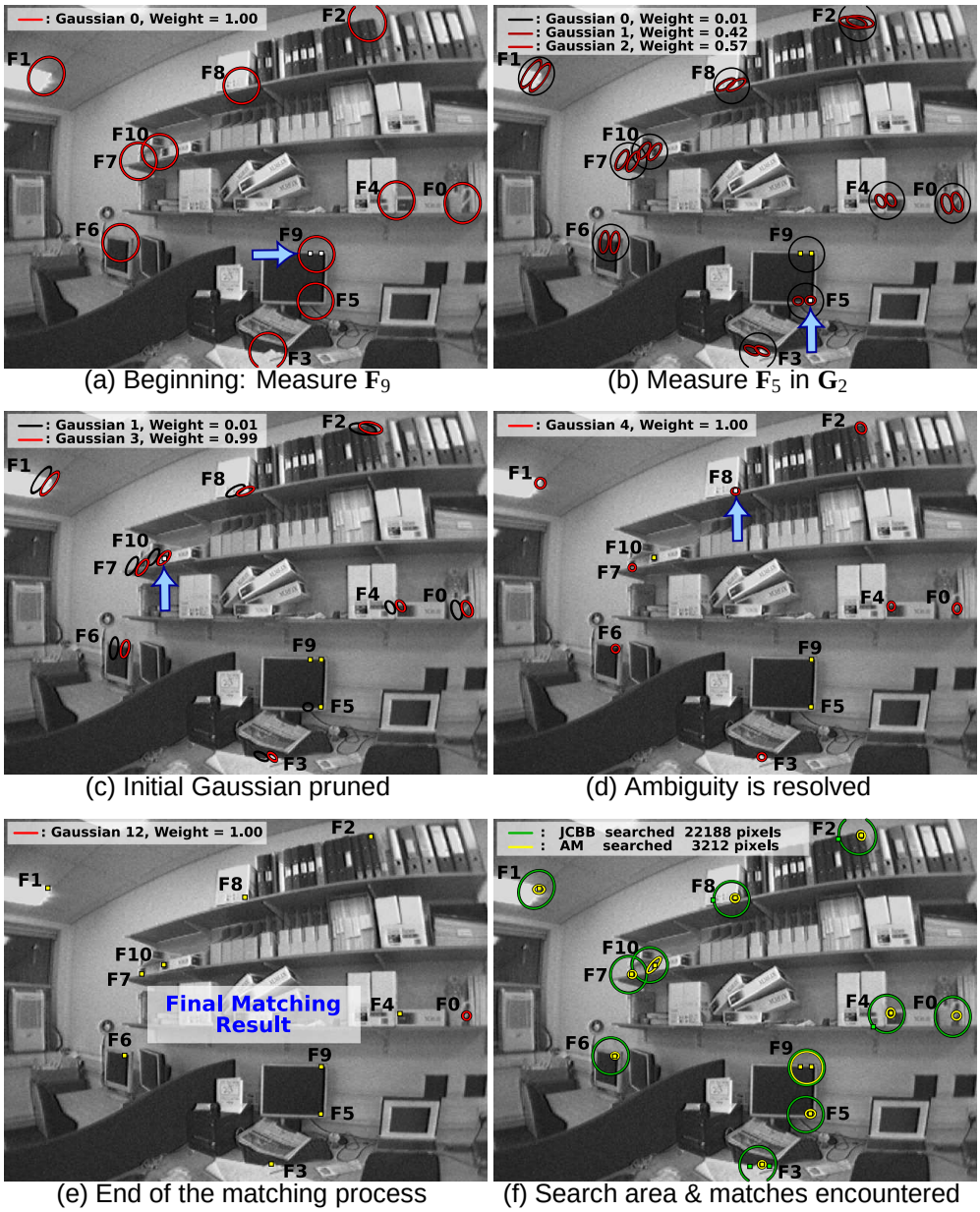
$$= \frac{1}{\sqrt{(2\pi)^D |\mathtt{P}_m|}} \; exp\{-\frac{1}{2}(\mathbf{x}_m - \hat{\mathbf{x}}_m)^\top \mathtt{P}_m^{-1} (\mathbf{x}_m - \hat{\mathbf{x}}_m)\} \; , \tag{5.3}$$

where D denotes the cardinality of vector $\mathbf{x}_m$. However, during Active Matching we represent the PDF over the estimates in $\mathbf{x}_m$ with a multi-variate MoG distribution formed by the sum of K individual Gaussians each with weight λ_i:

$$p(\mathbf{x}_m) = \sum_{i=1}^{K} p(\mathbf{x}_{m_i}) = \sum_{i=1}^{K} \lambda_i \mathbf{G}_i \; , \tag{5.4}$$

where we have now used the further notational shorthand $\mathbf{G}_i = \mathbf{G}(\hat{\mathbf{x}}_{m_i}, \mathtt{P}_{m_i})$. Each Gaussian distribution must have the same dimensionality and the weights λ_i must normalise to add up to 1 for this to be a valid PDF.

The current MoG search state forms the prior of the next step of Active Matching. This prior together with the likelihood and posterior distributions as shown in symbolic 1D form in Figure 5.5, are explained in the following sections. However, before looking into the details of the theoretical background, below we give an overview of the algorithm describing the processes involved from a high-level perspective.



(a) Beginning: Measure $\mathbf{F}_9$ (b) Measure $\mathbf{F}_5$ in $\mathbf{G}_2$

(c) Initial Gaussian pruned (d) Ambiguity is resolved

(e) End of the matching process (f) Search area & matches encountered

Figure 5.4: Resolving ambiguity using AM. Based on the input prior describing the joint probability distribution over the features' locations, the MI values are computed for each ellipse to describe the information each measurement is expected to provide to the rest of the features. As $\mathbf{F}_9$ achieves the highest MI score per pixel to search, it gets measured yielding two matches as shown in (a). Propagating this outcome, $\mathbf{G}_1$ and $\mathbf{G}_2$ are spawned in (b) and MI values are recomputed. The match found for $\mathbf{F}_5$ in $\mathbf{G}_2$ boosts the newly spawned $\mathbf{G}_3$, weakening $\mathbf{G}_0$ and $\mathbf{G}_2$ enough to get pruned off the mixture in (c). The match for $\mathbf{F}_{10}$ comes to resolve the ambiguity in (d) with $\mathbf{G}_4$ having dramatically reduced width with respect to the initial prior $\mathbf{G}_0$. Measuring the rest of the features in the same sequential manner, AM concludes in (e). Figure (f) superimposes of the elliptical areas searched to achieve data association with AM and traditional matching techniques. In this example, AM searches $7\times$ less image area than standard 'get matches first, resolve later' approaches like JCBB and RANSAC.

5.3.2 The Algorithm

The Active Matching process is initialised with a joint Gaussian prior over the features' locations in measurement space (e.g. prediction after application of a motion model). Hence, at start-up the mixture consists of this single, multivariate Gaussian. Every measurement candidate is evaluated based on the Mutual Information(MI) it is predicted to provide to the rest of the candidates. The candidate $\{$Feature, Gaussian$\}$ pair to achieve the highest MI-efficiency score is chosen for measurement. Essentially, a $\{$Feature, Gaussian$\}$ pair corresponds to an ellipse in image space, as shown in Figure 5.5. Section 5.4 is dedicated to explain how measurement selection is performed.

For every template match yielding from the search of the selected measurement pair, a new Gaussian is spawned with mean and covariance conditioned on the hypothesis of that match being a true positive — this will be more peaked than its parent. In both cases of either a successful or null template search, the weights of the existing Gaussians are redistributed to reflect the current MoG search state. The full description of the update step after a measurement is detailed in the rest of this section.

Finally, very weak Gaussians (with weight < 0.001) are pruned from the mixture after each search step. This avoids the otherwise rapid growth in the number of Gaussians such that in practical cases, fewer than 10 Gaussians are 'live' at any point, and most of the time much fewer than this. This pruning is the better, fully probabilistic equivalent in the dynamic MoG scheme of lopping off branches in an explicit interpretation tree search such as JCBB [Neira and Tardós, 2001].

Below, are the pseudo-code descriptions of the Active Matching algorithm and the mixture-updating procedure. While some of the notation is explained later in the section, these are really aimed at providing the reader with a general understanding of the processes involved.

ACTIVEMATCHING($\mathbf{G}_0$)

```
1  Mixture = [[1, G_0]]  // Each entry in the Mixture is a [weight, Gaussian] tuple
2  {F_c,G_c} = get_max_mi_efficiency_candidate(Mixture)
3  while (pair_not_yet_measured({F_c, G_c}))
4      Matches = measure({F_c, G_c})
5      UPDATEMIXTURE(Mixture, c, Matches)
6      prune_insignificant_gaussians(Mixture)
7      {F_c,G_c} = get_max_mi_efficiency_candidate(Mixture)
8  end while
9  G_best = find_most_probable_gaussian(Mixture)
10 return G_best
```

UPDATEMIXTURE(Mixture, i, Matches)

Propagate the result of a measurement of a feature **F** *in* $\mathbf{G}_i$*, following the update rule of Equation 5.10*

```
1  for k = 1 : K  // loop through all Gaussians
2      [λ_k, G_k] = Mixture[k]
3      if k = i then  // this is the measured Gaussian
4          for m = 1 : M  // for every match, spawn a new Gaussian
5              G_m = spawn_gaussian_and_fuse_match(G_k, Matches[m])
6              λ_m = λ_k × μ_match × prior(Matches[m], G_k)
7              Mixture = [Mixture, [λ_m, G_m]]
           end for
8          λ_k = λ_k × μ_in × (1 − prior_sum(Matches, G_k))
       else
   // Probability of G_k for the measured feature, summed over the region covered by G_i:
9          prob = prior_sum_under G_i(G_k)
10         sum = prior_sum(Matches, G_k)
11         λ_k = λ_k × [μ_match × sum + μ_in × (prob − sum) + μ_out × (1 − prob)]
       end if
12     Mixture[k] = [λ_k, G_k]  // reset entry to the updated state and weight of G_k
13 end for
14 normalise_weights(Mixture)
15 return
```

Note: prior(Matches[m], $\mathbf{G}_k$) *returns the prior probability of match* m *being a true match, in* $\mathbf{G}_k$ *(highest value at the centre of this Gaussian). Similarly,* prior_sum(Matches, $\mathbf{G}_k$) *returns the sum of all such prior probabilities for all elements of* Matches.

5.3.3 Likelihood Function

One step of Active Matching takes place by searching the region defined by the high-probability 3σ extent of one of the Gaussians in the measurement space of the selected feature. Suppose that $\mathbf{Z}^* = (\mathbf{z}_1^* \ldots \mathbf{z}_m^* \ldots \mathbf{z}_M^*)^\top$ is the outcome of this search for matches, meaning that template matching has been successful at M different pixel locations, but failed everywhere else in the region. The likelihood distribution of this result with respect to the current state $\mathbf{x}$ is defined as:

$$p(\mathbf{Z}^*|\mathbf{x}) = \mu_{\text{in}}\mathbf{T}_{\text{in}} + \mu_{\text{out}}\mathbf{T}_{\text{out}} + \sum_{m=1}^{M} \mu_{\text{match}}\mathbf{H}(\mathbf{z}_m^*) \;, \tag{5.5}$$

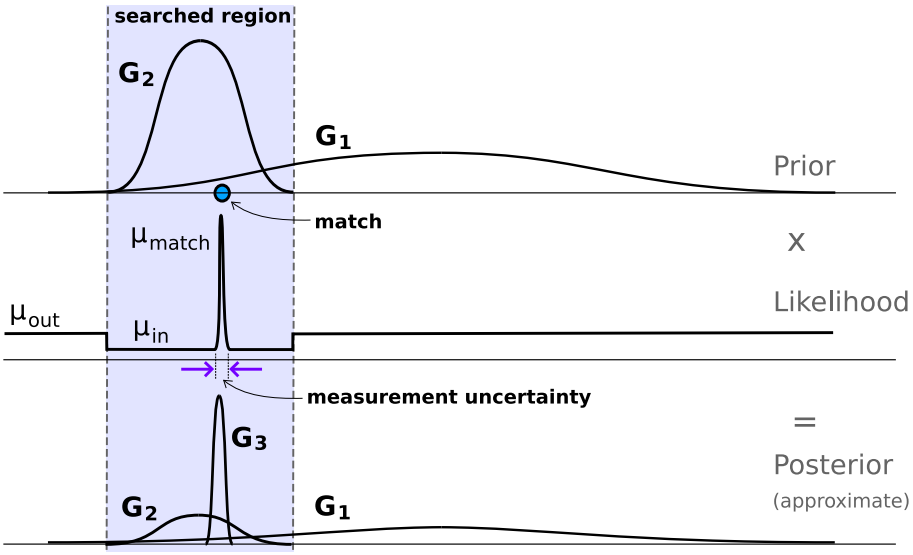


Figure 5.5: Updating the Mixture of Gaussians. In this example, the search in region $\mathbf{G}_2$ yields a match. Given this result, the likelihood function is formed as in Equation 5.5 and when multiplied with the prior mixture, leads to the estimated posterior MoG. The latter consists of a scaled version of the old mixture and a new $\mathbf{G}_3$ to represent the hypothesis that the match is a true positive. The updated distribution of weights depends on the statistical properties of the feature matched, the position of the match and the amount of overlap of the searched region with respect to each Gaussian. The closer the position of the match to the centre of the searched $\mathbf{G}_2$, the strongest the weight of the spawned $\mathbf{G}_3$. 'Third-party' Gaussians like $\mathbf{G}_1$, get scaled according the total belief they had that the match would lie in the searched region in the first place; if the overlap between $\mathbf{G}_1$ and $\mathbf{G}_2$ is small and the match occurs away from the centre of $\mathbf{G}_1$ then this Gaussian will become really weak in the posterior mixture.

where $\mu_{\text{in}}, \mu_{\text{out}}$ and μ_{match} are constants (defined later in Equations 5.6-5.8) capturing the matching characteristics of each feature. In essence, the likelihood function is modelled as a mixture of:

- M hypotheses $\mathbf{H}(\mathbf{z}_m^*)$, each to account for one candidate $\mathbf{z}_m^* \in \mathbf{Z}^*$ being the true **match** (considering all others as false positives) — these hypotheses are Gaussians having very small width, corresponding to the measurement uncertainty $\mathtt{R}_i$ as shown in the example of Figure 5.5, and

- two constant terms: $\mathbf{T}_{\text{in}}$ accounts for the hypothesis that the true match lies **in** the searched region but has not been recognised, while $\mathbf{T}_{\text{out}}$ supports the possibility of the true feature actually lying **out** of the region searched. In fact, these are both top-hat functions aimed at enforcing account for the spurious false positives in the measurement process: $\mathbf{T}_{\text{in}}$ and $\mathbf{T}_{\text{out}}$ have a value of one inside and outside of the searched Gaussian respectively and zero elsewhere, since the probability of a null search depends on whether the feature is really within the search region or not.

The μ-terms in the likelihood function expression (Equation 5.5)are introduced so that the individual feature characteristics are considered during the estimation process. Surely, the *distinctiveness* and the *measurability* of features varies depending on a wide variety of factors (e.g. the type of feature detector/descriptor, the repetitive structure in the scene, the lighting conditions). A match coming from a search for a unique feature should be trusted more than a match yielding from a search for a very common one. Conversely, a failed match for a feature that has been identifiable most of the times comprises stronger evidence than a failed search for a feature that has not been detected consistently throughout the sequence. Therefore, assessing the true-positive $P_{\mathbf{tp}}$, false-positive $P_{\mathbf{fp}}$, true-negative $P_{\mathbf{tn}}$ and false-negative $P_{\mathbf{fn}}$ probabilities of different features via 'statistical training' during tracking, the Active Matching methodology can inherently take them into account to enforce the robustness of the outcome based on the reliability of features.

Going back to the formation of the likelihood function upon the measurement of a $\{$Feature, Gaussian$\}$ pair, if N is the total number of pixels searched for this measurement, then the μ-terms of expression 5.5 can be computed as follows:

$$\mu_{\text{in}} = P_{\mathbf{fp}}^{M} P_{\mathbf{fn}} P_{\mathbf{tn}}^{N-(M+1)} \tag{5.6}$$

$$\mu_{\text{out}} = P_{\mathbf{fp}}^{M} P_{\mathbf{tn}}^{N-M} \tag{5.7}$$

$$\mu_{\text{match}} = P_{\mathbf{tp}} P_{\mathbf{fp}}^{M-1} P_{\mathbf{tn}}^{N-M} \, . \tag{5.8}$$

Given that there can only be one true match in the searched region, here the idea is to take account of all different possibilities:

- the true match fact lies **in** the searched region but does not correspond to any of the M matches, so μ_{in} is the probability of obtaining M false positives, a false negative and $N-(M+1)$ true negatives.
- the true match lies **out** of the searched region, so μ_{out} is the probability of M false positives and $N-M$ true negatives, and finally,
- one of the obtained matches is actually the true feature, so μ_{match} is the probability of a true positive occurring along with $M-1$ false positives and $N-M$ true negatives.

5.3.4 Posterior: Updating After a Measurement

The standard application of Bayes' Rule to obtain the posterior distribution for $\mathbf{x}$ given the new measurement, is:

$$p(\mathbf{x}|\mathbf{Z}^{*}) = \frac{p(\mathbf{Z}^{*}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{Z}^{*})} \, . \tag{5.9}$$

Substituting the mixture models from Equations 5.4 and 5.5, we get the posterior estimate:

$$p(\mathbf{x}|\mathbf{Z}^*) = \frac{\left(\mu_{\text{in}}\mathbf{T}_{\text{in}} + \mu_{\text{out}}\mathbf{T}_{\text{out}} + \sum_{m=1}^{M} \mu_{\text{match}}\mathbf{H}(\mathbf{z}_m^*)\right)\left(\sum_{i=1}^{K} \lambda_i \mathbf{G}_i\right)}{p(\mathbf{Z}^*)} . \tag{5.10}$$

The denominator $p(\mathbf{Z}^*)$ is a constant determined by normalising all new weights λ_i to add up to one. Figure 5.5 illustrates the formation of a posterior when the search outcome consists of a singe match ($M = 1$). This posterior will then become the prior for the next Active Matching step.

In the top line of Equation 5.10, the product of the two mixture sums will lead to K scaled versions of all the original Gaussians and MK terms which are the products of Gaussians with hypotheses, in essence yielding MK new Gaussians. However, we make the approximation that only M of these MK product terms are significant: those involving the prior Gaussian currently being measured. We assume that since the other Gaussians in the prior distribution are either widely separated or have very different weights, the resulting products will be negligible. Therefore there are only M new terms added to the mixture which are generally highly-weighted, spiked Gaussians corresponding to matches found in the searched region. These are considered to be '*children*' of the searched parent Gaussian. An important point to note, is that if multiple matches in a search region lead to several new child Gaussians being added, one corresponding to a match close to the centre of the search region will correctly have a higher weight than others, having been formed by the product of a prior and a measurement Gaussian with nearby means.

All other existing Gaussians get updated posterior weights by multiplication with the constant terms. Note that the information of making a null search where no template match is found, is fully accounted for in our framework — in this case we will have $M = 0$ and no new Gaussians will be generated, but the weight of the searched Gaussian will diminish.

Pruning Weak Gaussians From the Mixture

The nature of the MoG update implies that every time a match is encountered, a new Gaussian is spawned to represent the scenario that it is a true match, while the searched Gaussian is maintained to account for the case that the match is false — most probably assigned to a much lower weight. As a consequence, by the end of the matching process, the mixture is populated by as many matches as encountered plus the initial one. The vast majority of these Gaussians however, are usually ruled insignificant, quickly after they get spawned since they pass on most of their weight to their descendant,

newer Gaussians as demonstrated in the example of Figure 5.6.

This means that in essence, we carry an unnecessarily large mixture throughout the matching process, constantly updating them and evaluating the potential effect of measuring them for each feature (explained in detail in the following section), but without them having any impact on the decisions made throughout matching or the achieved accuracy. Thus, by pruning hypotheses whose weight falls below a certain threshold, the algorithm becomes a lot faster at no expense. Indeed, cutting off a Gaussian means that we can never go back to correct the matching scenario down that particular route, so it is vital not to cull a potentially true hypothesis — this accentuates the importance of a realistic weighting scheme within the mixture.

5.4 Measurement Selection

We assume that the input prior at the start of the search process is well-represented by a single Gaussian and therefore $\lambda_1 = 1$. As active search progresses and there is a need to propagate multiple hypotheses, this and subsequent Gaussians will divide as necessary, so that at a general instant there will be K Gaussians with normalised weights.

5.4.1 Search Candidates

At each step of the MoG Active Matching process, we use the mixture to predict the expected outcome of individual feature measurements, and thus decide on which action to take. In this sense our algorithm has been dubbed *active* matching suggesting a fully dynamic and automatic performance. At every instant, there are KF possible actions, where F is the number of measurable features. We rule out any $\{$Feature, Gaussian$\}$ combinations where we have already made a search. Also ruled out are 'child' Gaussians for a certain feature which lie completely within an already searched ellipse. Looking at Table 5.1 for example, if we have measured root Gaussian $\mathbf{G}_1$ at feature 1, leading to the spawn of $\mathbf{G}_2$ which after searching for feature 3 generates $\mathbf{G}_3$, then the candidates marked with '$*$' would be ruled out from the selection:

	$\mathbf{F}_1$	$\mathbf{F}_2$	$\mathbf{F}_3$
$\mathbf{G}_1$	$*$		

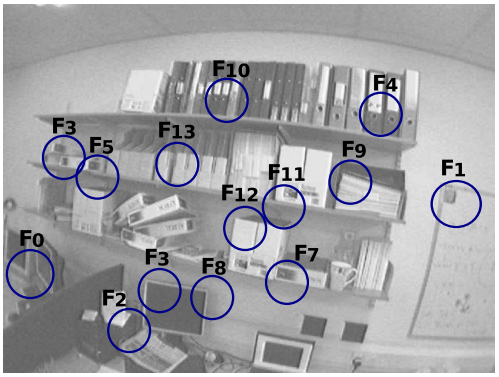
$\Rightarrow$

	$\mathbf{F}_1$	$\mathbf{F}_2$	$\mathbf{F}_3$
$\mathbf{G}_1$	$*$		
$\mathbf{G}_2$	$*$		$*$

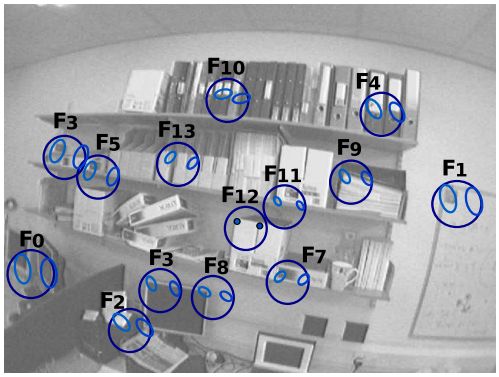
$\Rightarrow$

	$\mathbf{F}_1$	$\mathbf{F}_2$	$\mathbf{F}_3$
$\mathbf{G}_1$	$*$		
$\mathbf{G}_2$	$*$		$*$
$\mathbf{G}_3$	$*$		$*$

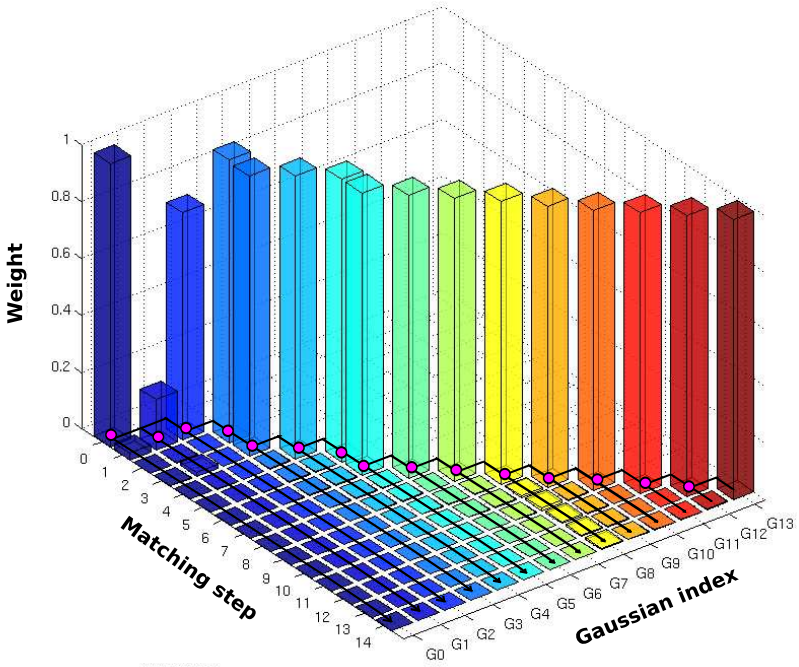
Table 5.1: Excluding measurement-pairs lying completely within searched Gaussians from the candidates selection



(a) Step 0: Beginning of search

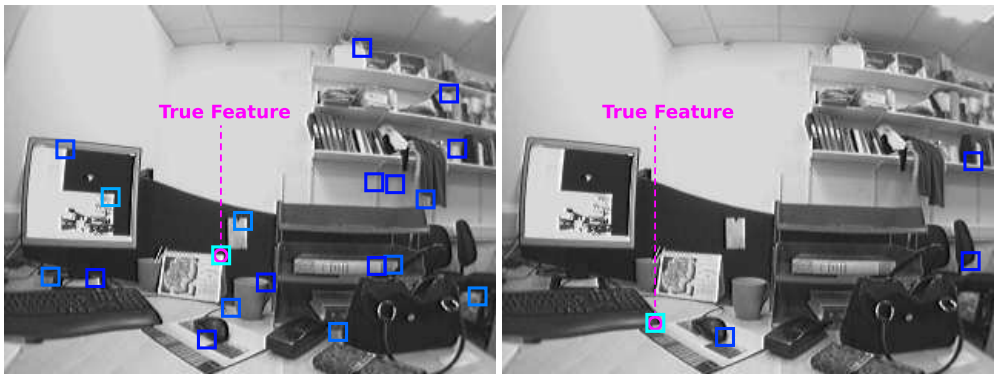


(b) Step 1: Measuring $\mathbf{F}_{12}$ yields 2 matches



(c) Distribution of weights in the MoG per matching step

Figure 5.6: The evolution of Gaussians and the distribution of their weights per search-step in a matching example. At the beginning of search in (a) there is only one Gaussian $\mathbf{G}_0$ present, which is the input joint distribution over the locations of the features. The search for $\mathbf{F}_{12}$ in $\mathbf{G}_0$ yields two matches, hence in (b) two new Gaussians are spawned ($\mathbf{G}_1$ and $\mathbf{G}_2$) each to represent that one of the matches is true. The distribution of weights in the mixture at every matching step, is shown in (c). The search in a Gaussian (denoted with pink blobs) at every step has two possible effects on the mixture: (i) one or more Gaussians get spawned depending on the number of matches found (e.g. search in $\mathbf{G}_0$ spawns $\mathbf{G}_1$ and $\mathbf{G}_2$) and weights get redistributed, or (ii) only the weights of the existing Gaussians are affected following an unsuccessful measurement attempt (e.g. at step 5 where $\mathbf{F}_7$ fails to match in $\mathbf{G}_5$). It is evident from this histogram that by the end of matching, the MoG gets cluttered with many, insignificantly weighted hypotheses which take up precious processing time without having any effect in the matching process.



(a) Common feature patch (b) Distinctive feature patch

Figure 5.7: Analysing feature false-positive rates within MonoSLAM, which detects features using the criterion proposed by Shi and Tomasi [1994]. The feature-patch queried in (a) describes a common structure in this scene with around 20 matches occurring per image, whereas the feature-patch in (b) is much more distinctive. In both examples, the brightness of the boxes indicates the strength of similarity. It is preferred to rely on distinctive features to resolve matching consensus for the obvious reason that the measurement result of such patches can be trusted more than others capturing more repeated scene structure. Therefore, if both features queried in (a) and (b) are candidates for measurement in a given frame, choosing to measure the feature of (b) first boosts the chances of the spawned Gaussian being a good start towards successful data association. The ability of Active Matching to incorporate statistical models for feature characteristics (rate of false positives, true positives, etc.) can potentially drive robust matching in highly challenging scenarios.

All of the remaining candidates are evaluated in terms of the Mutual Information they are predicted to provide to the estimates of the rest of the candidates in the mixture. The selection of which feature to measure next is based on the Information Efficiency scores defined as the Mutual Information value divided by the are of the region to be searched. The latter metric has been proposed by [Davison, 2005] and has been discussed in more detail in Chapter 4.

As demonstrated in Figure 5.7, some features can be matched more reliably than others. Our algorithm should automatically be able to benefit from the same properties, and probabilistically favour measurement of statistically trusted candidates with their ability to reduce ambiguity in hypotheses. We have implemented a straightforward feature statistics capability within MonoSLAM to sequentially record the average number of locations in an image similar to each of the mapped features, counting successful and failed match attempts in the feature's true location. This is used to assess false positive and false negative rates for each feature within the current type of scene (e.g. office, garden). The results shown in this chapter aim to demonstrate the effect of efficient and robust matching without prior knowledge of the environment that SLAM is performed, so the false-positive and false-negative rates which have been used are uniform for every feature considered.

It is worth noting that in all the experiments presented, the strength of the measurement noise ($\mathtt{R}_i$) is assumed to be constant across all features. While indeed any features with strong texture are likely to be matched with greater subpixel precision, this approximation is not expected to affect significantly the the final outcome or the order of measurement. Alternatively, one could determine an appropriate value for each $\mathtt{R}_i$ using the learned feature statistics. If however multi-scale features are used, then greater care should be taken in evaluating $\mathtt{R}_i$ to incorporate knowledge about the scale that each feature has been detected as $\mathtt{R}_i$ should no longer be assumed to be constant. In general, when using a realistic value for $\mathtt{R}_i$ in the measurement model then Active Matching is expected to provide an accurate matching outcome.

5.4.2 Mutual Information for a Mixture of Gaussians Distribution

In order to assess the amount of information that each candidate $\{$Feature, Gaussian$\}$ measurement pair can provide, we predict the post-search mixture of Gaussians depending on the possible outcome of the measurement:

1. A **null search**, where no template match is found above a threshold. The effect is only to change the weights of the current Gaussians in the mixture into λ_i'.

