Condensing A Priori Data for Recognition Based Augmented Reality

by

Matthew Glenn Watson BEng (Hons)

Submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy

Deakin University

November 2012
I am the author of the thesis entitled

Condensing A Priori Data for Recognition Based Augmented Reality

submitted for the degree of

Doctor of Philosophy

This thesis may be made available for consultation, loan and limited copying in accordance with the Copyright Act 1968.

'I certify that I am the student named below and that the information provided in the form is correct'

Full Name: Matthew Glenn Watson

Signed: [Signature Redacted by Library]

Date: 15/05/13
I certify that the thesis entitled

Condensing A Priori Data for Recognition Based Augmented Reality

submitted for the degree of

Doctor of Philosophy

is the result of my own work and that where reference is made to the work of others, due acknowledgment is given.

I also certify that any material in the thesis that has been accepted for a degree or diploma by any university or institution is identified in the text.

'I certify that I am the student named below and that the information provided in the form is correct'

Full Name.................................Matthew Glenn Watson

Signed .................................................................

Date...............................15/05/13
For my parents,
Their endless support and unconditional love knows no bounds
Abstract

Augmented Reality (AR) renders virtual information onto objects in the real world. A user should experience a seamless blend of the virtual and real, where the convergence of the two is difficult to discern. However, errors in the registration of the real and virtual worlds are common and often destroy the AR illusion. To achieve accurate and efficient registration, the pose of real objects must be resolved in a quick and precise manner.

Recognising and estimating the pose of discrete real world objects in real time imagery becomes viable when a system has prior knowledge of what it is looking for. Hence, the creation of detailed information of the appearance and structure of an object a priori is critical for an AR system. This thesis contributes a novel a priori knowledge generation and refinement methodology to increase the efficiency and accuracy of recognition based pose estimation for augmented reality applications. The tightly coupled nature between the construction and refinement of a priori data and the efficiency and accuracy of online pose estimation is explored in several experiments throughout the thesis.

The principal methods in this thesis detail the creation of Sparse Feature Model (SFM) a priori data from robust and repeatable features to comprehensively characterise an object. Multiple short-baseline stereo images are captured from different perspectives around an object. For each stereo pair, a 2.5D point cloud is generated by the triangulation of corresponding, highly descriptive object features. A unique raw-SFM is reconstructed from these multiple views by merging each 2.5D point cloud together using 3D-to-3D shape registration.

Without refinement, a raw-SFM can introduce time inefficiency and error with registration when used for online pose estimation. Hence,
statistical and geometric methods are employed to refine raw-SFM a priori data. The statistical analysis approach in this thesis identifies the strongest and most persistent local features of an object by sampling the distribution of corresponding features across multiple frames. Only these strong and persistent features are triangulated to produce 2.5D point clouds for each frame. Frame-wise 3D-to-3D registration is performed to unify all 2.5D perspective views into a single coordinate space.

The geometric analysis approach of this thesis designates a cluster of overlapping points (i.e. for a persistent feature, matched across multiple frames) with a representative point, and is described by a representative feature vector. A representative point is defined by the Gaussian weighted mean for all points in a cluster, and specifies the 3D location of an object feature. Similarly, a representative feature vector condenses multiple descriptors in a cluster into a single feature vector. Both of these approaches further reduce the cardinality of a raw-SFM to create unique, computationally efficient refined-SFM a priori knowledge.

The time efficiency and pose accuracy of a priori based online pose estimation using raw- and refined-SFMs are evaluated. This analysis shows considerable improvements in the time efficiency and pose accuracy during online estimation, whilst using the condensed and highly descriptive refined-SFM a priori data over raw-SFM a priori data.

The proposed methods to reduce the cardinality of a priori data and yet retain distinctive and persistent features are novel within the augmented reality community. This research will help reduce latency and increase accuracy in recognition based pose estimation systems, thus improving the user experience for augmented reality applications.
# Table of Contents

Abstract........................................................................................................................................ v  
Acknowledgements...................................................................................................................... xi  
Publications.................................................................................................................................. xii  
List of Figures................................................................................................................................... xiii  
List of Tables..................................................................................................................................... xv  

**Chapter 1: Condensing A Priori Data** ................................................................. 1  
1.1. Enhancing Our World with Augmented Reality .................. 2  
1.2. The Aims and Importance of this Research ....................... 4  
1.3. Assumptions and Applications .......................................... 6  
1.4. Road Map .......................................................................... 8  

**Chapter 2: Modern Pose Estimation** ................................................................. 10  
2.1. Genesis .................................................................................. 11  
2.2. Egomotion Base Pose Estimation........................................ 13  
2.2.1. Recursive Estimates ......................................................... 13  
2.2.2. Using Posterior Knowledge ............................................. 14  
2.2.3. Issues with Egomotion .................................................... 16  
2.3. Recognition Based Pose Estimation .................................. 16  
2.3.1. Using Markers ................................................................. 17  
2.3.2. Passively Creating A Priori Knowledge ......................... 19  
2.4. Sparse Feature Model A Priori Knowledge ......................... 21  
2.5. Refining A Priori Knowledge .............................................. 25  
2.6. Conclusion .......................................................................... 27  

**Chapter 3: Computer Vision Mathematics** ..................................................... 28  
3.1. Stereo Camera Calibration.................................................... 29
3.1.1. Intrinsic Camera Parameters ..................................... 29
3.1.2. Extrinsic Camera Parameters .................................... 31
3.2. Two-view Geometry ...................................................... 31
3.3. The Mathematics of Triangulation .................................. 33
3.4. Extracting Features from Images ................................... 35
  3.4.1. SURF’s Hessian Matrix Based Detector ....................... 36
  3.4.2. SURF’s Distribution Based Descriptor ......................... 38
3.5. 3D Registration Fundamentals ...................................... 41
3.6. Summary ....................................................................... 42

Chapter 4: Raw Sparse Feature Models .............................. 43
  4.1. The Raw Sparse Feature Model .................................... 44
    4.1.1. Test Bed and Dataset Generation ............................. 46
  4.2. 2.5D View Generation .................................................. 48
    4.2.1. Pre-processing and Data Structures .......................... 48
    4.2.2. Feature Extraction ................................................ 49
    4.2.3. Feature Correspondence ....................................... 52
    4.2.4. Feature Vector Allocation ..................................... 54
    4.2.5. Constructing All 2.5D Perspective Views .................. 56
  4.3. 2.5D View Registration ................................................. 56
    4.3.1. 3D Point Correspondence ..................................... 57
    4.3.2. Point Cloud Registration ...................................... 60
  4.4. Final Registration Results for All Views ....................... 63
  4.5. Conclusion ................................................................... 65

Chapter 5: Refining Sparse Feature Models ......................... 66
  5.1. Refinement Methodology ............................................. 67
  5.2. Tier One – Pairwise Analysis ...................................... 69
  5.3. Tier Two – Global Feature Vector Analysis .................. 69
Acknowledgements

First and foremost, I wish to thank my co-supervisor Dr. Hamid Abdi. My deepest gratitude goes to you. Your dedication, enthusiasm and boundless knowledge are profound. You have helped me reach this milestone in my life, through gentle guidance and enlightened discussions. I cannot express how grateful and humbled I am for your encouragement and support during the last few months of my research journey.

I would like to thank my supervisors Professor Saeid Nahavandi and Dr. Asim Bhatti for their guidance, insight, confidence and support during my time at Deakin University. I feel honoured and privileged to have the opportunity to learn from their great wisdom and experience.

I wish to thank my colleagues at Deakin University. I have shared many great discussions, interesting projects, fun games and plenty of laughs. It is the people that make Deakin a great place to work and play.

To my friends, I am back. The incompatibility between PhD life and actual real life has been resolved. I'll endeavour to become a functional human being once more!

Finally, I would be nowhere without my family. I cherish the love and support that you send my way every day. Mum and Dad, thank you.


# List of Figures

2.1: Milgrams and Kishino’s reality-virtually continuum ........ 12
3.1: Camera frame to image plane relations............................ 30
3.2: Projection of a point P onto two images planes of a stereo camera................................................................. 32
3.3: Second order Gaussian functions in xx, yy and xy directions .............................................................................. 37
3.4: Approximation of second order Gaussian functions in xx, yy and xy directions ................................................... 37
3.5: Haar wavelet filters to compute the response for the x and y directions ................................................................. 39
3.6: Orientation assignment.................................................... 39
3.7: The 4-by-4 quadratic grid consisting of 16 sub-regions, and a 2-by-2 sub-division of a sub-region ....... 40
4.1: Features, feature set, the k-th object coordinate system and the imaging device coordinate system ........ 44
4.2: Block diagram of the 3D SFM generation for the k-th object .................................................................................... 45
4.3: The virtual textured object and the virtual stereo camera system ................................................................................. 46
4.4: The left and right images of the first frame in the test dataset ..................................................................................... 47
4.5: Filtered SURF feature distributions for the left and right view in the first frame of the test data set ............. 50
4.6: SURF feature correspondence for the first frame in the test data set ........................................................................ 55
4.7: The 3D projection of the 2D corresponding points from Figure 4.6............................................................................. 57
4.8: XY plane view of the 3D point correspondence established between two 2.5D point clouds.................... 59
4.9: 3D perspective view of the 3D point correspondence established between two 2.5D point clouds........... 60
4.10: Two registered 2.5D point clouds........................................ 62
4.11: Final raw-SFM........................................................................ 64
5.1: Refined-SFM methodology............................................... 68
5.2: Occurrence rates for the first 150 features listed in frames 30 to 40 ................................................................. 72
5.3: The first corresponding pair in the 27th group set .......... 73
5.4: Cropped and zoomed regions highlighting the corresponding feature members in the 27th group set .... 74
5.5: The spread of grouped features and their details relative to the initial member ............................................. 75
5.6: Before and after perspectives of the 2.5D views generated for the first frame of the test data set ............ 78
5.7: Correspondence and merger of two refined 2.5D views ................................................................................ 79
5.8: A cluster of points that represent one object feature ...... 85
5.9: Descriptor elements for an object feature group and the boxplot of the distribution of the descriptor elements in this group ................................................................. 88
5.10: Comparison of a representative feature vs. the max/min ranges .............................................................................. 89
5.11: Error comparison of matching the respective representative feature vector to each of 53 groups ........ 91
5.12: Final refined-SFM ................................................................................................................................. 93
6.1: Sample images from the respective datasets .......... 98
6.2: The raw-SFM for the GingerBeer dataset .................. 102
6.3: The raw-SFM for the Nuts dataset ............................... 102
6.4: The raw-SFM for the Coffee dataset ............................ 103
6.5: The raw-SFM for the Wafer dataset ............................. 103
6.6: The raw-SFM for the Shapes ........................................ 104
6.7: The refined-SFM for the GingerBeer dataset ............... 106
6.8: The refined-SFM for the Nuts dataset ......................... 107
6.9: The refined-SFM for the Coffee dataset ...................... 107
6.10: The refined-SFM for the Wafer dataset .................... 108
6.11: The refined-SFM for the Shapes dataset .................... 108
6.12: The robotic system used to validate registration accuracy of both raw- and refined-SFM a priori data .... 114
6.13: Frames of reference in the test setup ......................... 115
List of Tables

4.1: Camera parameters for the virtual stereo camera system in Maya................................................................. 47
4.2: The size of the raw-SFM for the test dataset ............... 64
5.1: The members of the 21st group ($\hat{G}_{1,2}^{1}$) in the set $\hat{G}_{1}^{i}$ .......... 73
5.2: The size of the refined-SFM for the test dataset ........... 92
5.3: The size of the refined-SFM for the test dataset after point and vector refinement............................................. 92
6.1: Camera parameters for the real camera system ............ 97
6.2: Feature extraction results for each of the test datasets .. 100
6.3: Results of the selection procedure of the global feature vector analysis......................................................... 105
6.4: Size comparisons of the raw- and refined-SFM data...... 109
6.5: The raw- and refined-SFM element size per dataset ...... 121
6.6: The average percentage improvement in efficiency when matching features from each programmed test pose location .......................................................... 122
6.7: GingerBeer position accuracy .................................... 124
6.8: GingerBeer orientation accuracy ................................. 125
6.9: Nuts position accuracy .............................................. 126
6.10: Nuts orientation accuracy .......................................... 127
6.11: Coffee position accuracy .......................................... 128
6.12: Coffee orientation accuracy.................................... 129
Condensing A Priori Data

Augmented Reality (AR) renders virtual information onto objects in the real world. This new user interface paradigm presents a seamless blend of the virtual and real, where the convergence of the two is difficult to discern. Errors in the registration of the real and virtual worlds are common and often destroy the AR illusion. To achieve accurate and efficient registration, the pose of real objects must be resolved in a quick and precise manner. The research in this thesis addresses recognition based pose estimation and registration at its core, by condensing the a priori data used to represent real objects.
This chapter introduces the domain, research questions and novel contributions proposed and explored in this thesis.

1.1. Enhancing Our World with Augmented Reality

An augmented world is presented to a user through an interface such as a head mounted display (HMD) or tablet computer. To achieve the AR illusion, the relationship between viewing interface and the anchor on which to render information in freespace (the real 3D environment) must be calculated. This calculation of pose (position and orientation relative to the user) enables the world coordinates of the virtual content to be translated to match the real world coordinates of the render anchor so that the virtual content can be aligned or registered into reality. The term ‘registration’ refers to the precise alignment of one or several virtual coordinate system(s) to real world entities.

In the past, pose was estimated to be the actual orientation of a user’s head with regards to the surrounding environment [1]. The types of sensors used were typically hardware-based devices, including time of flight sensors, inertial sensors, mechanical linkages, phase-difference sensors and direct-field sensors [2-5]. These devices have the ability to resolve pose into fine resolutions, but are quite vulnerable to environmental anomalies and often tether a user to a limited working area.

Recent trends [6] in the AR community reinforce the departure from traditional hardware-based pose estimators towards those that process information from vision sensors. Vision sensors offer passive, detailed, non-invasive and low cost sensing of the natural world in a self contained package, without the need to engineer the surrounding environment [1, 3, 7].
There are two approaches of establishing a pose estimate from vision data:

1. via **Egomotion**
2. via **Recognition**

Egomotion establishes the 3D motion of a camera in freespace by monitoring visual flow or tracking salient but uncorrelated features in a scene frame by frame. Conversely, recognition estimates the pose of specific entities based on locally related and known features. Egomotion is a scene-based technique used to localise the pose of a camera from an arbitrary initial point, where as recognition detects and tracks local coordinate systems of independent, known entities in the scene relative to a current perspective. Egomotion and recognition are explored in greater detail in Chapter 2.

Egomotion-based systems only allow information to appear in user specified regions, with no synchronicity with objects in the real world. Pose established through recognition allows an AR system to register virtual information with context. A recognition system can perceive specific entities in an environment, and seamlessly augment information that directly corresponds to those entities. When a system knows what it is looking at, it can deliver contextual information to a user.

Humans have mastered this awareness. For example, a child can easily recognise a banana in a fruit bowl because at some prior stage in their life they have learnt the shape, colour, size, texture, and contextual circumstance (i.e. the fruit bowl) to be a true representation of what defines a banana. Although the human mind is a constantly learning with posterior experience, simplistically this pre-learnt data could be termed as a priori knowledge. A priori knowledge is assumed to be an accurate representation of the object, requiring no validation
or justification by further experience. People generally do not question the validity of their minds recognition of simple objects.

Imparting a computer system with a priori knowledge requires some anterior experience with the object. Typically, an offline learning stage is used to sample information from an object, which is stored in a database as a true representation of the object. When a recognition system runs online, the current data it is sampling from the world is referenced back to this database to see whether the object exists in the current environment. If recognised, the pose of that object can be determined through further processing. The accuracy of the pose estimate directly corresponds to the quality of representative data used for registration.

It has only been recently that raw processing power and efficient image processing algorithms have permitted real time pose estimation via recognition from passive vision data. Yet, even with the recent progress, the task remains challenging. Supplying robust and concise a priori data to a pose estimator is a critical step in achieving realistic registration in augmented reality applications, and is hence the focus of this research.

This thesis investigates the generation and refinement of a priori data for use in online pose estimation. Therefore it aims to address the data quality and subsequent performance efficiency in today’s a priori generation and recognition based AR routines.

1.2. The Aims and Importance of this Research

A priori data in the form of a sparse feature model (SFM) helps deliver metrically accurate, concise and descriptive object information that can be efficiently matched during online pose estimation. This thesis aims to…
generate and further refine SFM a priori knowledge in order to build compact, descriptive object representations, and to quantify how the refinement of this data will increase the time efficiency and pose estimate accuracy during online deployment.

This is research aim is significant because:

1. Approaches that use sparse feature model recognition have been shown to provide good recognition performance in the presence of clutter, noise, occlusion, perspective distortion and variable illumination [8].

2. Few current methods utilise complete local multi-view information to accurately build an object representation a priori.

3. Although it’s recognised that having more features in a dataset does not necessarily increase the accuracy of recognition [9], there has been little work in the augmented reality domain to retain distinctiveness of a database whilst reducing its cardinality during the database formation process.

4. Refining sparse feature a priori data is critical for efficient and accurate pose estimation, and the contextual augmentation of objects in augmented reality applications.

Further research is required to develop new offline a priori generation and refinement techniques to improve the efficiency and accuracy of online recognition based augmented reality. The original contributions to knowledge in this thesis are:

1. A methodology for the generation of sparse feature model a priori knowledge from multi-view data.

2. A methodology to refine sparse feature model a priori data using statistical analysis to analyse the persistence and strength of matching features across multiple perspectives.
3. The introduction of novel representative points and representative feature vectors to condense multiple features representing single object features into a unique 3D point, described by one unique descriptor.

4. Performance analysis of raw and refined sparse feature models during online pose estimation. This analysis shows that the refinement methods of this thesis significantly reduce the cardinality of an SFM thus improving matching efficiency, whilst also improving pose estimation accuracy.

1.3. Assumptions and Applications

Though this research aims to be extensible for many applications, there were two main assumptions put in place that influenced the overall implementation. The first is the choice of using a short-baseline stereo imaging system. For an AR application to be fully immersive, stereo perspectives conducive with a user’s vision system must be used. The main application of this work is to quickly and precisely register AR information onto stereoscopic displays; either head mounted or monitor based. From this premise, it’s assumed that...

... the imaging data used for offline and online processing will stream from two synchronised stereoscopic cameras, set at a horizontal baseline similar to that of a human vision system.

Stereoscopic telepresence provides important binocular disparity so that the depth distribution of entities within a scene is easily discernable [10-15]. The analysis of stereoscopic image data provides the important epipolar constraint, which allows for outliers (false correspondences) to be easily detected and rejected [10, 12]. Stereo cameras at a fixed short-baseline can be accurately calibrated together, resulting in more accurate correspondence and localisation
of interest points in 3D space [16]. Further more, when a recognition system runs online using stereo data, registration can be performed in a 3D-to-3D fashion, as opposed to the difficult 2D-to-3D registration problem of monocular AR implementations.

The second assumption that has shaped the implementation of the research in this thesis is the type of image information present in the input data, and the way in which the stereo camera system is used to capture the information. The main premise of the following assumption is that the objects being sampled are detailed and feature rich. Hence it is assumed that...

... the imaging data contains textural information suitable for blob and distribution based feature extraction, and that it is captured at a frame rate greater than an object’s motion to mitigate blur in the images.

This assumption is mainly driven by the choice of the Speed Up Robust Feature (SURF) detector and descriptor, described in Section 3.4 and in [17]. SURF uses regions of high pixel distributions to robustly classify points of interest in an image. The methods in this thesis are also extensible for other feature detection and matching routines. However, SURF is one of the best modern day methods of solving the correspondence problem, i.e. finding matching pairs of features in two images, and was hence chosen as the main feature detector to use.

The modality of capture was chosen to accurately reflect the typical capturing conditions for both offline and online inputs into the methods presented in this thesis. A typical offline-learning scenario involves the fluid pass of a stereoscopic camera system around an object. It would capture detailed images at a rate of around one image per 10° perspective of an object. The individual images of this streaming data
can be used as inputs into the a priori knowledge generation methods proposed in this thesis. When implemented into an online recognition based AR application, this a priori data can be efficiently matched to features streaming in from real time video, using 3D correspondence and 3D-to-3D registration.

It is possible for these assumptions to be somewhat relaxed. For example, a priori data generated in the form of a SFM can be used in an online monocular AR system, just with a greater time cost for robust pose estimation. The methods for generating this a priori knowledge are, however, dependent on short-baseline stereo data, in order to accurately localise and retain stronger and more persistent features across multiple views of an object. As mentioned, the use of SURF in the implementation of the methods presented in this thesis was a choice, and can be replaced with other feature classifiers if needed.

1.4. Road Map

This thesis investigates the generation and refinement of a priori data for use in online, recognition based augmented reality. Following this introductory chapter, the generation, refinement and analysis process is explored in the following chapters:

Chapter 2 outlines the evolution of pose estimation in online augmented reality systems, and highlights some previous methods of representing objects with a priori data. In this critical review of the literature, the existing theories on egomotion, recognition, and the generation and subsequent refinement of a priori knowledge are explored.

Chapter 3 establishes the general mathematics behind the computer vision and image processing techniques used throughout this thesis. The math behind the pinhole camera model, camera
calibration, two-view geometry and triangulation are summarised. General feature extraction and an overview of the SURF feature detector and descriptor method is also presented.

Chapter 4 introduces a novel methodology to generate a priori knowledge in the form of a Sparse Feature Model (SFM). The method details the construction of multiple 2.5D feature clouds from a short baseline stereo imaging system, and the 3D merger of these clouds into a unique, raw-SFM. This unique approach hybridises the best methods of 2D-to-3D and 3D-to-3D registration to further advance a priori generation from multi-view data.

Chapter 5 describes the process of refining a sparse feature model generated by the methodology in Chapter 4 into a more concise and computationally efficient a priori data set. Global statistical analysis is used to identify and retain strong and persistent features in the SFM. Representative points are introduced to approximate the optimal position of a feature from multiple overlapping corresponding points. The descriptive information from these overlapping points is used to build representative feature vectors to efficiently characterise an object feature with a hybrid vector.

Chapter 6 validates the proposed methods of Chapters 4 and 5 in two parts. Firstly, multiple raw- and refined-SFMs are generated for real objects, and the reduction in cardinality and data footprint are compared. Secondly, the efficiency and accuracy improvements of the refinement of sparse feature model a priori data vs. raw sparse feature model a priori data are analysed with an online pose estimation system.

Chapter 7 concludes the body of this thesis, with a look at future research opportunities for the proposed methods in this thesis.
Modern Pose Estimation

A stable anchor on which to render virtual information is essential for credible virtual- to real-world registration in Augmented Reality systems. This requires the determination of a coordinate system of interest in freespace with respect to the imaging device, e.g. the position and orientation of an object in a scene. An AR system uses this information to register virtual information into a user’s display. This chapter explores the evolution of pose estimation within the domain of image processing, and computer vision and how the quality of a priori knowledge is essential in determining pose for recognition based AR applications.
2.1. Genesis

The most common immersive, single user interface to augment a user’s reality is the head mounted display, or HMD for short. There exist two types: optical see-through devices [18-20], which project data onto a see-through medium, i.e. glass or plastic; and video see-through devices [21, 22], which stitch virtual content into a captured image stream and then display this information to a user on a video screen.

The first HMD to show real-time, computer-generated imagery on an optical see-through display projected 3D wireframe images onto prisms containing half-silvered mirrors. This allowed the user to see both the images from the cathode ray tubes and objects in the room simultaneously and in stereo [23]. This defining moment in HMD display technology featured head tracking and displayed content in real time, and is considered to be the birth of augmented reality [1, 24-27]. Though display technologies were being experimented with, it took some time for pose estimation technology to mature to a point where AR was feasible on consumer hardware.

Pioneering AR applications emerged in the early 1990s, with Caudell and Mizell describing an augmented manual manufacturing processes in 1992 [28], Bajura et al. overlaying ultrasound imagery onto a patient in 1992 [18], and Feiner et al.’s 1993 [20] system that projected maintenance instructions onto a laser printer. Advancements in processing power and visual tracking algorithm design were crucial enablers for these applications. They demonstrated the potential of augmented reality technology in real world scenarios whilst running on consumer hardware.

In 1994 Paul Milgram and Fumio Kishino [29] conceived a reality-virtuality continuum, outlining the transitional blend of virtual and real
world objects presented in one display, shown in Figure 2.1. Ronald Azuma [24] highlighted Augmented Reality at the centre left of Milgram’s continuum when forming the first comprehensive survey of AR research and applications in 1997, subsequently updated in [1].

![Figure 2.1: Milgram and Kishino’s reality-virtuality continuum (adapted by [1]).](image)

Within this paper, Azuma defined an AR system in a way that encompassed not only head mounted displays, but also monitor-based interfaces, monocular systems and see-through HMDs. In 2007 Michael Haller et al. [30] extended Azuma’s 1997 [24] definition such that an AR system has to fulfil the following three characteristics:

1. It combines both the real and virtual content.
2. The system is interactive and performs in real-time.
3. The virtual content is registered with the real world.

These characteristics separate Augmented Reality from Mixed Reality. Today, these terms are often mistakenly interchanged. The extent of real to virtual world registration is a clear divisor of MR and AR. On Milgram’s continuum shown in Figure 2.1, mixed reality encompasses all blends of real and virtual environments. However, as a subset under the MR domain, AR has strict requirements for seamlessly blending the virtual and real worlds. In AR the convergence of the two worlds should be imperceptible; however in MR applications this requirement is often quite relaxed.
For the seamless registration of virtual information onto real world entities, the position and orientation of a coordinate system for an entity of interest must be resolved. There are two approaches of establishing a pose estimate in real-time AR systems:

1. *via Egomotion* (Section 2.2)

2. *via Recognition* (Section 2.3)

Egomotion tracks the displacement of salient features frame-by-frame or frame-to-map to establish the 3D motion of a camera in freespace. Conversely, recognition detects and tracks the position and orientation of independent, known entities relative to a current perspective.

### 2.2. Egomotion Based Pose Estimation

Estimating object or scene's pose is an intensive process to run, so a common approach was to first detect an initial pose and then track and update the estimate through time. Egomotion (related to structure from motion and visual odometry in mobile robotics) establishes the pose of a camera from its 3D motion through freespace. There are two forms of egomotive systems that either rely on recursive estimates (Section 2.2.1), or require posterior information (Section 2.2.2).

#### 2.2.1. Recursive Estimates

Estimating pose via recursion or ‘chained transformations’ [31] was one of the first purely passive, ‘markerless’ augmented reality solutions. These systems can only access information obtained online, having sole reliance on matching information between temporally adjacent, pairwise frames. An initialisation stage is typically required to establish a ground truth from the scene. Once a stable initial configuration is met, features are followed recursively using information
from the preceding frames to estimate the state of the features in the current.

This method was long used [3, 32, 33] for its efficiency; i.e. given a fast enough frame rate compared to object motion, the search window for corresponding features narrows considerably. However, a major problem arises when dealing with long sequences. As all prior locations are implicitly located from the features of the previous frame, drift inevitably occurs due to an accumulation of error inherent in the recursive nature of this Bayesian technique [3, 34, 35]. Without some form of absolute information from which a reliable pose can be inferred, the tracker will eventually fail.

2.2.2. Using Posterior Knowledge

Tracking and resectioning a camera via its motion through an unknown environment can be considerably aided with the addition of posterior knowledge. During the online operation of an algorithm, a map of the surrounding environment is created and continually updated and reinforced with any new or superior data. This form of knowledge is known as *a-posteriori* (or posterior to) as the map’s validation is dependent on experience, and cannot be justified in relation to a known prior.

The primary example of this is Simultaneous Localisation and Mapping (SLAM) [36], successfully used in mobile robotics to localise a robotic platform in an unknown (but feature rich) environment. SLAM generates a scene model online from landmarks present in the visual field, starting at an arbitrary initialisation point. The position and orientation of the imaging device within the feature map is simultaneously localised using egomotion. See [37, 38] for more information.
Klein and Murray’s Parallel Tracking and Mapping (PTAM) [39] variant of this form successfully brought SLAM fundamentals into an AR tracking environment. The system runs parallel mapping and tracking threads, initialised by translating the camera to create a pseudo stereo pair. Parallel threads allow the map to be optimised using bundle adjustment, decoupled from the constraints of real time localisation. Keyframes are created and registered to the map online when certain acceptable conditions are met, to allow for easy detection when the tracker returns to a previous location. New points are initialised and added to the map using an epipolar search. Klein and Murray successfully ported PTAM to run on a camera phone in [40].

PTAM was originally designed to support tracking in unknown environments, limiting the user to where virtual information could be rendered. Typically, an arbitrary plane was established as a render platform. Robert Castle [41] extended PTAM by integrating a-priori information with the tracker. Castle extended his 2007 work [42] to include pose determination of planar targets by triangulating SIFT [43] features of the target to those in PTAMs keyframes [44]. This made PTAMM object aware. Similar work has been published by Lee and Hollerer [45]. In 2008 [46], Castle also built upon PTAM to include a multiple sub-mapping ability. This allowed users to move freely between multiple pre-mapped areas, re-establishing tracking after large motions. Multiple mapping PTAM was also incorporated in [44].

Algorithms that maintain and revisit posterior information typically assume that the scene is static – essentially one big object. If the scene dynamics change, the ability to resolve pose from past experience is compromised; i.e. a car moving in a previously mapped urban environment will alter a prior mapped area significantly.
2.2.3. Issues with Egomotion

Temporal tracking systems must be updated frequently to ensure that the refresh rate is greater than object or scene motion. With early systems, this was a large problem. Any motion blur or radically changing scene structure would cause tracking failure. Often the discernibility of an egomotion tracking system was directly limited by the amount of features a system could track frame by frame. Hence, it was simpler to estimate the motion of the camera to the scene rather than estimate the pose of specific objects in relation to the camera.

One major problem with egomotion based AR systems is that registration is typically ambiguous. The virtual coordinate system established does not directly map on to a specific entity. For example, PTAM [39] arbitrarily assigns or requires a user to specify a plane in the camera view on which to render virtual information upon. Objects can traverse or be offset from this plane, however they will not be able to be precisely mapped to a specific object on interest in the scene. This is a perfect example of a mixed reality application that does not have the registration characteristic of AR imposed. For precise registration, recognition of real world entities is required.

2.3. Recognition Based Pose Estimation

Through recognition, a system can perceive known entities in an environment, and seamlessly augment them with contextual information. The recognition component of these AR systems is often termed ‘tracking by detection’ [47, 48]. These systems use data or ‘models’ that have been pre-learnt. Barandiaran et al. [47] implemented a recursive tracker and a model-based tracker, concluding that, while the recursive tracker easily failed, tracking by detection of some a-priori information was robust.
A-priori (or prior to) knowledge is assumed to be an accurate representation of an object or scene. It requires no validation or justification. A-priori information allows a pose estimation system to recognise a known entity in a scene and estimates its pose. A-priori data can be of several forms: planar targets [49-55], full 3D CAD models [56, 57], pseudo CAD models [35, 58], line models [59-61], sparse feature models [62-64] etc.

This section explores two common approaches for utilising a-priori data in an Augmented Reality system: engineering the environment with known landmarks (section 2.3.1) or learning object or scene features or geometry in an offline, pre-processing stage (section 2.3.2).

### 2.3.1. Using Markers

The first vision based augmented reality systems required simple landmarks that could be easily detected and tracked. They used these landmarks to relate visual information extracted from a scene to a known prior in order to estimate pose. This prior would contain all the information required for a positive match, including geometry, scale, patterns and/or colour. The manageable computational expense and closed-loop operation allowed marker based systems to become the first prominent vision based technique for AR systems well before natural feature detection and tracking [7].

Markers can be classified as either non-fiducial or fiducial based on their type of information they can supply to an AR system. Non-fiducial marker based systems use simple patterns that act as a point of reference in an image, and typically require a particular viewing or measurement condition, e.g. a bar code will only work in a particular orientation. Though non-fiducial markers can be detected in an environment, the information that they hold cannot be used to determine their precise pose.
Conversely, fiducial marker systems use image processing and pattern recognition to find a distinctive arrangement of black and white, or coloured dots, blocks, lines and/or regions of an active target in an image. This information, e.g. the corners of a black square and the structure of an enclosed pattern, can be extracted from an image and compared directly to a known prior. With a match, the geometry of the prior and the current state of the marker in the scene can be used to calculate accurate pose. Markers are very susceptible to failure from occlusion of any part of the marker, and become difficult to detect in cluttered environments. However at the time of their introduction into the AR domain, the computational cost benefit far outweighed the negatives.