2. A **template match**, causing a new Gaussian to be spawned with reduced width as well as re-distributing the weights of the all Gaussians of the new mixture to λ_i''.

In a well-justified assumption of 'weakly-interacting Gaussians' which are either well-separated or have dramatically different weights, we separate the information impact of each candidate measurement into two components: (a) I_{discrete} captures the effect of the redistribution of weights depending on the search outcome and (b) $I_{\text{continuous}}$ gives a measure of the reduction in the uncertainty in the system on a match-search. Due to the intuitive, absolute nature of mutual information, these terms are additive:

$$I = I_{\text{discrete}} + I_{\text{continuous}} \tag{5.11}$$

One of either of these terms will dominate at different stages of the matching process, depending on whether the key uncertainty is due to discrete ambiguity or continuous accuracy. It is highly appealing that this behaviour arises automatically thanks to the MI formulation.

Mutual Information: Discrete Component

Following the introduction to the notion of Mutual Information in Chapter 4, we consider the effect of a candidate measurement purely in terms of the change in the weights

of the Gaussians in the mixture. Restating Equation 4.8 with the relevant symbols, the mutual information that a candidate (predicted) measurement $\mathbf{z}_j$ is predicted to provide is:

$$I(\mathbf{z}_j;\mathbf{z}_T) = H(\mathbf{z}_T) - H(\mathbf{z}_T|\mathbf{z}_j). \tag{5.12}$$

Given that the search outcome can have two possible states (null or match-search), then:

$$\begin{aligned} I_{\text{discrete}} = H(\mathbf{z}_T) \quad &- \quad P(\mathbf{z}_j = \text{null}) \quad \times H(\mathbf{z}_T|\mathbf{z}_j = \text{null}) && (5.13)\\ &- \quad P(\mathbf{z}_j = \text{match}) \times H(\mathbf{z}_T|\mathbf{z}_j = \text{match})\ , && (5.14) \end{aligned}$$

where

$$\begin{aligned} H(\mathbf{z}_T) &= \sum_{i=1}^{K} \lambda_i \log_2 \frac{1}{\lambda_i} && (5.15)\\ H(\mathbf{z}_T|\mathbf{z}_j = \text{null}) &= \sum_{i=1}^{K} \lambda_i' \log_2 \frac{1}{\lambda_i'} && (5.16)\\ H(\mathbf{z}_T|\mathbf{z}_j = \text{match}) &= \sum_{i=1}^{K+1} \lambda_i'' \log_2 \frac{1}{\lambda_i''}\ . && (5.17) \end{aligned}$$

While the weights of Gaussians currently in the mixture are denoted by λ_i, the notation λ_i' and λ_i'' stands for the predicted weights after a failed and a successful search, respectively. These predicted weights are calculated using the mixture-update Equation 5.10 with the only difference that the likelihood of a successful search is summed over all positions in the search-region that can possibly yield a match.

Mutual Information: Continuous Component

In Section 4.3 we have discussed the Mutual Information (MI) between one candidate and the rest visible in the scene as the essential probabilistic measure of measurement value. Following the single Gaussian formulation, we derived an efficient expression for the mutual information in bits between any two partitions of the state vector in Equation 4.25. Hence, the continuous component of the mutual information of a particular candidate pair $\{\mathbf{z}_j, \mathbf{G}_i\}$, is calculated using Equation 4.31:

$$I_{\text{continuous}} = \lambda_m''\, I(\mathbf{z}_j;\mathbf{z}_T) = \frac{1}{2}\lambda_m'' \log \frac{|\mathtt{S}_{jj}||\mathtt{S}_{T\neq j\ T\neq j}|}{|\mathtt{S}|}\ , \tag{5.18}$$

where λ_m'' denotes the predicted weight of the Gaussian to be spawned from the successful measurement of this candidate. Also, the entries of $\mathtt{S}$ and its sub-blocks in the above expression, correspond to the current innovation covariance matrix of $\mathbf{G}_i$. This

captures the information gain associated with the shrinkage of the measured Gaussian thanks to the positive match: if the new Gaussian has half the determinant of the old one, that is one bit of information gain. This was the only MI term considered by Davison [2005] but is now scaled and combined with the discrete component arising due to the expected change in the λ_i distribution.

Generally, an important aspect of the algorithm that is worth emphasising, is its fully dynamic nature allowing a general, adaptive behaviour. This is to be accounted to the fully probabilistic maintenance of the mixture, but also the way these predictions are made on the shape of the mixture to guide decisions for measurement. Defying the need for arbitrary scaling of weights, we combine the discrete and continuous terms of Mutual Information to take account of the expected variations in the distribution relying on the probabilistic predictions and Information Theoretic principles to drive the matcher towards efficient and robust performance.

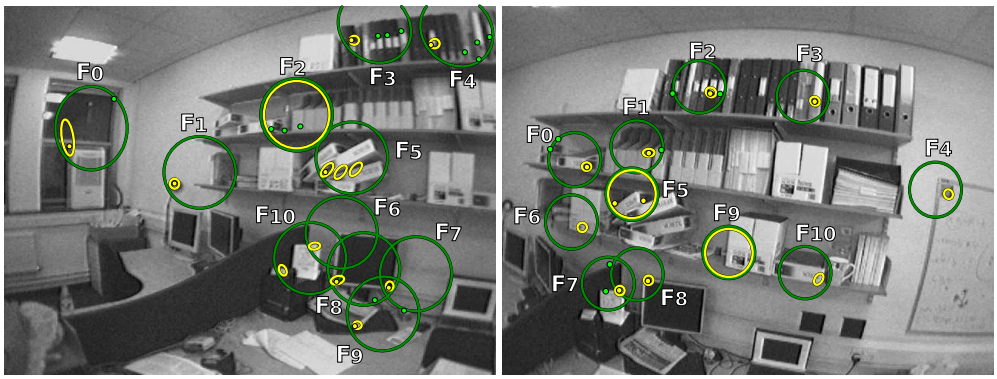
5.5 Results

We present results on the application of the algorithm to feature matching for several different situations within the MonoSLAM system of Davison et al. [2007] for real-time probabilistic structure and motion estimation, as discussed in Chapter 3. After discussing initial results in this section, we give a detailed analysis of how performance varies with different factors in Section 5.6.

In most cases where MonoSLAM has been applied (for example in tracking the motion of a hand-held camera in an indoor scene for use in augmented reality), the angular term is dominant in the motion uncertainty's effect on image search-regions, since clearly, it is much easier to induce fast feature motion through rotation than translation. Note that this fact has been harnessed directly in recent state of the art visual SLAM results like in the PTAM system of Klein and Murray [2008], where an explicit multi-stage tracking pipeline first performs simple but effective camera rotation estimation before tracking features to estimate pose. We would hope that Active Matching would be able to exhibit similar behaviour automatically.

5.5.1 Algorithm Characterisation

Our Active Matching algorithm simply takes as input from MonoSLAM the predicted stacked measurement vector $\mathbf{z}_T$ and innovation covariance matrix $\mathtt{S}$ for each image and returns a list of globally matched feature locations which are then digested by MonoSLAM's filter.



(a) Fast motion at $15Hz$ (b) Slow motion at $15Hz$

Figure 5.8: Active matching (AM) dramatically reduces image processing operations and mismatch encounters while still achieving global matching consensus. Here is a superposition of the individual gating ellipses searched in order to generate candidates for outlier rejection by JCBB (large, green ellipses) and the yellow ellipses searched for our Active Matching [Chli and Davison, 2008a] method. In these frames, joint compatibility needed to search $8.4\times$ more image area than active matching in (a) and $4.8\times$ in (b). Moreover, the 'intelligent' guidance of where to search in AM, pays off in terms of the matches encountered (yellow blobs) avoiding introducing unnecessary confusion in the system with the extra matches (green blobs) encountered in JCBB. Note that while typically AM needs to search at most one 'large' ellipse as shown in (a), in the case of a failed match-search like that of F_9 in (b) there is no evidence to reduce the rest of the search-regions further, resulting to template matching across another large ellipse for F_5. This demonstrates the adaptability of the methodology to different matching conditions, permitting the revisit of hypotheses upon lack of evidence.

5.5.2 Initial Sequence Results

Two different hand-held camera motions were used to capture image sequences at $30Hz$: one with a standard level of dynamics slightly faster than in the results of Davison et al. [2007], and one with much faster, jerky motion. MonoSLAM's motion model parameters were tuned such that prediction search regions were wide enough that features did not 'jump out' at any point — necessitating a large process noise covariance and very large search regions for the fast sequence. Two more sequences were generated by subsampling each of the $30Hz$ sequences by a factor of two. These four sequences were all processed for 11 features per frame using Active Matching. As a means of comparison, the same sequences have also been processed with the combination of full ellipse-searches of standard MonoSLAM and the JCBB method of Neira and Tardós [2001] to prune outliers. In terms of accuracy, Active Matching was found to determine the same set of feature associations as JCBB on all frames of the sequences studied. This observation confirms that the Gaussians spawned throughout the process of matching in each frame, were placed around the 'correct' matches, and also that the weight-scaling of the different hypotheses has been consistent with reality; if a Gaussian had got a low weight without enough evidence of it being an unlikely

	One tracking step	Matching only	No. pixels searched [relative ratio]		Max no. live Gaussians
Fast Sequence at 30*Hz (752 frames)*					
JCBB	56.8*ms*	51.2*ms*	40341	[8.01:1]	-
AM	21.6*ms*	16.1*ms*	5039		7
Fast Sequence at 15*Hz (376 frames)*					
JCBB	102.6*ms*	97.1*ms*	78675	[8.27:1]	-
AM	38.1*ms*	30.4*ms*	9508		10
Slow Sequence at 30*Hz (592 frames)*					
JCBB	34.9*ms*	28.7*ms*	21517	[6.89:1]	-
AM	19.5*ms*	16.1*ms*	3124		5
Slow Sequence at 15*Hz (296 frames)*					
JCBB	59.4*ms*	52.4*ms*	40548	[7.78:1]	-
AM	22.0*ms*	15.6*ms*	5212		6

Table 5.2: Statistical results of matching 11 features per frame with AM and JCBB. While both methods achieve successful resolution of consensus for all four sequences, Active Matching achieves fewer pixels(minimum of $\sim 7\times$ less) searched leading to lower matching timings and lower overall tracking timings. The 'Fast Sequence at 15*Hz*' is evidently the most challenging one, requiring a maximum of 10 Gaussians to represent the search-state at a particular instant.

scenario, then it could be mistakenly pruned off the mixture, resulting in missing some of the correct matches in the final, accepted result. This, highlights the importance of our fully probabilistic weighting scheme but also the guidance of matching using the mutual information cues to measure the most reliable and informative features first — it would not be a sensible strategy to search for a very common feature (with a high false-positive rate) when there are more distinctive features present, or implode the weight of the searched hypothesis after a null-search of a hardly recognisable feature (low true-positive rate).

The key difference of the two algorithms was in the computational requirements as shown in Table 5.2. The main result here is the ability of Active Matching to cope efficiently with global consensus matching at real-time speeds (looking at the 'One tracking step' total processing time column in the table) even for the very jerky camera motion which is beyond the real-time capability of the standard 'search all ellipses and resolve with JCBB' approach whose processing times exceed real-time constraints. This computational gain is due to the large reductions in the average number of template matching operations per frame carried out during feature search, as highlighted in the 'No. pixels searched' column — Global consensus matching has been achieved by analysing around one eighth of the image locations needed by standard techniques. (JCBB itself, given match candidates, runs typically in 1ms per frame.)

Testing fewer pixels for a template match, has the immediate effect of fewer matches being encountered. Guiding the matcher to 'look' at carefully selected (reduced) regions, we avoid introducing additional confusion to the system by extra false-positives improving the odds of converging to the true association scenario. A compar-

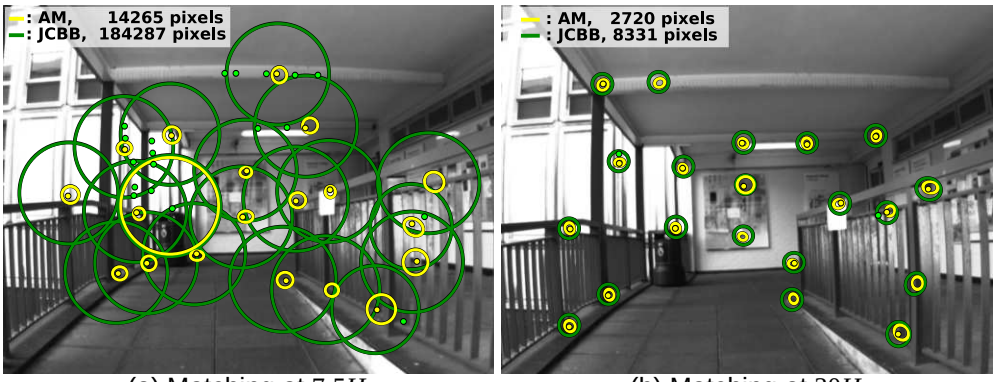
ison against RANSAC should have similar or worse than with JCBB, since the matcher would probably search into larger, fixed-sized windows to achieve a globally consistent outcome. The dramatic reduction in the area searched together with the matches encountered by the two techniques are overlaid on frames from two of the sequences in Figure 5.8.

In all the experiments presented in this work, we have used the Shi-Tomasi criterion [Shi and Tomasi, 1994] to extract the features tracked. However, our Active Matching algorithm is not specifically tied to any particular feature detector/descriptor. While SIFT [Lowe, 2004] or SURF [Bay et al., 2008] features would be particularly useful for matching due to their highly descriptive and distinctive nature (especially in the presence of only weak priors) the cost associated with their extraction renders them unsuitable for frame-rate matching (depending on the number of features tracked per frame). However, Active Matching could potentially allow standard use of sophisticated descriptors in tracking, since they need only be applied locally in small search-regions. Despite the somewhat lower quality alternatives like Shi-Tomasi, FAST [Rosten and Drummond, 2005, 2006] features or the randomised ferns classifier [Lepetit and Fua, 2006] as used by Williams et al. [2007], could be used equally effectively in matching — allowing denser frame-to-frame correspondence scenarios studied in Chapter 7.

5.5.3 Computational Complexity

We have seen that active matching will always reduce the number of image processing operations required when compared to blanket matching schemes, but it requires extra computation in calculating *where to search* at each step of the matching process. The sequence results indicate that these extra computations are more than cancelled out by the gain in image processing speed, but it is appropriate to analyse of their computational complexity.

Each step of the algorithm first requires MI efficiency scores to be generated and compared for up to the KF measurable combinations of features with current live Gaussians (note that K is the total number of Gaussians live at any instant during matching, while F denotes the corresponding total number of measurable features). Each such combination is evaluated for the MI it is predicted to provide, requiring computation of order $O(K)$ for the discrete component and $O(F^3)$ for the continuous component using the expression of Equation 5.18 (the determinants can be computed by LU decomposition or similar). The constants of proportionality are small here and these evaluations are cheap for low numbers of feature candidates. However, this complexity becomes the weakness of the algorithm with regard to high numbers of features, as addressed in the following section. The number of steps required to achieve



(a) Matching at $7.5Hz$ (b) Matching at $30Hz$

Figure 5.9: Typical images from the detailed performance analysis test-bed. Both (a) and (b) illustrate the search-regions for matching with AM and JCBB when tracking at $7.5Hz$ and $30Hz$ respectively. At low frame rates the search-regions are large to allow for bigger prediction error, resulting to more matching ambiguity. In both cases, AM shrinks the searched area with respect to the initial prior that methods like JCBB need to search, however this reduction is less evident at higher frame rates.

global matching of all features will be around $\bar{K}F$, where $\bar{K}$ is the average number of live Gaussians after pruning.

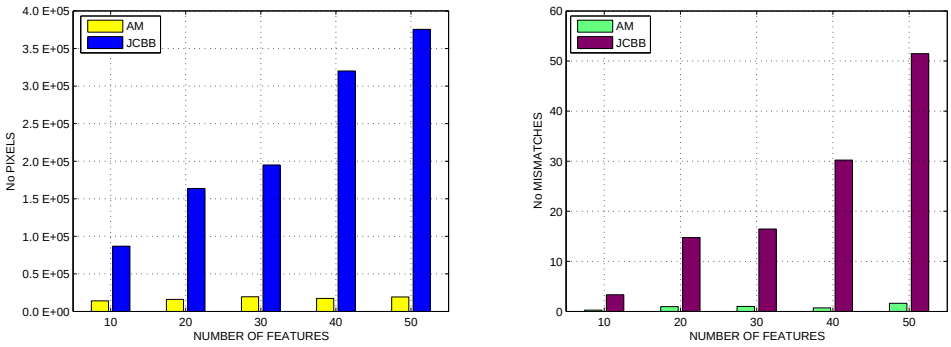
5.6 Detailed Performance Analysis

In order to assess the performance of Active Matching in detail, we have generated a set of experimental sequences by taking a high frame-rate image sequence (with resolution of 512×384) and down-sampling temporally to generate reduced versions. Varying both the frame-rate and the number of features being tracked per frame, we generate a matrix of experiments to form a test-bed of the performance of Active Matching. Typical images are shown in Figure 5.9.

5.6.1 Performance with Varying Frame-Rate and Number of Features

In this analysis of the computational performance of Active Matching, we consider the average time consumed per frame in terms of the main stages of the algorithm. Namely, within each matching step it is necessary to (i) **evaluate** the mutual information that each candidate measurement is predicted to provide followed by (ii) **measurement** of the selected candidate (by correlation) and finally (iii) the **update** of the mixture of Gaussians according to the measurement result.

For the sake of comparison with the 'get candidates first, resolve later' methods, we monitor the computational time needed to perform JCBB. Again, the timings are considered in terms of the time consumed to perform the two main steps of the method,



(a) Pixels searched per frame (b) Mismatches encountered per frame

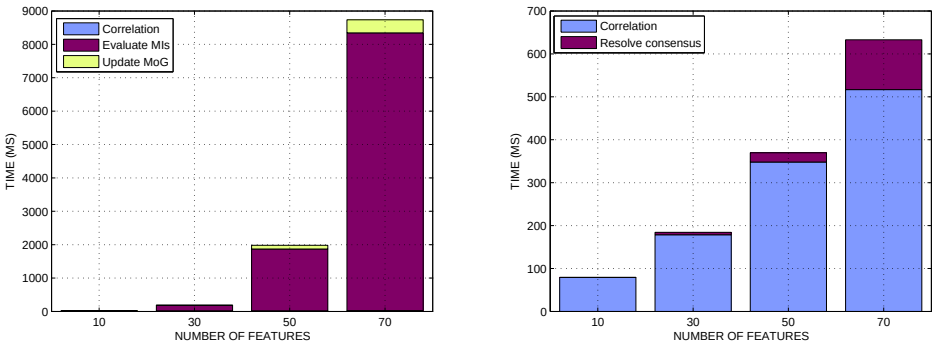
Figure 5.10: Statistics gathered while tracking at $3.75Hz$ using Active Matching (AM) and Joint Compatibility Branch and Bound (JCBB). Carefully selecting where to look for matches pays off for Active Matching which needs to search dramatically fewer pixels per frame than JCBB as demonstrated in (a). Also, constantly refining the search region for each feature avoids encountering unnecessary false positives, which is the case with Joint Compatibility as shown in (b).

namely to (i) **get the candidate matches** for each feature (by correlation) and (ii) **resolve** their consensus.

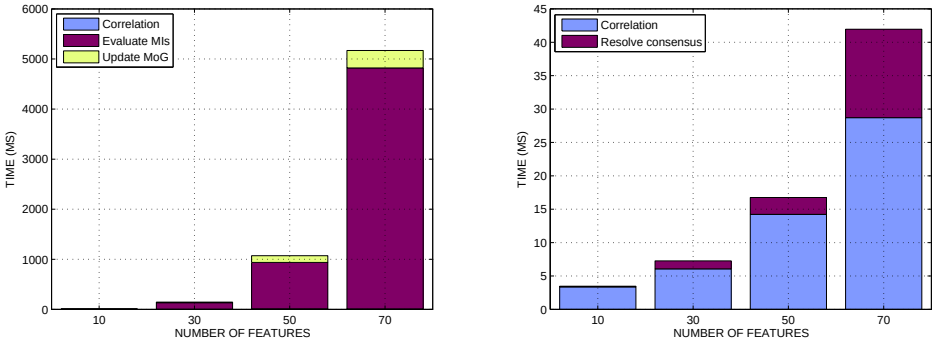
Fixed Frame-Rate; Varying Number of Features

Increasing the number of features tracked per frame means that the matcher is equipped with more evidence to aid the resolution of ambiguities, and in general it has been shown that tracking many features is key in obtaining more precision in pose estimation [Klein and Murray, 2008] and therefore is clearly desirable. On the other hand, more time needs to be consumed to process the extra information available. In order to study how much more time is needed we recorded timings while varying the number of features matched per frame when tracking a particular sequence. Time breakdowns for both Active Matching and Joint Compatibility are shown in Figure 5.11.

Our results show that Active Matching scales badly with increasing number of features and the step dominating the time consumed is the mutual information calculation of the candidate measurements in order to select which one to measure next. This is explained by the fact that every new feature added in the system introduces a new candidate measurement for **each** Gaussian present in the mixture. Therefore, Active Matching has more candidates to choose from, especially in a highly ambiguous scene where there are many Gaussians present (i.e. in the low frame-rate case in Figure 5.11(a)). Evaluating the MI of each candidate involves a prediction of how the MoG will evolve in both cases of a successful and a failed measurement of the current candidate. The estimation of the continuous MI part in particular, translates into



(a) Active Matching at $3.75Hz$ (b) Matching with JCBB at $3.75Hz$

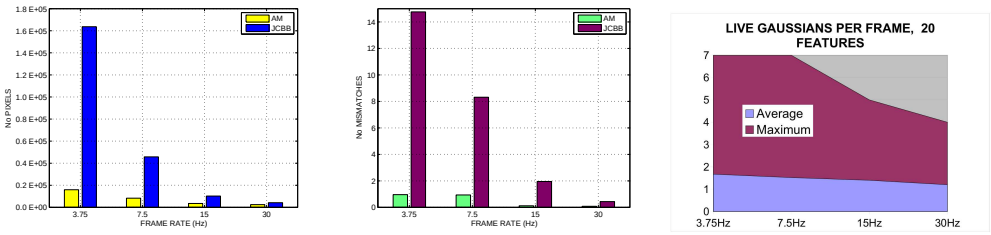


(c) Active Matching at $30Hz$ (d) Matching with JCBB at $30Hz$

Figure 5.11: Computational time breakdown for AM and JCBB while varying the number of features matched in the $3.75Hz$ (top row) and at $30Hz$ (bottom row) sequences. Active Matching scales badly with increasing number of features mainly due to the constantly expanding cost of the evaluation of mutual information of all the measurement candidates. Joint compatibility on the other hand maintains better performance when more candidate measurements are available but its performance is also far from real-time due to the increasing number of pixels needed to test for a template match.

the potentially costly handling of big innovation covariance matrices — which expand linearly in dimension with the number of features.

Joint Compatibility performs better with increasing number of features, but is still far from real-time performance. Measuring more features translates into more image regions we need to search for template matches but also potentially more false-positives — hence the constantly increasing time needed to perform correlation and resolve consensus. In Active Matching on the other hand, since we are very selective in the areas we look for matches, both the number of mismatches encountered and the number of pixels searched remain very low even for large numbers of features matched as demonstrated in Figure 5.10.



(a) Pixels searched (b) Mismatches encountered (c) No. Live Gaussians

Figure 5.12: Tracking 20 features per frame. Decreasing the frame rate more pixels need to be tested for a match as shown in (a). This also means that more ambiguity is present during matching as more mismatches are likely to occur as demonstrated in (b). When tracking highly ambiguous sequences, more matching scenarios arise per frame, hence the mixture of Gaussians needs to be populated with more members as confirmed in (c), in order to accurately reflect the search-state at every instant.

Coping with Ambiguity: Varying Frame-Rate; Fixed Number of Features

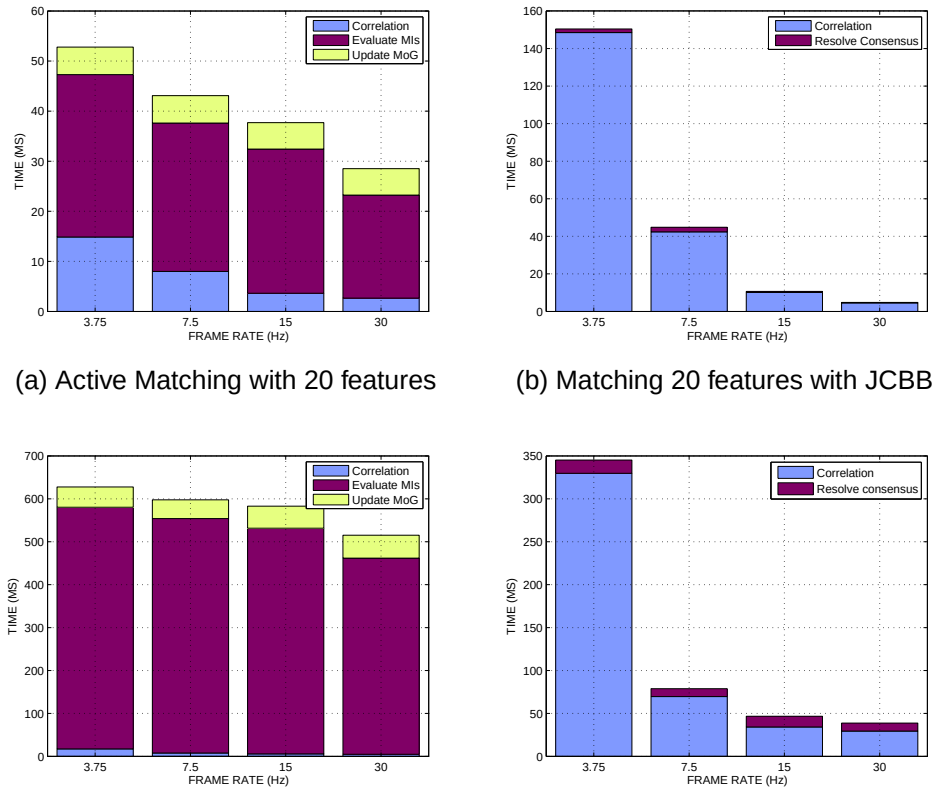
As the frame rate decreases and the search-regions of features cover bigger image area it becomes more likely to encounter more mismatches per feature, therefore complicating the process of discovering consensus in the prediction error. This is evident in Figure 5.12 where again, the number of pixels searched is dramatically reduced using Active Matching and as a result so is the number of mismatches encountered. As matching becomes more ambiguous with decreasing frame rate, we need more Gaussians in the mixture to accurately represent the different hypotheses arising, hence the negative slope in the maximum and average number of live Gaussians in Figure 5.12(c).

Tracking a scene with a low frame-rate camera is the real challenge for data association algorithms since the amount of time elapsing between consecutive frames is increasing, introducing larger uncertainty into the system. The uncertainty in the camera position translates into inflated search regions for each feature in the image plane.

5.6.2 Evolution of Mutual Information

Mutual information is what guides our matcher to select potentially more informative measurements, avoiding areas of high ambiguity. Since the process of evaluating the discrete and continuous parts for every candidate has been proven to be the main computational bottleneck of our algorithm, here we study the evolution of the mutual information throughout the matching steps of each frame to uncover the true value it has at different stages during matching.

As demonstrated in Figure 5.14 at the beginning of matching there is no ambiguity



(a) Active Matching with 20 features

(b) Matching 20 features with JCBB

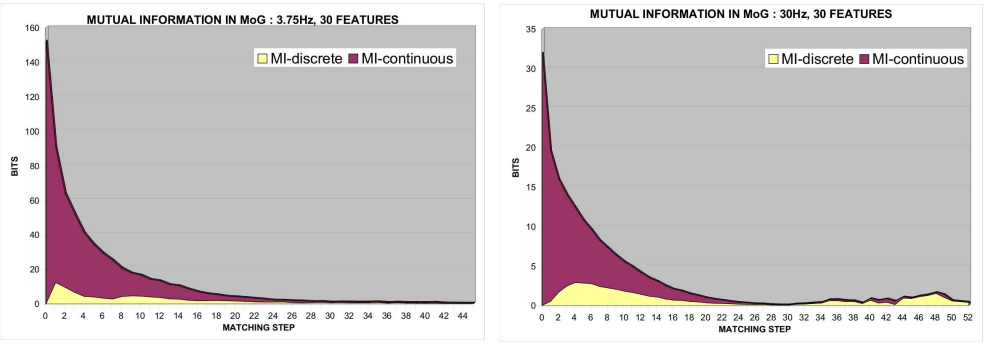
(c) Active Matching with 40 features

(d) Matching 40 features with JCBB

Figure 5.13: Timings breakdown for variable frame rate matching of a constant number of features using Active Matching and JCBB (tracking 20 features per frame in the top row and 40 in the bottom row). For around 20 features per frame, Active Matching is entirely within real-time limits for all frame-rates whereas JCBB's performance degrades at low frame-rates since more time is needed to find the correlation matches. When tracking 40 features per frame though, the costly evaluation of MIs pushes the time performance of Active Matching lower.

in the mixture since we start off with one Gaussian with high uncertainty (which is directly related to the frame-rate of tracking). This is represented by the dominant MI-continuous presence during the initial steps of matching, since this part of MI takes account of the desire to improve the accuracy of the most probable Gaussian. As we obtain matches for more features, the MI-continuous decreases dramatically and if any of the matches encountered is inconsistent with existing Gaussians, new ones are spawned to accurately reflect the ambiguous search-state. In such cases, the MI-discrete part comes in and sometimes takes over until both resolution of ambiguity and high accuracy are achieved.

The more features we match, the more information we expect to gain, always at the expense of computational time. So is it really worth the effort measuring one more feature? How much more information lies in this candidate measurement? A good



(a) MI components at $3.75Hz$ (b) MI components at $30Hz$

Figure 5.14: Evolution of the continuous and discrete components of MI for different frame rates, throughout the matching steps followed during AM in an average frame. In both (a) and (b) the two MI parts are shown stacked on top of each other to demonstrate the contribution that each has to the total MI in the mixture at any given step. The Continuous-MI is the dominant factor during the initial steps of matching, especially when tracking at $3.75Hz$ in (a) where there is more uncertainty present. As features get localised one-by-one, the uncertainty in the MoG decreases, but as soon as we start encountering inconsistent measurements, more Gaussians are spawned resulting to an increase in the Discrete-MI part which aims at resolving ambiguity. In both (a) and (b), the total MI tails off smoothly (notice the difference in scale) as the matcher encounters more measurements.