The most prominent marker based system in the AR community is Mark Billinghurst’s and Hirokazo Kato’s ARToolkit. ARToolkit was developed in 1999 [65] at the Human Interface Technology Lab (HITLab) in the University of Washington. ARToolkit uses planar markers together with global template matching to reliably extract large single features from which pose can be extracted. It brought a fast and simple to use augmented reality solution, freely accessible to the research community, and could be deployed on consumer level hardware. Similar toolkits such as ARTK+ [66], MXR [67], ARTag [68], and Reactivision [69] have also gained momentum in the AR domain.

The main problem with marker-based systems is the intrusive nature of the markers in an environment. For some restricted applications this may be acceptable, i.e. in an industrial environment where the presence of a marker will not impact on the function or aesthetics of the plant. However, for augmented reality to become pervasive in society this intrusion into the environment is not acceptable.
2.3.2. Passively Creating A Priori Knowledge

Tracking natural features as they traverse through a scene invokes many challenges. As the environment is not engineered [1, 6, 70] for their benefit, these systems have to extract visual cues from image data passively [3, 70, 71] and match them to an a priori database of known features. The first systems used local patches of pixels, e.g. a unique arrangement of pixels on some planar target, or global ‘keyframe’ images to represent known viewing angles. Keyframes can be quite limited in their flexibility as they operate on a global scale, i.e. a whole view of a scene must be matched. Local patches are robust to partial occlusions and have higher recognition rates.

Keyframe based techniques have similarities to most markerless based systems that perform pose determination on planar targets with natural features. Where planar-based trackers determine pose of a known object based on some planarity constraint, keyframe techniques estimate pose of the current view in relation to a known keyframe configuration.

Vacchetti et al. [35] combined offline information from one or more keyframes, together with online information from the current and previous frame in a model-based, monocular, real-time 3D object tracker. Their system incorporated an offline training stage in which keyframe views were registered to a ground truth: a 3D CAD model. Vacchetti et al. [35] used this data to specifically eliminate tracking error inherently accumulated in chained transformations. This was one of the first systems to integrate a-priori information in this manner. This merger of techniques was invariant to large camera motions, whilst also being free from jitter. However, as the keyframes were related globally, the system had to generate ‘online’ keyframes when the camera views underwent an aspect change quite different from the
original keyframe set. Hence, the dataset restricted the possible motion of a camera during online operation.

Lowe [72] introduced a method called ‘view clustering’ that integrates any number of training images from different locations around a view sphere into a complete object model view. Images of similar viewpoints were grouped by the quality of feature matches into a single model view. The object model view consisted of several single model views, associated by matching features between sets.

Schaffalitzky and Zisserman [73] implemented a similar methodology to Lowe [72] to spatially organise multiple unordered views of a scene into similar view clusters. They used the ‘now standard’ [73] wide baseline approach of matching invariant descriptors between images using a binary space partition tree. After a clean up stage to remove incorrect matches and outliers between the sets, a greedy algorithm was used to join a subset of images together.

Takacs et al. [74] focus on handling large databases of reference images for image retrieval on mobile phone using a server/client model. As the processing power and memory on mobile device is prohibitive for large scale, location based image retrieval, Takacs et al. used the physical location of the device and the current camera image to find a set of reference images from the database that match the current view. This occurred through the matching of current viewpoints from the device’s camera to features that have been extracted and processed a priori on a server from the reference images.

The methods of Lowe [72], Schaffalitzky and Zisserman [73] and Takacs et al. [74] are useful for spatially organising or retrieving images relevant to a specific viewpoint of an object. Though they match and cluster features points between similar, they do not store this point information directly to form a priori information.
Techniques from the statistical classification domain have been applied to natural feature detection and tracking by Vincent Lepetit [50] and further extended by Mustafaf Ozuysal [52, 75]. They treat wide baseline matching of local patches as a classification problem, in which each class corresponds to the set of all possible views of such a point.

Given one or more images of a target object, their system synthesizes a large number of views or image patches of individual keypoints to automatically build the training set. This method reduces the global template matching approach of keyframe-based techniques to a local patch classifier. This work was implemented into the Garfield module of the BazAR Toolkit [76] to quickly detect and register pre-learnt planar objects in images. Wagner et al. [77] modified randomized ferns and SIFT feature descriptor [43] to enable real-time detection and tracking on low power devices.

2.4. Sparse Feature Model A Priori Knowledge

Rather than matching keyframes or local patches, some systems aim to build a-priori information directly from pixel level features like [17, 43] or as described in [63, 78, 79]. Often these features are processed in multi-view object learning systems that follow the notion of Rothganger et al. [54] in which the integration of features from multiple views is more complete and robust than any single view. Using a-priori information built from multiple views greatly assists recognition and pose estimation, and allows for auto-initialisation and recovery from failure [62].

Multiple-view, feature based methods generate data sets that contain the location and descriptive information of local points of interest, and are hence considered sparse. Throughout this research, these data sets are termed Sparse Feature Models (SFM) [80], and are the main focus of this thesis.
Irschara et al. [81] use large, city sized points clouds to recognize the location of a camera in a known area. Though the application is technically egomotive in nature, Irschara et al. demonstrate the construction of an SFM from the extraction of many interest points from multiple views. Using calibrated single cameras and the methods described in [82], they extract and localise SIFT features from multiple similar perspectives. Once again, the wide-baseline technique requires the estimation of the rotation and translation of each reference image for every matching pair, however using calibrated single cameras helps them obtain greater accuracy over other unordered views (i.e. [73]) at a cost of extra calibration and processing time.

In order to keep the SFM manageable due to the large number of views used, Irschara et al. [81] applied a mean-shift clustering technique [83] to quantize all the SIFT descriptors that represent one point. This compression technique results in a reported 40% reduction in the SFM size, speeding up matching time and lowering the SFMs data footprint. As their clouds are in the order of hundreds of thousands to millions of points, this reduction is significant.

Given the large scale of the SFMs of Irschara et al. [81] the data is only useful for determining the 3D motion of a camera through the cloud. Similar methods can be applied in a more contained manner to build sparse feature models for local objects. A local object SFM represents the 3D distribution of sampled points of interest that lie on an object. These points can be generated from the appearance of an object’s texture and geometry. Constructing an SFM requires the extraction and localization of the most robust and identifiable features from an object into a common coordinate system.

In 1999 Piotr Jasiobedzki [84] presented a method for determining the 3D pose of a known object for autonomous space hardware systems
from local object SFM data. In his system, an a priori model was represented as a hierarchy of line segments that had been triangulated from a stereo camera. In order to match current scene data to the model, a triangular mesh of the a priori data was iteratively matched to the scene structure in 3D. This was one of the first examples of 3D point data derived from real 2D stereo images being matched in 3D to determine the pose relative to a pre-learnt prior.

Gordon and Lowe [62] built upon [73]’s framework, establishing multi-view correspondence of highly descriptive SIFT [43] features from unordered images to generate a metrically accurate 3D model of an object and all its feature locations. Due to the nature of this wide-baseline method, the similarity between the unordered views must be established before points can be triangulated. To improve the matching speed and eliminate the matching outliers, they employed a Best Bin First (BBF) search algorithm on image features (ordered in k-d trees) to find intra-image correspondences.

Epipolar constraints were considered to clean up the outliers from this matching stage. However, due to time constraints, the epipolar constraint was only considered for images that had a high number of matches. Selection of the images with high match was performed using the greedy algorithm of [73] on the spanning tree of the image set. RANSAC [85] outlier elimination was used to further eliminate the outliers. From the final correspondence sets, Gordon and Lowe estimated the projective parameters between views and established the 3D location of the points using the Levenberg-Marquardt algorithm [86].

Gordon and Low provide promising results, however the wide-baseline methodology can limit its effectiveness as the estimation of camera parameters and point locations must occur for every image pair. This
can introduce further uncertainty in the localisation of the points. In their work, imprecise feature localisation resulted in inaccurate pose estimations (jitter) when tracking online. Hence, later in the paper [62] they discussed jitter reduction to solve this symptom.

More recently Fenzi et al. [87] used a wide-baseline, structure from motion approach to generate a sparse feature model of SIFT features from multiple single view training images. Similar to Irschara [81], mean-shift clustering was used to compress the cloud in the high-dimensional feature space. Though the focus of Fenzi et al.'s paper [87] mostly focused on an online recognition system, the description of their offline methodology is a prime example of the limited attention given to a priori SFM generation.

Most existing methods in the literature today use a wide-baseline, structure from motion method to determine the arrangement of multiple training images around an object. The main issue with wide-baseline approaches is accuracy in triangulating feature points. Images taken from calibrated hardware, e.g. [81, 87], claim to generate more stabilised and geometrically accurate a priori data those who use uncalibrated hardware, e.g. [62, 73].

In the research of this thesis, short-baseline stereo has been used to eliminate the need for establishing the relationship between image pairs that can introduce error in triangulation. As the camera parameters of a calibrated short-baseline camera only have to be established once and the stereo views are always of similar appearance, matched points can be triangulated with higher accuracy.

The choice of using stereo is motivated by the fact that for augmented reality on a HMD to be immersive, one requires a stereoscopic view of the world. As multiple camera streams should be present, it is foolish to not take advantage of the additional perspectives. The a priori
methods presented in this thesis use a short-baseline stereo system for offline SFM generation, but as the data locates 2D image features with 3D geometry, the a priori knowledge can be used for both 2D-to-3D and 3D-to-3D registration online.

2.5. Refining A Priori Knowledge

It is generally accepted [9] that having more features to represent an object does not necessarily increase the accuracy of recognition. In fact, it can be detrimental in terms of efficiency and accuracy especially when matching to a large database in real time. Limited work has been done within the AR domain to condense an a priori database (i.e. reduce its cardinality) whilst retaining its uniqueness.

As previously mentioned, Irschara et al. [81] and Fenzi et al. [87] use mean-shift clustering [83] to compress their features. The mean shift is defined by the difference between a point in the feature space and the weighted mean of feature points in a sphere around that point. The method converges like points to dense points (hills), and was proven mathematically and through several experiments. The method assumes an interest point as a dense point or hill and iteratively updates it through the mean shift method. The main advantage of this method is that it is parameterless, hence suitable for many applications. For the case of this thesis, the feature set is clustered from known correspondences through sum of squared differences (SSD) feature matching, therefore parameterless clustering is not suitable.

An interesting note from [74] is the way they cluster features across similar images on the server processing side. Features are extracted and matched between all the reference images from a similar view. For features that span multi-images (with a high occurrence rate), a ‘meta-feature’ was constructed by averaging and renormalizing the descriptors of that group. Unlike mean-shift, this method is simple and
non-iterative, however it must conform to the assumptions inherent in the features used.

Baheti et al. [8] demonstrate how the reduction of the number of features in a database, using an information theory approach, can equal and improve the accuracy online matching to an a priori database. They have offered three levels of pruning including intra-object, inter-object pruning, and key points clustering to reduce the cardinality of a database of multiple objects after the feature points have been extracted for all input data.

For intra-object pruning, the most relevant work to this thesis, a group of matching feature vectors representing one object feature (sampled from multiple views) was established based on the Euclidean distance between the points. One vector from this group was chosen to represent the group. Additionally, all the x,y point locations, scale, object ID and view ID are retained for geometric checks later on. Each descriptor was given a weight based on the ratio of the number of descriptors representing one point vs. the total number of descriptors across all views of one object.

More recently, there is new research on compressing current features descriptors or defining new feature descriptors with smaller dimensions. Takacs et al. [74] briefly address the dimension reduction of a group of SURF features by mapping them to lower dimensional space. Chandrasekhar et al. [88] introduce CHoG, a feature descriptor built on gradient histograms. CHoG has a high level of compressibility, without the need for decoding. Even for these compressed and low dimensional features, a method of pruning less consistent and less accurate features in a database is valid as, once again, having more features in a dataset does not necessarily increase the accuracy of recognition [9].
2.6. Conclusion

Augmented Reality is an application domain for pose estimation technologies. Solving the registration problem for real-time systems is one of the most popular and challenging problems within the imaging community and is yet to have a general unified solution. Furthermore, as most research in markerless systems is driven by applications of the technology, published solutions are often customised and unscaleable, permitting many solutions for similar tasks.

Generating compact, descriptive object representations for recognition based AR is a solution for time inefficient and inaccurate registration. Approaches that use sparse feature model recognition have been shown to provide good recognition performance in the presence of clutter, noise, occlusion and perspective/illumination [8]. A priori data in the form of a SFM helps deliver metrically accurate, concise and descriptive object information that can be efficiently matched to online visual information.

There are many gaps in the methods of a priori knowledge generation for online recognition tasks. Few methods utilise complete local multi-view information to accurately build an object representation a priori. Although it’s recognised that having more features in a dataset does not necessarily increase the accuracy of recognition [9], there has been little work in the augmented reality domain to retain the distinctiveness of a database whilst reducing its cardinality during the database formation process.

Further research is required to develop new offline a priori generation and refinement techniques to improve the efficiency and accuracy of online recognition based augmented reality. This thesis aims to fill this gap by generating and further refining sparse feature model a priori knowledge.
“There is geometry in the humming of the strings, 
there is music in the spacing of the spheres”
Pythagoras

Computer Vision Mathematics

The primary concern of this thesis is the analysis of multiple-view imaging data. The mathematical nature of multiple-view computer vision and image processing is a mature topic of research [16, 89, 90]. Following an overview of the calibration of short-baseline stereo cameras, the mathematics behind two-view geometry, triangulation and registration in this chapter introduce the fundamental principles behind the research in this thesis. General feature extraction and an overview of the SURF feature detector and descriptor method is also introduced.
3.1. Stereo Camera Calibration

A short-baseline stereo system consists of two spatially displaced monocular cameras, typically located horizontal to each other at the same height. The cameras can be generalised as two pinhole type cameras [16]. In order to accurately use the image information from the camera system, they must be calibrated.

Calibration mathematically defines the true imaging properties and structure of a camera system that produces a given image set. Intrinsic parameters define the internal optical and sensor specific characteristics of each pinhole type camera in a stereo pair. For a short baseline stereo camera, extrinsic parameters denote the orientation of one camera to the other.

3.1.1. Intrinsic Camera Parameters

In general, the camera coordinate system and the world coordinate system are related by a set of physical parameters specific to the imaging device. The internal or intrinsic parameters define the focal length, pixel size and principle point from the structure of the sensor and its location respective to the focal/image plane [16].

In Figure 3.1, the camera centre \(c\) denotes the coordinates of the optical centre in pixels with respect to the origin of the image. The line from the camera centre perpendicular to the image plane is called the principle axis, and its intersection with the image plane is called the principle point \(o_p = \left( o_x, o_y \right)^T\). Focal length \(f\) defines the distance between the optical centre of the camera sensor and the image plane. Lets call \(p_x\) and \(p_y\) the respective width and the height of a pixel on the chip. Thus, the focal length of a camera in terms of pixels is defined by:
$f_x = \frac{f}{p_x}$ and $f_y = \frac{f}{p_y}$.  

Figure 3.1: Camera frame to image plane relations

A ray projected from point $p$ intersects the image plane at $[u,v]$ on a path towards the camera centre as shown in Figure 3.1. The camera calibration matrix $K$ maps the $x,y$ value of the point $p$ in the camera coordinate system to a pixel value intersection $[u,v]$ on the image plane:

$$
\begin{bmatrix}
  u \\
  v \\
  1 
\end{bmatrix} = K
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
$$

(2)

where

$K =
\begin{bmatrix}
  f_x & s & o_x \\
  0 & f_y & o_y \\
  0 & 0 & 1
\end{bmatrix}$

(3)
The $s$ parameter denotes the amount of skew that a non-square pixel has. Modern production methods have all but eliminated pixel distortion, so generally $s$ is near zero. See [16] for more information on deriving these camera calibration parameters.

### 3.1.2. Extrinsic Camera Parameters

The extrinsic parameters of a single camera indicate the transformation of the 3D camera frame to the origin of a world frame, i.e. how a camera is oriented and positioned in freespace relative to any arbitrary reference points in a scene.