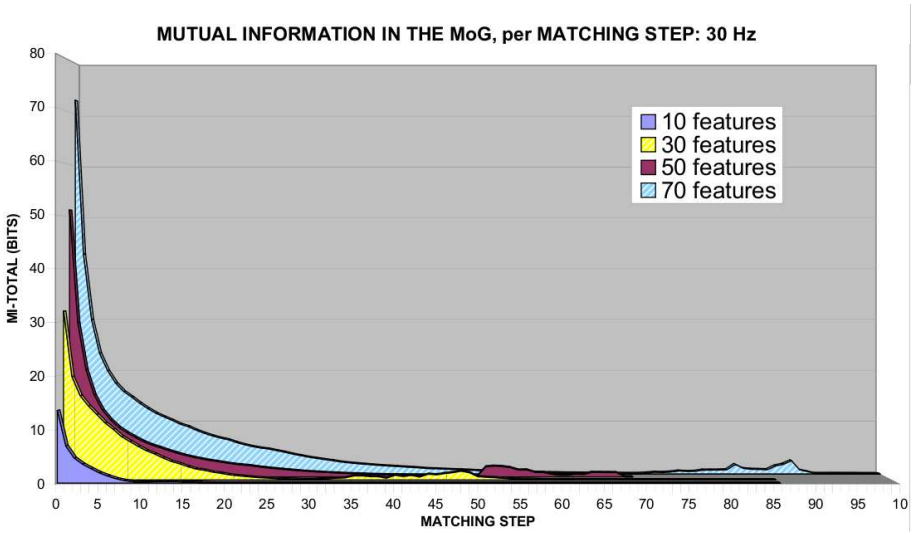


Figure 5.15: Matching many features is informative. But how much more information is a new measurement expected to give? This figure shows that the more the features we match per frame, the more information we expect to get during the initial steps of matching. After matching has progressed for a number of steps though, the MI present in the mixture does not decrease significantly.

answer to this question depends on a plethora of factors; feature characteristics, camera dynamics, speed of processor, etc. The evolution of the total mutual information in the mixture can be a representative measure of the value that an extra measurement can have in the current search-state. Figure 5.15 demonstrates that although initially there is higher mutual information to be gained for a bigger numbers of features, as we proceed with matching features one-by-one the total-MI decays exponentially. During the initial steps of the process, the evaluation of predicted MIs is key to the algorithm since most of the uncertainty and ambiguity in the scene get resolved. Measuring an extra feature after a certain stage though does not reduce the uncertainty of the current search state very much more. Thus, predicting which feature will provide the most information to measure next does not have any significant effect on the subsequent result of the algorithm.

These observations and conclusions are exploited in Chapter 7 to refine our Active Matching method so that it can dynamically adapt its performance according to the number of features and ambiguity in tracking, achieving improved computational performance.

5.7 Conclusions

This chapter has demonstrated how Active Matching, using a mixture of Gaussians formulation, allows global consensus feature matching to proceed in a fully sequential, Bayesian framework. Information theory plays a key role in guiding highly efficient image search and we can achieve large factors in the reduction of image processing operations.

While our initial instinct was that the algorithm would be most powerful in matching problems with strong priors such as high frame-rate tracking due to the advantage it can take of good predictions, our experiments with lower frame-rates indicate its potential also in other problems such as recognition. The priors on absolute feature locations will be weak but priors on relative locations may still be strong.

In an attempt to unveil the bottlenecks of the algorithm in comparison to standard ‘get candidates first, resolve later’ approaches like JCBB, we discussed an evaluation of the performance of Active Matching via extensive testing for variable number of features tracked per frame and different frame-rates. Briefly, our results indicate that the full Active Matching algorithm, despite maintaining real-time performance for different frame-rates and relatively low numbers of features per frame (around 20), scales badly when this number increases mainly due to the manipulation of large matrices during the calculation of mutual information.

Following a detailed discussion of the value of mutual information in the course of

the algorithm, we observed that carefully selecting which feature to measure at each step (guided by mutual information) plays a key role during the initial steps of matching where most of the uncertainty and ambiguity in the systems gets resolved. Based on this extensive analysis on the performance of Active Matching and the conclusions drawn, Chapter 7 explores new avenues towards a more scalable matching algorithm able to track even more features, faster.

From a more general point of view, we believe that mutual information has yet a lot to provide in high frame-rate tracking — the motion priors are indeed stronger then but the limited processing time available makes the task of resource allocation in matching even more challenging. Our long-term aim is to develop fully scalable algorithms via the active matching approach which will be able to perform the best matching job possible given a certain computational budget. For instance, state of the art optical flow algorithms [Zach et al., 2007] are now able to produce real-time matching for every pixel in an image when running on the latest GPU hardware. A hierarchical active approach may permit such dense matching performance to be approached with much reduced computational requirements.

6

Inferring the Hierarchical Structure of Visual Maps

In SLAM, it is well known that probabilistic filtering approaches which aim to estimate the robot and map state sequentially suffer from poor computational scaling to large map sizes. Various authors have demonstrated that this problem can be mitigated by approximations which treat estimates of features in different parts of a map as conditionally independent, allowing them to be processed separately. When it comes to the choice of how to divide a large map into such 'submaps', straightforward heuristics may be sufficient in maps built using sensors such as laser range-finders with limited range, where a regular grid of submap boundaries performs well. With visual sensing, however, the ideal division of submaps is less clear, since a camera has potentially unlimited range and will often observe spatially distant parts of a scene simultaneously.

This chapter presents an efficient and generic method for automatically determining a suitable submap division for SLAM maps as proposed in [Chli and Davison, 2009a], and demonstrates the application of this partitioning criterion to visual maps built with a single agile camera. The mutual information between predicted measurements of features is used as an absolute measure of correlation, and highly correlated

features get clustered into groups. Via tree factorisation, we are able to determine not just a single level division into submaps but a powerful, fully hierarchical correlation and clustering structure. The analysis and experiments discussed reveal particularly interesting structure in visual maps and give pointers to more efficient approximate visual SLAM algorithms.

6.1 Introduction

As a moving camera (or multi-camera rig) explores its environment, each measurement of the image location of a repeatably observable scene feature provides a probabilistic constraint on its location relative to the camera. It is well understood that many such measurements captured over a long image sequence, in combination with the assumption that most elements of the scene are static, suffice to permit stable estimates of the camera's 3D trajectory as well as a 3D map of the locations of the observed features. The most accurate solution to this estimation problem will be obtained by a batch optimisation approach which seeks the estimates which are most globally consistent with the measurements. This methodology is known as bundle adjustment in the photogrammetry and computer vision communities, as discussed in Chapter 2, and has been generalised by SLAM researchers in graph optimisation frameworks which are able to incorporate all types of sensory input [Thrun et al., 2005; Dellaert, 2005].

6.1.1 Sparsification for Real-Time Visual Mapping

The natural emphasis in robot vision has been on visual localisation and mapping methods which are able to run not as off-line optimisation but as sequential procedures potentially implementable in real-time on modest computing hardware. Real-time operation inevitably requires some form of approximation or sparsification of full global optimisation, since it soon becomes infeasible to repeatedly find a globally optimal solution based on the ever-growing volume of data acquired from a live camera. Moreover, the increasing demand for denser maps and extended coverage over larger areas, drives research towards more agile manipulation of big data loads since the complexity of maintaining full probabilistic maps threatens real-time performance.

Real-time methods for camera motion estimation like the works of Mouragnon et al. [2006] and Nistér et al. [2004], can be classed as visual odometry, choosing to 'forget' information from past measurements beyond a sliding time window. In essence, the focus there is on local motion estimation rather than the maintenance of a complete probabilistic map of all visited scenes. The result is indeed highly accurate local motion estimates due to the ability to cope with a large number of feature measurements per frame, but on the other hand, such methods suffer from drift over

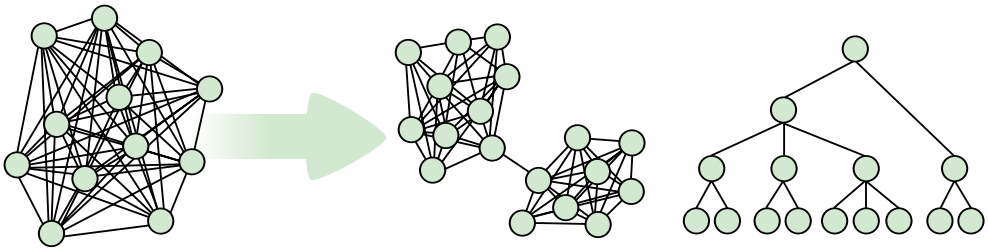


Figure 6.1: Sparsifications of the complete graph for large-scale SLAM. The most accurate solution to SLAM is global off-line optimisation (bundle adjustment), but real-time systems need sparsifying approximations to cope with the expanding map. Solutions include splitting the map into local submaps to process separately like in [Estrada et al., 2005; Bosse et al., 2003] or discovering approximate global tree-like structure (e.g [Frese et al., 2005; Paskin, 2003]). These methods work well when they capture the most important correlation structure – but the question of identifying suitable sparsifications in visual maps remains.

extended sequences. This problem has recently been successfully mitigated by the use of 'keyframes': a subset of representative images and camera poses selected from the continuous stream and subject to global optimisation with the rest of the trajectory related to these. Both the PTAM monocular system of Klein and Murray [2007] and the FrameSLAM stereo system of Konolige and Agrawal [2008] have demonstrated impressive performance.

Alternative real-time methods for visual mapping based on sequential probabilistic filtering (e.g. [Davison, 2003; Eade and Drummond, 2006b]) aim to 'summarise' the information gained from past images with a probabilistic state. This uncertain estimate of the camera and map state can be combined with the information from each new image in a weighted average of fixed complexity at each time-step. It turns out however, that the accurate probabilistic representation of uncertainty which is required here is computationally expensive in a way which scales poorly with the number of mapped features. For this reason methods such as in [Davison, 2003] only map features relatively sparsely. The most successful solution to this problem has been, as in real-time SLAM research using other sensors (e.g. [Bosse et al., 2003; Bosse and Roberts, 2007; Chong and Kleeman, 1999; Kaess et al., 2008; Tardós et al., 2002]), to split a large map into several conditionally independent visual submaps which can be processed separately (e.g. [Clemente et al., 2007; Eade and Drummond, 2007; Piniés, 2009]). Perhaps the most successful approach has been the sophisticated real-time monocular SLAM system of Eade and Drummond [2007, 2008] which connects submaps (here called 'nodes') with a higher level graph structure estimating their relative locations. Their method has been discussed extensively in Chapter 2.

So keyframes or submaps are sparsifying approximations which permit real-time implementation of globally consistent mapping. But in the case of either keyframes or

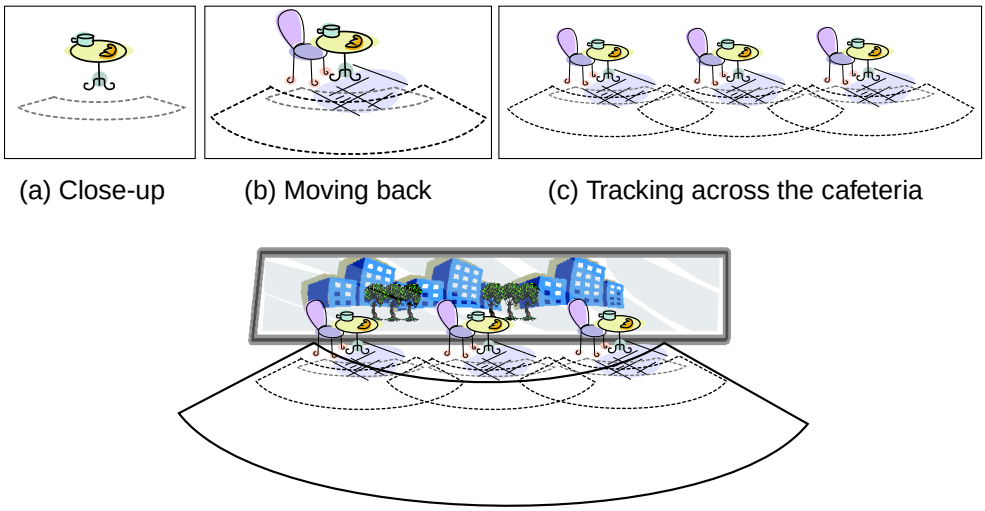


Figure 6.2: Visual tracking in a cafeteria, studying the regions of camera motion within which features are observable. The features on a table can aid tracking within a small distance from it as in (a). Moving back a little in (b), some more features come into the field of view permitting camera motion estimation over a bigger area. In the same way, the camera can track its path along the cafeteria in (c) using the features observable from each pose. This might seem a straight-forward sub-mapping scenario, but what happens when there is also a window as in (d)? The camera's infinite range makes features in both the foreground and the far background visible at the same time, which makes an ideal submap division less clear.

submaps, there remains the question of how to choose their locations and scope.

6.1.2 The Special Character of Visual Maps

A little consideration makes it clear that visual sensing is not in general conducive to a straightforward division of a scene into block-like submaps for the purposes of efficient map processing, as has proven successful with other sensors. Laser range-finders and sonar sensors have strictly limited ranges of measurement, and setting submap sizes which relate closely to this is a sensible strategy — features located farther apart than this range will not be simultaneously observed. There are other potential heuristic strategies for the choice of submap boundaries: an upper bound on the number of features, or bounded uncertainty (or deviation from linearity as in [Eade and Drummond, 2007]) within a submap. In keyframe approaches, all of the scene elements visible from a particular camera pose are implicitly grouped together for the purposes of estimation, independent of their distance from the camera. This can cope with a range of depths, but is still a somewhat arbitrary grouping.

Consider for instance the example of a large, cluttered room (perhaps a cafeteria), browsed and mapped by a mobile camera carried by a robot or person as in Figure 6.2. The camera will view parts of the room from different distances to obtain different

levels of detail: a table may be framed from close-up, or the camera may move even closer to inspect particular objects. The periphery of these views though may simultaneously be filled with distant walls or even the outside scene beyond the windows. Different features in a scene tend to have more strongly correlated estimates in a map when they are regularly co-observable by the moving sensor, but this is not always the case if they give different information about camera location. Similarly, features in almost the same scene location but measured from different camera positions may be uncorrelated.

6.1.3 Determining Hierarchical Map Structure

The simple tracking scenario of Figure 6.2, demonstrates the need for a general and dynamic criterion to place submap divisions when dealing with visual maps. Basing decisions solely on the co-observability of features is clearly insufficient, while more ad hoc thresholds limit the applicability of algorithms and most often lack the insight of a theoretical investigation. Here, we explore the benefits of the application of Information Theory in this context, to drive map partitioning.

Our investigation leads to the emergence of a straightforward and absolute measure for the level of correlation between features in a mapping scenario based on the mutual information of predicted measurements. We show that this automatic inference of structure can easily go beyond a single level of submaps to deduce a full hierarchy of correlation relations via a tree decomposition. In fact, many of the most exciting recent approximate but super-efficient SLAM algorithms [Frese, 2006; Paskin, 2003; Paz et al., 2007a] are tree-like in nature, showing the power this gives.

The tree structure encodes a hierarchy of correlation levels between features which permits their grouping into sets with a user or application-settable coarseness or fineness, from one extreme where all features are considered as independent and unrelated to the other where they will all be grouped together. In between, features will be accumulated into clusters which gradually join into a single whole.

It is important to understand that the hierarchical structure which this method discovers is that of the *probabilistic map*, not a fundamental property of the scene itself. The structure depends on the motion of the camera, priors which we have about how the camera moves, and its imaging properties such as resolution and field of view. In a map built using an omnidirectional camera, for instance, we might expect simultaneously observable features on opposite sides of a robot to be regularly highly correlated in measurements and that they would be clustered together, while in a map built using a camera with a narrow field of view they would be distant in the tree. We consider that this dependence on the specifics of the camera and motion is a strength of the approach, not a weakness.

6.2 Feature Correlations in Mutual Information Space

In order to achieve high quality approximations, we must think in terms of preserving the most important correlation structure of the SLAM map. Using the Information Theoretic framework discussed in Chapter 4, here we translate correlations between individual image feature measurements into the mutual information (MI) space. MI can be understood as an absolute, normalised measure of degree of correlation and more precisely, the pairwise MI as defined in Section 4.3.2 describes the amount of common uncertainty (entropy) shared between two candidate measurements.

Following the notation for probabilistic SLAM filtering of image sequences introduced in Chapter 3, we consider making image measurements of a scene of which the current state of knowledge is modelled by a probability distribution over a finite vector $\mathbf{x}$ stacking camera and map parameters. When a new image arrives, we can project the current probability distribution over the state parameters $\mathbf{x}$ into measurement space to *predict* the candidate feature measurements from the new viewpoint. This joint probability distribution $p(\mathbf{z}_T)$ describes the entries of stacked vector $\mathbf{z}_T = \begin{pmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \dots \end{pmatrix}^\top$ containing all predicted candidate feature measurements via a mean vector $\hat{\mathbf{z}}_T$ containing their predicted image positions and the innovation covariance matrix $\mathtt{S}$ encoding the variance in these predictions.

Equipped with this set of predictions we can assess the strength of correlation between individual candidates — essentially asking 'how much on average, does measuring one feature tells us about the others?'. In Chapter 5 we have seen how this joint prediction can be used for probabilistic data association (matching), in either batch [Neira and Tardós, 2001] or sequential [Chli and Davison, 2008a] forms. However, here we are interested in the *pairwise* relationships between the candidate feature measurements (rather than the relationship of one feature and the set of the rest of the visible features in the frame) with the aim of understanding the *structure* of correlations and therefore the scene.

In order to convert the correlations between candidate measurements $\mathbf{z}_i$ and $\mathbf{z}_k$ encoded in $\mathtt{S}$ into the pairwise MI space, we use the expression derived in Equation 4.35 and restated below:

$$I(\mathbf{z}_i;\mathbf{z}_k) = \frac{1}{2}\log\frac{|\mathtt{S}_{ii}||\mathtt{S}_{kk}|}{|\mathtt{S}_{i,k}|}\ . \tag{6.1}$$

Evaluating this quantity for every pair of candidate measurements, we can build the MI matrix as described in Section 4.3.2. If the pairwise MI $I(\mathbf{z}_i;\mathbf{z}_k)$ is large, a measurement for $\mathbf{z}_i$ is expected to tell us a lot about the predicted location of $\mathbf{z}_k$ (and vice versa) suggesting the presence of a strong correlation link between them. On the other hand, if the two features have never been predicted to co-occur then their common information content in measurement space will be zero. Generally, when two

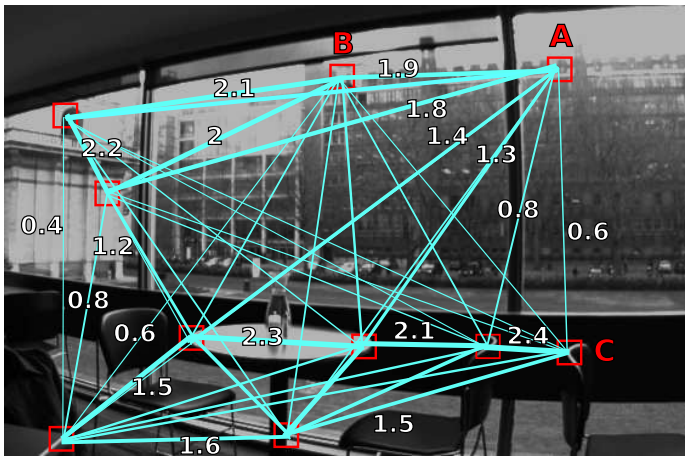


Figure 6.3: The projection of the MI matrix during real single-camera-tracking in a cafeteria. Depending on the covisibility and the coherence of motion throughout the sequence, some features build stronger correlations than others, as evident in the projected MI links (quantities shown in bits). For example, a potential measurement of feature A is predicted to give 1.9 bits of information to its neighbour B, whereas C is expected to gain only 0.6 bits from this measurement. Indeed, due to the fact that both A and B lie in the far background, they have been moving consistently throughout the sequence building a strong correlation bond. However, while C has also been visible in all frames, its correlation with A has weakened through time since it lies closer to the camera and moves incoherently with A.

features have been regularly covisible throughout a substantial number of frames in the tracked video sequence they are usually strongly correlated. However, this stops being true when the two features do not move *coherently*: each individual measurement then gives different information about the camera motion, hence their correlation link gradually weakens despite being co-observed. This fact confirms that the elements of the MI matrix encode the scene structure as perceived by the camera throughout its path — hence all processing to reveal the scene structure can be based on the manipulation of this matrix. Figure 6.3 demonstrates the visual projection of the MI matrix built using real data captured while tracking the camera motion in a cafeteria.

6.2.1 State Space vs. Measurement Space

All our Information Theoretic analysis of the value of elements in the SLAM map is performed in the measurement space (i.e. image space). While this is an obvious choice when applied to feature matching where all actions are taken in measurement space, the most natural choice when dealing with the actual map and its structure would be to perform calculations in state space. However, our experimentation with state space data has revealed drawbacks in this approach affecting the quality of performance. In fact, the problem relies in taking into account the global uncertainty in the map estimates. As the camera keeps exploring new areas, this uncertainty in constantly

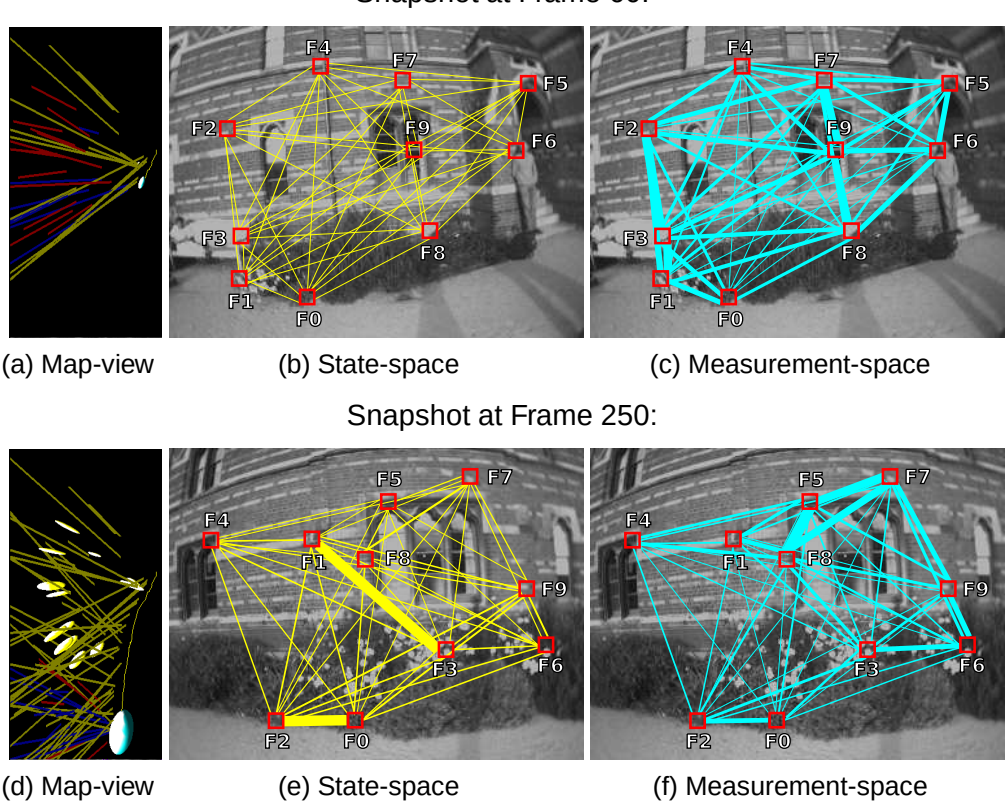


Figure 6.4: Comparison of pairwise MIs in state and measurement spaces. The top row images are snapshots taken at Frame 60 while the bottom row corresponds to Frame 250. The map-view in (a) and (d) are shown to scale in an attempt to demonstrate the growth in uncertainty of the camera throughout the intermediate frames (the camera here follows an exploratory trajectory). The rest of the figures illustrate the projections of the pairwise MI links as calculated in either state-space or measurement-space. It is important to note that while the thickness of lines corresponds to the strength of links, the MI scores computed in state-space are two orders of magnitude larger than in measurement-space. So for the sake of clarity, (b) and (e) share the same thickness scale but this is different from the scale shared by (c) and (f). While the link structure in (b) is fairly homogeneous, in (c) we observe a more interesting distribution of MI strength. As the uncertainty in the camera pose grows as shown in (d), the state-space estimates of feature locations in (e) are heavily affected, resulting to great variations in link strengths due to discrepancy in global position estimates of feature locations. In measurement space however, the *relative* structure remains much more stable throughout the sequence while illustrating some detail of feature correlations still. Since we are interested in the relative uncertainties in the estimates of feature locations in order to understand the scene structure, we conduct all our Information Theoretic analysis of maps in measurement space.

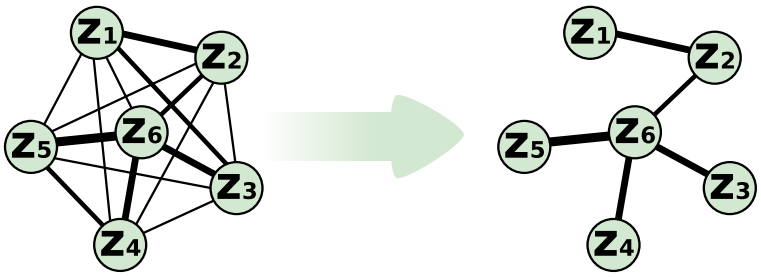
growing with newly initialised features inheriting most of it. Moreover, the inevitable depth uncertainty of features in the map influences heavily the structure of the correlations and subsequently the pairwise MI links (computed based on state-space data). As a result, while the structure of the MI graph of pairwise MI links in measurement space exhibits relative stability throughout the camera trajectory, the corresponding link structure based on state-space data is far more irregular — the strength of links is progressively enforced as more uncertainty accumulates into the global feature estimates, and there is also a large variance of correlation strength in these links resulting from the partially initialised features (i.e. those with large depth uncertainty).

Figure 6.4 depicts a comparison of the MI graph as obtained using both state-space and measurement-space estimates, for two different frames of an exploratory sequence. As the camera moves away from its initial position, its state estimate progressively becomes more uncertain and so is the probabilistic map, having a big impact on the state-space computations. Here, we are interested in the *relative* uncertainties and correlations of features with the aim of maintaining a stable 'representation' of the scene structure, in order to understand it. Therefore, we conduct our Information Theoretic investigation in the measurement space, which usually has the additionally property of manipulating smaller matrices than in state-space.

6.2.2 From a Single Frame to a Sequence

Our analysis on inferring the scene structure begins by investigating the distribution of correlations between features in a single frame. However, since the goal is to expand our understanding to the whole map, we can easily extend our methodology to cover all the data gathered in the system throughout the sequence of frames used to build this map. Keeping a running average of the MI links between features in the map, we can accumulate information on features that were co-observed at any instant. It is worth noting that at any frame we only need to calculate the MI matrix of the measurable features in that frame, therefore the cost is tractable, since all data needed is evaluated in image space.

Depending on the nature of the scene and the camera motion, the structure of this *full* MI matrix (containing average MI links between all features in the map) will be different — for example, in the case of a purely exploratory sequence this matrix will be sparse since features viewed from spatially distant poses will never be co-observed. On the contrary, when the camera browses a small scene in a loopy manner, we expect denser configuration of MI links as most features will be co-observed. Following this formulation, we can then inspect the MI matrix built over the whole map to automatically discover areas of high mutual information density.



(a) Complete MI graph (b) Chow-Liu tree approximation

Figure 6.5: The approximation of a joint PDF $p(\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_6)$ by second-order and marginal distributions yields a tree. Here, (a) shows the complete MI graph where thicker links represent higher mutual information between the nodes they connect, and (b) is the optimal such approximation maximising the total information preserved in the tree as suggested by Chow and Liu [1968].

6.3 Tree Factorisation

With the goal of unveiling the hierarchy of correlations encoded in the MI graph, one can consider different graph structures that can potentially provide a 'meaningful' insight into the structure of the complete graph. The tree being the simplest such structure, here we consider how we can decompose the MI matrix via a tree factorisation.

A probabilistic estimate of the values of a set of variables $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N\}$ given background information I is most generally specified by a joint density function over all of those variables:

$$p(\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N) = f(\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N) \; . \tag{6.2}$$

One possible approximation to a general joint probability density is the factorised form below:

$$\begin{aligned} p(\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N) &= p(\mathbf{z}_N) \prod_{i=1}^{N-1} p(\mathbf{z}_i | \mathbf{z}_{i+1} \ldots \mathbf{z}_N) & (6.3) \\ &\approx p(\mathbf{z}_N) \prod_{i=1}^{N-1} p(\mathbf{z}_i | \mathbf{z}_{i+1}) \; . & (6.4) \end{aligned}$$

Figure 6.5 shows that this approximation can be interpreted as a tree-shaped model of probabilistic links between variables (each link representing a conditional density function over just the two connected variables). Out of all possible such tree approximations, Chow and Liu [1968] showed that the one closest to the full, joint probability density can be found by considering the Mutual Information between pairs of variables in this distribution: the optimal approximation of the complete distribution via a

first-order dependency tree corresponds to the maximum spanning tree[1] in the Mutual Information space. The application of this conclusion in our scene structure analysis is straightforward, as we already have the MI graph of links between the features in the SLAM map. The optimal tree approximation in the MI space, will be subsequently referred to as the 'Chow-Liu tree'.