The extrinsic parameters denote a 3-by-4 transformation matrix that consists of a 3-by-3 rotation $R$ and 1-by-3 translation vector $t$ as shown by $\begin{bmatrix} R & t \end{bmatrix}$. The orientation component $R$ details the rotations about each 3D axes. The translation vector $t$ denotes the distance between the two camera centres.

Therefore a point in the space is first projected to the camera frame by $\begin{bmatrix} R & t \end{bmatrix}$ and then into the image plane by $K$. The combined projection by $\begin{bmatrix} R & t \end{bmatrix}$ and $K$ is shown by $P$ and is:

$$P = K \begin{bmatrix} R & t \end{bmatrix}$$

### 3.2. Two-view Geometry

The axioms of two-view geometry describe the intrinsic relationship between two images taken from slightly different perspective views of a 3D scene, highlighted in Figure 3.2. In this figure, the left and right image planes are shown in a 3D coordinate system $X,Y,Z$. A 3D interest point of $p=(x_k,y_k,z_k)$ of the k-th object has a 2D projection in
the left and right images denoted as \((u,v)\) and \((\hat{u},\hat{v})\) where the ray intersects the image plane on a path towards the camera centre. These 2D projections are obtained from the two projection matrices that map the interest point \(p\) to the left and right image planes. These projection matrices come from the camera calibration parameters (Section 3.1).

\[
\begin{bmatrix}
\xi_L \\
\eta_L \\
1
\end{bmatrix} = P_L \begin{bmatrix} p \\
1
\end{bmatrix}
\]
for the left image \( (5) \)

\[
\begin{bmatrix}
\xi_R \\
\eta_R \\
1
\end{bmatrix} = P_R \begin{bmatrix} p \\
1
\end{bmatrix}
\]
for the right image \( (6) \)

where \( \xi_L \) and \( \xi_R \) is the distance of the interest point from the focal plane of the left and right cameras respectively.

\( Figure 3.2: \) Projection of a point \( p \) onto two image planes of a stereo camera
3.3. The Mathematics of Triangulation

Triangulation localises a point in 3D space by analysing its 2D projections in a stereo pair (as in Figure 3.2). The projection points for an interest point \( \mathbf{p} = (x_k, y_k, z_k) \) for the k-th object were shown in Equations (5) and (6) as \((u, v)\) and \((\hat{u}, \hat{v})\) respectively. With the intrinsic and extrinsic parameters from the calibration of the stereo camera, triangulation can be used to calculate the position of \( \mathbf{p} = (x_k, y_k, z_k) \) from the locations of \((u, v)\) and \((\hat{u}, \hat{v})\), and the difference in disparities from the camera centres \( d_L \) and \( d_R \) in Figure 3.2.

The triangulation of sparse salient 2D image features is a little bit different from general dense disparity estimation in stereo image processing. Following the same rules, the sparse triangulation procedure should estimate the depth of matched points that have been localised with sub pixel accuracy. This can be achieved by merging Equations (5) and (6) in a homogenous Equation of \( \mathbf{A} \mathbf{x} = 0 \), where \( \mathbf{x} = \left[ \hat{\mathbf{p}}^T \quad w \right]^T \). \( \hat{\mathbf{p}} \) is the 3D position of the point, scaled by \( w \). The homogenous linear Equation \( \mathbf{A} \mathbf{x} = 0 \) can be simply obtained noting the cross product of any vector with itself is a zero vector. Therefore,

\[
\begin{bmatrix}
  u_i & v_i & 1
\end{bmatrix}^T \mathbf{P}_L \begin{bmatrix}
  \mathbf{p} \\
  1
\end{bmatrix} = 0 \quad (7)
\]

\[
\begin{bmatrix}
  \hat{u}_i & \hat{v}_i & 1
\end{bmatrix}^T \mathbf{P}_R \begin{bmatrix}
  \mathbf{p} \\
  1
\end{bmatrix} = 0 \quad (8)
\]
If the vectors of $p_{Lj}^T$ and $p_{Rj}^T$ are defined from the j-th rows of the known projection matrices $P_L$ and $P_R$, then the expansion of cross products in Equations (7) and (8) result to

$$ A = \begin{bmatrix}
    u_j p_{Lj}^{3T} - p_{Lj}^{1T} \\
    v_j p_{Lj}^{3T} - p_{Lj}^{2T} \\
    \hat{u}_j p_{Rj}^{3T} - p_{Rj}^{1T} \\
    \hat{v}_j p_{Rj}^{3T} - p_{Rj}^{2T}
\end{bmatrix} \quad (9) $$

where the first two rows of $A$ are associated with the left image and the second two rows are associated with the right image.

The non-zero solution of the Equation $Ax = 0$ belongs to null space of $A$, but because $A$ is a 4-by-4 matrix, in general it does not have a null space. In this case, $x$ can be obtained by minimization of the following least square Equation:

$$ \min \| Ax \|^2 \quad (10) $$

This optimisation is solved by the eigen vector associated to the minimum singular value of $A$. Hence,

$$ x = \begin{bmatrix}
    \hat{p} \\
    w
\end{bmatrix} = eigv(A) \quad (11) $$

for the minimum eigen value of $A$.

Finally the unscaled 3D position of the corresponding points $(u,v)$ and $(\hat{u},\hat{v})$ is obtained by:

$$ p = \frac{1}{w} \hat{p} \quad (12) $$
3.4. Extracting Features from Images

There are various considerations when selecting a suitable feature extraction method, including accuracy, distinctiveness and repeatability. Image features should be robust to rotation, scaling, illumination and perspective distortion. To achieve a more discernable and repeatable feature, researchers [2, 17, 43, 63, 78] have looked at ways of adding extra information after feature detection. A description stage constructs a high dimensional feature vector by sampling the pixel neighbourhood around a detected feature. If the vector is unique enough compared to the rest of the feature vectors, a descriptor is appended to the sampled feature. Substantially increasing the uniqueness of a detected feature with a descriptor returns a higher likelihood of a positive match during correspondence, however at a cost of time through the extra processing.

One such detector and descriptor scheme is Speeded Up Robust Features [17] or SURF for short. SURF has demonstrated remarkable repeatability, distinctiveness, robustness and efficiency when compared [17, 91] to other such features types like SIFT [43]. Though SIFT was the forbearer for descriptive feature matching, SURF has succeeded SIFT as a more robust and efficient description algorithm in recognition applications. For these reasons, SURF has been chosen as the feature extraction method in this work.

SURF uses a Hessian matrix based detector to find blob like textures in an image, and a distribution based descriptor to construct high dimensional vectors around detected interest points. The SURF descriptor is explained in [17], and is summarised in the following sections.
3.4.1. SURF’s Hessian Matrix Based Detector

**Integral Images**

The fast computation time of SURF interest points is largely contributed to the use of integral images. The intensity calculations for the box type convolution filters used in SURF are easily calculated once an integral image has been computed. An integral image $\text{Im}_\Sigma$ for an input image $\text{Im}$ is generated by

\[
\text{Im}_\Sigma(x, y) = \sum_{i=0}^{x} \sum_{j=0}^{y} \text{Im}(i, j)
\] (13)

The value of any pixel in the integral image $\text{Im}_\Sigma(x, y)$ at each point $(x, y)$ is the sum of pixels above and to the left of that point [17, 92].

**Hessian Matrix**

SURF detects blob-like structures at locations and scales where the determinate of the Hessian matrix is maximum [17]. Given a point $p = (x, y)$ in an integral image $\text{Im}_\Sigma$, the Hessian matrix $H(p, \sigma)$ in the space $p$ and at scale $\sigma$ is:

\[
H(p, \sigma) = \begin{bmatrix}
    l_{xx}(p, \sigma) & l_{xy}(p, \sigma) \\
    l_{xy}(p, \sigma) & l_{yy}(p, \sigma)
\end{bmatrix}
\] (14)

where $l_{xx}(p, \sigma)$ is the convolution of the Gaussian second order derivative with the integral image $\text{Im}_\Sigma$ in point $p$, and similarly for $l_{xy}(p, \sigma)$ and $l_{yy}(p, \sigma)$ [17, 92].
These Gaussian second order functions in $xx$, $yy$ and $xy$ are shown in Figure 3.3 (left to right) below:

![Figure 3.3: second order Gaussian functions in $xx$, $yy$ and $xy$ directions [17]](image)

These functions are convolved with integral images to produce $l_{xx}(p, \sigma)$, $l_{xy}(p, \sigma)$ and $l_{yy}(p, \sigma)$ in the Hessian matrix. Although the Gaussian second order functions are optimal for scale space analysis, they are discretised and cropped for the approximate SURF algorithm to make the calculations more efficient.

The SURF uses an approximate for the second order Gaussian functions, denoted $d_{xx}$, $d_{yy}$ and $d_{xy}$, and are shown in Figure 3.4.

![Figure 3.4: Approximation of second order Gaussian functions in $xx$, $yy$ and $xy$ directions [17]](image)

The approximation of second order Gaussian functions over the integral image using box filters allows the Hessian matrix to be computed at very low cost. The approximation for the Hessian matrix $\tilde{H}$ is obtained by applying a simple relative weight to the Hessian matrix as:
\[
\tilde{H} = \begin{bmatrix}
    d_{xx}(p,\sigma) & wd_{xy}(p,\sigma) \\
    wd_{xy}(p,\sigma) & d_{yy}(p,\sigma)
\end{bmatrix}
\]  
(15)

where \( w \) is a relative weight

The relative weight of the filter responses is used to balance the expression for the Hessian's determinant. This is needed for the energy conservation between the Gaussian kernels and the approximated Gaussian kernels. It has been shown in [17] that the appropriate value for the relative weight is 0.912, therefore

\[
\det(\tilde{H}) = d_{xx}d_{yy} - (0.9d_{xy})^2
\]  
(16)

The above determinant of the approximated Hessian represents the blob response in the image at location \( p \) [17].

### 3.4.2. SURF’s Distribution Based Descriptor

#### Orientation Assignment

The description stage in SURF samples the pixel neighbourhood surrounding a detected feature to create a high dimensional vector. This vector greatly increases the uniqueness associated with detected features, and allows like features to be filtered out of the final data set.

To assign a descriptor to a blob feature, the Haar wavelet responses in the \( x \) and \( y \) directions within a circular neighbourhood of radius \( 6s \) around the interest point \( p = (x, y) \) is calculated for different scales of \( \sigma \), where \( s \) is the scale at which the interest point is detected.

Figure 3.5 shows the Haar wavelet filters that are applied to the integral image, where the response in \( x \) or \( y \) direction is quickly calculated.
The wavelet responses are weighted by a second order Gaussian with $\sigma = 2s$. The responses are represented as points in a coordinate system centred at the interest point, with the horizontal and vertical directions aligned to the image coordinate system. The dominant orientation is estimated by calculating the sum of all responses within a 60° sliding orientation window [17], as shown in Figure 3.6. In this figure, the scattered blue points are the Haar wavelet responses for different scales. The red arrow indicates the assigned direction.
**Generation of the SURF Descriptor**

To build a 64 dimensional SURF descriptor, a quadratic grid with 4-by-4 square sub-regions is laid over the interest point. The quadratic grid is aligned to the orientation estimate calculated in the previous section. Each square of the quadratic grid is further divided into 2-by-2 sub-divisions, as shown in Figure 3.7, where the sub region squares and sub division squares are indicated.

![Figure 3.7: The 4-by-4 quadratic grid consisting of 16 sub-regions (left), and a 2-by-2 sub-division of a sub-region (right) [17]](image)

For each sub-division, the $x,y$ response of the Haar wavelet filters are calculated to obtain a vector located at the centre of each square. The horizontal and vertical components of these vectors in the coordinate system of the quadratic grid are depicted as $\rho_{xi}$ and $\rho_{yi}$, where $i = 1,2,3,4$. Based on these components, four values are calculated as

$$\sum \rho_{xi}, \sum \rho_{yi}, \sum |\rho_{xi}|, \text{ and } \sum |\rho_{xi}|.$$ (17)
These four values represent the actual fields in the SURF descriptor for one sub-region. With 16 sub-regions of the quadratic grid there will be 64 individual values for the SURF descriptor for any sampled interest point.

### 3.5. 3D Registration Fundamentals

Registration is an iterative procedure that merges two sets of point clouds into one single structure. A transformation that optimally maps the points of one cloud to corresponding points in another cloud must be obtained. The transformation follows the standard 3D rotation and translation designated \( [R \mid t] \). If the transformation matrix \( [R \mid t] \) is known, then we simply multiply the rotation \( R \) component of the transformation to a cloud’s points to orient them to the base cloud, and then add the translation \( t \) to merge the clouds together into one base coordinate space.

To determine an optimal transformation between two point clouds with no known relation, a set of at least 4 corresponding points of each cloud must be established. Assuming the set \( f^D \) from the first cloud and the set \( f^M \) from the second cloud include known corresponding points, the transformation that relates these clouds is given as

\[
f^M_i = R f^D_i + t + e_i
\]

(18)

where the \( e_i \) vector indicates the Euclidian difference between the projected point of the first cloud with its corresponding point in the second cloud.

An ideal transformation will result to zero difference in the distance between the projected points and their corresponding points in the second cloud. When clouds are noisy, the ideal is impossible; therefore
an optimal transformation that will produce minimum error is suitable. For this minimization, the sum of the square norm for all the pairs is used:

\[ \sum_{i=1}^{n} ||e_i||^2 \]  

(19)

for \( n \) points.

Therefore, the minimization problem is defined by

\[ \Psi = \min_{\hat{R}, \hat{t}} \sum ||p_i - \hat{R}q_i - \hat{t}||^2 \]  

(20)

There are different methods to solve this optimization problem. A solution presented by [93] has been implemented in this thesis, and is explained in Program 1, Section 4.3.2.

### 3.6. Summary

A mathematical background of computer vision theory, including the calibration of stereo cameras, the principles of two-view geometry, the math behind triangulation, an introduction into the inner workings of the SURF feature detector and descriptor, and the fundamentals of registration, have been provided in this chapter to prepare the reader for more complete understanding of the original and novel contributions in this work.

The methods introduced in the next chapters build upon the foundations presented in this chapter, to hybridise both 2D-to-3D and 3D-to-3D a priori data generation and registration procedures in order to generate sparse feature model a priori data.
Raw Sparse Feature Models

This chapter introduces a method of generating a raw Sparse Feature Model (SFM) to comprehensively characterize an object a priori. When used as a priori knowledge, SFM delivers the unique 3D position of many points of interest that lie on an object.

In this method, multiple short-baseline stereo images are captured from different perspectives around an object. For each stereo pair, a 2.5D point cloud is generated by the triangulation of corresponding, highly descriptive object features. A unique, raw-SFM is reconstructed from these multiple views by merging each 2.5D point cloud together using 3D-to-3D shape registration.
4.1. The Raw Sparse Feature Model

This chapter presents a method of generating descriptive a priori data in the form of a raw Sparse Feature Model (SFM). To construct a SFM, the k-th object $O_k$ from a collection of $n$ objects (classified as $O_1, O_2, ..., O_n$) is imaged from multiple perspectives using a short baseline stereo camera $C$. For each perspective, a group $F_k$ of features $f = [f_1, f_2, ..., f_m]^T$ is extracted, where $m$ is the dimension of the feature vector. Figure 4.1 shows a set of features $f$, grouped as $F_k$, with reference to the k-th object's coordinate system $(O_k, \hat{i}_k, \hat{j}_k, \hat{k}_k)$ and the imaging device coordinate system $(C, \hat{i}_c, \hat{j}_c, \hat{k}_c)$ in freespace.

![Figure 4.1: Features, feature set, the k-th object coordinate system and the imaging device coordinate](image-url)

---

44
To generate the feature sets for each perspective, a feature extraction method locates robust and repeatable interest points as $F^L_{k,i}$ and $F^R_{k,i}$, where $L$ and $R$ represent left and right images of the stereo pair, and $i$ is the $i$-th view of the $k$-th object. Correspondence between features in $F^L_{k,i}$ and $F^R_{k,i}$ is established for each $i$-th view. These corresponding features are triangulated to generate a 2.5D perspective view $M^L_{k,i}$. Finally, 3D shape registration is used to merge each 2.5D perspective view $M^L_{k,i}$ into a unified 3D representation $M_k$, termed the Sparse Feature Model.