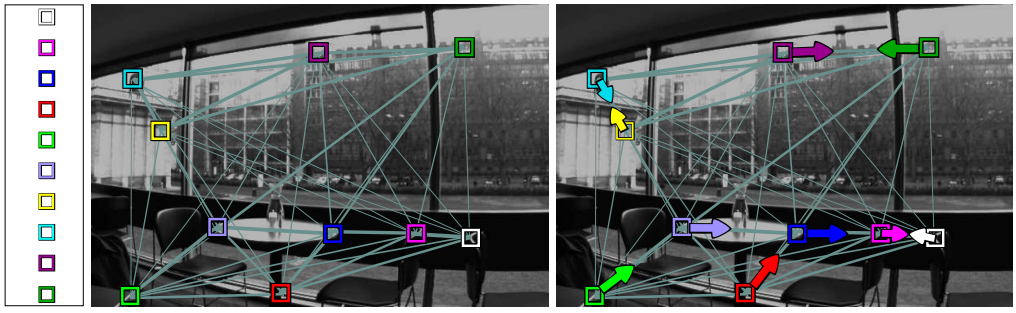
6.4 Inferring Hierarchical Structure from the Tree

Have deduced an undirected tree structure linking the features, there remains the question of how to use this to infer hierarchical clusters. One option would be to somehow choose a root feature and then 'hang' the tree from this. By choosing a number of levels down from this root we could fix where to lop off branches, all nodes further down each branch forming a cluster. Alternatively, we could use a threshold on the MI scores on branches of the tree, cutting all those weaker than a certain value to divide the tree into clusters. In experimentation, while this approach has some nice properties it tends to leave many features alone in clusters of one. We have also considered expanding this idea to progressively identify cliques of features while lowering the threshold of MI link strength. However, despite the theoretical justification, in practice features most often team up to a single cluster before constructing the first clique resulting to a single cluster.

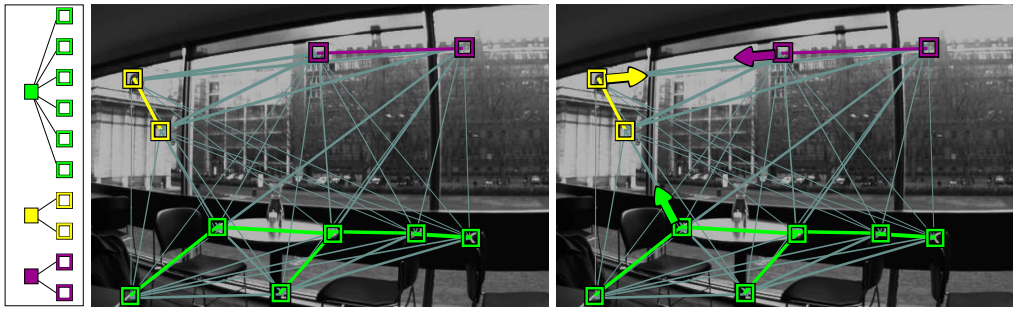
Instead, here we propose a simple bottom-up procedure where features are progressively grouped in a manner similar to Chow and Liu's original algorithm to build the spanning tree. The goal being to identify image regions of high mutual information density, we consider an example where N features have been tracked in a sequence of frames and start joining features together. Figure 6.6 shows a real, simplified example of the step-by-step building process of this tree.

We start off with every predicted-to-be-visible feature lying in a separate cluster (or submap), as if these features were completely uncorrelated. All off-diagonal entries in the MI matrix would then be zero and therefore at this stage we have N different clusters. Jumping a level up the hierarchy, the aim is to link each tree to the tree with which it is sharing the strongest tie so that no cycles are introduced. Therefore, we progress by examining all MI links spanning out of each cluster, and fuse the strongest such link (avoiding loops) and the node it is connected to into the cluster in question. Identifying the new clusters forming from joining features together in the previous step, we define a new level in the hierarchy containing all nodes and links participating into clusters. Proceeding in the same manner, we reach the top level of this hierarchy where all features are connected into a single cluster. Interestingly, all links 'activated'

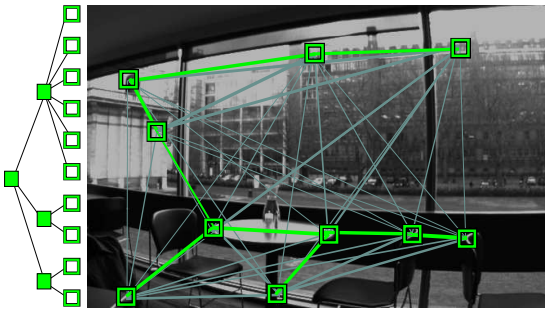
[1] The acyclic path connecting all nodes in a weighted graph which yields the maximal sum of weights.



(a) Level 0: Each feature in a different cluster (b) Level 0: Grow clusters



(c) Level 1: New clusters created (d) Level 1: Grow clusters



(e) Level 2: Features joint in a single cluster, Chow-Liu tree is created

Figure 6.6: Discovering regions of high MI density by progressively building the Chow-Liu tree in a real scene. Having computed the links of the complete MI graph, initially each feature is set to a different cluster as in (a). (Note: different clusters are denoted by different colours and on the left of each row is a diagrammatic representation of the features' memberships into clusters). In (b) each cluster attempts to join with the rest following the strongest, outgoing MI link. This growing process results in the 'activation' of a subset of the links of the MI-graph which is then used to define new clusters for the new hierarchy level as in (c). Since the features have not yet joined into a single map, another growing process takes place in (d) which leads to the completion of the Chow-Liu tree joining all features to a single map, signifying the end of the procedure.

throughout this process and present in this final cluster correspond to the links forming the Chow Liu Tree. Moreover, the clusters in intermediate hierarchy levels correspond to smaller parts of the Chow Liu Tree, where the density of MI strength is high.

DISCOVERCLUSTERS

1 Hierarchy level $h = 0$: each feature in different cluster
2 **while** (number of clusters > 1)
3 GROW each cluster following the strongest, outgoing MI link
4 DEFINE new level: $h = h + 1$ corresponding to new clusters
5 **end while**

Following the analysis to infer the scene structure in a single frame, we can expand this idea to a sequence of frames, keeping a running average of all MI links, as described in Section 6.2.2. We can then build the Chow Liu tree over the whole map to automatically discover areas of high mutual information density in a hierarchical manner.

6.5 Results

We demonstrate this algorithm on several different visual maps generated from a handheld camera using a standard configuration of MonoSLAM [Davison, 2003; Davison et al., 2007]. MonoSLAM uses the Extended Kalman Filter (EKF) to incrementally construct a probabilistic map of visual point features represented by a single joint Gaussian distribution as described in Chapter 3. At each new frame a subset of features is selected for measurement based on whether they are predicted to lie within the camera's field of view and whether the camera is predicted to be within a set of bounds for each feature on motion (inducing scale changes and warping) where correlation matching is expected to be possible. The innovation covariance matrix S is calculated at every frame of during tracking with MonoSLAM as part of the active feature matching (data association) process so there is little additional computational cost incurred by our tree construction algorithm.

The following is an analysis of several single frames and extended sequences at 30Hz which draws attention to the behaviour of the algorithm to infer submap divisions. While maintaining a full EKF map in MonoSLAM, we apply our Information Theoretic framework on a variety of tracking scenarios to demonstrate the generality of the approach in providing an insight to the current map structure. Finally, we compare the quality of submap partitions suggested via our Chow-Liu submapping to the Naive approach through a quantitative analysis.

(a) Planar scene (b) Scene with different depths

Figure 6.7: Clustering features based on data obtained from a single frame. (a) is a typical frame of the Keble College Sequence, tracking features on the wall. Since this is a planar scene, the distribution of the mutual information between features, and therefore the clustering result, is mostly according to image proximity. On the other hand in (b) is a scene with more interesting structure hence the clusters are also based on the depth of features.

6.5.1 Single Frame Analysis

As a proof of the concept of the work in this chapter, we performed the simplest application of our tree-based clustering; a Chow-Liu tree is built using data from a single frame only.

Once the correlations between features have been settled and the map has converged, then so have the mutual information links between them. Therefore, by building the Chow-Liu tree we can infer clusters that are conceptually consistent. Figure 6.7 demonstrates the application of our methodology to group of features based on the distribution of the pairwise MI links: while correlation structure is dominated by the features' proximity in image space in the case of a planar scene, the MI links capture some more interesting scene structure in the presence of significant difference in features' depth positions resulting to that distinction of background/foreground clusters.

However, the main interest here is to infer meaningful and consistent submaps through time, where features are constantly added, deleted and updated in the map. Hence in the following, we conduct an analysis on sequences of frames.

6.5.2 Sequence Analysis

Sideways Exploration

Here we analyse a segment of the image sequence of Clemente et al. [2007] taken by a hand-held camera moving sideways around a large college quadrangle, moving at a steady walking speed while observing a wall at approximately constant depth. We call the sequence exploratory because the camera moves progressively and does not return to previously visited positions in the segment (we do not consider large loop closure

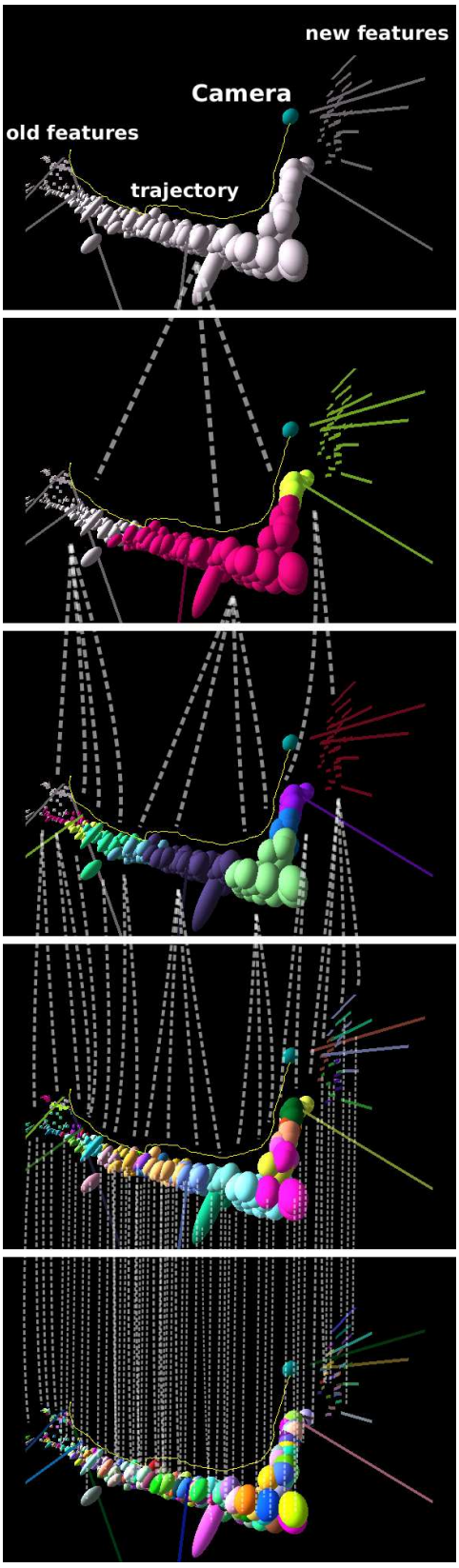


Figure 6.8: Discovering the map structure of the sideways exploratory Keble College sequence (left). This hierarchy tree is build after tracking for 2500 frames. At the root of the hierarchy tree (top image) all features lie in a single map. Moving deeper in the hierarchy each map progressively splits into several submaps. Here, the roughly uniform distribution of features across the image and the constant speed of the camera cause a uniform distribution of MI links between features. Hence the formation of regular-sized submaps, as expected.

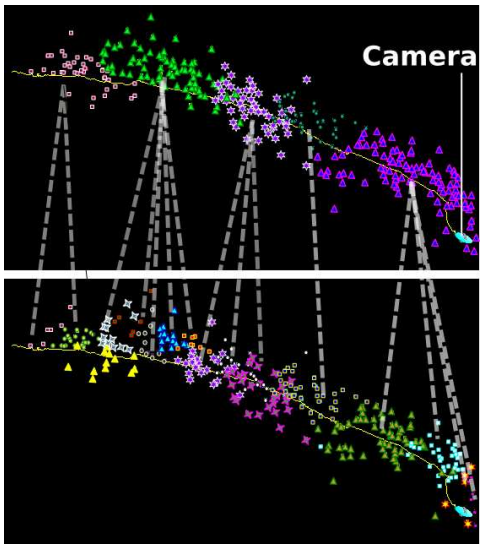
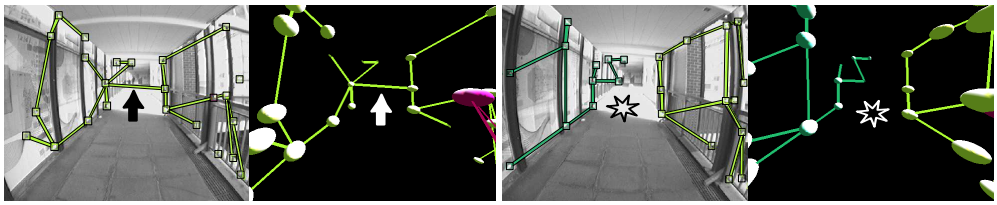


Figure 6.9: Submapping in a corridor sequence tracked with a forward-looking camera (above). Here are two intermediate levels of the hierarchy tree where features are visible on both sides of the camera's trajectory, so left and right hand side features are grouped into the same submap. The submaps formed here have overlaps due to the covisibility of features belonging to different submaps during tracking, in contrast to the Keble sequence clusters illustrated in Figure 6.8 where submapping is straightforward.

Note that the feature uncertainties are not displayed here for the sake of clarity.



(a) Corridor Clusters in frame 184 (b) Corridor Clusters separate in frame 208

Figure 6.10: Typical clusters in the corridor sequence (note: in both (a) and (b) the left image corresponds to the camera view and on the right is the 3D perception of the map). In this type of sequence, the features on the walls closer to the camera move quickly in the image (and past the camera) being observed from a wide baseline, hence at a given frame their depth estimates are much more accurate than the ones further away. Here, the MI links of the features visible in (a) hold them together into a single cluster (bright green). As the camera moves closer to the end of the corridor, the depth estimates of the furthest features are gradually refined, causing the break of this cluster into two separate ones in (b) — this is the point of 'realisation' that the visible features no longer move coherently.

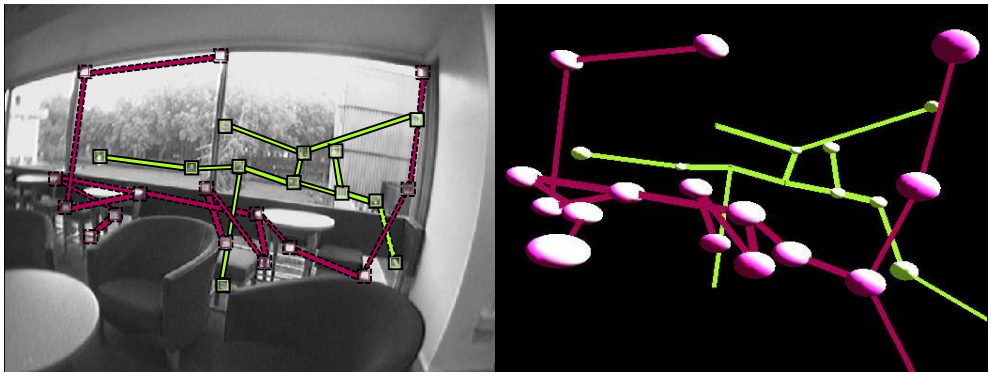
here). This sequence is of interest because its simple nature makes the 'ideal' map structure a clear case of approximately regular metric division, as implemented explicitly in [Clemente et al., 2007] by bounding the number of features in each submap at a fixed value.

Figure 6.8 shows the grouping of features at all levels of the Chow-Liu tree formation; each feature belongs to a different submap at the leaves of the hierarchical tree, and then they gradually team-up to form a single map. Due to the roughly constant speed of the camera and the regular presence of features on the observed wall, the distribution of mutual information links is uniform and therefore the clusters forming in the intermediate levels of the hierarchy are fairly similar in size. This result agrees our initial expectation, demonstrating the intuitive soundness of this approach.

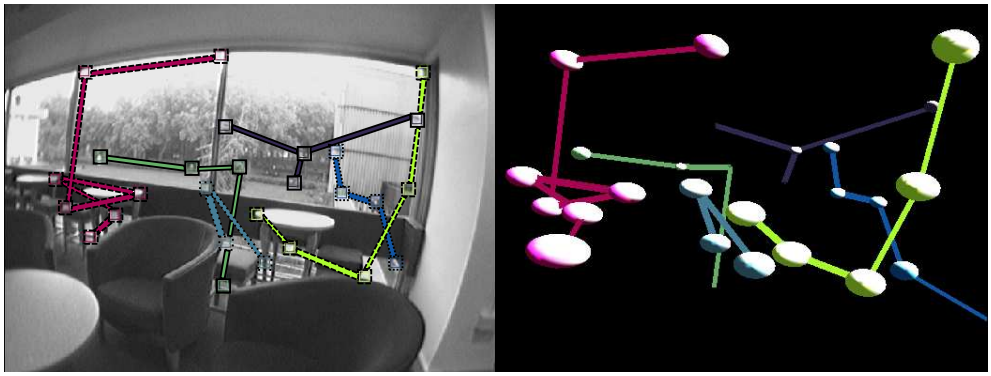
Forward Exploration

The next example is an exploratory sequence from a forward-moving hand-held camera. The additional interest here is in the presence of features close to the centre of expansion in the middle of the image which are very distant and therefore remain visible for long periods of time while the majority of features towards the edges of the image quickly pass out of the field of view. This means that at a given frame, the depth estimates of the features on the walls closer to the camera are more accurate (as they have been moving more quickly in the image) in comparison to the ones further away — these in turn do become more certain when the camera moves closer. Now, since the cluster memberships are purely a function of the correlations between features, unless these correlations have converged the submaps will keep changing meaning.

Figure 6.10 shows an example where distant features near the centre of the image



(a) Hierarchy Level 3 of 4



(b) Hierarchy Level 2 of 4

Figure 6.11: Discovering the scene structure of a loopy-browsing sequence in a cafeteria with a window. (Note: left is camera view, right is the 3D map as perceived by MonoSLAM at the particular instant). Despite the apparent proximity in image space of all the visible features in (a), their incoherent motion throughout the sequence due to the difference in parallax has weakened some MI-links, leading to a clear distinction between background and foreground features. In fact, it is evident in (a) that image proximity is overpowered here by consistency of motion. In (b) we see the same frame of this sequence, but we project the clusters from a level deeper in the hierarchy tree. Here, more cluster 'granularity' is observable, indicating regions of even higher MI density. Note here that features out of the window are also visible below the level of the table.

appear moving consistently in image space from the camera pose in (a), and therefore they belong to the same cluster. As the camera moves closer, the extra information has refined the position and uncertainties of these features. As a result, some pairwise correlations have changed causing a broken link right in the middle of the cluster separating the features on the left and right walls. Clearly at this point, the MI-links reflect the fact that the two clusters no longer move coherently and hence they get clustered separately. Figure 6.9 is a full 3D-map view of all the features tracked along the corridor and the clusters formed. The difference with the map formed in the section above is that features appear on both sides of the trajectory here as the camera is facing forward, and also there is overlap between clusters in state space due to their covisibility in image space. In this scenario the ideal submapping solution is less clear and the MI cues are truly helpful to uncover the scene structure.

Loopy Browsing of a Scene with Various Depths

Going back to the tricky cafeteria scenario used to explain our method in the beginning of this chapter, here is an example of our clustering method applied on a real cafeteria sequence, which is indeed a scene with a substantial disparity in depth. Figure 6.11 shows a typical frame of this sequence where both close-by and distant features are co-visible from the same camera pose. Despite these two types of features being covisible and appearing close in image space from a single view, throughout the sequence their difference in depth causes them to move incoherently (in image space). Hence the correlation between the two groups is gradually weakened over time due to their difference in parallax and therefore they get clustered in separate background/foreground submaps, as a human would very easily perceive. We consider that this example highlights the true power of our method, since it demonstrates how the Chow-Liu tree of the MI graph captures consistently the scene structure taking account of a plethora of interfering factors (i.e. camera movement, feature depth, image proximity, etc) at the same time.

Figure 6.12 demonstrates the clusters discovered in two adjacent hierarchy levels which interestingly, seem to agree with the semantic meaning of the features they contain. As a future avenue of investigation, we would like to employ our submap inference method described in this chapter, learn some labels for different parts of the SLAM map.

6.5.3 Quantitative Analysis

As a means of demonstrating the effect of selecting the cluster partitions carefully, we compare the results of Naive decisions to split the map with our Chow-Liu tree based inference method. The comparison is conducted for different levels of the clustering

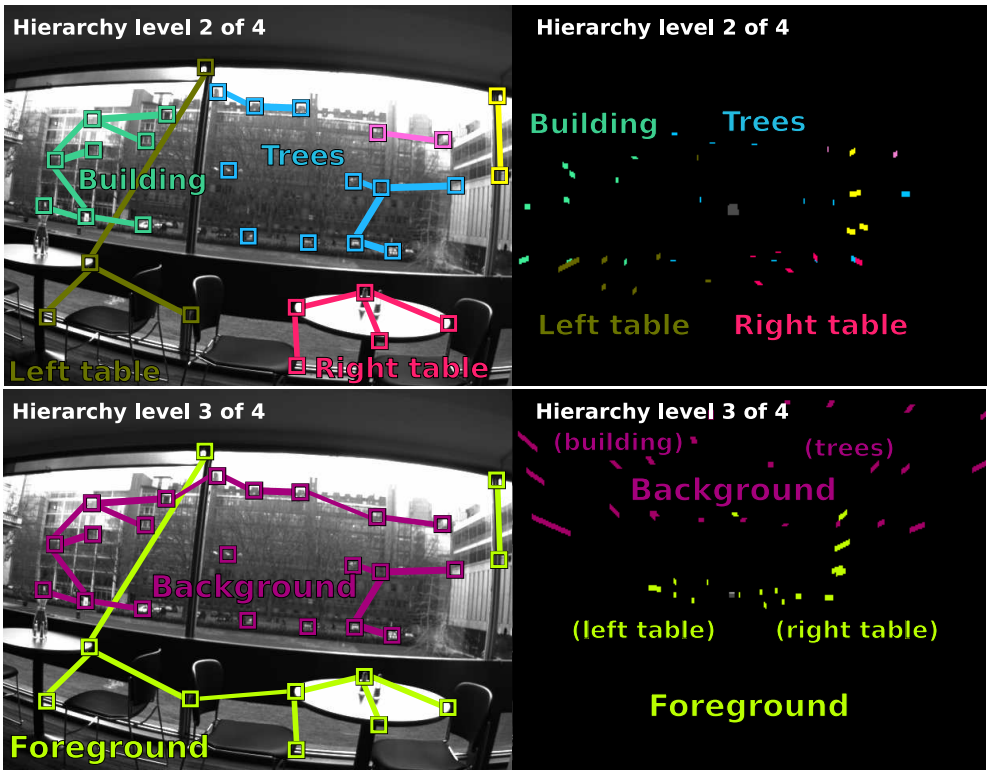


Figure 6.12: Giving a semantic meaning to submaps. Here are two hierarchy levels of clusters discovered while tracking the cafeteria scene. On the top row, the structure of clusters seems to agree with the semantic meaning of the type of features they contain, as suggested in the manually added labels. On the bottom row, we can see how some of these clusters joined together to form bigger sets which again can be labelled as shown. In the future, we would like to use the map-partitioning approach we propose here to *learn* a semantic meaning of clusters fusing appearance information in this process. Note that in the camera view we only project links shared between the features displayed, though more links have been active for the discovery if these clusters.

hierarchy when tracking for 1000 frames in each of the sequences discussed in Section 6.5.2. At the end of each sequence, we record the effect of partitioning the map into an equal number of submaps using both clustering schemes. Note that here we use 'Naive' clustering to refer to the method of splitting the map into regular-sized clusters of features, following the order that they were initialised into the system. The result obtained following either approaches, comprises of clusters within which all MI links between member features are preserved. Any links between features of different clusters are 'cut' except from the strongest one, in order to preserve some relationship between clusters.

In essence, each scheme provides an approximation of the complete, joint distribution of the pairwise MI links. As a means of comparing the correlation structure

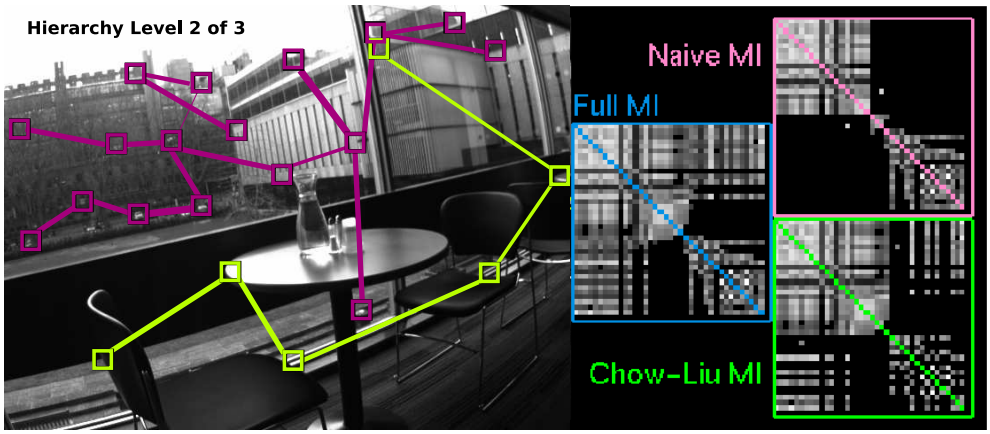
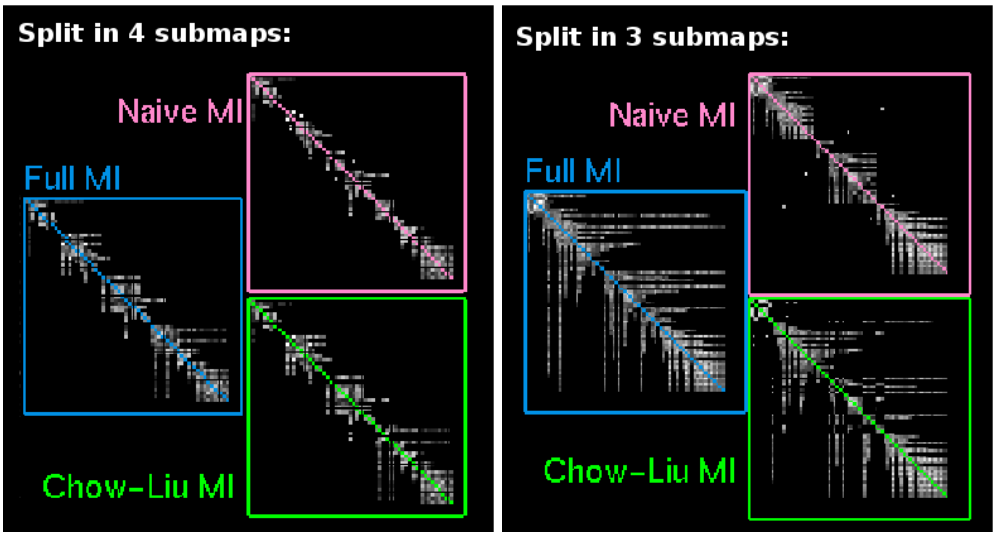


Figure 6.13: Comparing the quality of our clustering method with Naive submapping. On the left is the camera view with the features tracked in this frame (colour depicts cluster membership) and on the right are the matrices of pairwise mutual information links between all features in the map built so far. Brighter pixels denote stronger mutual information links in measurement space. The true matrix of all such links is displayed in the blue box as the 'Full MI'. The other two matrices display approximations to the Full MI according to the submapping scheme used. It is evident that our clustering method preserves far more structure in the distribution of MI links rather than the Naive approach due to the careful selection of the cluster partition. Here, splitting the map in two clusters with the Naive approach we capture 55% of all the links of the Full MI whereas using our Chow-Liu tree based method we capture 81%.

preserved after the application of the two clustering schemes, we superimpose the ratio of the total pairwise MI preserved over the total pairwise MI present before the approximation, by summing the MI links maintained in each case. Table 6.1 shows these ratios as percentages at each level of the hierarchy built using our clustering approach.

For all three sequences, the highest hierarchy level corresponds to all the features lying in a single map preserving all pairwise MI links, resulting to no approximation at all. In the lowest level of the hierarchy each feature comprises a different cluster. However, since we are preserving the strongest MI links between any two clusters, the result is the same, preserving again all of the pairwise MI links present in the initial MI graph. The results for the intermediate levels in the hierarchy however demonstrate the ability of the Chow-Liu based clustering scheme to capture most of the MI structure and, as a consequence, most of the correlation structure present in the full distribution. In all cases of different sequences, the quality of the approximation obtained using the Naive scheme is inferior achieving lower percentages of captured MI. The biggest difference between the two schemes is recorded when splitting in two the map built for the scene with various depths: when submapping with the Naive approach the preserved MI links sum up to 51% of the total MI present in the initial MI graph, whereas our approach preserves 85% of the initial distribution. Figure 6.13 shows an



(a) Sideways exploration (b) Forward exploration

Figure 6.14: The distribution of pairwise MI links before and after sparsification using either Naive or our Chow-Liu tree based clustering methods. In the sideways exploration sequence the camera is constantly visiting new areas resulting to block-like correlation of features within the Full MI matrix as shown in (a). As expected, the Naive approximation to the true distribution captures most of the map structure, however it does provide similar quality of results in the forward exploration example in (b). There, the features initialised early in the sequence remain visible for a long time, progressively building correlations with features seen later on. Evidently, our Chow-Liu based submapping approach captures much more correlation structure than the Naive case.

example frame of the sequence along with a visual representation of the matrix of MI links before and after each approximation. The mostly exploratory nature of the other two sequences results in a sparser distribution of links in the map as demonstrated in Figure 6.14 therefore the losses recorded for both approximation schemes are smaller in comparison to the 'scene with various depths' sequence. Still, even in the case of the sideways exploration where our Chow Liu based approach results into regular-sized clusters resembling the Naive scheme, the careful consideration of where to place the submap partitions results to a better quality approximation as demonstrated in the results of Table 6.1

6.6 Conclusion

The need for the capability of large-scale mapping and denser map representations in modern system pushes algorithms towards more efficient manipulation of large data sets. Motivated by the demand for sparsifying approximations of the SLAM map by current state-of-the-art systems, this chapter analysed the correlations of features in a monocular visual map with the aim of arriving to effective approximations. Apply-

Hierarchy Level	No. Submaps	Pairwise MI captured in Approximation	
		Naive	Chow-Liu-based
Scene with various depths			
4 of 4	1	100 %	100 %
3 of 4	2	50.78 %	84.50 %
2 of 4	7	21.43 %	35.67 %
1 of 4	38	100 %	100 %
Sideways exploration			
4 of 4	1	100 %	100 %
3 of 4	4	74.70 %	93.83 %
2 of 4	10	51.67 %	75.38 %
1 of 4	60	100 %	100 %
Forward exploration			
4 of 4	1	100 %	100 %
3 of 4	3	74.51 %	85.47 %
2 of 4	18	33 %	43.70 %
1 of 4	111	100 %	100 %

Table 6.1: Quantitative comparison of Naive and Chow-Lie based clustering. Results here are quoted as percentages of the total pairwise MI preserved with respect to the initial complete graph of pairwise MI links. The Naive scheme partitions the map into regular sized clusters, where the number of these clusters is chosen to be equal to the number of clusters identified by the Chow-Liu based approach. At the top hierarchy level, all features lie within a single map, preserving 100% of the pairwise MI links. Since we preserve the strongest link between clusters to approximate their relationship, in the bottom level of the hierarchy where each feature lies in a separate cluster, all links are preserved using this scheme, resulting to the full 100% percentage of total MI captured. The results in the intermediate hierarchical levels, demonstrate the power of our Chow-Liu based scheme to capture most of the MI structure in the approximation.

ing our Information Theoretic framework we studied the structure of pairwise feature relationships in the Mutual Information space and our analysis has revealed that the strength of 'bonds' between features is not only a function of co-observability, but also coherency of motion within the image space. Via a straightforward calculation of the Mutual Information of feature measurements followed by temporal averaging and tree construction we can achieved a computationally efficient way of automatically extracting the full, hierarchical correlation structure of a visual map as it is built.