If triangulation is shown by $\oplus$ and the multi-view registration process is represented by a union $\cup$ then

$$M^L_{k,i} = F^L_{k,i} \oplus F^R_{k,i}$$  \hspace{1cm} (21)$$

$$M_k = \bigcup M^L_{k,i}$$  \hspace{1cm} (22)$$

where $M_k$ is the SFM representation of the $k$-th object $O_k$.

This procedure is shown graphically in Figure 4.2.
### 4.1.1. Test Bed and Dataset Generation

Throughout Chapter 4 and Chapter 5, a synthetic dataset is used to demonstrate the behaviour of the following algorithms under controlled conditions. As the highlight of this research is the feature point processing methodology, the use of a consistent dataset helps emphasise the steps in the algorithm, rather than introducing abnormal behaviour from a real camera system.

A virtual textured object, shown in Figure 4.3, was generated in Autodesk’s Maya software [94]. A virtual short-baseline stereo camera system (shown in green) was created to render images from the virtual scene. Of the three cameras shown in Figure 4.3, only the left and right render images. The middle camera is used for visualisation, alignment and control assistance only. The intrinsic and extrinsic parameters for the virtual camera system can be precisely controlled, and are listed in Table 4.1.

![Figure 4.3: The virtual textured object and the virtual stereo camera system.](image)

This dataset had to be created, as there are no standard datasets available that contain multiple short-baseline stereo views around a single textured object.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Extrinsic Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>Om</td>
<td>[0 ; 0 ; 0]</td>
</tr>
<tr>
<td>T</td>
<td>[75 ; 0 ; 0]</td>
</tr>
<tr>
<td><strong>Intrinsic Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>Focal Length L</td>
<td>[1200 ; 1200]</td>
</tr>
<tr>
<td>Focal Length R</td>
<td>[1200 ; 1200]</td>
</tr>
<tr>
<td>Principle Point L</td>
<td>[400 ; 250]</td>
</tr>
<tr>
<td>Principle Point R</td>
<td>[400 ; 250]</td>
</tr>
<tr>
<td>α L (pixel skew)</td>
<td>0</td>
</tr>
<tr>
<td>α R (pixel skew)</td>
<td>0</td>
</tr>
<tr>
<td>Image Distortions L</td>
<td>[0 ; 0 ; 0 ; 0 ; 0]</td>
</tr>
<tr>
<td>Image Distortions R</td>
<td>[0 ; 0 ; 0 ; 0 ; 0]</td>
</tr>
</tbody>
</table>

Table 4.1: Camera parameters for the virtual stereo camera system in Maya. Om are vectors related to rotation via the Rodrigues formula [95]

The object was subjected to a complete 360° out of plane rotation concurrently in both pitch and yaw. A global stream of 100 RGB frames (left and right stereo images) was rendered from the stereo cameras at a resolution of 800x500 pixels per image for one complete rotation of the object. Along with the camera parameters, these 100 frames are the only inputs used in the proposed methodologies. The 1st stereo pair is shown in Figure. 4.4 below.

![Figure 4.4: The left and right images of the first frame in the test dataset](image-url)
4.2. 2.5D View Generation

Data from any single view of a three-dimensional object is not representative of the object as a whole [54]. This is a consequence of self-occlusion, where the object’s geometry inherently obstructs information from a single perspective. For example, there is a region of the moon that permanently faces away from our viewpoint from earth. Man did not know what was on the far side of the moon until the Soviet space probe Luna-3 [96] returned images of it in 1959. Additional perspectives were needed to form a complete map.

Due to occlusion, the 3D data obtained from a single stereo pair is termed as a 2.5D representation (or view) of an object. Later in this chapter (Section 4.4) multiple 2.5D views taken from different perspectives are merged into a single 3D object representation. This section overviews how image information is extracted and classified as features, how these features are matched between images, and how these matches are filtered and triangulated to create a 2.5D view for one stereo pair.

4.2.1. Pre-processing and Data Structures

A key advantage of camera based a priori generation and object recognition is the passivity of the approach. Objects are characterised and recognised in real world configurations without the need to engineer the environment or to use active sensors or landmarks. For this methodology, only the camera parameters and the stereo images are required as inputs.

With the intrinsic and extrinsic matrices of the stereo cameras $C$ (Figure 4.1) known (Table 4.1), pre-processing requirements for the dataset inputs are few. The most common manipulation of the input data is to convert the stereo images to intensity or grey-scale images
[97]. In the case of the test dataset, the images were originally rendered with the RGB colour model, where the chromaticity of each pixel is defined by a Red Green and Blue component.

Through the development of this thesis an evolved data structure has been used to contain the input information, associated parameters and outputs for each frame (stereo pair). This data structure includes an array of these records. Each record has the following components:

- Original left and right grey-scale images for the frame
- A structure that contains the geometric information about the extracted points in the frame
- An array of the associated descriptive information associated to each feature extracted in the frame.
- The final parameters (with updates if any have been modified)

This data formatting compartmentalised all the framewise information, simplifying and improving the code that was developed during the progression of this research.

4.2.2. Feature Extraction

As the input images are obtained passively, the only information that can be extracted is held in each 2D perspective of the 3D world. For each i-th stereo pair, the SURF algorithm (Section 3.4) is used to generate feature sets $F^L_{k,i}$ and $F^R_{k,i}$, for the left $L$ and right $R$ images (Figure 4.2). Each salient blob feature in any of the left and right images is assigned a 64 dimensional descriptor based on its surrounding pixel neighbourhood. Figure 4.5 shows the filtered SURF feature distributions for the left and right view in the first frame of the test data set (Figure 4.4).
Figure 4.5: Filtered SURF feature distributions for the left and right view in the first frame of the test data set (Figure 4.4). The + in the green circles indicates the location at which a SURF feature has been extracted. The size of the circle indicates the scale of each feature. The orientation for each feature is shown by the directional line in each corresponding circle.
Originally this work was developed with Petter Strandmark’s SURFmex [98] implementation of Bay’s [17] original SURF algorithm. SURFmex is a MATLAB® interface that uses precompiled C++ binaries and OpenCV. The closed nature of this implementation made it difficult to manipulate the code to fit this methodology. Data formatting and structure control were similarly restricted.

One of the new features in MATLAB 2011b [99] was the integration of SURF as a core MATLAB function in the Computer Vision System Toolbox. This allowed greater control over the behaviour of SURF and possible internal modifications to fit this methodology. Also, this implementation allowed for better management of the SURF point information generated. The parameters and SURFPoints structure that contains the SURF point data are described below.

**Blob Parameters in the code:**

Three tuneable parameters are required when calling the SURF feature detection routine. A metric threshold controls the number of points that are returned. The number of octaves and the number of scale levels control the size of the scale-space used by the Hessian matrix.

SURF classifies blob interest points based on the maximum of the determinant of the Hessian matrix described in Section 3.4.1. When this maximum is greater than a specified threshold, a blob feature is extracted. The threshold is a positive integer parameter that is set at 1000. A reduction in this threshold will yield more detected interest points.

The number octaves specify the range that the approximate second order Gaussian functions are applied. An octave is a discrete jump in the scale-space for which a number of filters are applied. In this implementation, 3 octaves are used. The number of scales, i.e. the size
of these filters, is set at 4. Hence, 27-by-27, 51-by-51, 75-by-75, and 99-by-99 sized filters are used. Higher octaves use larger filters to find larger blobs. Increasing the number of scale levels increases the number of blobs detected at finer scale increments per octave.

**SURFPoints Data Structure**

MATLAB returns a SURFPoints data structure for any sampled image. It contains the following entries:

- **Location**: The x,y location of a feature in the image.
- **Count**: number of points held in the structure
- **Scale**: specifies scale at which the interest points were detected.
- **Metric**: a value that describes the strength of the detected feature. This value is specified by the determinant of the approximated Hessian used for each feature.
- **Sign Of Laplacian**: is specified from the trace of the Hessian matrix. This helps identify when a dark blob on a light background or vice versa is detected.
- **Orientation**: describes the orientation of a feature (see Section 3.4.2). This angle is measured from the X-axis with the origin at the point’s x,y location and is given in radians.

**4.2.3. Feature Correspondence**

Each image in a short-baseline stereo pair is captured synchronously with its partner with limited perspective distortion, hence determining feature correspondence is far easier than in wide-baseline methods. After extracting features for each of the left and right images, the feature correspondence block of Figure 4.2 finds feature matches between each image of a stereo pair. There are different methods to
calculate correspondence, though matching high dimensional data like the SURF descriptor is time consuming. The previously established methods for the correspondence of simple features do not perform efficiently for high dimensional data.

Linear methods try to establish the best match for each feature, for example, in the left image with all features in the right. For a small number of simple features, linear methods will return the best answer, however they become extremely time consuming when dealing with large amounts of features [62], especially if the matching stage has to deal with large vectors. More advanced binary search structures like k-d trees and variants [62, 100] allow searches in large data sets to be implemented with great efficiency for simple features. These structures often have trouble dealing with high dimensional data, potentially deteriorating to a time cost equivalent to a linear method.

Approximate nearest neighbour searches can run significantly faster for high dimensional vectors than linear and nearest neighbour methods. Muja and Lowe’s [101] Fast Library for Approximate Nearest Neighbour matching (FLANN) has been designed to automatically select either a hierarchal k-means structure or a randomised kd-tree with optimal parameters based on the input data. Although FLANN can return matches for large data sets many orders of magnitude faster than a linear search, the matches are less than optimal. This library is ideal for real time feature matching of many high dimensional features, however this benefit is not critical in the execution of this methodology. Finding the highest number of optimal matches is important; hence a linear search with some modifications is implemented.

A useful product of the SURF feature detection stage is the trace of the Hessian matrix (sign of the Laplacian). This is calculated automatically during the detection phase. It distinguishes light blogs on dark
backgrounds and vice-versa. During correspondence, the signs of the traces of the Hessian matrices for each pair of features are compared. This check can significantly reduce the time it takes for correspondence. From the sign of the Laplacian, a ‘meaningful hyperplane’ [17] can automatically be defined in the search strategy to significantly reduce the time it takes for feature correspondence; an advantage the SIFT based techniques [43] do not have. In addition to this check, a best to second best threshold is enforced to ensure that a current match is somewhat better than the previous estimated match.

Feature matching was performed on the stereo pairs shown in Figure 4.5, and the result is shown in Figure 4.6.

4.2.4. Feature Vector Allocation

For the i-th matched pair of features $f_i^L$ and $f_i^R$ in the feature set $F_{k,i}^L$ and $F_{k,i}^R$, an estimate for the descriptor to be appended to the matched points in the stereo pair is generated, based on weighted average of the matched descriptors. The weight is obtained from the strength value in the description stage of the SURF algorithm by

$$f_i = \frac{S_i f_i^L + S_i f_i^R}{S_i^L + S_i^R}$$  \hspace{1cm} (23)

where $f_i$ is the descriptor chosen to represent the matched points, and $s_i^L$ and $s_i^R$ are the strength values of the descriptors in the left and right image.
Figure 4.6: SURF feature correspondence for the first frame in the test data set. The left (coloured cyan) and right (coloured red) perspectives from the stereo pair have been concatenated into a single figure. The positions of the extracted features in the left (cyan) and right (red) images are indicated with red circles and green diamonds marks respectively. The disparity between the points is shown with the blue dotted lines.
4.2.5. Constructing All 2.5D Perspective Views

Applying the triangulation procedure from Section 3.3 for any corresponding pair in a feature set $F^L_{k,j}$ and $F^R_{k,j}$, a 2.5D perspective view $M_{k,j}$ can be produced. Each point will represent the 3D coordinates of a highly distinctive 2D SURF descriptor, relative to the imaging device. The descriptor for this 3D point is obtained with Equation (23). Given that the extrinsics are calculated only once for the stereo camera rig, time is saved by not having to estimate $[R | t]$ for every two-view comparison as in wide baseline approaches [62, 73].

An example of the 2.5D view based on the stereo pair represented in Figure 4.6 is shown in Figure 4.7. Each red point represents the 3D position of a point in the camera coordinate system. As this point data is quite sparse, visualizing this data on 2D page is quite difficult. Hence, a mesh has been used to convey its structure. The structure of the mesh has been interpolated from the sparse point data using the griddata command in MATLAB, and is not entirely indicative of the 3D geometry of the object. It is used only for illustrative purposes.

Once repeated for all frames, the resulting data set contains a sparse 2.5D point cloud for each frame. Each point represents the 3D coordinates of a highly distinctive 2D SURF descriptor, relative to the imaging device (i.e. in the camera coordinate system relative to that perspective view).

4.3. 2.5D View Registration

Once a series of $i$ 2.5D perspective views $M_{k,j}$ have been built from an ordered set of stereo images, each 2.5D must be registered into a single coordinate space. To achieve this, correspondence must be established between matching features of overlapping 2.5D views. To
merge one 2.5D perspective view on to another, an error metric is assigned to estimate an initial coarse geometric transformation of the two clouds. Minimising this error metric brings these clouds into alignment. Fine adjustment of the merger is achieved using an iterative refinement routine. Once two views are merged, this process is repeated for the initial merged set and another similar view so that all perspectives are registered into a single coordinate system. These procedures are explored in the following sections.

![Figure 4.7: The 3D projection of the 2D corresponding points from Figure 4.6.](image)

### 4.3.1. 3D Point Correspondence

Identical to the correspondence problem in Section 4.2.3, the goal is to find which points in two overlapping 2.5D perspective views match each other. One 2.5D cloud is defined as the model $M_{k,i}^M$ and the 2.5D
cloud to be merged on to the model is defined as the data $M_k^D$. Correspondence of 3D points is quite often more difficult than 2D feature matching, as the primary data in the cloud are single points with only 3D coordinates.

Similarities in the arrangement of these points can be used to drive some method of surface matching, however with sparse data this becomes challenging. One advantage of this methodology is that every point in $M_k^M$ and $M_k^D$ has been triangulated from highly descriptive 2D image features. Given that the model and data should have overlapping regions, it can be assumed that they have been taken from similar perspectives. Therefore, as every point in the 2.5D perspectives has a high dimensional feature vector appended to it, this extra information can be used to identify matching points.

The same linear correspondence technique in Section 4.3.3 is used to find SURF features in the model feature set $F_{k,j}^M$ that match to SURF features in data feature set $F_{k,j}^D$. Again, the sign of Laplacian can be used to reduce the breadth of the search. With the addition of 3D displacement of points, a geometric constraint is used to reject pairs with a distance greater than a measure of the median distance, as in [102]. Outliers can have a substantial effect when performing the following least squares minimisation, therefore the aforementioned filtering steps are essential in reducing the prevalence of outliers in the final correspondence set.

Figures 4.8 and 4.9 show the 3D point correspondences established between two 2.5D views. In both cases the correspondence (shown by a green line) maps the absolute 3D distance between the respective points when both are brought into one common coordinate space.
Once again, both clouds in Figure 4.9 are rendered with an interpolated mesh for ease of visualization.

Figure 4.8: XY plane view of the 3D point correspondence established between two 2.5D point clouds. The data (shown in red) represents the point cloud that is to be merged onto the model (shown in blue). The best points to use for correspondence have been highlighted with larger markers.
Figure 4.9: 3D perspective view of the 3D point correspondence established between two 2.5D point clouds. The data (shown in red) represents the point cloud that is to be merged onto the model (shown in blue). The best points to use for correspondence have been highlighted with larger markers. An interpolated mesh has been fitted to each cloud for ease of visualization.

4.3.2. Point Cloud Registration

Registration is an iterative procedure that merges the points of the data $M^D_{k,j}$ onto the model $M^M_{k,j}$. The geometric relationship between corresponding points $f^D_i$ and $f^M_i$ in $M^D_{k,j}$ and $M^M_{k,j}$ is given in [93] and in Equation (18), in Section 3.5. As overview in Section 3.5, the optimal rigid transformation parameters $[\hat{R}^T|\hat{t}]$ between the two clouds can be estimated by minimising the distance error $\Psi$ in Equation (20) in Section 3.5, as in [93].
In the test dataset, Equation (20) is explicitly minimised using the singular value decomposition (SVD) approach in [93]. The implementation is summarised below:

Program 4.1: SVD Minimisation for point cloud registration

Inputs: set of corresponding points $f_i^D$ and $f_i^M$.

Output: optimal rigid transformation parameters $\hat{[R | \hat{t}]}$.

1: Centralize points in each cloud respective to the centroid:
   
   $$C_p = \sum \rho$$
   $$n_p = (\rho - C_p)$$

2: Calculate the covariance matrix by the inner product of the normalized pairs:
   $$Q = n_{px} \cdot n_{tx}$$

3: Perform SVD:
   $$\|e\|Q = USV'$$

4: The rotation matrix becomes:
   $$\hat{R} = VU'$$

5: Check for Reflection
   
   if $\det = -1$
   
   $$\hat{R} = \hat{R} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

6: Obtain the Translation vector:
   $$\hat{t} = M^M_{avg} - \hat{R}xM^D_{avg}$$

7: Apply the transformation and check errors:
   
   if $e_i > \text{thresh}$
   
   Remove point giving with largest error
   
   goto 1

   else
   
   return $\hat{[R | \hat{t}]}$
The final output of this procedure is shown in Figure 4.10. Here, the two corresponding point clouds from Figures 4.8 and 4.9 are merged together, i.e. the red onto the blue.

![Two registered 2.5D point clouds](image)

Figure 4.10: Two registered 2.5D point clouds

A merging routine is used to bring all the 2.5D clouds into alignment. The union of all the generated 2.5D clouds begins with the selection of one view, typically the first, to become the seed or reference cloud. The 2\textsuperscript{nd}, 3\textsuperscript{rd} to \( n \textsuperscript{th} \) 2.5D views are registered to this reference using the known correspondence established in Section 4.4.1 together with Equation (20). Any 2.5D view can act as a seed so long as it has well-established correspondence between its neighbours.

No matter which coordinate system is selected, an object coordinate system for the final SFM is defined from the centroid of the complete
3D point cloud. This ensures that the origin for the SFM is contained within the point cloud, allowing a simple anchor to be used when registering a virtual coordinate system online on which to render information.

4.4. Final Registration Results of All Views

The result of the final registration routine for the test dataset is shown in Figure 4.11. This figure represents the registration of 100 2.5D views into a common coordinate system. In this case, the first 2.5D view has been selected as a reference frame. The pairwise correspondence established for the dataset in Section 4.4.1 is used to register each current 2.5D onto the previous and seat them into the common coordinate space. In this figure, each of the 100 2.5D clouds was assigned a separate colour from MATLAB’s Jet colour map. Given the amount of information present in this figure, it is difficult to discern the individually coloured clouds. The colour gradient is used only as a reference in this image to show the propagation of pairwise registration of the individual 2.5D views. A final SFM is treated as one single 3D structure; therefore this colouration is only for illustrative purposes.

The final 3D point cloud and the associated descriptors for each 3D point are indexed into one data structure. This includes all triangulated points from all 2.5D views regardless of strength and persistence in the final SFM. However, this raw-SFM is the most comprehensive object representation that can be constructed with the selected parameters. Given its density, any features queried online will easily find a match, however the data size and the time cost in querying this cloud with a linear search increases with the order of 2 as the size of the SFM increases.
Table 4.2 below shows the element size and data footprint of the raw-SFM built for the test dataset.

<table>
<thead>
<tr>
<th>Item</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D Point Cloud</td>
<td>14174x3 elements</td>
</tr>
<tr>
<td>Descriptor Database</td>
<td>14174x64 elements</td>
</tr>
<tr>
<td>Data footprint</td>
<td>4.19MB</td>
</tr>
</tbody>
</table>

*Table 4.2: The size of the raw-SFM for the test dataset*
4.5. Conclusion

Recognition based augmented reality requires pre-learnt a priori knowledge that accurately represents the most robust features that lie on an object. The creation of a priori knowledge from many perspectives is critical to accurately and comprehensively characterise an object. In this chapter a method of generating a complete 3D sparse feature model from multi-view data was developed.

Multiple short-baseline stereo images were captured from different perspectives around an object. For each stereo pair, a 2.5D point cloud was generated by the triangulation of corresponding, highly descriptive object features. A unique, raw-SFM was reconstructed from these multiple views by merging each 2.5D point cloud together using 3D-to-3D shape registration. This work represents a synergy of 2D and 3D computer vision and image processing techniques that together produce comprehensive a priori data.

Examples from a test dataset of 100 short-baseline stereo images were used to highlight each stage of the methodology. Throughout the chapter, each example demonstrated typical outputs of the respective algorithms. The final raw-SFM resulted in a 3D point cloud, where each point represents the 3D location of a 2D object feature characterised with a 64 dimensional descriptor.

Though this cloud is descriptive given its size, the time cost when using this data in an online recognition based AR scenario is restrictive. As the size increases, the time taken to match to this cloud increases, with the order of two (On^2). Hence, a refinement process to reduce the size of the SFM whilst retaining only the most distinctive and persistent object features is introduced in Chapter 5.

Early versions of this work were published in [64, 80].
“One should engage himself in self study to refine his intelligence and to acquire knowledge”
Atharva Veda

Refining Sparse Feature Models

Typically, multi-view a priori generation (including the methods in Chapter 4) consists of a union of all features from all perspectives to produce a Sparse Feature Model (SFM). Often there are several weak or poorly localized features in a view, or features that are not persistent in a number of consecutive frames. If all of these weak features are retained in the a priori data set, they will cause time inefficiency and error online [8]. Baheti et al. [8] demonstrate how the reduction of the number of features in a database, whilst retaining the distinctiveness of the overall a priori data, is an important step to increasing the efficiency of online AR applications.
In this chapter, additional processing steps to the methods of Chapter 4 are proposed to refine a sparse feature model a priori representation of an object.

5.1. Refinement Methodology

For this three-tiered methodology, a stereo camera records a global stream of frames (each containing a stereo pair) in a fluid pass around an object. In tier one, highly descriptive triangulable 2D features are extracted from the left and right images for all frames. The proposed statistical analysis in tier two identifies the strongest and most persistent local features of an object by sampling the distribution of corresponding features across multiple frames (i.e. varying perspectives). In tier three, these strong and persistent features are triangulated to produce 2.5D point clouds for each frame. Frame-wise 3D-to-3D registration is performed to unify the 2.5D perspective views into a single coordinate space. Geometric analysis clusters overlapping features to produce single representative points. Further feature analysis of the components of SURF descriptors is used to generate representative features.

The final a priori data set is termed the refined Sparse Feature Model. Refined-SFM a priori data is more distinctive, with less data than traditional sparse feature models, reducing the time cost of feature matching when used online.

Figure 5.1 shows an overview of this method.
Figure 5.1: Refined-SFM methodology
5.2. Tier One - Pairwise Analysis

The input into Tier one is a set of frames, each containing a short-baseline stereo pair. To capture this data, a stereo camera is moved around an object in a fluid motion whilst capturing detailed images. This Tier performs pairwise feature extraction on the left and right stereo image for each frame in the global stream. This tier identifies robust descriptive 2D image features that can be triangulated to produce a 2.5D perspective map of their locations on an object. Though quite similar to the first steps of the method in Chapter 4 (Section 4.3.2), an initial correspondence check retains only triangulable features, rather than all features.

For each $i$-th view of the $k$-th object, the SURF algorithm is used to generate feature sets $F^L_{k,i}$ and $F^R_{k,i}$, for the left $L$ and right $R$ images. Each salient feature in any of the left and right images is assigned a 64 dimensional descriptor. Initial correspondence between features $F^L_{k,i}$ and $F^R_{k,i}$ is established for each $i$-th view. The matched features of $F^L_{k,i}$ and $F^R_{k,i}$ are reduced to $\hat{F}^L_{k,i}$ and $\hat{F}^R_{k,i}$, producing new sets that only hold features that comply with epipolar geometry [16].

5.3. Tier Two - Global Feature Vector Analysis

Tier two is a global analysis and refinement process that identifies strong and persistent feature vectors across multiple perspectives.

5.3.1. Framewise Correspondence

The feature sets of $\hat{F}^L_{k,i}$ and $\hat{F}^R_{k,i}$ contain features from one perspective $i$ that can be triangulated to localize their 3D position. Let’s introduce two new sets $G^L_k$ and $G^R_k$, where:
\[ G_k^L = \bigcup_i F_{k,i}^L \]  
\( \text{and} \)
\[ G_k^R = \bigcup_i F_{k,i}^R \]  

\( G_k^L \) and \( G_k^R \) include all triangulable features in all frames of the global stream. As the global stream of frames is quite dense in its coverage and the similarity between neighbouring frames is quite high, features that appear in only one frame are often anomalies. These weak features are either unique to a highly constrained viewpoint (i.e., susceptible to minor perspective distortion), background or foreground features that are not native to the object, or are simply features extracted from noisy artefacts in the image.

### 5.3.2. Statistical Analysis

In order to identify strong and persistent features and eliminate weak features from the initial sets \( G_k^L \) and \( G_k^R \), statistical analysis is used to sample the distribution of features and their occurrence rate across multiple frames. The mean and variance of the feature distributions are used to group corresponding features together as \( g_{k,i}^L \) and \( g_{k,i}^R \) by matching features to neighbouring frames. Through the identification of strong and persistent features in the initial sets \( G_k^L \) and \( G_k^R \) with groups \( g_{k,i}^L \) and \( g_{k,i}^R \), they can be reduced to \( \hat{G}_k^L \) and \( \hat{G}_k^R \) with groups \( \hat{g}_{k,i}^L \) and \( \hat{g}_{k,i}^R \).

To perform the analysis, feature vectors are matched across neighbouring frames to assemble corresponding points into feature groups \( \hat{g}_{k,i}^L \) and \( \hat{g}_{k,i}^R \). As typical, the matching process analyses the variance of the descriptor components to determine a good match.
The measure of variance indicates the confidence of a match. Any match with a low confidence level is not included into a feature group.

Of the 14,174 individual triangulable features in the 100 frames of the test data set, there were 22,135 interframe features matches. This initial correspondence set was then clustered into 1,536 groups. A window can be applied to the grouping process that restricts the range of interframe matches to a few frames before and after the current frame being analysed. In the test data set, 10% (36º) of the entire global stream was used. This eliminates false positives from impossible matching events, like a feature on the front of the object being matched to one on the back of the object. As the test data set had the same beginning and end point, this window wraps back to Frame 1 from frame 100, as the views leading up to frame 100 are similar to the views proceeding frame 1.

Figure 5.2 shows a snapshot of the occurrence rate of a range of features in the set \( G^i_k \) from the test data. To keep the graph meaningful (due to the high number of groups returned by the clustering routine), the occurrence rate of the first 150 features of frames 30 to 40 are shown as an example.

Each element in this grouping array holds a structure that contains an index to all of the features that one feature \( \hat{g}_i \) has been matched to. An example of the members in group 21 for the test data is shown in Table 5.1, and in Figures 5.3 and 5.4. These results show that the 27th feature of the first frame was matched to features in 7 neighbouring frames. Note that the origin of the image is in the upper left corner.
Figure 5.2: Occurrence rates for the first 150 features listed in frames 30 to 40. For each feature along x, the height of the bar in y represents the number of features it is matched to in neighboring frames (i.e. its occurrence).
Table 5.1: The members of the 21st group ($\hat{G}_{12}^1$) in the set $\hat{G}_1^L$. The features $x,y$ coordinate is given in the square parentheses.

<table>
<thead>
<tr>
<th>Frame</th>
<th>Sampled Feature</th>
<th>Frame</th>
<th>Matched Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>27</td>
<td>2</td>
<td>27 - [520.13 ; 409.33]</td>
</tr>
<tr>
<td></td>
<td>[519.2251 ; 408.5290]</td>
<td>3</td>
<td>25 - [521.65 ; 411.04]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>28 - [523.69 ; 413.09]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>97</td>
<td>26 - [516.07 ; 405.51]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>98</td>
<td>24 - [517.68 ; 407.08]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>99</td>
<td>23 - [518.59 ; 408.58]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100</td>
<td>31 - [518.99 ; 408.27]</td>
</tr>
</tbody>
</table>

In the following images, the left frame from the initial member [$F(1),f(27)$] are superimposed to the left frames (in cyan) of the other group member (in red). As the views are quite similar, this is a little hard to distinguish from a full-framed image, as in Figure 5.3. The first correspondence is shown in Figure 5.3 where the original feature 27 of Frame 1 [$F(1),f(27)$] (in red) was matched to feature 27 in Frame 2 [$F(2),f(27)$] (in green).

Figure 5.3: The first corresponding pair in the 27th group set. Note: the origin is in the top left corner of the image.
Figure 5.4 shows a cropped version of the other six matches for this feature group. Once again, the original feature is highlighted in red and the corresponding feature $f(n)$ from frame $F(n)$ is shown in green.

Figure 5.4: Cropped and zoomed regions highlighting the corresponding feature members in the 27th group set.

Figure 5.5 shows the spread of the features matched to feature 27 of frame 1 as they traverse from Frame 97 through to Frame 4. The coordinates from each feature are taken from their respective location on the test image from which they were extracted, and are drawn in a common reference frame. This figure clearly highlights the path of the features as the object rotates in both pitch and yaw. Once again, note that the origin is in the top left hand side corner of the image.
5.3.3. Selection Criteria for Refinement

To refine the feature sets $G_k^L$ and $G_k^R$ to $\hat{G}_k^L$ and $\hat{G}_k^R$, the following selection criteria are used. Firstly, feature groups with occurrence rates above a persistence threshold are selected. These repeatable features will have a higher possibility of being observed across multiple perspectives. Secondly, feature groups with low variance are selected. These grouped features remain distinctive across multiple viewpoints and are robust to perspective distortions. The size of the refined-SFM can be tuned from these parameters.

Retaining features with lower persistence levels increase the resolution of perspectives represented in the refined-SFM at a cost of a higher amount of features that have to be matched online. Restricting or relaxing the presence of feature groups with low or high variance,
scales the distinctiveness and robustness of the refined-SFM. Any feature group with a number of matched features less than the persistence threshold (set at 3 for the test dataset) is pruned from the final group set $\hat{G}_k^L$ and $\hat{G}_k^R$. This ensures that a feature is persistent across at least three frames, or approximately 10.8° of the 360° scan pattern for the test global stream.

This refinement process reduced the size of $G_k^L$ (initially 14,174 unique triangulable features with 22,135 interframe matches) to a set $\hat{G}_k^L$ with 6420 interframe feature matches from 5936 unique triangulable points. In this case $g_{ki}^L$, with 1,536 feature groups, was reduced to $\hat{g}_{ki}^L$, with 582 feature groups.

**5.4. Tier Three - Localisation**

Tier three has two purposes. Firstly a framewise 3D-to-3D registration process unifies the locations of all strong and persistent features in all frames together into a single 3D sparse feature model. Next, the point geometry and the associated feature vectors of the resulting SFM are analysed to intelligently remove/reclassify repeated features and outliers.

**5.4.1. 2.5D Localisation and Registration**

As with section 4.4 in the raw-SFM method, a multi-view representation of an object is achieved through the 3D-to-3D registration of all 2.5D frames in the global stream. To generate the 2.5D cloud for a frame $i$ in the global stream, all the points related to $i$ from $\hat{G}_k^L$ and $\hat{G}_k^R$, are extracted. As correspondence has already been established in Section 5.2, the refined features for frame $i$ in $\hat{G}_k^L$ and
\( \hat{G}_k^R \) are assigned 3D points in the camera coordinate system for frame \( i \) using the triangulation procedure in Chapter 3.

Figure 5.6 shows a comparison between the 2.5D view generated from the first frame of the test data set before and after refinement. The points in the refined 2.5D view represent strong and persistent features that occur across more than three frames in the test dataset. An interpolated mesh has been fitted to each cloud for ease of visualization.

For each frame in the global stream, a 3D rotation and translation transformation is required to merge the 2.5D point clouds of two neighbouring frames together into a single coordinate space. In the statistical analysis process of Section 5.3, feature groups \( \hat{g}_{k,i}^L \) and \( \hat{g}_{k,i}^R \) with members that span multiple frames provide interframe correspondences that can be used to initialize iterative 3D-to-3D registration. As the clouds are quite sparse yet each point represents highly descriptive 2D image features, this method once again has the unique ability to generate an initial correspondence between interframe points directly from the feature vectors without restrictively relying on the 3D geometry of the cloud.