Our experiments show that the resulting hierarchical structure displays characteristics which agree with the expected behaviour in 'obvious' cases such as simple exploration where regular division is appropriate, but also captures much more subtle effects in scenes and camera motions with large ranges of depth or level of detail. Future work involves developing a filter based on this submapping approach for efficient SLAM. Another intriguing prospect is to fuse appearance information along with geometry to refine the definition of submaps as a means of perhaps understanding the semantic nature of each submap.

7

Scalable Feature Matching

Following the maturity of basic real-time monocular SLAM algorithms there is now a rising trend towards more accurate performance. This growing need for bigger and better solutions is fuelled by the increasing hardware capabilities of modern systems which provide the extra computational power much needed when handling tens and hundreds of features per frame. Brute-force matching algorithms like RANSAC constitute the dominant, if not imperative choice at present due to their inherent ability to efficiently manage large data sets in straightforward scenarios. Aiming to bring sequential probabilistic solutions to the same basis of applications, this chapter discusses ways of speeding-up the fully Bayesian framework of Active Matching leading to fast algorithms able to deliver real-time, multi-hypothesis dense matching.

In Chapter 5, we have seen that Active Matching clearly reduces the image processing through guided search, but at the cost of a large overhead in updating the probability distributions determining "where to look next". Here, we present two variations ('FAM' and 'CLAM') which aim to keep the sequential probabilistic search of Active Matching but approximate the inference process to attack computational cost. While the FAM approach has been proposed in [Chli and Davison, 2009b], it is the first time the CLAM methodology is presented, which brings together the ideas evolved in this

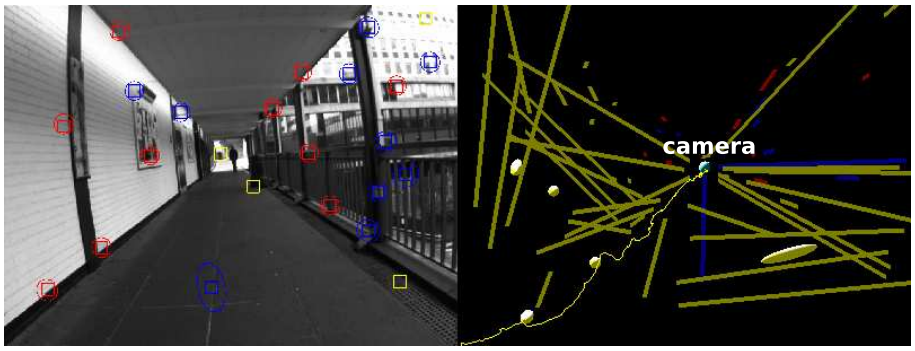
thesis to achieve matching of a hundred features in 125*ms* per frame. To our knowledge is the best that a fully probabilistic method has ever achieved and we hope that this method will scale much further still.

7.1 Introduction

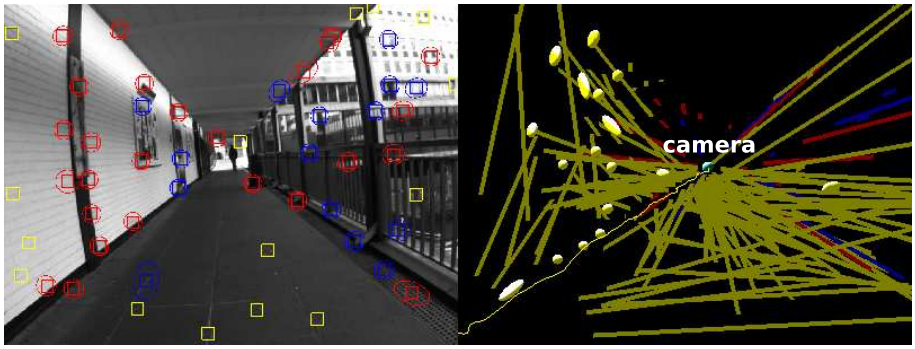
Tracking several features from one frame to the next means that more evidence is available to check matching consensus hypotheses upon, and therefore outlier rejection can potentially be more robust. Moreover, the availability of more candidate measurements means there can be greater reductions in uncertainty, overall leading to greater accuracy of estimates. Indeed, as illustrated in Figure 7.1 tracking a few extra features in a SLAM setup, the estimated camera trajectory becomes significantly smoother implying that consecutive pose estimates become more compatible with each other. However, as pointed out earlier in this thesis, every extra bit of incoming information comes at the cost of processing time. As a result, the vast majority of modern monocular SLAM systems [Davison et al., 2007; Eade and Drummond, 2006b, 2007] are limited to matching a couple of tens of features per frame, restricting their overall performance in accuracy.

The stand-out exception exception is the work of Klein and Murray [2007]. Their approach to tracking and mapping allows searching for far more features per frame than all other similar works (also, these are run in separate, parallel threads leaving more computational time for individual processes). Their coarse-to-fine two-stage tracking consists of a search for a set of map features (around 50) on a low resolution image to compute the new camera pose and then reproject up to a thousand other patches on finer scales to complete matching. While RANSAC [Fischler and Bolles, 1981] is employed during the initialisation of the system, during tracking they do not use any clever data association technique which means that inevitably, outliers are bound to be used into the map. The robustness of this method relies on bundle adjustment to recover consistent pose estimates however, as the authors admit, this approach makes their system prone to repeated structure.

The rising programmability of the graphics processing unit (GPU) often available in modern machines is bound to change the landscape of tracking capabilities. Notable is the work of Sinha et al. [2007] who implemented a GPU-based KLT tracker [Lucas and Kanade, 1981; Tomasi and Kanade, 1991] recording speed-ups of a factor of 20 allowing real-time tracking of a thousand features. However, GPUs are mostly unavailable on embedded devices yet, limiting the applicability of GPU-based implementations. Therefore, the general research interest is still focused on cost-effective CPU-programmable algorithms for more modest computation times, with the prospect



(a) Matching 20 features per frame



(b) Matching 40 features per frame

Figure 7.1: Trajectory smoothness as affected by the number of tracked features per frame. In this example a hand-held camera is tracked while moving along a corridor using 20 features per frame in (a) and 40 in (b). As implied in this superposition of estimated trajectories, more feature measurements provide stronger evidence about the way the camera moves from one position to the next enforcing the robustness of outlier rejection. As a result, consecutive pose estimates are more consistent with each other leading to a smoother trajectory. It is important to note that the rate with which the trajectory becomes smoother is not linear with respect to the number of features; matching 60 features for example would have a much smaller difference with the trajectory in (b).

of faster performance once GPU platforms become widely established.

Dense image matching and consensus resolution using RANSAC [Fischler and Bolles, 1981] is by far the dominant approach at present, having received more than 3000 citations. This popularity is to be accredited to the simplicity of the algorithm and its low requirement for processing resources. With the ability to adjust the maximum number of iterations of the algorithm, the time to completion can be adapted accordingly to achieve a solution within prespecified limits, though the matching outcome might not be the optimal one. However, it was soon realised that the combinatorial explosion of possible inlier choices to draw an initial hypothesis from (subject to voting later) is very wasteful, so recent variants have been proposed to cut down the number of tested hypotheses [Nistér, 2003; Chum and Matas, 2008].

Attempts to fuse some probabilistic predictions in the loop for more informed

decisions throughout matching have led to the branching of another set of semi-probabilistic variants like KALMANSAC [Vedaldi et al., 2005] and guided-MLESAC [Tordoff and Murray, 2005]. Most recently, Civera et al. [2009c] proposed a 1-point RANSAC method which looks for agreement with matches constrained within the predictions' initial innovation regions only shifted towards their EKF-updated means, achieving online matching of around a hundred features at 1 frame per second. Inspired by the exploitation of the EKF correlations in Active Matching [Chli and Davison, 2008a] they constrain the possible locations of feature matches based on the hypothesis generated from the 1-point selection, though to cut down costs they ignore the reduction in the innovation regions of features upon an EKF-update.

Fully probabilistic algorithms, despite eliminating the need to adjust problem-specific thresholds (e.g. number of iterations to perform), are yet to prove their ability to handle hundreds of matches in real-time. Their main hold-back is the cost involved in processing all the available priors (correlations and uncertainty) and input information (matches). RANSAC and variants on the other hand resort to the statistical fairness of randomness to select which hypotheses to test and accept. As a result, these methods gain ground on speed of processing while sacrificing valued cues essential to discover consensus robustly in the presence of a large proportion of outliers.

Each supported by a solid probabilistic framework, Joint Compatibility Branch and Bound (JCBB) [Neira and Tardós, 2001] and Active Matching are bound to make more knowledgeable choices during the resolution of consensus. Chapter 5 has illustrated how both methods successfully reject outliers in the presence of different levels of ambiguity and input priors. Our analysis in Section 5.6 however has revealed that Active Matching scales poorly with the number of features. Suffering similar difficulties (though at smaller scales as suggested in Section 5.6) JCBB, is bound to become "computationally intractable when the number of matches grows near a hundred", quoting Civera et al. [2009c].

More generally, the main distinction between conventional matching techniques and Active Matching is this trade off between spending time to 'think' of the *best* way to exploit the available information versus brute-force exhaustive search for matches first and resolution of consensus later (JCBB included). As demonstrated in the analysis of Section 5.6, carefully selecting where to concentrate processing resources leads to less contaminated data boosting the odds of accepting the correct matching scenario, albeit with the risk of exceeding the real-time limit. The bottleneck in JCBB on the other hand is the growth of the interpretation tree which makes the task of searching for consensus more time consuming. The objective of this chapter is to achieve a better balance between thinking and acting in an attempt to improve the processing speed of Active Matching making it more adaptable to larger data sets, achieving better perfor-

mance than any other fully probabilistic matching technique.

Looking at ways of approximating the full solution of Active Matching, Section 7.2 details an initial attempt leading to the emergence of Fast Active Matching (FAM) as proposed in [Chli and Davison, 2009b]. At every matching step, FAM makes a random preselection of measurement candidates to enter the evaluation of mutual-informations (MIs) stage. Then, following the standard Active Matching procedure of measuring the candidate with the highest MI-efficiency and updating the mixture of Gaussians (MoG), all features are matched until a jointly compatible scenario is achieved. Aiming for a fully probabilistic method, section 7.3 describes how the joint probability distribution of features can be approximated by the Chow-Liu tree in a similar manner as introduced in Chapter 6, to achieve scalable performance with our newly-proposed Chow-Liu Active Matching (CLAM) algorithm. Updates are propagated via messages passed across this tree using the principles of Belief Propagation described in Section 7.4 leading to dramatically reduced prediction and update timings for the shape of the mixture in every matching step. Finally, experiments and results are presented in Section 7.5 for our CLAM algorithm, leading to the conclusions in Section 7.7.

7.2 Fast Active Matching

Chapter 5 discussed the methodology of Active Matching (AM) for establishing pairings between predictions and sensor observations using a mixture of Gaussians to represent the search state at each instant throughout matching. The fully probabilistic maintenance of this mixture permits the use of Information Theoretic measures to guide the decisions of either continuing to explore a promising hypothesis or moving on to check an alternative. Predicting the shape of the mixture after a potential measurement makes AM very selective in the areas it looks for matches and this really pays off in terms of the number of mismatches encountered, aiding the robust resolution of ambiguity. The quality of the accepted scenario in JCBB like in AM, is bound to be the most compatible one irrespective of the proportion of outliers present as long as these outliers do not jointly ‘agree’ in consensus which is highly unlikely. RANSAC-like techniques on the other hand are more prone to data contamination especially when a limit is imposed on the number of iterations where convergence to the optimal solution can be interrupted.

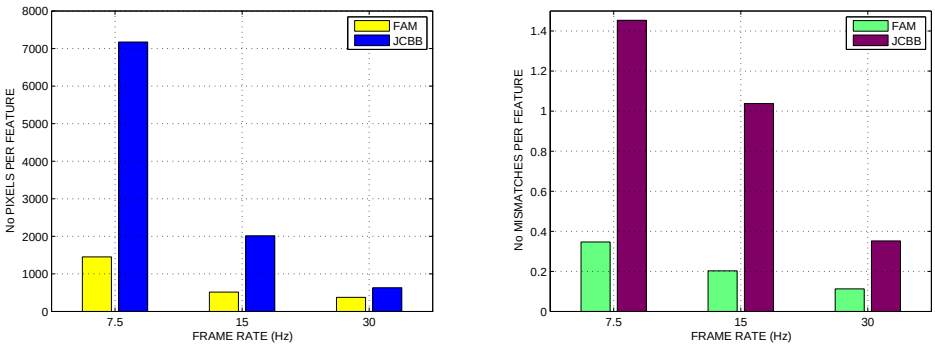
The process of estimating the MI value of *every* potential measurement in AM is the main bottleneck of the algorithm driving performance out of the bounds of the real-time requirements when it comes to matching more than around 20 features per frame. Aiming to tackle this drawback, here we present a variant of the algorithm, first

introduced in Chli and Davison [2009b]. Following the detailed performance analysis of full AM algorithm in Section 5.6, we have come to the conclusion that the first steps of matching are indeed the most crucial in decreasing the variance and resolving the ambiguity in the new frame. The suggestion is therefore to stop evaluating MIs once the maximum such score in the mixture drops below a threshold (i.e. the choice of which feature to measure, becomes unimportant past this point). As demonstrated in Figure 5.15 where the total MI is shown to tail off relatively early during matching, this approximation should have negligible effect on the course of the algorithm.

Although aborting the evaluation of MIs after a certain stage will have a big impact on the computation time, if the goal is to track many features this might not be enough. The biggest source of delays then comes from dealing with big matrices, primarily during the evaluation of MIs (even after the first few steps) but also during the update of the mixture alone. Therefore, to cut down the computational costs further Fast Active Matching (FAM) works by pre-selecting a certain number of candidates to evaluate their MIs rather than evaluating all of them. It is most likely that we will no longer be able to discover the *best* candidate to measure next, but provided that the pre-selected candidates are evenly spread across all Gaussians (one can easily select a certain number of candidates from each Gaussian), the candidate that gets selected for measurement should be a fairly good approximation to the globally optimal choice. Within each Gaussian, the pre-selection is random.

FASTACTIVEMATCHING($\mathbf{G}_0$)

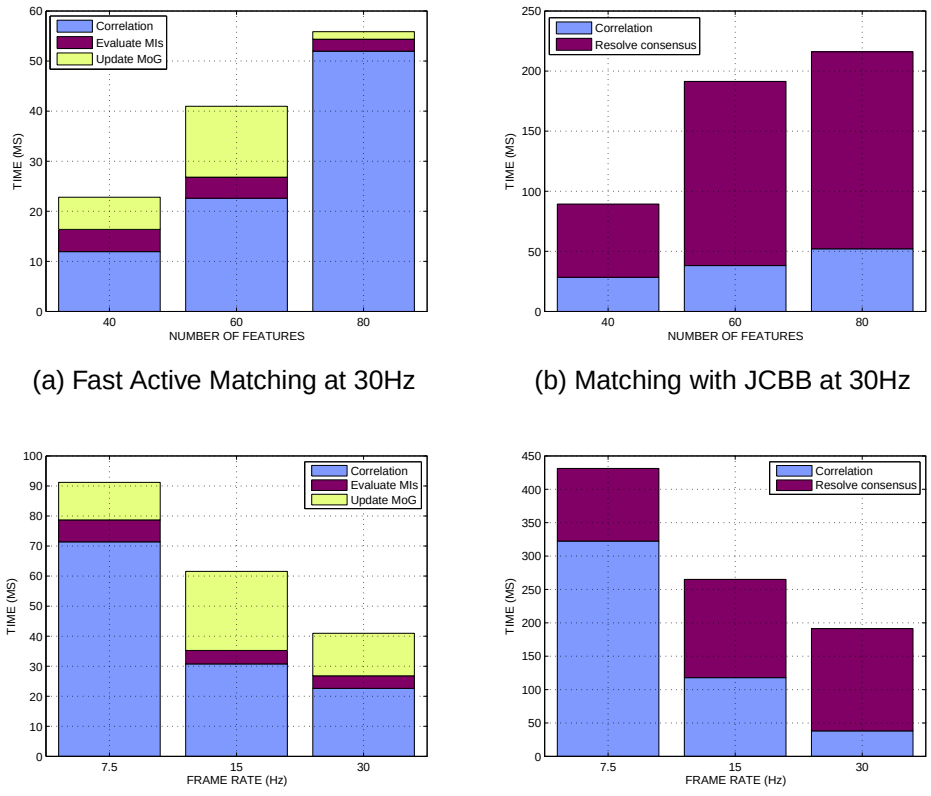
```
1  Mixture = [[1, G_0]]  // each entry in the Mixture is a [weight, Gaussian] tuple
2  preselection = draw_evenly_spread_candidates(N_preselection)
     // 'preselection' consists of {Feature, Gaussian} tuples
3  [f_c,G_c] = get_max_mi_efficiency_candidate(preselection)
4  while (pair_not_yet_measured({F_c, G_c})
5      Matches = measure({F_c, G_c})
6      UPDATEMIXTURE(Mixture, c, Matches)
7      prune_insignificant_gaussians(Mixture)
8      if (MI_max < MI_threshold)
9          {F_c,G_c} = get_max_mi_efficiency_candidate(Mixture)
10     else
11         {F_c,G_c} = pick_next_unmeasured_candidate(Mixture)
12     end if
13 end while
14 G_best = find_most_probable_gaussian(Mixture)
15 return G_best
```



(a) Pixels searched per feature (b) Mismatches per feature

Figure 7.2: Comparing Fast Active Matching (FAM) with JCBB. Correlation is usually a dominant factor in both methods, but using FAM the number of pixels searched is reduced significantly at low frame rates as demonstrated in (a), explaining the superior performance of the algorithm. (b) shows the difference in mismatches encountered per feature matching using either of the two methods. The larger proportion of outliers in JCBB is the main cause of inflated timings to resolve consensus in JCBB. Despite randomising the selection of the candidate to measure, these figures suggest that FAM still reduces search-regions enough to encounter much fewer mismatches than JCBB.

At the stage where enough measurements have been made to decrease the total MI in the mixture sufficiently, any further MI evaluation of candidates is terminated. In the case that matching has reached this state with a Gaussian dominating in the mixture (i.e. having high probability of being the true hypothesis) then this Gaussian will have very low variance left (of the order of measurement uncertainty). The total MI value comprised by the sum of the discrete and continuous MI parts becomes small when both addends are also small. More intuitively, the absence of competing hypotheses means there is no ambiguity in the result which translates into a low discrete-MI value but also the fact that a sufficiently large subset of features have been measured enforcing the dominance of that Gaussian implies that the residual variance in that hypothesis is is also low, hence a small continuous-MI value. In image space, this means that the search regions for any yet unmeasured features in that Gaussian will have very small ellipses so fusing the nearest neighbour matches with respect to the predicted positions of these features, is guaranteed to create a compatible scenario. This is also apparent when thinking in terms of the chi-squared compatibility test used in JCBB: a match within the confidence limit imposed on the distance of the observation from the hypothesised feature position is compatible with the queried hypothesis. Since this confidence limit describes a small ellipse (whose size is defined by the confidence level) centred on the hypothesised feature position, it can easily be visualised that a nearest neighbour match within that circle is bound to be consistent with the rest of the pairings in this hypothesis.



(a) Fast Active Matching at 30Hz

(b) Matching with JCBB at 30Hz

(c) Fast Active Matching 60 features

(d) Matching 60 features with JCBB

Figure 7.3: Comparison of time needed per frame on average to perform Fast Active Matching and JCBB (note the difference in scale). (a) and (b) show the breakdown for FAM and JCBB respectively when tracking at 30Hz; the time spent in evaluation of MIs here is significantly reduced maintaining almost constant overall time adapting to the number of features whereas the resolution of consensus in JCBB deteriorates performance with increasing number of features. In (c) and (d) are the timings for tracking 60 features at different frame rates where it is evident that for lower frame rates, correlation is takes up the most significant part of the computation.

Testing FAM on the same setup and the (512×384) image-sequences used in the analysis of AM in Section 5.6, our aim is to perform a direct comparison with the original AM, gradually increasing the number of tracked features for a constant frame rate and vice versa. Figures 7.2 and 7.3 demonstrate how the proposed refinements can dramatically improve the computation time recorded for AM to the extend FAM achieving faster operation than JCBB. All the results shown in this section have been taken by pre-selecting 15 random candidates evenly spread across all Gaussians present in each matching step. The evaluation of MIs aborts when the total-MI per feature drops below 0.5 bits. Moreover if there is a dominating Gaussian with more than 70% probability of being correct we accept it as the true scenario, fusing the nearest neighbour matches to the remaining unmeasured features. Note that since we prune weak Gaus-

sians throughout matching and renormalise the weights, a Gaussian with probability 70% by the end of the matching is actually a lot more certain.

Observing the histograms in Figure 7.3, it is evident that imposing a limit on the number of candidates evaluated for their MIs before measurement is attempted bounds the otherwise exploding time requirements of that step of the algorithm. The time spent in correlating patches to discover matches allows the observer to see how FAM manages to bound the correlation time when the frame rate is kept lower than JCBB for a constant number of features (note the difference in scale of Figures 7.2(c) and (d)), by dramatically reducing the number of pixels searched. Joint Compatibility on the other hand struggles to resolve the consensus of increasing numbers of features due to the extra mismatches it needs to consider.

A point to note here is that despite using the same test-bed as in Section 5.6, there is some difference in the timings recorded for JCBB. The reason for this is that the tracker is selecting different features to track in each experiment hence there is a difference in the level of ambiguity involved between two different runs using exactly the same settings. However, the comparison of JCBB with AM in Chapter 5 or with FAM here is based on exactly the same input data per frame.

In the future, we can go a step even further and stop measuring features when the MI in the mixture becomes very low. This is expected make use of the fully adaptive nature of AM and can prove particularly beneficial in high-frame rate tracking with a lot of features. In such cases, the uncertainty in the camera pose can be very small leaving little room for ambiguities during matching. Also, the expected improvement in accuracy when incorporating new measurements can soon tail off therefore, aborting matching at that stage translates into reducing redundancy with potentially big savings in computation time.

7.3 Active Matching Using the Chow-Liu Tree

While FAM presented above proves much more cost efficient than the AM of Chapter 5, the introduction of randomness in the loop of search for consensus annihilates the true, fully probabilistic power of the original method imposing the need of application-specific thresholds. Indeed, bounding the number of candidates evaluated for their MI achieves the reduction of the ‘thinking’ process as desired, though this happens at the expense of accuracy: the order by which features are measured is especially important during the first few steps of matching where the variance and ambiguity are expected to reduce most. Due to the fact that once a Gaussian gets pruned from the mixture means that this hypothesis can never be examined again, any such decisions made prematurely or based on false evidence can be fatal. Of course measurement-candidates can be

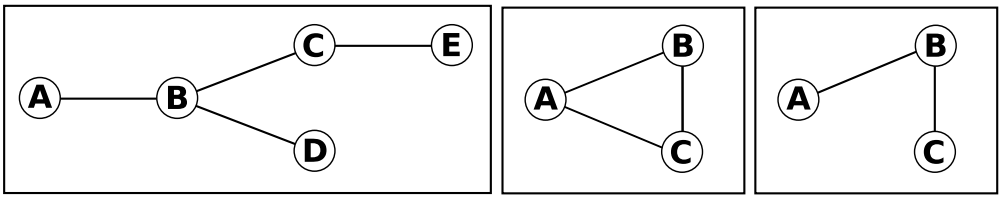
'ranked' according to their false-positive and true-positive rates when evaluating their expected MI values (as explained in Section 5.4.2), but when the all the members in the subset preselected for MI-evaluation are poor then even the best candidate selected for measurement is likely to produce a false result. In essence, this means that while the semi-randomised preselection guarantees better timings it makes the algorithm more prone to data contamination, like most non-probabilistic techniques. This realization, led to the observation of the problem of speeding up matching from a different, fully probabilistic perspective.

Chapter 6 has introduced the notion of the Chow-Liu tree as the least lossy approximation of a joint probability distribution by a singly-connected tree. Inspired by the power of this tree to capture the most representative correlation structure in the scene, here we propose using it to represent the distribution of expected feature locations input in AM, leading to the emergence of our new Chow-Liu Active Matching (CLAM) algorithm. Since all processing in a matching scenario here is done based on the data of a single frame, it is only necessary to build the Chow-Liu tree out of this frame's MI graph. While in Chapter 6 the aim was to expose persistent relationships between features obtained across the sequence and therefore the MI-graph of the whole map was used, here we are only interested in the 'local' relationships of the visible features which will help us discover matching consensus. This immediately means that the time needed to build the Chow-Liu tree is now bounded by the number of features we are aiming to match per frame rather than the cardinality of the whole map. Moreover, using a tree in the context of AM has many attractive properties analysed in the next section allowing efficient processing while maintaining high levels of accuracy.

7.4 Belief Propagation

The intermediate steps in AM involve propagating matching associations (during both evaluation of MIs and update of the mixture) for each feature to the rest of the graph to either obtain an updated or a predicted mixture by fusing an association in the examined hypothesis. In Chapter 5, all such updates reducing the variance of the examined Gaussian were made following the EKF-update rules which involve the use of the innovation covariance matrix since the joint probability distribution of feature positions was a fully-connected, complete graph. Here however, the tree is a much simpler graph-structure allowing updates of order O(n) in the number of features in the worst case as the analysis in this section suggests.

Belief propagation (BP) is a method for efficient and exact inference in a tree structure with a tractable computational cost first proposed by Pearl [1988]. Given observations for a subset of the graph, the algorithm computes marginals for all other



(a) (b) (c)

Figure 7.4: Examples of graphical structures used as a reference for discussing the flow of information in the BP methodology. An observation for node A in (a) can only yield an update to node E following the path A-B-C-E. This is done by sequentially forming the conditional distributions $P(B|A)$, $P(C|A)$ so that $P(E|A)$ can be calculated as outlined in Equation 7.2. The tree in (c) is an approximation to the fully-connected graph in (b).

variables by recursively propagating local messages along the edges of the tree. Appendix A gives a brief outline of the more general sum-product algorithm using parts of the derivation of Bishop [2006] in an attempt to give a deeper understanding of how this methodology works.

7.4.1 CLAM: Estimating Posteriors Upon a Successful Measurement

The key idea behind the BP methodology is the exhaustive exploitation of the tree structure and the properties of d-separation: there is only one path between any two nodes of the tree, hence an update-message originating from an observed node A is bound to update the probability distributions of any intermediate nodes in the way until it reaches its final destination, node E for the tree example of Figure 7.4(a). The conditional distribution of $P(E|A)$ can only be computed if $P(C|A)$ is available which in turn is a function of $P(B|A)$:

$$P(E|A) = P(E|C) \times P(C|A) \tag{7.1}$$

$$= P(E|C) \times [P(C|B) \times P(B|A)] \tag{7.2}$$

Following the BP methodology, this computation can be interpreted in terms of messages: node A passes its observed value to B, which sends off the $P(B|A)$ message towards C (and D if the aim is to discover all conditionals), which in turn can now compute $P(C|A)$ to propagate to E.

Considering now the nodes of this tree as features we aim to match with CLAM, we can say that every node J describes the Gaussian probability distribution of the position of that feature in the image in terms of a mean $\mathbf{j}$ and an associated innovation covariance matrix S_{JJ}. Relying on the Chow-Liu tree to pick out the edges of the MI-graph that approximate it best, a link between nodes J and K is described by a cross-

covariance block S_{JK} and its transpose S_{KJ} referring to the relationships of $P(J|K)$ and $P(K|J)$, respectively. At the beginning of matching, these parameters are set to the respective values input into the feature matcher. Strictly speaking, when approximating the complete, joint distribution of features with a tree the uncertainty encoded in each S_{JJ} increases to reflect the effect of the approximation, but here we can directly read out these values from the innovation covariance matrix since the confidence of the tracker (providing these values) is not affected by this approximation.

Using this notation we will evaluate each of the probabilities involved in Equation 7.2 for the special case where all distributions are Gaussians (as used in AM) to derive the expressions for the BP-messages passed along the branches of our Chow-Liu tree upon a successful measurement. However, we first analyse the basic rules of conditioning in multivariate Gaussian distributions to derive expressions for posteriors in tree-structured distributions, from first principles.