The registration algorithm detailed in Section 4.4.2 and in [93] is used to register the multiple 2.5D views generated for the frames in the refined dataset. From 16 corresponding points, the rotation and translation parameters \( [\hat{R} | \hat{t}] \) of the red cloud with respect to the blue cloud was estimated and applied to the red cloud in order to bring them into alignment. Figure 5.7 shows the correspondence of points in two 2.5D views in (a), and both merged together in (b). Once again, for visualisation purposes, an interpolated mesh has been fitted to each cloud.
Points in the first frame **before** refinement

Points from the first frame **after** refinement

*Figure 5.6: Before and after perspectives of the 2.5D views generated for the first frame of the test data set.*
Figure 5.7: Correspondence and merger of two refined 2.5D views

a – 3D point correspondence

b – two merged clouds

Figure 5.7: Correspondence and merger of two refined 2.5D views
5.5. Tier Three - Geometric Error Analysis

When merging multiple 2.5D point clouds together, corresponding points are often misaligned. There are a few possible sources for this error. In the construction of each 2.5D point cloud, error in the \( x, y, z \) estimate of the point data will cause small discrepancies in the localization of points. Consequently, the registration of the clouds is inherently impacted by these errors, with rotation and translation \( [R|t] \) estimates also introducing alignment errors.

5.5.1. Errors in the Construction of a 2.5D View

The calibration of a stereo camera rig is critical \([103, 104]\) for accurate depth estimation. Stereo calibration errors can introduce uncertainty into the 3D projection of 2D points. When estimating depth from disparity, the alignment and distortive effects of the imaging system must be compensated for. \([104]\) discuss the projection error probabilities associated with stereo camera systems. The optics and hardware of the imaging system can also influence triangulation. For example, pixel level noise from the imaging sensors and image compression artefacts can influence the location estimate of identifiable features and thus compromise a features 3D projection.

Imaging an object from different perspectives can also inherently affect the \( x, y \) point location of a feature. As in this thesis, descriptors that are constructed from information in a region or patch of pixels help increase the robustness of interest point correspondence. The angle in which a patch of pixels is viewed at will affect its geometry, modifying its discernibility. For example, the centroid (reported \( x, y \) location) of a blob shaped feature used in the SURF detector and descriptor \([17]\) can vary with changes in perspective. As can the coverage of pixels for a descriptor. The variance in the \( x, y \) estimate can directly affect the \( z \)
depth estimate of a point. Confidence in the triangulation of two corresponding points is entirely dependent on the disparity that the depth estimator is fed.

5.5.2. Errors in 3D-to-3D Registration

The whole premise of iterative 3D-to-3D registration is the minimization of the distance error between matching feature points. In the case of this work, the SVD method described in [93] and Chapter 4.4.2 is used. Firstly, exact correspondence must be established between the two clouds. This is a fundamental problem in the computer vision community. This method implements many filters and checks, and utilise high dimensional feature descriptors to help mitigate correspondence errors.

To estimate the optimal rigid transformation \( [R \mid t] \) between two clouds, a distance error metric is minimized until the error between the two reaches a threshold or there is an iteration time out. Ideally the minimized error should be zero, however inaccuracies in the 3D projection of the 2D object features ultimately influence this best possible alignment.

When registering multiple clouds together, it’s typical to only feed the minimization routine with a subset of points from the correspondence set between each cloud. Using these ‘best points’ helps to reduce the likelihood of outliers affecting the final transformation. However, as this subset drives the foundation of the transformation, the distance between other corresponding points is not taken into account and these points are thrust (hopefully) into alignment. If the transformation is compromised by poor localization of the best corresponding features, then there can be quite a large margin of error on these other points.
5.6. Tier 3 - Representative Points

Theoretically, if the 3D projection of points from stereo views and the registration of multiple 2.5D perspective views have no error, multiple samples of the same object feature should be localized at the same 3D position in the final SFM. In reality, small errors introduced in these methods disperse these matching points around the real location of an object feature [103].

Generating an SFM from a stream of images that are extensively overlapped will often result in many features being grouped into unique clusters of like feature points. If these points were reduced to a representative point that optimally matches to the location of a real object feature, then the total amount of 3D point elements in a final refined-SFM can be greatly reduced. More so, the location of an object feature can be used more confidently as an accurate representation of the location of that feature. In this section, a Gaussian weighted mean reduction technique is used to establish the position of the representative points.

5.6.1. Constructing the Points

A Gaussian weighted mean approach is used to condense a group of 3D points that represent one feature to a representative point. By weighting the location of each point, there is potential to eliminate the influence of some of the high error outliers and therefore obtain a more accurate location estimate for the representative point.

Let’s assume that $G$ is the raw-SFM that includes multiple feature groups. Each group consists of similar features that are each attempting to represent one object feature. The feature groups of $G$ are introduced as $G_k$, where $k$ is the group number. If $s_{ki}$ represents
the location of the \( i \)-th feature in the \( k \)-th group, then the mean estimate of all locations in the group \( k \) is expressed as

\[
\hat{S}_k = \sum_{i=1}^{n_k} \frac{S_{k,i}}{n_k}.
\]  

(26)

where \( n_k \) is the number of elements in the \( k \)-th cluster.

When taking a typical mean estimate, all points are weighted equal regardless of the accuracy of the points. This can introduce considerable error if outliers on the boundaries of the datasets have introduced inaccurate estimates. A better approach should factor in information from which weighting can be obtained.

The Gaussian weighted average method is expressed as

\[
\hat{S}_k = \sum_{i=1}^{n_k} w_{k,i} S_{k,i},
\]  

(27)

where \( 0 \leq w_{k,i} \leq 1 \) and \( \sum_{i=1}^{n_k} w_{k,i} = 1 \).

The validation of this method has limitations imposed by the SURF feature extraction method used. Although SURF is robust to illumination, scaling and rotation, its creator considered perspective effects to be a second order concern [17]. SURF has not been designed to compensate for perspective distortion, however throughout this work, SURF has demonstrated a robustness to out of plane rotations of up to 30° as supported by [17]. Drawing from this knowledge, weights can be assigned to each \( i \)-th point \( S_{k,i} \) in the \( k \)-th group based on how much an individual point’s normal deviates from
the camera view normal. The normal of each point relative to the camera view of the object is encoded during the SURF description stage.

With this method, the weight values are calculated by:

\[ w_k = \frac{b}{\sqrt{8\pi^3 \sigma_x \sigma_y \sigma_z}} \exp \left( -\frac{(x_k - \mu_x)^2}{2\sigma_x^2} - \frac{(y_k - \mu_y)^2}{2\sigma_y^2} - \frac{(z_k - \mu_z)^2}{2\sigma_z^2} \right) \] (28)

where \( b \) is a scalar that is defined to have \( \sum_{i=1}^{n} w_i = 1 \)

If the coordinate system for Equation (28) is placed on the object feature point and its z-axis is in the same direction of the normal camera view direction then \( \mu_x = \mu_y = \mu_z = 0 \). Hence the weights simplify to:

\[ w_k = \frac{b}{\sqrt{8\pi^3 \sigma_x \sigma_y \sigma_z}} \exp \left( -\frac{x_k^2}{2\sigma_x^2} - \frac{y_k^2}{2\sigma_y^2} - \frac{z_k^2}{2\sigma_z^2} \right) \] (29)

### 5.6.2. Representative Points Results

Figure 5.8 shows a group of nine location estimates, clustered around one object feature in the test set. All members of this point group attempt to localise and represent one object feature. They were clustered together based on the similarity of their feature vectors, as in Section 5.3.2. The points in the blue/green shade have a stronger influence on the weights than the points shaded red/yellow.

Theoretically with zero localization and registration error, these points should converge to the real location of the object feature. As this is not the case, a representative point, as shown in Figure 5.8 by the cyan
star with blue outline, was calculated using the methods proposed in Section 5.6.1. This point represents the Gaussian weighted mean location from the nine sample points in the group. As a comparison, the yellow hexagon with a red outline represents the standard mean location of the nine sample points. Clearly, the representative point (cyan) has been less influenced from the red/yellow points than the mean estimate (yellow).

![Figure 5.8: A cluster of points that represent one object feature. The yellow hexagon with a red outline represents the mean estimate. The cyan star represents the estimated location for the representative point using the Gaussian weight approach in Section 5.6.](image)

The representative point method has subsequently reduced the number of elements representing this one object feature by a factor of 9:1. Applied across an entire raw-SFM, this can significantly condense a dataset’s footprint.
5.7. Representative Feature Vectors

Each point in an SFM is localized and matched using descriptive feature information. Inherently, the feature vectors of a cluster of matched points representing one object feature are closely related/near identical within a factor of confidence; a product of the feature matching algorithm. To reduce the number of descriptors populating an SFM, a representative feature vector is substituted for each cluster. The distribution of each element in the 64-dimensional descriptor is analysed to condense multiple matching descriptors into a unique representative feature vector. Due to the initial similarity of the vectors, the generalisation of the feature vector is small.

The new representative feature vector holds the descriptive information used for feature correspondence between a sample scene and the refined-SFM, and is defined in 3D space by the representative point. As discussed in Section 4.3.3, there are ways of speeding up correspondence estimation between high dimensional feature vectors. Structured methods such as approximate nearest neighbour searches can run significantly faster for high dimensional vectors than linear and nearest neighbour methods. As mentioned previously, Muja and Lowe’s [101] FLANN library is quite useful for returning an approximated set of possible matches for any queried feature.

These methods are useful for online searching, especially on low powered devices, but they return less than optimal matches within a given threshold. They also often require the database to be restructured to facilitate these structured methods. This representative feature vector method aims to first reduce the cardinality of the descriptor database whilst remaining robust and distinctive. Thus, performance benefits can be introduced before resorting to approximated matching.
5.7.1. Constructing the Feature Vectors

Let’s assume that $F$ is the descriptive data component of a raw-SFM. Each object feature has an associated group $F_k$ of similar vectors that are each a description of the same object feature. $F$ is the union of all feature groups $F_k$. If $f_{k,i}$ represents the 64-dimensional descriptor of the $i$-th SURF vector in the $k$-th feature group $F_k$, then the Gaussian weighted mean for a descriptor of a representative feature vector is expressed as:

$$\hat{f}_k = \sum_{i=1}^{n_k} W_{k,i} f_{k,i}$$  \hspace{1cm} (30)

where $0 \leq w_{k,i} \leq 1$ and $\sum_{i=1}^{n_k} w_{k,i} = 1$.

The weights of these groups $w_{k,i}$ are obtained from Equation (29). This procedure ensures that perspective effects on the SURF descriptors have limited influence on the representative feature’s descriptor.

5.7.2. Representative Feature Vector Results

Figure 5.9 shows the boxplot of the descriptor elements of the SURF feature vectors that have been calculated for the object feature used in Section 5.6.2. The boxplot shows the 50th percentile of the data in the box area. The black whiskers represent the max and min of the data. The weighted mean values (green plus signs, calculated using the method in Section 5.7.1, represent the descriptor elements of the new representative vector.
Figure 5.9: Descriptor elements for an object feature group and the boxplot of the distribution of the descriptor elements in this group. In this figure the red marks represent the medians. The green plus represents the Gaussian weighted means that are used to construct the representative feature vector for this object feature.
5.7.3. Representative Feature Vector Matching

There is a concern that generalising a feature vector will have a detrimental effect on its matching ability to the vectors that make up the group. Contrary to this belief, the next figures prove how the creation of a representative vector using the methods of Section 5.7 generates an optimal feature vector that best matches to all features in the group.

Figure 5.10 compares the matching error of a representative feature against all the features of its seed group. In this figure, each feature was matched to all other figures in the group. The maximum and minimum error metric was chosen for each feature to highlight a matching range (i.e. the coloured bars). The red markers highlight the error of the match between the representative feature vector and each feature vector of the group.

Figure 5.10: Comparison of a representative feature vs. the max/min ranges
Clearly, Figure 5.10 shows that the representative feature vector has a minimal error when matched to each feature vector of the group. This indicates that any external features that were to find a match to any of the vectors in the group representing this object feature will match optimally with the representative feature vector.

Figure 5.11 shows the error comparison of the representative vectors for 53 groups of features. Each of these 53 groups contained 8 feature vectors. Each x element in the figure represents average min/max range for inter-group matching. The red marker indicates the maximum error obtained when matching the representative feature to all vectors in a group. Clearly, this representative vector matching error computed for each feature group demonstrates that the generalised vector has a minimal error compared to the rest of the elements in the group.

The data from these two experiments indicate that the creation of a representative feature vector does not generalise its discernibility. A feature vector built from the Gaussian weighted mean of a group of matched features retains a high similarity against all group members and will match optimally to all members. This gives a better balanced feature to describe a representative point, rather than just choosing one vector out of the group as in [8].

5.8. From Raw-SFM to Refined-SFM

The final 3D point cloud and the associated descriptors for each 3D point are indexed into one data structure. This includes only features with the highest strength and persistence from all 2.5D views. This refined-SFM is comprehensive and distinctive, yet with a reduction in its cardinality over the raw-SFM of about 42% before representative point and vector association.
Figure 5.11: Error comparison of matching the respective representative feature vector to each of 53 groups
Table 5.2 below shows the element size and data footprint of the refined-SFM built for the test dataset, before multiple entities describing one object feature have been condensed into representative points and features.

<table>
<thead>
<tr>
<th>Item</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D Point Cloud</td>
<td>5936x3 elements</td>
</tr>
<tr>
<td>Descriptor Database</td>
<td>5936x64 elements</td>
</tr>
<tr>
<td>Data footprint</td>
<td>1.79MB</td>
</tr>
</tbody>
</table>

*Table 5.2: The size of the refined-SFM for the test dataset*

Applying the representative point and feature vector methods of Section 5.6 and 5.7, this data was further condensed to that of Table 5.3. Overall, an approximate 3.4x reduction in elements was found, as most groups contained between 3-5 features.

<table>
<thead>
<tr>
<th>Item</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D Point Cloud</td>
<td>1749x3 elements</td>
</tr>
<tr>
<td>Descriptor Database</td>
<td>1749x64 elements</td>
</tr>
<tr>
<td>Data footprint</td>
<td>682KB</td>
</tr>
</tbody>
</table>

*Table 5.3: The size of the refined-SFM for the test dataset after point and vector refinement*

Figure 5.12 shows the complete refined sparse feature model. As with Figure 4.11, each of the 100 2.5D clouds was assigned a separate colour from MATLAB’s Jet colour map. Given the amount of information present in this figure, it is difficult to discern the individually coloured clouds. The colour gradient is used only as a reference in this image to show the propagation of pairwise registration of the individual
2.5D views. A final refined-SFM is treated as one single 3D structure; therefore this colouration is only for illustrative purposes.

**Figure 5.12: Final refined-SFM**

### 5.9. Conclusion

The methodology proposed in this chapter can be used to refine sparse feature model a priori data. In the three-tiered approach, statistical analysis was used to measure the strength and persistence of a set of triangulable features extracted from multiple short-baseline images of an object. A refinement process used the occurrence rates and variance of persistent feature vectors to condense an initial sparse feature set down to a refined sparse feature model.
The geometric structure and descriptive feature component of a refined-SFM was analysed to condense the number of elements present in the a priori data. Representative points were introduced as a Gaussian weighted mean of multiple point locations for individual object feature groups. Representative feature vectors were introduced to condense multiple descriptors representing the same object feature into a single vector.

There was a 62% reduction in the cardinality of the refined-SFM over raw-SFM data generated for the test dataset. The cardinality of the refined data was then further reduced by approximately 70%, by condensing multiple entities describing single object features into representative points and representative feature vectors. It was also shown that the creation of a representative feature vector does not generalise its discernibility. In the test dataset, feature vectors built from the Gaussian weighted mean of a group of matched features retained a high similarity and optimal match against all group members.