Mathematical Derivation of Conditionals in Gaussian Tree Structures

Let us consider the simple graph of Figure 7.4(b). The joint distribution of $P(A,B,C)$ can be expressed as

$$P(A,B,C) = P(A) \times P(B|A) \times P(C|A,B)\ . \tag{7.3}$$

Since we are interested in Gaussian distributions of variables, let this distribution be described by a mean vector $\hat{\mathbf{x}}$ stacking all vectors $\hat{\mathbf{a}}$, $\hat{\mathbf{b}}$, and $\hat{\mathbf{c}}$, and a corresponding covariance matrix S. Explicitly,

$$p(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^{3n}\ |S|}}\ e^{-\frac{1}{2}(\mathbf{x}-\hat{\mathbf{x}})^\top S^{-1}(\mathbf{x}-\hat{\mathbf{x}})}\ , \tag{7.4}$$

where $3n$ is the dimension of $\mathbf{x}$, n being the dimension of each of $\hat{\mathbf{a}}$, $\hat{\mathbf{b}}$, and $\hat{\mathbf{c}}$. If variable B is now observed with a value $\mathbf{b}$, we can obtain expressions for the posterior mean and covariance of $P(A,C|B)$ by considering the quotient of distributions $P(A,B,C)$ over $P(B)$ and rearranging in the form of Equation 7.4. The covariance of this conditional can be described using Schur's complement which briefly is the result of applying Gaussian elimination to the rows and columns corresponding to B.

Schur's complement

Let T be partitioned as $T = \begin{bmatrix} P & Q \\ R & U \end{bmatrix}$. Given that matrices T and P are square and nonsingular, then the Schur complement of P in T is $(U - RP^{-1}Q)$.

Denoting the parameters of the conditional distribution on B with the subscript '$_{|B}$' we reorder and partition S so that matrix P of the inset above corresponds to the innovation covariance of B, S_{BB}. Applying Schur's complement we arrive at the posterior innovation covariance of the distribution (conditioned on the observed value of B):

$$\begin{aligned}\begin{bmatrix} S_{AA|B} & S_{AC|B} \\ S_{CA|B} & S_{CC|B} \end{bmatrix} &= \begin{bmatrix} S_{AA} & S_{AC} \\ S_{CA} & S_{CC} \end{bmatrix} - \begin{pmatrix} S_{AB} \\ S_{CB} \end{pmatrix} S_{BB}^{-1} \begin{pmatrix} S_{BA} & S_{BC} \end{pmatrix} \\ &= \begin{bmatrix} S_{AA} - S_{AB}S_{BB}^{-1}S_{BA} & S_{AC} - S_{AB}S_{BB}^{-1}S_{BC} \\ S_{CA} - S_{CB}S_{BB}^{-1}S_{BA} & S_{CC} - S_{CB}S_{BB}^{-1}S_{BC} \end{bmatrix} . \end{aligned} \tag{7.5}$$

The expressions along the diagonal demonstrate how the individual variances of nodes A and C get reduced by the measurement of B, whereas the off-diagonal blocks show that the correlation-link between nodes A and C also gets affected by this measurement. The means vector can also be shown to obey:

$$\begin{pmatrix} \hat{\mathbf{a}}_{|B} \\ \hat{\mathbf{c}}_{|B} \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{a}} \\ \hat{\mathbf{c}} \end{pmatrix} + \begin{pmatrix} S_{AB} \\ S_{CB} \end{pmatrix} S_{BB}^{-1}(\mathbf{b} - \hat{\mathbf{b}}) = \begin{pmatrix} \hat{\mathbf{a}} + S_{AB}S_{BB}^{-1}(\mathbf{b} - \hat{\mathbf{b}}) \\ \hat{\mathbf{c}} + S_{CB}S_{BB}^{-1}(\mathbf{b} - \hat{\mathbf{b}}) \end{pmatrix} . \tag{7.6}$$

These are the standard update equations used to evaluate the parameters of posteriors in Gaussian joint probability distributions. If however we now consider approximating the fully connected distribution of Figure 7.4(b) with the tree of Figure 7.4(c), evaluating the terms in these expressions is no longer obvious. Cutting the link between nodes A and C means that we no longer have explicit values for the cross-correlation terms S_{AC} and S_{CA}. The variables are still however correlated and in fact d-separated by B: any information originating from node A has to pass through B to reach C and vice versa. Once however B gets measured, then nodes A and C become completely independent and uncorrelated. Therefore the off-diagonal blocks in equation 7.5 should be equal to zero to enforce this independence:

$$S_{AC|B} = 0 \Rightarrow S_{AC} - S_{AB}S_{BB}^{-1}S_{BC} = 0 \Rightarrow S_{AC} = S_{AB}S_{BB}^{-1}S_{BC} . \tag{7.7}$$

Similarly, we can find an expression for S_{CA} or any other cross-correlation block in longer tree paths in terms of known matrix blocks, using the same principles. Attempting to give an intuitive understanding of the expression in equation 7.7 above, we can see how the information leaving node A has to pass through the link AB (using the term S_{AB}) and be converted to the frame of reference of B (simulating the effect of this information on B's variance with S_{BB}^{-1}) so that it can be passed on to C (via S_{BC}). Supposing now that we have further obtained a measurement for node A, then applying

Schur's complement again the updated variance of C will be:

$$S_{CC|A,B} = S_{CC|B} - S_{CA|B} S_{AA|B}^{-1} S_{AC|B} \ . \tag{7.8}$$

This confirms that if the initial joint distribution is represented by the tree of Figure 7.4(c) then the measurement of A has no further impact on C (given that B had already been measured) since $S_{AC|B} = S_{CA|B} = 0$. Therefore, cutting the direct link AC is equivalent to setting $S_{CC|A,B} \approx S_{CC|B}$, which reveals the effect of a tree approximation of a fully-connected graph: the reduction in variance upon a measurement in the tree approximated structure is bound to be less than or equal to the reduction in the true, fully-connected distribution.

Evaluation of Messages Passed In Gaussian Tree Structures

Having obtained expressions for the parameters of conditional distributions we now go back to our initial example of Figure 7.4(a), aiming to evaluate the messages passed along the links of that tree upon a successful measurement of a node within the context of CLAM. Supposing that an image-search for feature A has yielded the match $\mathbf{a}$, the new distribution conditioned on this measurement will have the position of A assigned to this match with its innovation covariance set to measurement noise. From then on, the parameters of any neighbours of A can be directly updated using update rules of equations 7.5 and 7.6. Since here B is the only neighbour of A:

$$\hat{\mathbf{b}}_{|A} = \hat{\mathbf{b}} + S_{BA} S_{AA}^{-1} (\mathbf{a} - \hat{\mathbf{a}}) \tag{7.9}$$

$$S_{BB|A} = S_{BB} - S_{BA} S_{AA}^{-1} S_{AB} \ . \tag{7.10}$$

Given that node B is now updated it can disseminate information regarding the measurement of A to all of its neighbours (excluding A). Considering node C and using the equation 7.7 to express S_{AC} and S_{CA},

$$\hat{\mathbf{c}}_{|A} = \hat{\mathbf{c}} + S_{CB} S_{BB}^{-1} \left(S_{BA} S_{AA}^{-1} (\mathbf{a} - \hat{\mathbf{a}}) \right) \tag{7.11}$$

$$S_{CC|A} = S_{CC} - S_{CB} S_{BB}^{-1} \left(S_{BA} S_{AA}^{-1} S_{AB} \right) S_{BB}^{-1} S_{BC} \ . \tag{7.12}$$

Comparing the updated parameters of C (in Equations 7.11, 7.12) with the updated parameters of B (in Equations 7.9, 7.10), it is evident that the influence on both the mean and variance of C is a function of the influence the measurement had on B, indicating the recursion which is to become more evident as we evaluate expressions for variables deeper in the tree. In a similar manner, the updated parameters of node D are:

$$\hat{\mathbf{d}}_{|A} = \hat{\mathbf{d}} + S_{DB}S_{BB}^{-1}\left(S_{BA}S_{AA}^{-1}(\mathbf{a}-\hat{\mathbf{a}})\right) \quad (7.13)$$

$$S_{DD|A} = S_{DD} - S_{DB}S_{BB}^{-1}\left(S_{BA}S_{AA}^{-1}S_{AB}\right)S_{BB}^{-1}S_{BD}\ . \quad (7.14)$$

Lastly, the updated node C can now issue a message to update E. Using the update rules of equation 7.5 and the rationale used to derive equation 7.7:

$$S_{EE|A} = S_{EE} - S_{EA}S_{AA}^{-1}S_{AE} \quad (7.15)$$

$$= S_{EE} - \left(S_{EC}S_{CC}^{-1}S_{CA}\right)S_{AA}^{-1}\left(S_{AC}S_{CC}^{-1}S_{CE}\right) \quad (7.16)$$

$$= S_{EE} - \left(S_{EC}S_{CC}^{-1}\left(S_{CB}S_{BB}^{-1}S_{BA}\right)\right)S_{AA}^{-1}\left(\left(S_{AB}S_{BB}^{-1}S_{BC}\right)S_{CC}^{-1}S_{CE}\right)\ . \quad (7.17)$$

In a similar manner, it can be shown that:

$$\hat{\mathbf{e}}_{|A} = \hat{\mathbf{e}} + S_{EC}S_{CC}^{-1}\left(S_{CB}S_{BB}^{-1}\left(S_{BA}S_{AA}^{-1}(\mathbf{a}-\hat{\mathbf{a}})\right)\right)\ . \quad (7.18)$$

Based on this brief derivation of the expressions used to update the nodes of the tree in Figure 7.4(a) upon an observation of a node, we can now arrive at more general BP rules for updating and message-passing in trees.

BP Messages and Update Rules in Gaussian Tree Structures

Given an observation of some node F, node K lying on the same tree gets updated upon the receipt of mean and covariance messages ($\mathbf{m}_{\text{msg } J\to K}$ and $S_{\text{msg } J\to K}$ respectively) from its direct neighbour, node J by:

$$\hat{\mathbf{k}}_{|F} = \hat{\mathbf{k}} + \Delta\hat{\mathbf{k}} \quad \text{, where} \quad \Delta\hat{\mathbf{k}} = S_{KJ}\mathbf{m}_{\text{msg } J\to K} \quad (7.19)$$

$$S_{KK|F} = S_{KK} - \Delta S_{KK} \quad \text{, where} \quad \Delta S_{KK} = S_{KJ}S_{\text{msg } J\to K}S_{JK}\ . \quad (7.20)$$

Once updated, node K can issue similar messages as below for all its other neighbours M such that $M \neq J$:

$$\mathbf{m}_{\text{msg } K\to M} = S_{KK}^{-1}\Delta\hat{\mathbf{k}} \quad (7.21)$$

$$S_{\text{msg } K\to M} = S_{KK}^{-1}\Delta S_{KK}S_{KK}^{-1}\ . \quad (7.22)$$

Starting off with $\Delta\hat{\mathbf{a}} = \mathbf{a} - \hat{\mathbf{a}}$ and $\Delta S_{AA} = S_{AA}$ and passing messages formed as in Equations 7.21 and 7.22 along the links of the tree of Figure 7.4(a), we can derive the same expressions for conditional distributions of all nodes as above in a recursive manner.

If only one node is ever observed in the tree, then the update rules of Equations 7.19, 7.20 and the messages in 7.21, 7.22 are enough to get expressions for the con-

ditioned marginals of the variables in the tree. If however another variable is further observed then our expressions will no longer be valid because we have not taken into account the updated values of the conditioned cross-covariance terms (corresponding to the correlation-links between variables) which are indeed affected upon a measurement as suggested in our mathematical derivation in Equation 7.5. Since in our matching paradigm we are obtaining observations for features sequentially, we also need to carry link messages (as well as the mean and covariance messages) to update these cross-covariance terms. Using the same notation and rationale as before, when node K receives BP messages from J and gets updated, then their link JK is also updated by:

$$S_{KJ|F} = S_{KJ}\, L_{\text{msg } J\to K} \tag{7.23}$$

$$S_{JK|F} = L_{\text{msg } J\to K}{}^{\top} S_{JK}\ . \tag{7.24}$$

Then, along with outgoing messages for the mean and variance, node K also issues link messages for its neighbours M:

$$L_{\text{msg } K\to L} = I - S_{KK}^{-1}\Delta S_{KK}\ , \tag{7.25}$$

where I in the above expression refers to the identity matrix.

Given that these updates can be done recursively, we no longer need to carry around the big innovation covariance matrix in every new hypothesis formed to represent the search state of the matching procedure (as was necessary in AM). Therefore, every Gaussian in CLAM now consists of a linked-list of the means and innovation covariance sub-blocks representing the marginal distribution of each predicted feature position, along with the off-diagonal sub-blocks of the covariance matrix corresponding to the links preserved in the tree structure.

Once the measurement of a feature (node) yields a set of matches, the new Gaussians spawned from it (each to represent the hypotheses one of these matches is the true feature) will inherit a copy of the linked-list of the Gaussian just measured, only isolating the measured feature so that any links connected to it will be cut (since there can be no more information passed though them once the feature is observed). This means that over the course of matching, the initial tree spanning all features present in the frame is progressively thinned depending on the matching results of features. Hence in the worst case, an update upon a measurement is of order O(n) in the number of nodes.

7.4.2 CLAM: Computing MIs

Following the derivation of the update rules and message passing in CLAM, here we derive expressions for efficiently computing the Mutual Information that each mea-

surement is expected to provide. Once again, the d-separation properties of the tree allow for short-cuts in this otherwise explosive computation. Considering again the tree example of Figure 7.4(a) we aim to find expressions for the MI that a node is predicted to give aiming to gain a general understanding to the flow of information in the tree so that we can expand this to a more general tree structure.

Using the basic rules of Mutual Information introduced in Chapter 4, let us consider the information value of a supposed measurement of node A:

$$\begin{aligned} I(B,C,D,E;A) &= H(B,C,D,E) - H(B,C,D,E|A) \\ &= \int_{A,\ldots,E} P(A,B,C,D,E) \log_2 \frac{P(B,C,D,E|A)}{P(B,C,D,E)} d(A,\ldots,E) \,. \end{aligned} \tag{7.26}$$

The division inside the logarithm can be simplified using the factorised expression for probability distribution of the tree:

$$P(A,B,C,D,E) = P(A) \times P(B|A) \times P(C|B) \times P(D|B) \times P(E|C) \,. \tag{7.27}$$

Therefore, using Bayes' rule and equation 7.27,

$$\begin{aligned} \frac{P(B,C,D,E|A)}{P(B,C,D,E)} &= \frac{P(A,B,C,D,E)}{P(A)P(B,C,D,E)} \\ &= \frac{P(A)P(B|A)P(C|B)P(D|B)P(E|C)}{P(A)P(B)P(C|B)P(D|B)P(E|C)} \\ &= \frac{P(B|A)}{P(B)} \,. \end{aligned} \tag{7.28}$$

Hence, substituting now Equation 7.28 into 7.26:

$$\begin{aligned} I(B,C,D,E;A) &= \int_{A,\ldots,E} P(A,B,C,D,E) \log_2 \frac{P(B|A)}{P(B)} d(A,\ldots,E) \\ &= \int_{A,B} P(A,B) \log_2 \frac{P(B|A)}{P(B)} d(A,B) \equiv I(B;A) \,. \end{aligned} \tag{7.29}$$

In words, the MI that node A is predicted to provide to the rest of the tree is equivalent to the MI it is expected to provide to node B alone. Considering instead a possible measurement of node B,

$$\begin{aligned} I(A,C,D,E;B) &= H(A,C,D,E) - H(A,C,D,E|B) \\ &= \int_{A,\ldots,E} P(A,B,C,D,E) \log_2 \frac{P(A,C,D,E|B)}{P(A,C,D,E)} d(A,\ldots,E) \,. \end{aligned} \tag{7.30}$$

The division inside the logarithm can again be simplified using a different factorisation

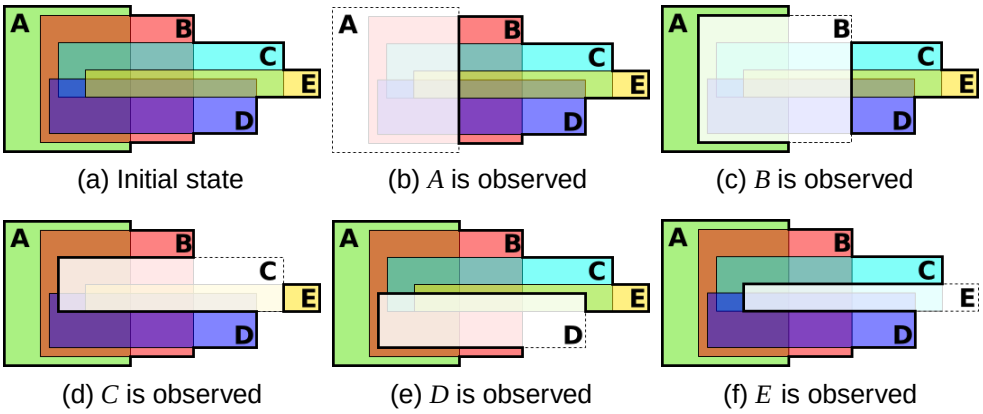


Figure 7.5: Visualising the entropies of variables in the tree of Figure 7.4(a) and their pairwise relationships. Supposing each variable has an entropy proportional to the area of the rectangle used to represent it, then the total entropy of the distribution $P(A,B,C,D,E)$ is equivalent to the area enclosed in the outer, bold border of (a). Figures (b) - (f) depict the effect that each possible measurement has on the marginal and conditional entropies of the distribution.

for the tree distribution:

$$\begin{aligned}\frac{P(A,C,D,E|B)}{P(A,C,D,E)} &= \frac{P(A,B,C,D,E)}{P(B)P(A,C,D,E)} \\ &= \frac{P(B)P(A|B)P(C|B)P(D|B)P(E|C)}{P(B)\int_B P(B)P(A|B)P(C|B)P(D|B)P(E|C)dB} \\ &= \frac{P(A,C,D|B)}{P(A,C,D)} \ . \end{aligned} \tag{7.31}$$

Substituting now back in equation 7.30:

$$\begin{aligned} I(A,C,D,E;B) &= \int_{A,\ldots,E} P(A,B,C,D,E)\log_2\frac{P(A,C,D|B)}{P(A,C,D)}d(A,\ldots,E) \\ &= \int_{A,B,C,D} P(A,B,C,D)\log_2\frac{P(A,C,D|B)}{P(A,C,D)}d(A,B,C,D) \\ &\equiv I(A,C,D;B) \ . \end{aligned} \tag{7.32}$$

Equation 7.32 states that the MI that node B is predicted to give to the rest of the tree is equivalent to the MI it is predicted to give to its neighbours alone, analogously to the conclusion for the MI of node A. Deriving expressions for every other node in a similar manner and also considering the case of more complex tree structures, it can be shown that this observation is indeed general: the information gain that a node is predicted to give to the rest of the variables in a tree is equivalent to the MI it is expected to give to its immediate neighbours alone. Considering the way updates are propagated across the tree upon the observation of one of the nodes, one can under-

stand that a measurement for a particular node K will cause progressive, breadth-first updates, first updating its immediate neighbours, followed by updates of their neighbours and so on. Therefore, any information flowing from this node K reaching an indirectly linked node F in the tree is bound to be less than or equal to the information passed on to K's immediate neighbour.

In an attempt to give a deeper understanding of information flow in a tree, Figure 7.5 depicts a visualisation of the information shared between the variables of the tree of Figure 7.4(a). All relationships between variables can be derived in the same way as above, though we chose to illustrate them visually to give a better intuition. On the assumption that a measurement of a given node causes zero or positive reduction to the variances of the rest of the variables, we can safely represent marginal and conditional entropy in terms of area [Mackay, 2003]. Without loss of generality, the entropy of each variable in this tree is depicted by the area enclosed in the corresponding rectangle in Figure 7.5(a). There is of course overlap between these rectangles representing the shared information content between nodes. The arrangement of these overlaps however is very significant because it is chosen to reflect the effect of each measurement to the rest of the distribution. Given that the area enclosed inside the outline of this composition of rectangles (shown with a bolder line), corresponds to the total entropy of the distribution, we now consider a supposed observation of each of the variables and discuss their effect on every other conditional distribution:

- A **is observed:** the posterior entropies conditioned on the observation of A can be visualised as cutting A's marginal entropy out of the composition as shown in (b). As expected, all other marginal and conditional entropies are reduced following this measurement, including the total entropy in the tree.

- B **is observed:** the updated distribution conditioned on this observation as shown in Figure 7.5(c) reflects the properties of d-separation breaking the tree in two so that node A becomes completely disjoint from the set $\{C,D,E\}$ since B was the pathway of A to propagate any information to the rest of the tree and vice versa.

- C **is observed:** this is a similar case as above, only E is now the node completely separated out from the rest of the variables as suggested in 7.5(d).

- D **is observed:** node D being a leaf in the tree, it did not d-separate any sets of variables in the first place so the remaining variables can still share information between them as suggested in Figure 7.5(e). Considering the posterior tree structure, a measurement for node D therefore makes redundant only links coming out of it, and finally:

- E **is observed:** again, a measurement of this leaf node preserves some correlation between the rest of the variables as shown in 7.5(f), though their correlation is reduced depending on how correlated they were with node E in the first place.

The key result of this analysis is that:

> The MI gain that a node is predicted to provide to the rest of the distribution upon successful measurement is **equivalent** to the MI it is predicted to give to its neighbours alone.

This follows from the fact that the information content that this node has in common with the rest of the variables is exactly equal to the information it shares with its immediate neighbours. As a result, the evaluation of MIs in CLAM becomes trivial: the costly manipulation of the full covariance matrix (as was necessary in AM) gets replaced by a few fast message-passing operations within the sub-tree spanning the candidate node and its immediate neighbours only. Moreover, the tree representation allows for further short-cuts in evaluating the nodes' MIs: due to the fact that upon successful measurements some links become redundant and the tree breaks into smaller sub-trees, not only update operations are then confined within the sub-tree a measurement is made, but also the MI values of any features not been updated within a particular matching step can be carried forward to the next one since they will still be valid.

7.5 Experimental Results

In order to demonstrate the power of this fully probabilistic adaptation of Active Matching this section discusses assessment experiments performed using the MonoSLAM system. The quality of matching is tightly coupled with the quality of features selected for tracking used as discussed earlier in this thesis which in turn depends on the feature detector and descriptor used. In order to remain consistent with the rest of the results presented in this thesis, here we use Shi-Tomasi features [Shi and Tomasi, 1994] saving the 11×11 patch surrounding the detected peak as their descriptor. However, in order to achieve both high numbers of Shi-Tomasi peaks in the image while also achieving patches of acceptable quality to track, we have increased the resolution of images used to 1024×768. This indeed makes searching for matches more costly since more pixels ($4\times$ more) correspond to the same uncertainty region for each feature, meaning that more normalised cross-correlation operations have to be performed in order to discover matches. However, in case a different detector-descriptor combination can provide high numbers of features in a lower resolution image, then all the timings presented in this section (for all the superimposed matching algorithms) will be reduced by a significant amount, improving the overall speed of performance.

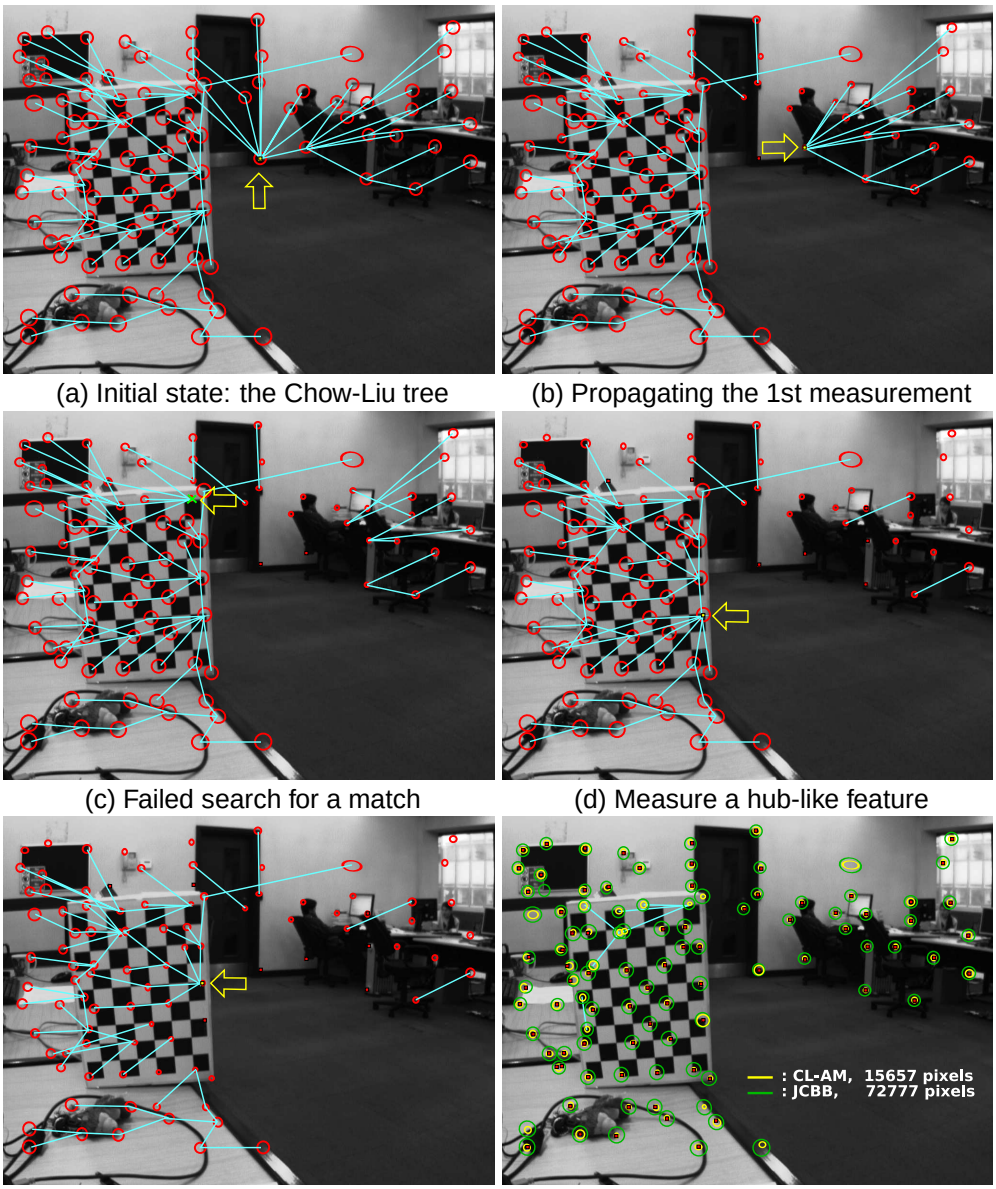
7.5.1 CLAM: A Step-By-Step Example

Our CLAM algorithm, which is mainly based on the AM algorithm of Chapter 5, follows the same principles but as described in detail previously in this chapter, the difference is the approximation of the joint distribution of visible features with the Chow-Liu tree factorisation as introduced in Chapter 6. This tree factorisation allows speed-ups in the time needed for both updating the tree and evaluating the MI that each feature is predicted to give.

Figure 7.6 illustrates a step-by-step example of the way Chow-Liu Active Matching works within a given frame. The tracker provides the matcher with the means vector and the dense innovation covariance matrix describing the joint distribution of the features predicted to be visible in the new image. Computing all the pairwise MI links between features based on the innovation covariance entries, we can form the complete MI graph which is then sparsified so that the links forming the Chow-Liu tree can be identified. Figure 7.6(a) shows this tree in a typical matching example while tracking in an office scene. As discussed in Chapter 6, the Chow-Liu tree has the power of capturing the most representative correlation structure in the scene, preserving links between strongly correlated features. As evident in the example of Figure 7.6(a), features that have been tracked consistently and moved coherently throughout the sequence share strong correlations hence they lie close to each other in the tree space (e.g. the features on the checker-board), whereas less correlated features lie more 'steps' away in the tree (e.g. features on the left part of the image with features on the right).

According to the correlation structure preserved in the Chow-Liu tree and the predicted MIs of each of feature, the hub-like feature in the middle of the image is estimated to provide the biggest MI gain per pixel searched. Propagating the successful measurement result of this feature along the branches of the tree causes reductions in uncertainty of different magnitude to all other features in the image: the features directly connected to it become more certain than the rest due to the fact that by construction of the tree, these features share stronger correlations with the one measured.

As illustrated in Figure 7.6(b), conditioning the distribution on the measurement result of (a) not only has an effect on the variance of each feature but also the tree structure itself: since measuring the feature has pinned down its exact position to the obtained match, then all information that could be passed though this node has already been disseminated to its neighbours upon the update of the distribution, so these links have now become redundant. Cutting these links as shown in 7.6(b) the problem of matching is now essentially partitioned in sub-trees corresponding to different parts of the image, which are highly intercorrelated. Yet another successful measurement in (b) causes further disappearance of links in (c) and a variance reduction only for the



(a) Initial state: the Chow-Liu tree (b) Propagating the 1st measurement

(c) Failed search for a match (d) Measure a hub-like feature

(e) Updated state (f) Superposition of searched areas

Figure 7.6: Matching using CLAM within a frame while tracking an office scene. In (a) is the initial state of matching, illustrating the variance of each of the 97 features and their joint distribution as the Chow-Liu tree. The arrow points at the feature predicted to provide the most MI per pixel searched. Propagating the successful measurement of (a) in (b) cuts any links directly connected to it and reduces the variance of all other features. The successful measurement in (b) yields updates in (c) for that subtree only. In (c) however, the selected feature yields no match preserving the same search state and tree structure. The match in (d) causes new divisions into subtrees in (e), partitioning the problem of matching further. Finally (f) demonstrates that CLAM searched almost 5 times fewer pixels than JCBB in this frame.

features that were part of the subtree where the measured feature was lying. In (c) however, the feature searched for did not yield a successful match which causes neither reduction in uncertainty nor changes in the tree structure. As detailed in Chapter 5, a failed match has an effect only on the distribution of weights in the mixture. The matching scenario of Figure 7.6, however, has been chosen to be straightforward for the sake of clarity: since matches occur mostly right at the predicted locations of features, each new Gaussian spawned upon a successful match inherits most of its parent's weight. As a result, the parent (measured) Gaussian gets pruned out of the mixture (this is true for the search state of all images displayed, however more Gaussians have been live during the course of matching in this frame). In effect, the mixture of Gaussians in (c) contains a single live Gaussian hence a failed match has no effect on the search state (aside from removing this ellipse from the candidates for measurement).

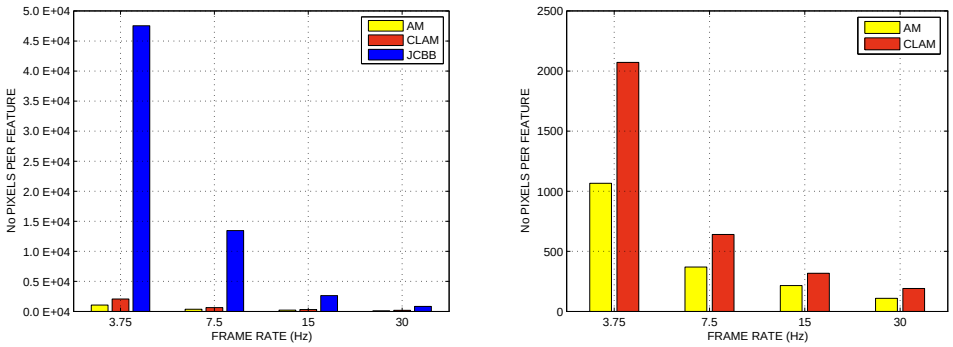
Figure 7.6(d) shows how the successful measurement of another hub-like feature yields great uncertainty reduction and breaks the problem of matching down even more in (e). Finally, Figure (f) illustrates a superposition of areas searched for until all features have been matched using CLAM as opposed to conventional 'get matches first, resolve later' techniques like JCBB. More specifically, in this example CLAM searched almost 5 times fewer pixels than JCBB. A minor comment is that some features have not returned any successful matches, so any of these features sharing a link in the initial Chow-Liu tree, they continue to share it until the end of matching.

7.5.2 Sequence Results

Testing the capabilities of the CLAM algorithm, we generated a test-bed of sequences to span different frame rates by progressively subsampling an office sequence captured at $30Hz$. Since the resolution of all captured and generated sequences is 1024×768 not only more pixels have to be searched per feature as mentioned before but also more mismatches are likely to occur rendering these sequences significantly hard to track. A point to note for the figures presented in this section is that the comparisons made between the different matching techniques used are based on runs using the exact same tracking settings, however the choices made in each run for which features to track can differ though this should have a negligible effect on the recorded results.

Area Searched for Matches

Tracking at low frame rates, the 'blind' interval in between frames becomes larger allowing for more unpredictable behaviour hence there is more uncertainty in the estimated position of the camera and each feature measured. As a result, the search region for each feature increases with decreasing frame rate as suggested in Figure 7.7. Chapter 5 discussed how AM outperforms JCBB when tracking at lower frame

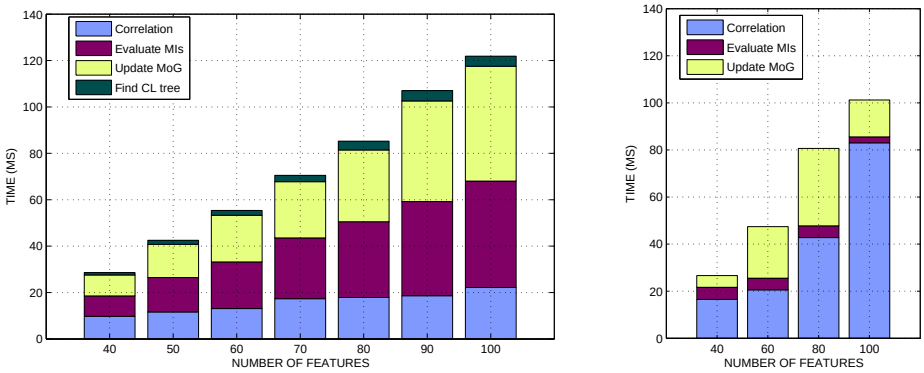


(a) Searched area per feature (b) Isolating AM-based methods

Figure 7.7: Comparison of the number of pixels searched per feature when tracking the office sequence. In (a) is a comparison to scale of the numbers of pixels searched when using either the original AM, CLAM or JCBB. It is evident that both AM-based methods search significantly fewer pixels so for the sake of clarity (b) illustrates the same data only for these two methods. As expected, as CLAM is an approximation of AM it exhibits less reduction in search-areas however, the difference becomes less obvious in higher frame rates. In essence, the variance reduction to scale as shown in (a) suggests that approximating the joint distribution of features with the Chow-Liu tree indeed captures the most significant correlation structure.

rates since it achieves large reductions in search regions avoiding further confusion of the matcher with unnecessary mismatches. Here, Figure 7.7(a) demonstrates that despite CLAM being an approximation of AM it still reduces the searched areas significantly with respect to the initial search regions (necessary to search when matching with any conventional matching method like RANSAC).

During the derivation of the BP messages passed along the Chow-Liu tree branches in section 7.4.1, it became apparent that approximating the joint distribution of features with a tree will lead to less reduction in variance upon a successful measurement. Figure 7.7(b) demonstrates exactly this statement, also suggesting that the difference in variance reduction becomes less apparent with increasing frame rate. This is due to the fact that as we are moving towards higher frame rates, the uncertainty ellipse of each feature becomes smaller (bound to reach measurement error at minimum) as the predictions become more and more accurate, therefore there is not much room for further reduction then. However, when observing this variance reduction to scale (i.e. with respect to the initial search regions as shown in Figure 7.7(a)) it is evident that using CLAM search regions are still reduced dramatically which in a way demonstrates the good quality of the approximation: the Chow-Liu tree, despite an approximation preserves the most dominant correlation structure (if this was not the case, then search regions wouldn't reduce as much).



(a) CLAM timings (b) FAM timings

Figure 7.8: Breakdown of computation time for CLAM and FAM when tracking the office sequence. Both figures (a) and (b) here are shown on the same scale for the sake of comparison. Observing (a) it is evident that CLAM maintains a better balance between the main stages of the algorithm as opposed to the original AM where evaluating MIs the dominant and limiting factor of the algorithm. While FAM seems to perform marginally better than CLAM, the robustness and accuracy that CLAM provides makes it a better option.

Time Requirements

As interpreted before, Figure 7.8 illustrates the breakdown of computation time consumed within each step of the matching procedure on average, per frame. Observing the histogram bars in 7.8(a) it is evident that the use of the tree to evaluate MIs and update the mixture in CLAM achieves the maintenance of a better balance of resources between the different parts of the algorithm. As the number of features tracked increases, the time consumed by each procedure also increases though it a dramatically better rate than the original AM algorithm. Still this rate is a little worse than linear, however we can match 100 features per frame in less than 125*ms* per frame. While Civera et al. [2009c] do not mention explicitly the time needed for matching alone, however they record tracking timings of 1 frame per second for the same number of features and much lower resolution of images (320×240) which in principle makes the task of identifying matches a lot faster.

Figure 7.8(b) superimposes the timings of Fast Active Matching for increasing numbers of features per frame for the same office sequence. While correlation is a more costly process in FAM, overall the timings recorded are marginally better than those of CLAM. However, as discussed later on FAM proves to be significantly less robust and accurate than CLAM which in the end makes CLAM a much more attractive approach to fast matching.

As a means of comparison between all the different methods discussed in this chapter, Table 7.1 below illustrates the breakdown of the time consumed per frame

METHOD	CORRELATION	RESOLVE CONSENSUS			TOTAL
		Evaluate MIs	Update MoG	CL-tree	
JCBB	136.2*ms*	23786.6*ms*			23.9*s*
AM	19.7*ms*	22531.1*ms*	2653.4*ms*	-	25.2*s*
FAM	83.1*ms*	2.5*ms*	15.6*ms*	-	101.2*ms*
CLAM	22.2*ms*	45.9*ms*	49.5*ms*	4.4*ms*	122.0*ms*

Table 7.1: Time needed by each method to complete matching of 100 features in a typical frame of the office sequence. Note the difference in units of these figures.

when tracking a typical frame of the office sequence with 100 features (notice the difference in the measurement units in some cases).

Albeit achieving the lowest correlation time due to its ability to reduce search regions most, the original AM algorithm takes a ridiculously long time to discover the matching consensus for 100 features and the main factor is the evaluation of MIs, taking almost 23*s* per frame. While it is the guiding force in AM, the evaluation of MIs comprises a huge bottleneck which we specifically tackled with FAM. Indeed, randomising the choice of candidates to evaluate, reduces the timing of this step drastically in FAM while it loses from the reduction in the search regions leading to longer correlation time. The fully probabilistic approximation of CLAM on the other hand selects more carefully which relationships between features to preserve, achieving lower correlation timings though taking a little longer to complete. Joint Compatibility, as expected, has the longest correlation time but its main bottleneck is the resolution of consensus rendering it well out of real-time bounds, while in some frames it requires significantly longer time than recorded here (the timings here are shown on a basis of a small portion of the sequence in an attempt to give a general understanding of how the processing times compare).

Comparing Trajectories

While outstanding time performance is a very significant asset that a matching algorithm has to have, accuracy is equally important. So the question is what can each algorithm achieve within the time it takes to complete? Here, in an attempt to assess the quality of the matching outcome of each technique, we superimpose the estimated trajectories when tracking both the captured office sequence and a more challenging sequence of an outdoor walkway for different numbers of features.

Since ground truth is not available, here we critique the quality of matching with respect to the trajectory estimated while tracking using our original AM algorithm. We assume that this method will provide the most consistent estimates given the current tracking settings and available information. This assumption is made on the grounds that despite taking a long time to complete, AM's fully probabilistic way of discovering

consensus makes it resilient to significant presence of outliers and moreover, it uses all available prior information.

Figure 7.9 superimposes the estimated trajectories obtained when tracking the captured office sequence with different settings. Tracking using 100 features per frame we expect to get the best of every matching technique due to the presence of significant amounts of data upon which a hypothesised matching scenario can be checked. Indeed looking at Figures (c) and (d) for CLAM and FAM respectively, we can see that the trajectories obtained are generally very similar to those obtained using AM in (a). However, looking more closely at (d) we can see that the most recent camera pose estimates do not seem very consistent leading to a jerky end which contrasts the smoothness throughout the trajectories of AM and CLAM in (a) and (c) respectively. This suggests that FAM fuses some outliers in the system estimate which at this time in the sequence are generated by mismatched features lying either on the carpet or outside the window where patch-correlation is likely to fail or produce multiple matches. As a result, despite achieving the fastest performance, FAM proves less robust to outliers than CLAM.

Looking at Figure 7.9(b) which illustrates the trajectory obtained while tracking 40 features per frame using CLAM, it seems that despite the relatively low number of features, CLAM still performs very well. Despite the fact that it is indeed an approximation of AM, these figures suggest that this is an approximation worth making since the quality of tracking is not put at risk while the time performance improves drastically.

Pushing the algorithm to the limits, we tested the quality of tracking on the outdoor walkway sequence. This is a particularly challenging sequence to track since lowering the Shi-Tomasi thresholds to allow for tracking a hundred features leads to many Shi-Tomasi peaks occurring on edge-like patches, since edges are a very dominant structure in the obtained images. Edge features are a major source of outliers when matching using normalised cross correlation since they can very easily slide along the edge producing multiple similar matches, making robust resolution of consensus a very hard task even to the human eye. In fact, FAM really fails to track this sequence as illustrated in Figure 7.10 where the estimated trajectory does not resemble at all the roughly straight path of the hand-held camera.

Due to the fact that the quality of Shi-Tomasi features detected in this sequence is very limited, tracking is generally not advised using this choice of feature detector and descriptor. For this reason and due to the fact that FAM fails to produce consistent estimates, the statistical results presented in this Section so far, have solely been based on the test-bed generated for the office sequence. However, our objective here is to demonstrate the quality of approximation of CLAM, therefore in Figure 7.11 we

(a) **AM tracking 100 features per frame**: used as the model trajectory for this sequence.

(b) **CLAM tracking 40 features per frame**: even with much lower number of features, the algorithm achieves consistent estimates resulting to a trajectory very similar to (a).

(c) **CLAM tracking 100 features per frame**: despite an approximation of AM, the obtained trajectory is almost the same as in (a).

(d) **FAM tracking 100 features per frame**: trajectory becomes jerky towards the end which is a sign of some mismatches incorporated in the system.

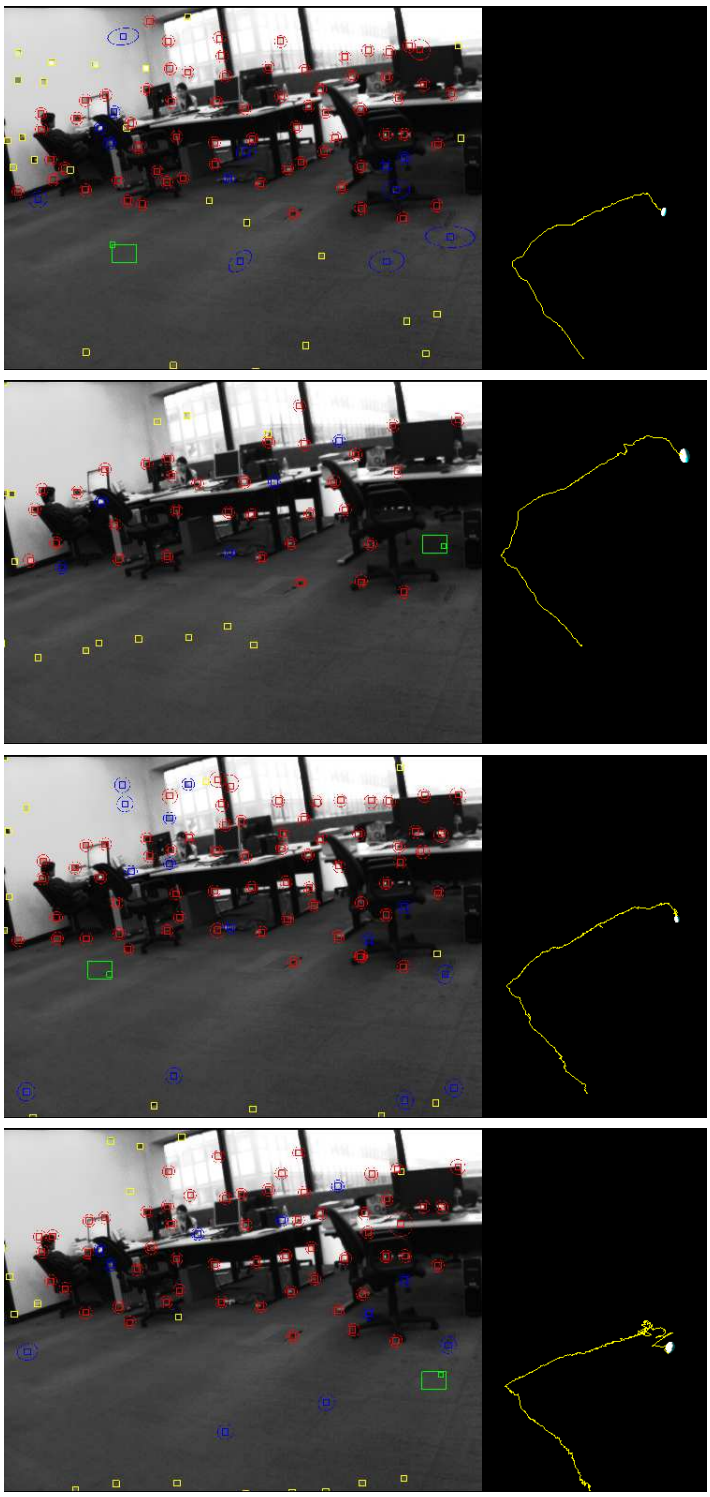


Figure 7.9: Comparison of trajectories using different matching techniques to track the office sequence. Images on the left correspond to the camera view and images on the right display the camera trajectory with the accumulated uncertainty in the camera position. Note: absolute scale is not estimated here, so differences in the scale of trajectories are a matter of display.

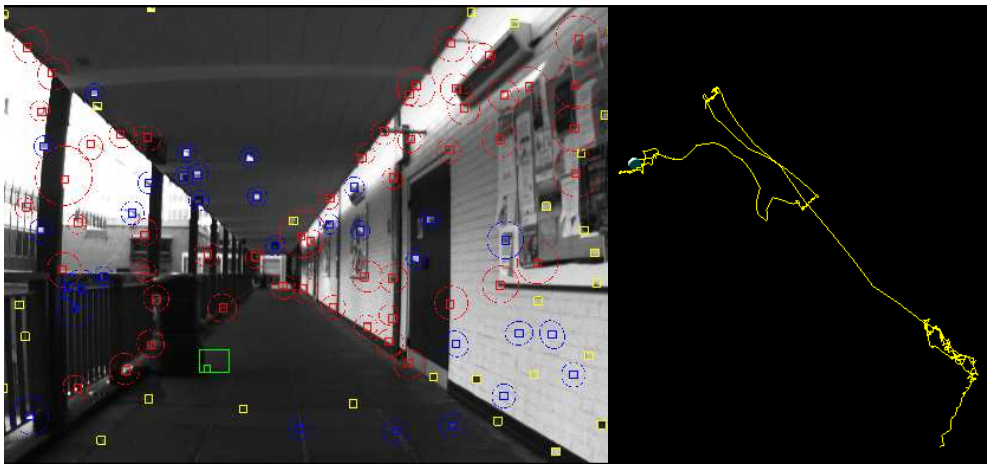


Figure 7.10: Tracking the outdoor walkway sequence using FAM with 100 features per frame. The multiple edge-like features selected for tracking produce a high number of outliers which FAM is not able to cope with as demonstrated in the estimated trajectory which deviates significantly from the actual straight path followed by the hand-held camera.

compare the quality of trajectories obtained using AM and CLAM while tracking the outdoor walkway sequence.

Figure 7.11(a) which illustrates the trajectory obtained when matching 40 features per frame using AM, suggests that the bad quality of features has affected the quality of tracking leading to a slight curve in the estimated camera path. It is important to note here that in order to isolate the effect of the approximation of AM with CLAM, the capability of both algorithms to handle features with variable false-positive and true-positive rates has been switched off, essentially assuming that all features share the same matching characteristics. Looking at Figure 7.11(b) which depicts the trajectory obtained for matching 40 features with CLAM, it is evident that the estimates that the tracker has made are very similar to the case of matching using AM with the same number of features as shown in (a): despite the high data contamination, the approximation in CLAM maintains the same quality of tracking as AM at a much lower computational budget.

When raising the number of tracked features per frame to a hundred however, the quality of tracking is improved achieving straight trajectories for both AM and CLAM as illustrated in Figures (c) and (d). Despite the significant presence of outliers, both methods manage to recover what seems to be a very close representation of the true camera trajectory, due to the presence of extra data to check matching hypotheses upon.

(a) **AM tracking 40 features per frame**: the estimated trajectory has a slight curve due to mismatched features.

(b) **CLAM tracking 40 features per frame**: a similarly shaped trajectory for the approximated feature distribution.

(c) **AM tracking 100 features per frame**: denser matching allows better rejection of outliers, achieving a straight trajectory.

(d) **CLAM tracking 100 features per frame**: approximating the distribution still achieves a straight trajectory.

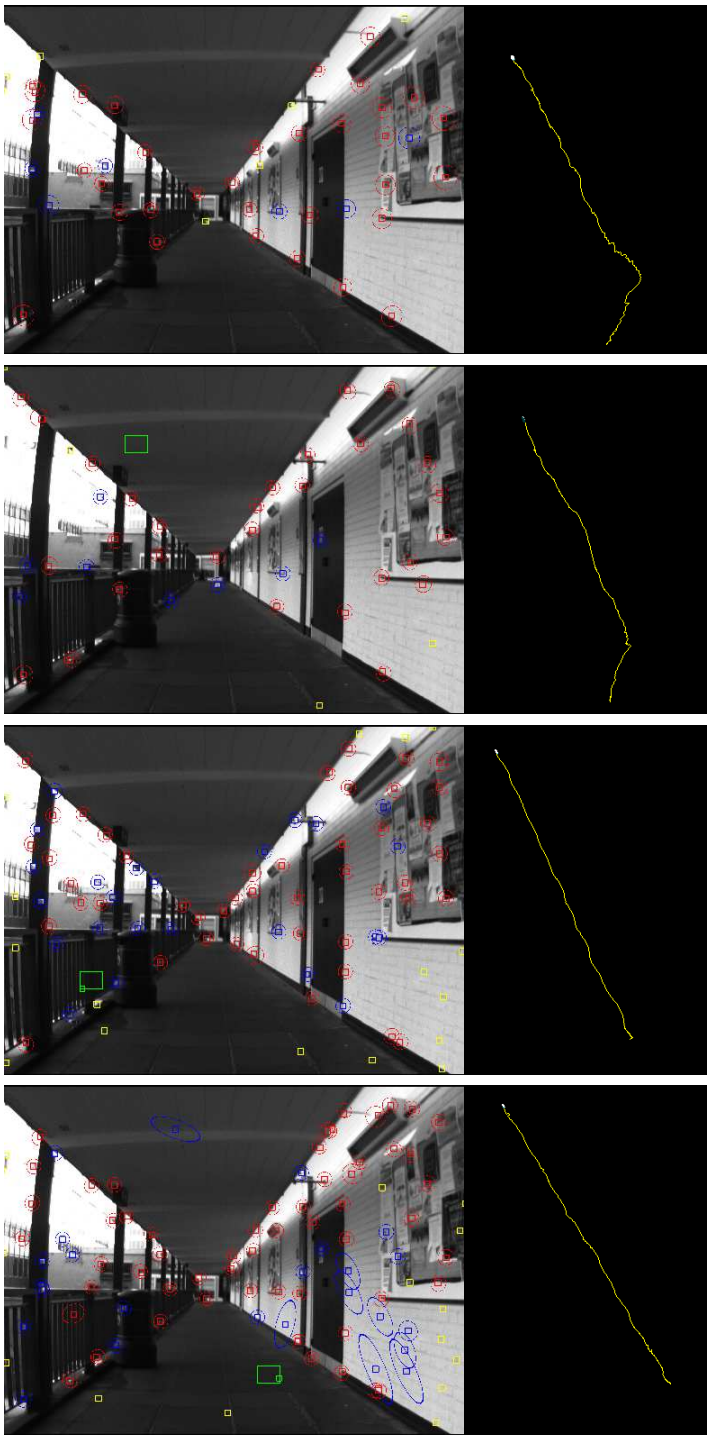


Figure 7.11: Comparison of trajectories obtained for the walkway sequence taken with a hand-held camera. This is a particularly challenging sequence at high resolution since many features snap on to edges which are a major source of outliers. Note that all features have been assumed to share the same matching characteristics.

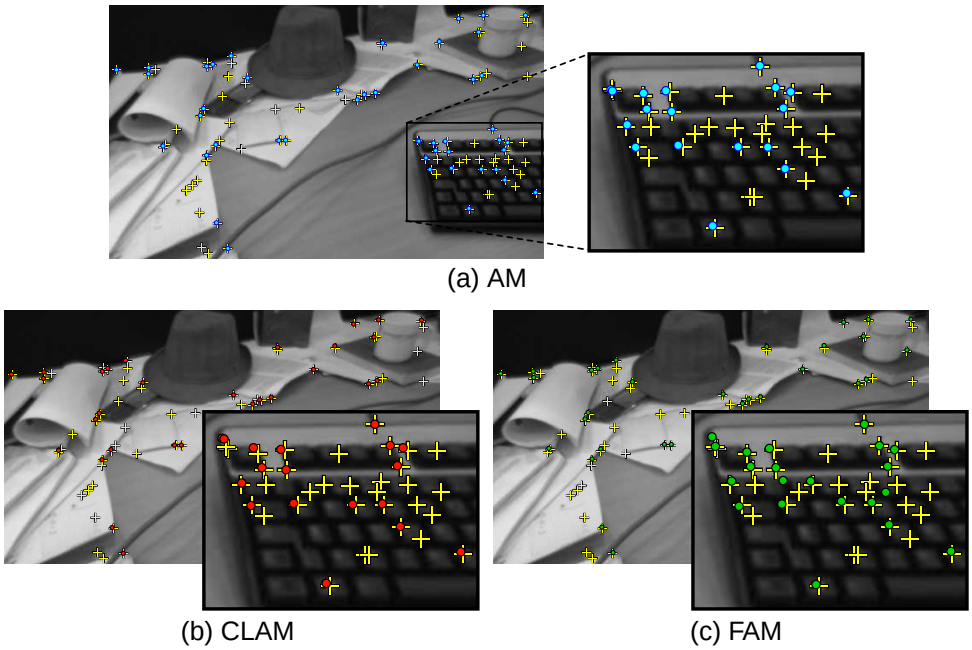


Figure 7.12: The matches obtained with each of the AM-based methods when given query-patches covering 3% of the area enclosed in the original template patches (i.e. using 4×4 pixels^2 patches). In this frame, AM's matches deviate by 0.8 pixels from the reference matches (shown in yellow crosses) on average while CLAM's matches exhibit an error of 0.9 pixels and FAM's matches a corresponding error of 1.3 pixels.

7.6 Quantitative Results

Following the qualitative analysis above, this section presents a quantitative comparison of AM, FAM and CLAM against the output of an independent reference matcher which employs sophisticated state-of-the art techniques to achieve very high accuracy performance albeit at high computational cost. Running within a keyframe optimisation framework which follows very much the design of PTAM [Klein and Murray, 2007], the reference matcher works by performing dense optical flow as proposed by Pock et al. [2008] between the current and the last frame to compute an initial guess for the camera pose. This pose is then used to guide matching by projecting the bundle-adjusted 3*D* feature positions in image space and updating their predicted appearance by warping the feature patches accordingly. The matches accepted in this scheme (referred to as 'reference matches' hereafter) correspond to the patches scoring the highest response within a small fixed-size window centred at each projected feature location.

In order to assess the performance of our AM-based algorithms against the reference matcher we construct the innovation covariance matrix $\mathtt{S}$ on every frame as this is not explicitly maintained in keyframe optimisation tracking frameworks. This computation turns out to be straightforward in this particular setup as at every instant the

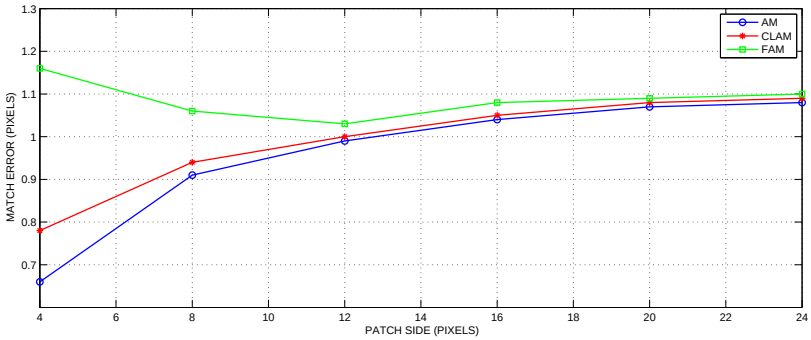


Figure 7.13: Assessing the performance of our AM-based algorithms with respect to a high-quality reference matcher. This figure illustrates the average distance of the obtained matches from the reference matches for variable-sized input templates (note that the templates are square patches). AM achieves the lowest match error confirming the power of the algorithm to robustly reject outliers. CLAM's performance is closer to AM than FAM which indicates that CLAM is a more accurate approximation to AM.

pose of the previous frame has already been optimised with respect to the 3*D* map. Hence only the relative uncertainty between the previous and the current frame $\mathtt{P}_{\mathbf{x}_c}^{(rel)}$ will have an effect on the feature predictions; this uncertainty is essentially equal to the process noise ($\mathtt{Q}$) of the camera motion model. Hence, using the notation introduced in Chapter 3 we can evaluate:

$$\mathtt{S} = \frac{\partial \mathbf{h}(y_{1:n})}{\partial \mathbf{x}_c} \mathtt{P}_{\mathbf{x}_c}^{(rel)} \frac{\partial \mathbf{h}(y_{1:n})}{\partial \mathbf{x}_c}^{\top} + \mathtt{R}\,, \tag{7.33}$$

where $\mathbf{h}$ denotes the measurement model of all map features $\mathbf{y}_i$. Note that the resulting $\mathtt{S}$ is dense even though the inter-feature covariances come solely from motion uncertainty.

The keyframe-based tracker (which incorporates the reference matcher) is run once on a 400-frame sequence to gather all the data necessary for the quantitative analysis: on every frame, we store the 24×24 patches of the features expected to be visible along with their predicted image locations and associated innovation covariance matrix $\mathtt{S}$. Following the completion of feature matching, we also store the reference matches encountered on every frame.

During testing, each AM-based algorithm is evaluated on the same video sequence and on every frame it is fed with the data corresponding to the features for which a reference match exists. Before processing a new frame, we record the distance of the matches discovered with respect to the positions of the reference matches. This score is then averaged over the whole sequence of frames to give the *'match error'* for each of AM, FAM and CLAM. In an attempt to assess the relative behaviour of these algorithms with increasing ambiguity, we also run tests where the input feature-patches

correspond only to a sub-region of the original templates (cropping off equal blocks of pixels from each side of the original 24×24 template patch). Patches at smaller sizes are less individually distinctive and as a result, consensus matching becomes increasingly important then, imposing a greater challenge in outlier rejection.

Figure 7.12 illustrates the matches obtained using all three methods for the case of matching 4×4 patches using each of the AM-based algorithms on a typical frame, whereas Figure 7.13 summarises the results over all test-runs. As expected, the distance of the matches obtained with respect to the reference matches, AM seems to be the most accurate of the three methods achieving the lowest match error in all cases. CLAM follows very closely the performance of AM whereas FAM exhibits larger deviations and bigger match errors confirming that CLAM is a better approximation to AM than FAM.

7.7 Conclusions

Imagine that a tracking system had all the time in the world to perform matching, what would be the best method of discovering matching consensus upon the reception of a new image?

One could argue that the answer is to attempt matching of every single pixel of the previous image with the new one employing all available priors together with probabilistic inference, resembling the philosophy behind optical flow techniques. However, due to the very high costs involved, real-time solutions aim to approximate this procedure. The key then is in the balance between the quality that one is prepared to sacrifice and the time consumed to completion. While super-dense matching is yet far from real-time (Zach et al. [2007] record around 30fps for 256×256 images on a GPU implementation, however this would be many times slower on CPU), RANSAC-based techniques are dominating the landscape of modestly dense matching algorithms at present. Their reliance on statistical fairness however is both the reason for their speedy performance but also the source of tracking failure in challenging scenarios. Approaching the problem of matching from a fully probabilistic perspective, more informed choices can be made on the course of matching based on the prior information available. This can lead to more robust solutions, suffering however from poor scaling with expanding data sets. The purpose of this chapter has been to tackle this problem in an attempt to bring probabilistic techniques a step closer to dense real-time matching.

Using the fully probabilistic framework of Active Matching as a basis, the first attempt has been to attack the most time consuming part of the algorithm: substituting part of the processing of available priors with randomness, decisions within matching have been made less informed which in a way is similar to the RANSAC philoso-

phy. This semi-randomised Fast Active Matching algorithm indeed proved much more cost effective than Active Matching but our experiments later demonstrated how this randomness can significantly affect the quality of tracking.

Taking a different route to approximating the procedure of Active Matching, this chapter discussed how the notion of the Chow-Liu tree can be used to sparsify the joint distribution of predicted features while preserving the most important correlation structure, leading to the emergence of the Chow-Liu Active Matching (CLAM) algorithm. Exploiting the benefits of the tree structure, CLAM is shown to accomplish both high tracking quality and competitive timings.

While CLAM is indeed a breakthrough in fully probabilistic dense matching, there remains yet a lot to be done to reach the optimal, online dense matching solution. Following the same path of probabilistic inference and Information Theory, in the future we aim to look deeper into quality approximations of the full solution to the problem of pixel-by-pixel matching using more general inference techniques.

8

Conclusion

8.1 Summary of Contributions

This thesis has explored the application of an Information Theoretic framework to guiding efficient and robust estimation within the context of SLAM. Driven by the demand for agile manipulation of data in current state of the art systems, this research has employed Information Theory to direct decision-making towards more effective approximations to the full SLAM problem. The analysis of the inherent relationships between the members of a SLAM map from an Information Theoretic perspective provides a transparent understanding of the scene structure which in turn is key to the development of powerful algorithms.

8.1.1 Active Matching

Addressing one of the major sources of errors during tracking, we have presented a novel Bayesian algorithm for step-by-step active search and data association in images. Our Active Matching methodology gets to the heart of the importance of priors in maintaining consistency, engaging them in an Information Theoretic manner to guide search for matching consensus. Essentially, the algorithm harnesses the fact that corre-

lations between different candidate feature measurements encode the expected impact of each measurement in achieving a globally consistent outcome. A dynamic mixture of Gaussians represents the uncertain search-state of matching at each instant and is maintained in a fully probabilistic manner to account for the multiple hypotheses naturally arising in real images. Information Theory then used to combine the influence of a candidate measurement on both the convergence to a single hypothesis and the achievement of high precision within that hypothesis. Upon completion, the algorithm gives a list of matching scenarios surviving the consistency check, along with their estimated probabilities of reflecting the true solution.

Our experimental analysis has demonstrated the robustness of Active Matching to variable camera dynamics, and different levels of input priors and ambiguity. The sequential evaluation of 'where to look next' not only reduces dramatically the number of pixels searched for matches, but most importantly enforces resilience to repetitive structure; the matcher is guided towards more promising areas resulting to the encounter of fewer false positives than traditional matching techniques. With the aim of revealing the strengths and weaknesses of the algorithm we also presented an extensive performance analysis varying both the frame rate and the number of features being tracked. While across the span of the scenarios tested the accuracy of the algorithm has not been compromised, the computational scaling to increasing numbers of features, comprises a significant limitation to the applicability of this methodology. However, studying the evolution of the Mutual Information of candidate measurements throughout matching has opened up the route to explore more scalable matching algorithms.

8.1.2 Map Management for Large Data Sets

In the context of effective manipulation of data for increased versatility of solutions, we have proposed an automatic and efficient methodology to infer the hierarchical structure of visual maps. Following the need for bigger and denser maps, researchers have long been using map-sparsification techniques to approximate the otherwise fully-connected graph of feature relations, aiming for reduced management costs. However, while submapping criteria have most often been based on a variety of implementation-specific thresholds, we have demonstrated how our Information Theoretic framework can be employed to dynamically guide quality partitioning of maps. We have illustrated the application of our method on the particularly complex case of visual maps, where the infinite range of the camera makes the 'ideal' submap divisions less clear.

Translating the probabilistic priors available in sequential tracking into Mutual Information, we have showed how we can quantify the information content shared between individual entries in a general SLAM map, as perceived through a camera lens. Studying the correlation-links into the Information space, we illustrated how

we can progressively identify regions of high Mutual Information density, suggesting strong correlation structure. Our experiments have demonstrated how this fairly simple analysis can provide meaningful clusters of features (e.g. separating foreground from background) for different types of scene and camera motions. This comprehensive insight into the hierarchy of relationships inherent in visual SLAM maps we believe has great potential in enhancing the quality of performance in modern systems.

8.1.3 Scalable Feature Matching

Following the understanding gained by the detailed performance analysis of Active Matching and the study of approximations of the map structure, we have presented a new, fully probabilistic feature-matching algorithm able to achieve online matching in dense tracking scenarios. While fast data association for a large number of features has previously been made possible with random-sampling techniques, our Chow-Liu Active Matching (CLAM) algorithm aims to bring the robustness promised by fully probabilistic approaches to the applications. By making decisions solely based on concrete probabilistic estimates and Information Theoretic measures, CLAM defies the need for implementation-specific thresholds and the reliance on randomness.

CLAM essentially comprises an approximation to the problem addressed in Active Matching, only approximated enough to permit online performance while preserving the precision of the outcome at the same time. Approaching the prior probability distribution of visible features from an Information Theoretic perspective, we have shown how this can be approximated with a tree in Mutual Information space. Belief Propagation is employed to propagate predicted or observed matches across the branches of this tree. As a result, exploiting the computational shortcuts and complexity benefits of this tree structure, we have demonstrated how the robustness of Active Matching can be achieved in a much more efficient way.

8.2 Future directions

The extensive Information Theoretic analysis of the relationships between SLAM estimates discussed within the body of this thesis leads to a broad understanding of the problem we are trying to solve. The generality of this investigation suggests that it can be applied to the wide variety of SLAM estimation implementations currently in the literature, providing an insight into the effectiveness of the approximations performed. With the ultimate goal of high performance and generally applicable algorithms, this work has been a small contribution along a longer chain of research towards practical and theoretically justified methods.

We believe that Information Theory still has many answers to offer on the primary

trade-off of 'quality versus cost' challenging today's systems. Is it worth matching every single pixel in the image? Will a keyframes-approach prove more beneficial than traditional filtering given this tracking scenario? Following the general acceptance of probabilistic techniques in manipulating real-world data, Information Theory can provide the complementary framework to guide dynamic decision-making. Either offline or at runtime, these decisions can provide the key to the goal performance of our algorithms.

Getting to the heart of the estimation problem in SLAM, our general aim is to gain deep understanding of the complexity involved and the processing resources available. With the target of fully adaptable algorithms, below we give a more specific description of future research directions following from the conclusions drawn in this thesis.

8.2.1 Understanding the Graphical Representations of the World

Filtering approaches have brought implementations a long way providing good estimates for small-scale environments while requiring sparsifying approximations for larger amounts of data. Keyframes solutions on the other hand have been more successful in dense maps. Moreover, the recent trend towards relative representations and bundle adjustment methods on selected sets of nodes seem to suggest improved overall timings. It would be interesting to investigate the strengths and weaknesses of each representation with the dual aim of identifying the best option given a particular tracking scenario, and also revealing avenues for improvement.

8.2.2 Quality and Speed in Frame to Frame Processing

Our investigation of feature matching has revealed the power of 'thinking' how to process the input data to achieve robustness and efficiency in local motion estimation. However, we have seen that the overhead involved in this assessment can take a significant part of the available processing time. Moreover, while Information Theory has indicated successful approximations towards scalable feature matching, we would like to investigate further simplifications of the frame-estimation process with the goal of really dense matching. A more practical analysis of the computational costs involved within every individual assessment and estimation process can provide a measure of the effort implied by the employment of a particular approach. We believe that through this investigation estimation algorithms can potentially have the ability to dynamically assess the stability and effectiveness of the approximations suggested and adapt the computation process accordingly.

A

Appendix

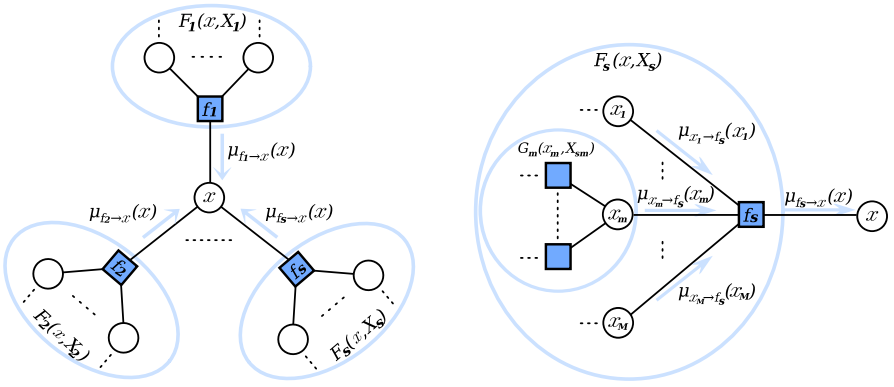
A.1 The Sum-Product Algorithm

Following the notation of Bishop [2006] we consider the joint probability distribution $\mathbf{x}$. Computing the marginal of a particular variable x involves a summation of the joint distribution over all variables except x:

$$p(x) = \sum_{\mathbf{x}/\,x} p(\mathbf{x}) \; . \tag{A.1}$$

The distribution $p(\mathbf{x})$ can be expressed as a product of 'local' functions describing the relationships (factors) of member-variables or 'nodes' when considering the diagrammatic representation of the distribution. This is known as the bipartite *factor graph* in the literature of graphical models. In turn, $p(\mathbf{x})$ can be expressed in terms of groups of factors $F_s(x, X_s)$ such that each group contains the factors relating the set of nodes X_s in the subgraph neighbouring node x as illustrated in Figure A.1(a). Hence, if $ne(x)$ is the set of factor nodes neighbouring with x we can write:

$$p(\mathbf{x}) = \prod_{s \in ne(x)} F_s(x, X_s) \; . \tag{A.2}$$



(a) Evaluating $p(x)$ (b) Evaluating $\mu_{f_s\to x}(x)$

Figure A.1: The nested nature of messages passed in the sum-product algorithm. (a) is a visualisation of the factorisation of the marginal $p(x)$ in terms of factor messages $\mu_{f_s\to x}(x)$ as defined in Equation A.3. Each factor message coming from a subgraph neighbouring with node x is evaluated in a recursive manner as suggested in (b) where a close-up of one such subgraph is depicted. Any factor f_s can only propagate a message to x once it has collected all other node messages $\mu_{x_m\to f_s}(x_m)$ as defined in equation A.7.

Substituting Equation A.2 into A.1 and interchanging the product and sum operators, the marginal of x can be re-written as:

$$p(x) = \prod_{s\in ne(x)} \left[\sum_{X_s} F_s(x,X_s) \right] = \prod_{s\in ne(x)} \mu_{f_s\to x}(x) \ , \tag{A.3}$$

where each $\mu_{f_s\to x}(x)$ stands for the message passed from factor node f_s to node x defined to be:

$$\mu_{f_s\to x}(x) = \sum_{X_s} F_s(x,X_s) \ . \tag{A.4}$$

Now in order to see how these messages get evaluated we have to look 'deeper' into each subgraph as depicted in Figure A.1(b). Each F_s being a group of factors, itself can be factorised in terms of f_s and sub-groups of factors $G_m(x_m,X_{sm})$. In order to motivate recursion, each x_m (which by definition is a member of X_s) is chosen so that it is an immediate neighbour of f_s. Therefore,

$$F_s(x,X_s) = f_s(x,x_1,\ldots,x_M) \prod_{m\in ne(f_s)/\ x} G_m(x_m,X_{sm}) \ , \tag{A.5}$$

where X_{sm} comprises the set of nodes in X_s relating with the rest of the graph through x_m. Having defined F_s we can now evaluate the factor message $\mu_{f_s\to x}(x)$. Given that

$X_s = \{x_1, \ldots, x_M, X_{s1}, \ldots, X_{sM}\}$, we expand the summation in A.4 and substitute for F_s:

$$\begin{aligned}
\mu_{f_s \to x}(x) &= \sum_{x_1} \ldots \sum_{x_M} \sum_{X_{s1}} \ldots \sum_{X_{sM}} f_s(x, x_1, \ldots, x_M) \prod_{m \in ne(f_s)/\, x} G_m(x_m, X_{sm}) \\
&= \sum_{x_1} \ldots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in ne(f_s)/\, x} \left[\sum_{X_{sm}} G_m(x_m, X_{sm}) \right] \\
&= \sum_{x_1} \ldots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in ne(f_s)/\, x} \mu_{x_m \to f_s}(x_m) \; , \qquad \text{(A.6)}
\end{aligned}$$

where $\mu_{x_m \to f_s}(x_m)$ is the message issued from node x_m to factor f_s, defined to be equal to:

$$\mu_{x_m \to f_s}(x_m) = \sum_{X_{sm}} G_m(x_m, X_{sm}) \; . \qquad \text{(A.7)}$$

Equation A.6 demonstrates the recursive nature of the evaluation of messages passed along the branches of the tree. A factor node f_s wishing to issue a message for node x collects the node-messages coming from all other neighbour-nodes x_m each evaluated recursively and in the same manner based on the subgraph they d-separate from the rest of the graph. Messages are initially issued from the leaves of the graph and are then progressively propagated to the root of the tree (here node x), essentially marginalising variables one-by-one so that by the time the root has received messages from all its variables, the summation of Equation A.1 will have been achieved.

Given now that a set of nodes is observed and the goal is to compute the new marginal distribution of every other node in the tree, we can arbitrarily designate a root-node and propagate messages from the leaves to the root and back, so that every node will have efficiently received updates from all its neighbours. In the special case where only one node is observed, this process can become even more efficient by issuing outward messages directly from that node towards all its neighbours until updates reach the ends of the tree. This is immediately applicable to our Active Matching paradigm where only one feature is matched at every matching step. Note that although discrete probability distributions have been assumed in this derivation, the methodology is general and can easily be adapted to continuously-distributed variables, by essentially replacing the summation operators by integration. As a side note, while the sum-product algorithm provides exact inference in tree structures, researchers have been using it also for graphs containing cycles (loops) leading to the emergence of Loopy Belief Propagation (LBP). However, since here we aim to use the Chow-Liu tree, LBP is out of the scope of this chapter.

Bibliography

M. Agrawal, K. Konolige, and M. R. Blas. CenSurE: Center surround extremas for realtime feature detection and matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2008.

D. Alspach and H. Sorenson. Nonlinear bayesian estimation using gaussian sum approximations. *IEEE Transactions on Automatic Control*, 17(4):439–448, 1972.

A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions on Robotics (T-RO)*, 24(5):1027–1037, 2008.

A. Azarbayejani and A. P. Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 17(6):562–575, 1995.

T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot. Consistency of the EKF-SLAM algorithm. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2006a.

T. Bailey, J. Nieto, and E. Nebot. Consistency of the FastSLAM algorithm. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006b.

H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 110(3):346–359, 2008.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., 2006.

M. Bosse and J. Roberts. Histogram matching and global initialization for laser-only SLAM in large unstructured environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

M. Bosse, P. Newman, J. J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

M. Bosse, P. Newman, J. J. Leonard, and S. Teller. Simultaneous Localisation And Mapping in Large-scale Cyclic Environments using the Atlas Framework. *International Journal of Robotics Research (IJRR)*, 23(12):1113–1139, 2004.

T. J. Broida, S. Chandrashekhar, and R. Chellappa. Recursive 3-D motion estimation from a monocular image sequence. *IEEE Transactions on Aerospace and Electronic Systems*, 26:639–656, 1990.

M. Brown and D. G. Lowe. Recognising panoramas. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2003.

M. Bryson and S. Sukkarieh. An information-theoretic approach to autonomous navigation and guidance of an uninhabited aerial vehicle in unknown environments. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2005.

P. Bunnun and W. W. Mayol. OutlinAR: an assisted interactive model building system with reduced computational effort. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 61–64, 2008.

J. Campbell, R. Sukthankar, and I. Nourbakhsh. Techniques for evaluating optical flow for visual odometry in extreme terrain. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2004.

D. Capel and A. Zisserman. Automated mosaicing with super-resolution zoom. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1998.

R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray. Video-rate recognition and localization for wearable cameras. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2007.

D. Chekhlov, M. Pupilli, W. W. Mayol, and A. Calway. Real-time and robust monocular slam using predictive multi-resolution descriptors. In *Proceedings of the 2nd International Symposium on Visual Computing*, 2006.

A. Chiuso, P. Favaro, H. Jin, and S. Soatto. Structure from motion causally integrated over time. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(4):523–535, 2002.

M. Chli and A. J. Davison. Active Matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2008a.

M. Chli and A. J. Davison. Automatically and efficiently inferring the hierarchical structure of visual maps. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009a.

M. Chli and A. J. Davison. Active Matching for visual tracking. *Robotics and Autonomous Systems*, 57(12):1173 – 1187, 2009b. Special Issue 'Inside Data Association'.

M. Chli and A. J. Davison. Efficient data association in images using active matching. In *Robotics: Science and Systems (RSS) Workshop on Inside Data Association*, 2008b.

K. S. Chong and L. Kleeman. Feature-based mapping in real, large scale environments using an ultrasonic array. *International Journal of Robotics Research (IJRR)*, 18(2): 3–19, January 1999.

C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

O. Chum and J. Matas. Optimal randomized RANSAC. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(8):1472–1482, 2008.

J. Civera, D. R. Bueno, A. J. Davison, and J. M. M. Montiel. Camera self-calibration for sequential bayesian structure from motion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009a.

J. Civera, A. J. Davison, J. A. Magallón, and J. M. M. Montiel. Drift-free real-time mosaicing. *International Journal of Computer Vision (IJCV)*, 81(2):128–137, 2009b.

J. Civera, O. G. Grasa, A. J. Davison, and J. M. M. Montiel. 1-point RANSAC for EKF-based structure from motion. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2009c.

L. A. Clemente, A. J. Davison, I. Reid, J. Neira, and J. D. Tardós. Mapping large loops with a single hand-held camera. In *Proceedings of Robotics: Science and Systems (RSS)*, 2007.

A. I. Comport, E. Malis, and P. Rives. Accurate quadri-focal tracking for robust 3D visual odometry. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, second edition, 2006.

M. Cummins and P. Newman. Probabilistic appearance based navigation and loop closing. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

M. Cummins and P. Newman. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research (IJRR)*, 27(6): 647–665, 2008.

M. Cummins and P. Newman. Highly scalable appearance-only SLAM — FAB-MAP 2.0. In *Proceedings of Robotics: Science and Systems (RSS)*, 2009.

A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2003.

A. J. Davison. Active search for real-time vision. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2005.

A. J. Davison and N. Kita. Sequential localisation and map-building in computer vision and robotics. In *Proceedings of the 2nd Workshop on Structure from Multiple Images of Large Scale Environments (SMILE), in conjunction with ECCV 2000*. Springer-Verlag LNCS, 2000.

A. J. Davison and D. W. Murray. Mobile robot localisation using active vision. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 1998.

A. J. Davison, W. W. Mayol, and D. W. Murray. Real-time localisation and mapping with wearable active vision. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2003.

A. J. Davison, N. D. Molton, I. Reid, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(6):1052–1067, 2007.

F. Dellaert. Square root SAM. In *Proceedings of Robotics: Science and Systems (RSS)*, 2005.

H. Durrant-Whyte and T. Bailey. Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. *IEEE Robotics and Automation Magazine*, 13(2): 99–110, 2006.

E. Eade. *Monocular Simultaneous Localisation and Mapping*. PhD thesis, University of Cambridge, 2008.

E. Eade and T. Drummond. Edge landmarks in monocular SLAM. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2006a.

E. Eade and T. Drummond. Unified loop closing and recovery for real time monocular SLAM. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2008.

E. Eade and T. Drummond. Scalable monocular SLAM. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006b.

E. Eade and T. Drummond. Monocular SLAM as a graph of coalesced observations. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2007.

E. Eade and T. Drummond. Edge landmarks in monocular SLAM. *Image and Vision Computing*, 27(5):588 – 596, 2009. Special Issue for the 17th British Machine Vision Conference (BMVC 2006).

C. Estrada, J. Neira, and J. D. Tardós. Hierarchical SLAM: Real-time accurate mapping of large environments. *IEEE Transactions on Robotics (T-RO)*, 21(4):588–596, 2005.

R. M. Eustice, H. Singh, and J. J. Leonard. Exactly sparse delayed state filters. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2005a.

R. M. Eustice, H. Singh, J. J. Leonard, M. Walter, and R. Ballard. Visually navigating the RMS titanic with SLAM information filters. In *Proceedings of Robotics: Science and Systems (RSS)*, 2005b.

R. M. Eustice, H. Singh, J. J. Leonard, and M. R Walter. Visually mapping the RMS titanic: Conservative covariance estimates for SLAM information filters. *International Journal of Robotics Research (IJRR)*, 25(12):1223–1242, 2006.

O. D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In *Proceedings of the European Conference on Computer Vision (ECCV)*, 1992.

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

A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 311–326. Springer-Verlag, June 1998.

U. Frese. Treemap: An $O(logn)$ algorithm for indoor simultaneous localization and mapping. *Autonomous Robots*, 21(2):103–122, 2006.

U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics (T-RO)*, 21(2): 196–207, 2005.

W. E. L. Grimson. *Object Recognition by Computer: The Role of Geometric Constraints*. Cambridge, MA: MIT Press, 1990.

J. E. Guivant and E. M. Nebot. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242–257, 2001.

M. Habbecke and L. Kobbelt. A surface-growing approach to multi-view stereo reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

C. G. Harris. Tracking with rigid models. In A. Blake and A. Yuille, editors, *Active Vision*. MIT Press, Cambridge, MA, 1992.

R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

S. Holmes, G. Sibley, G. Klein, and D. W. Murray. A relative frame representation for fixed-time bundle adjustment in SFM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 1996.

M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics (T-RO)*, 24(6):1365–1378, 2008.

J. H. Kim and S. Sukkarieh. Real-time implementation of airborne inertial-SLAM. *Robotics and Autonomous Systems*, 55(1):62–71, 2007.

J. H. Kim, S. Sukkarieh, and S. Wishart. Real-time navigation, guidance, and control of a UAV using low-cost sensors. In *Field and Service Robotics, Recent Advances in Research and Applications (FSR)*, 2003.

G. Klein and D. W. Murray. Improving the agility of keyframe-based SLAM. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2008.

G. Klein and D. W. Murray. Parallel tracking and mapping for small AR workspaces. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

K. Konolige and M. Agrawal. FrameSLAM: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics (T-RO)*, 24:1066–1077, 2008.

K. Konolige, M. Agrawal, and J. Solà. Large scale visual odometry for rough terrain. In *Proceedings of the International Symposium on Robotics Research*, 2007.

J. J. Leonard, H. Durrant-Whyte, and I. J. Cox. Dynamic map building for an autonomous mobile robot. *International Journal of Robotics Research (IJRR)*, 11(4): 286–298, 1992.

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

R. Lienhart and J. Maydt. An extended set of Haar-like features for rapid object detection. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2002.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4(4):333–349, 1997.

B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1981.

D. Mackay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

J. Manyika. *An Information-Theoretic Approach to Data Fusion and Sensor Management*. PhD thesis, University of Oxford, 1993.

C. Mei, S. Benhimane, E. Malis, and P. Rives. Efficient homography-based tracking and 3-D reconstruction for single-viewpoint sensors. *IEEE Transactions on Robotics (T-RO)*, 24(6):1352–1364, 2008.

C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. A constant time efficient stereo SLAM system. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2009.

Live Labs Microsoft©. Photosynth™, 2008. URL `http://www.photosynth.net`.

N. D. Molton, A. J. Davison, and I. Reid. Locally planar patch features for real-time structure from motion. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2004.

M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

J. M. M. Montiel, J. Civera, and A. J. Davison. Unified inverse depth parametrization for monocular SLAM. In *Proceedings of Robotics: Science and Systems (RSS)*, 2006.

E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real-time localization and 3D reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

P. Moutarlier and R. Chatila. Stochastic multisensory data fusion for mobile robot location and environement modelling. In *Proceedings of the International Symposium on Robotics Research*, 1989.

A. G. O. Mutambara. *Decentralized Estimation and Control for Multisensor Systems*. CRC Press, Inc., 1998.

J. Neira and J. D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, 2001.

J. Neira, M. I. Ribeiro, and J. D. Tardós. Mobile robot localisation and map building using monocular vision. In *Proceedings of the International Symposium on Intelligent Robotics Systems*, 1997.

P. Newman, D. Cole, and K. Ho. Outdoor SLAM using visual appearance and laser ranging. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

D. Nistér. Preemptive RANSAC for live structure and motion estimation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2003.

D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

E. Olson, J. J. Leonard, and S. Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

M. A. Paskin. Thin junction tree filters for simultaneous localization and mapping. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1157–1164, 2003.

L. M. Paz, P. Jensfelt, J. D. Tardós, and J. Neira. EKF SLAM updates in O(n) with divide and conquer SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007a.

L. M. Paz, J.Guivant, J. D. Tardós, and J. Neira. Data association in *O(n)* for divide and conquer SLAM. In *Proceedings of Robotics: Science and Systems (RSS)*, 2007b.

J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

P. Piniés. *SLAM in Large Environments with Wearable Sensors*. PhD thesis, Universidad de Zaragoza (Spain), 2009.

P. Pinies and J. D. Tardós. Large scale SLAM building conditionally independent local maps: Application to monocular vision. *IEEE Transactions on Robotics (T-RO)*, 24 (5):1094–1106, 2008.

T. Pock, M. Unger, D. Cremers, and H. Bischof. Fast and exact solution of total variation models on the GPU. In *Proceedings of the CVPR Workshop on Visual Computer Vision on GPU's*, 2008.

C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. Technical report, Computer Science Department, Carnegie Mellon University, 1993.

M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 1998.

M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3D reconstruction from video. *International Journal of Computer Vision (IJCV)*, 78 (2-3):143–167, 2008.

D. Ribas, P. Ridao, J.D. Tardós, and J. Neira. Underwater SLAM in man-made structured environments. *Journal of Field Robotics*, 25(11-12):898–921, 2008.

E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2006.

E. Rosten and T. Drummond. Fusing points and lines for high performance tracking. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2005.

D. Scaramuzza and R. Siegwart. Appearance guided monocular omnidirectional visual odometry for outdoor ground vehicles. *IEEE Transactions on Robotics (T-RO), Special Issue on Visual SLAM*, 24(5), 2008.

J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994.

G. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Proceedings of Robotics: Science and Systems (RSS)*, 2009.

S. N. Sinha, J. M. Frahm, M. Pollefeys, and Y. Genc. Feature tracking and matching in video using programmable graphics hardware. *Machine Vision and Applications*, 2007.

J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2003.

P. Smith, I. Reid, and A. J. Davison. Real-time single-camera SLAM with straight lines. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2006.

R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In *Uncertainty in Artificial Intelligence*, pages 435–461. Elsevier, 1988a.

R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In *Proceedings of the International Symposium on Robotics Research*, pages 467–474, 1988b.

N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3D. In *ACM Transactions on Graphics (SIGGRAPH)*, 2006.

R. Szeliski and S. B. Kang. Recovering 3D shape and motion from image streams using non-linear least squares. Technical report, Robotics Institude, 1993.

R. Szeliski and H. Y. Shum. Creating full view panoramic image mosaics and environment maps. In *ACM Transactions on Graphics (SIGGRAPH)*, 1997.

J. D. Tardós, J. Neira, P. Newman, and J. J. Leonard. Robust mapping and localization in indoor environments using sonar data. *International Journal of Robotics Research (IJRR)*, 21(4):311–330, 2002.

C. J. Taylor, D.J. Kriegman, and P. Adandan. Structure and motion in two dimensions from multiple images: a least squares approach. In *Proceedings of the IEEE workshop on Visual Motion*, 1991.

S. Thrun and M. Montemerlo. The graphSLAM algorithm with applications to large-scale mapping of urban structures. *International Journal of Robotics Research (IJRR)*, 25(5-6):403–429, 2006.

S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and A. Y. Ng. Simultaneous mapping and localization with sparse extended information filters. In *Proceedings of the Fifth International Workshop on Algorithmic Foundations of Robotics*, 2002.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Cambridge: MIT Press, 2005.

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization approach. *International Journal of Computer Vision (IJCV)*, 9(2): 137–154, 1992.

C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report, Technical Report CMU-CS-91-132, Carnegie Mellon University, 1991.

B. J. Tordoff and D. W. Murray. Guided-MLESAC: Faster image transform estimation by using matching priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(10):1523–1535, 2005.

P. H. S. Torr and A. Zisserman. MLESAC: a new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding (CVIU)*, 78 (1):138–156, 2000.

A. Vedaldi, H. Jin, P. Favaro, and S. Soatto. KALMANSAC: Robust filtering by consensus. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2005.

T. Vidal-Calleja, A. J. Davison, J. Andrade-Cetto, and D. W. Murray. Active control for single camera SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2006. URL `http://www.doc.ic.ac.uk/~ajd/Publications/vidal_etal_icra2006.pdf`.

G. Vogiatzis, C. H. Esteban, P. H. S. Torr, and R. Cipolla. Multiview stereo via volumetric graph-cuts and occlusion robust photo-consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(12):2241–2246, 2007.

J. Weingarten and R. Siegwart. EKF-based 3D SLAM for structured environment reconstruction. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2005.

B. Williams, G. Klein, and I. Reid. Real-time SLAM relocalisation. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2007.

S. B. Williams and I. Mahon. Simultaneous localisation and mapping on the great barrier reef. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2004.

S. B. Williams, G. Dissanayake, and H. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2002.

C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. In *Proceedings of the DAGM Symposium on Pattern Recognition*, 2007.