Refinement by statistically analysing the most persistent and strong object features, and the generation of representative points and representative feature vectors will condense a raw-SFM to a refined-SFM. These refinement procedures build highly descriptive yet computationally efficient a priori knowledge. With a major reduction in cardinality, and the removal of weakly localised and non-persistent points, refined sparse feature model a priori knowledge is superior to unprocessed SFM data.
Application to Real Objects

This chapter applies the methods presented in the earlier sections of this thesis, to analyse the performance and efficacy of refined sparse feature models versus raw sparse feature models built from real objects. It is divided into two complimentary parts. Part A uses the methodologies presented in Chapters 4 and 5 to generate raw and refined sparse feature model (SFM) a priori data, and compares their cardinality and their data footprint. Part B analyses the matching performance of the raw- vs. refined-SFM methodologies, and their recognition performance during online pose estimation.
Part A: Raw and Refined Real Object SFMs

Theoretically, the methods of Chapters 4 and 5 are applicable for any object as long as the object is textured (a restriction from SURF) and its size is conducive to the measurement setup. SURF can be tuned to process features at multiple scales, dependent on the image size, though in practical terms, an object will have a limited range of detection based on these scales.

In Part A, five objects are processed with the methodologies presented in Chapters 4 and 5 to produce sparse feature model type a priori data.

6.1. Camera Calibration and Datasets

As mentioned in Chapter 4, there are no standard datasets available that contain multiple short-baseline stereo views around a single textured object. Most of the available datasets contain either multiple single images around an object, or one synchronous short-baseline stereo view from a single perspective. The methods of this thesis use multiple stereo views taken from many perspectives, so that this multi-view information can be used to form a SFM from all facets of an object. Integrating features from many views into a single metric model requires the construction of an input dataset.

The datasets in this chapter were created from short-baseline stereo images of real objects. For this implementation, each object was placed in a static scene with minimal background noise. A Fujifilm FinePix REAL 3D W3 [105] camera was used to image the test objects from multiple perspectives. This off the shelf, point and shoot camera features twin 10 megapixel CCDs horizontally displaced with a 75mm baseline. This device was quick to deploy and easy to integrate into the implemented workflow.
6.1.1. Stereo Camera Calibration

To determine the precise arrangement of the camera centres of both of the FinePix’s CCDs, the device had to be calibrated. The calibration of two pinhole type cameras in a fixed baseline stereo arrangement is a common procedure. There are many freely available toolkits, including the camera calibration toolbox for MATLAB [106] and calibration routines in OpenCV and in standalone C++ [107].

The camera calibration parameters for the Fujifilm FinePix REAL 3D W3 are listed in Table 6.1. The camera was calibrated using Jean-Yves Bouguet Camera Calibration Toolbox for MATLAB [106].

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Extrinsic Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>Om</td>
<td>[0.0178 ; 0.0073 ; -0.0034]</td>
</tr>
<tr>
<td>T</td>
<td>[-76.121 ; 1.8080 ; -1.1026]</td>
</tr>
<tr>
<td><strong>Intrinsic Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>Focal Length L</td>
<td>[2414.4308 ; 2428.7415]</td>
</tr>
<tr>
<td>Focal Length R</td>
<td>[2414.0457 ; 2423.9092]</td>
</tr>
<tr>
<td>Principle Point L</td>
<td>[954.0479 ; 787.1866]</td>
</tr>
<tr>
<td>Principle Point R</td>
<td>[1030.9011 ; 814.6964]</td>
</tr>
<tr>
<td>α L (pixel skew)</td>
<td>0</td>
</tr>
<tr>
<td>α R (pixel skew)</td>
<td>0</td>
</tr>
<tr>
<td>Image Distortions L</td>
<td>[-0.1910 ; 0.2407 ; 0.0033 ; -0.0019 ; 0]</td>
</tr>
<tr>
<td>Image Distortions R</td>
<td>[-0.2058 ; 0.3114 ; 0.0088 ; -0.0063 ; 0]</td>
</tr>
</tbody>
</table>

*Table 6.1: Camera parameters for the real camera system. Om are vectors related to rotation via the Rodrigues formula [95]*

6.1.2. Image Dataset Overview

Multiple short-baseline stereo images were captured around several real objects to form image input datasets for the methods described in Chapters 4 and 5. Stereo images were captured at a resolution of 2048x1536 pixels. This data was imported from the camera in a .mpo format, from which left and right images were extracted. Upon loading
into MATLAB, the only image manipulation during this process was the conversion of each full colour image to grey-scale. Sample images from theses datasets are shown in Figure 6.1 (a) – (e).

(a) GingerBeer sample image  (b) Nuts sample image

(c) Coffee sample image  (d) Wafer sample image

(e) Shapes sample image

Figure 6.1 (a) – (e): Sample images from the respective datasets
The first image dataset was of a box of ginger beer drinks, shown in Figure 6.1a. This object had a feature rich appearance and sharp geometric structure that highlights the geometric performance of the triangulation and registration methods of this thesis. Having sharp, 90° faces, the reconstruction of the salient features on this object demonstrates the ability of the registration routine to handle abrupt changes in the surface angles. 81 stereo image pairs were taken for this dataset.

The second image dataset was of a round tub of mixed fruit and nuts, shown in Figure 6.1b. This object also presented many varying textures for the SURF routine to extract blob like features from. Being cylindrical, the reconstruction of this object highlights the registration of features on non-planer surfaces. 101 stereo images were taken for this dataset.

The third image dataset was a packet of coffee, as shown in Figure 6.1c. Though technically not rigid, the bag was tightly packed and held its shape well. With a well-textured odd shape, the use of this dataset was interesting to see how the online pose estimation routine would work with semi-deformable structures. As a part of the online implementation, the geometric distributions of regions of features are analysed to make sure that a feature is seated correctly with respect to its surrounding neighbours. This dataset contained 55 stereo images.

The fourth image dataset was of a cylindrical wafer tin, as shown in Figure 6.1d. This object has well transitioned colours with large shapes in the background artwork. The text was also quite fine, limiting the amount of features that could be extracted from the object. This dataset comprised of 49 stereo images. Although the object is geometrically similar to 6.1b, this object demonstrates the repeatability of the method when generating SFMs of similar shaped objects with
different textures, using different imaging paths and dataset size (101 images for 6.1b vs. 49 images for 6.1d)

The fifth image dataset was of a rectangular Shapes biscuit box, shown in Figure 6.1e. This object was chosen to show the repeatability of the methods for different sized objects with sharp, angular structure. The texture on this object was less repetitive than 6.1a. There were 98 images in this dataset.

### 6.2. Feature Extraction

Each object’s sequence of images was subjected to the SURF feature extraction and description algorithm built into MATLAB 2011b, as described in Section 4.3.2. A series of filters were implemented during this procedure to limit the introduction of outliers. The main check was a vertical displacement test. This determined whether the coordinates of a matching pair of features were parallel to the x-axis. The 2D projection of a 3D point in each stereo image should lie in the same horizontal plane of each image, assuming each camera centre is aligned and there is minimal lens distortion. For these tests, a vertical displacement of ± 5 pixels was used.

Table 6.2 below shows the distribution of features extracted for each dataset, and the number of actual matched features and triangulable features per image.

<table>
<thead>
<tr>
<th>Dataset name- [number of images]</th>
<th>Average number of features per image</th>
<th>Average number of matched features</th>
<th>Average number of triangulable features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Left</td>
<td>Right</td>
<td>Left</td>
</tr>
<tr>
<td>Beer-[81]</td>
<td>1699</td>
<td>1456</td>
<td>682</td>
</tr>
<tr>
<td>Nuts-[101]</td>
<td>1542</td>
<td>1528</td>
<td>714</td>
</tr>
<tr>
<td>Coffee-[55]</td>
<td>1626</td>
<td>1633</td>
<td>653</td>
</tr>
<tr>
<td>Wafer-[49]</td>
<td>1189</td>
<td>1207</td>
<td>547</td>
</tr>
<tr>
<td>Shapes-[98]</td>
<td>1406</td>
<td>1398</td>
<td>602</td>
</tr>
</tbody>
</table>

*Table 6.2: Feature extraction results for each of the test datasets*
6.3. Raw-SFM Processing

This section explores the results for building sparse feature model a priori data from stereo images using the procedures outlined in Chapter 4. The results from the feature extraction component of the algorithm were used to directly build raw-SFMs. Correspondence was established for each stereo pair, using a linear search to find the best possible match for all extracted features. For all best matching features, their depth was estimated using the triangulation procedure of Section 3.3. Each 2.5D cloud was then merged together using the 3D-to-3D registration methods of Section 4.4. The results for each object in the captured datasets are explored below.

In the following figures, each 2.5D cloud from each stereo pair in the respective dataset was assigned a separate colour from MATLAB’s Jet colour map. Given the amount of information present in the figures, it is difficult to discern the individually coloured clouds. The colour gradient is used only as a reference in these images to show the propagation of pairwise registration of the individual 2.5D views. The results for each object in the captured datasets are explored below.

Figure 6.2 shows the raw-SFM point cloud for the GingerBeer Dataset (sampled in 6.1a). Figure 6.3 shows the results for the raw-SFM output of the Nuts dataset (sampled in 6.1b). Figure 6.4 shows the results for the raw-SFM of the Coffee dataset (sampled in 6.1c). Figure 6.5 shows the results for the raw-SFM output of the Wafer dataset (sampled in 6.1d). Figure 6.6 shows the results for the raw-SFM output of the Shapes dataset (sampled in 6.1e).

During the construction of each of these clouds, all triangulable points are localised into 3D space. The top 10% of all pairwise interframe matching points are used to seed the SVD registration algorithm that merges each 2.5D view into one single 3D sparse feature model.
Figure 6.2: The raw-SFM for the GingerBeer dataset. The colour variations highlight the different 2.5D views used to construct the full 3D SFM.

Figure 6.3: The raw-SFM for the Nuts dataset.
Figure 6.4: The raw-SFM for the Coffee dataset

Figure 6.5: The raw-SFM for the Wafer dataset
6.4. Refined-SFM Processing

This section explores the results of the refinement procedure outlined in Chapter 5 for the five test objects. The feature extraction results of Section 6.2 were first analysed globally to find the most persistent and strong features across multiple views of the objects. Only these robust features were used to generate the full 3D SFMs. Representative points and feature vectors were then constructed and the final refined-SFMs were generated.

6.4.1. Statistical Analysis

The global feature vector analysis of Section 5.3 was applied to the features of the image datasets extracted in Section 6.2. The distribution of the feature groups were obtained for each dataset and the results are presented in Table 6.3
<table>
<thead>
<tr>
<th>Image Dataset</th>
<th>Number of Global Matches</th>
<th>Number of Groups</th>
<th>Mean number of features per group</th>
<th>STD (σ)</th>
<th>Number of selected groups</th>
<th>Mean number of features for the selected groups</th>
<th>STD (σ) for the selected groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beer-[81]</td>
<td>11803</td>
<td>2848</td>
<td>7.50</td>
<td>3.45</td>
<td>1350</td>
<td>5.99</td>
<td>1.43</td>
</tr>
<tr>
<td>Nuts-[101]</td>
<td>13276</td>
<td>19073</td>
<td>6.51</td>
<td>3.44</td>
<td>11601</td>
<td>6.50</td>
<td>1.12</td>
</tr>
<tr>
<td>Coffee-[55]</td>
<td>7380</td>
<td>844</td>
<td>6.33</td>
<td>2.32</td>
<td>650</td>
<td>5.50</td>
<td>1.11</td>
</tr>
<tr>
<td>Wafer-[49]</td>
<td>15746</td>
<td>3569</td>
<td>6.53</td>
<td>2.29</td>
<td>1831</td>
<td>5.51</td>
<td>1.12</td>
</tr>
<tr>
<td>Shapes-[98]</td>
<td>13729</td>
<td>4231</td>
<td>7.52</td>
<td>2.87</td>
<td>1766</td>
<td>5.98</td>
<td>1.39</td>
</tr>
</tbody>
</table>

Table 6.3: Results of the selection procedure of the global feature vector analysis
Table 6.3 shows an initial average number of features per group of between 6.33-7.52 with a standard deviation of between 2.29-3.45. After the selection of the most promising groups, the average group size was between 5.50-6.50 features, with a variance of 1.11-1.43. A lower STD value highlights the selection a set of groups that consistently contain the most strong and persistent features.

6.4.2. Final Refined-SFMs

Figure 6.7 shows the refined-SFM point cloud for the GingerBeer Dataset. Figure 6.8 shows the results for the refined-SFM output of the Nuts dataset. Figure 6.9 shows the results for the refined-SFM of the Coffee dataset. Figure 6.10 shows the results for the refined-SFM output of the Wafer dataset. Figure 6.11 shows the results for the refined-SFM output of the Shapes dataset.

Figure 6.7: The refined-SFM for the GingerBeer dataset.
Figure 6.8: The refined-SFM for the Nuts dataset

Figure 6.9: The refined-SFM for the Coffee dataset
Figure 6.10: The refined-SFM for the Wafer dataset

Figure 6.11: The refined-SFM for the Shapes dataset
6.5. Raw- and Refined-SFM Comparisons

Table 6.4 shows the reduction in size of the refined sparse feature model a priori data over the raw-SFM data. Overall, there was approximately an 85%±10% reduction in cardinality and data footprint of the refined-SFM vs. the raw-SFM a priori data. This demonstrates a significant element wise reduction, resulting in faster search times when used in time critical, online applications.

<table>
<thead>
<tr>
<th>Item</th>
<th>Raw</th>
<th>Refined</th>
<th>% Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>GingerBeer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D Point Cloud</td>
<td>11803x3</td>
<td>1350x3</td>
<td>88.56%</td>
</tr>
<tr>
<td>Descriptor Database</td>
<td>11803x64</td>
<td>1350x64</td>
<td></td>
</tr>
<tr>
<td>Data footprint</td>
<td>3.05MB</td>
<td>341KB</td>
<td></td>
</tr>
<tr>
<td>Nuts</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D Point Cloud</td>
<td>65233x3</td>
<td>11601x3</td>
<td>82.22%</td>
</tr>
<tr>
<td>Descriptor Database</td>
<td>65233x64</td>
<td>11601x64</td>
<td></td>
</tr>
<tr>
<td>Data footprint</td>
<td>16.74MB</td>
<td>2.98MB</td>
<td></td>
</tr>
<tr>
<td>Coffee</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D Point Cloud</td>
<td>13276x3</td>
<td>650x3</td>
<td>95.10%</td>
</tr>
<tr>
<td>Descriptor Database</td>
<td>13276x64</td>
<td>650x64</td>
<td></td>
</tr>
<tr>
<td>Data footprint</td>
<td>3.62MB</td>
<td>177KB</td>
<td></td>
</tr>
<tr>
<td>Wafer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D Point Cloud</td>
<td>7946x3</td>
<td>1831x3</td>
<td>76.96%</td>
</tr>
<tr>
<td>Descriptor Database</td>
<td>7946x64</td>
<td>1831x64</td>
<td></td>
</tr>
<tr>
<td>Data footprint</td>
<td>2.43MB</td>
<td>560KB</td>
<td></td>
</tr>
<tr>
<td>Shapes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3D Point Cloud</td>
<td>9567x3</td>
<td>1766x3</td>
<td>81.54%</td>
</tr>
<tr>
<td>Descriptor Database</td>
<td>9567x64</td>
<td>1766x64</td>
<td></td>
</tr>
<tr>
<td>Data footprint</td>
<td>2.64MB</td>
<td>490KB</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.4: Size comparisons of the raw- vs. refined-SFM data
Part B: Online Performance Comparisons

In Part B, the raw- and refined-SFM a priori data built for each object in Part A are analysed for improvements in matching efficiency and position and orientation accuracy in online pose estimation. Online AR implementation has a host of problems that exist beyond the scope of this thesis. A few of these are briefly discussed next.

6.6. Problems with Online Augmented Reality

Augmented Reality is required to run in real-time and at high frame rates so that the co-existence of the real and virtual worlds are realistic and believable [24]. However, the notion of ‘real-time’ performance in an online augmented reality system is a complex issue. As the processing time inherent in the system is non-zero, a virtual augmentation will never be synchronous to the movement of a user, object or scene. This section explores the sources of error that the methods in this thesis aim to reduce during online recognition using sparse feature model a priori data.

6.6.1. The Human Factor

In terms of visual perception, the human vision system does not behave like camera. The brain receives and interprets visual data as a continuous flow of information rather than static snap shots in time [108]. This non-zero processing time is highly subjective; with the brain using many different perceptual tricks maintain a flow of acceptable visual information [109]. This perception of motion governs the amount of latency a user could tolerate in an online AR scenario.

The quality of registration of virtual information onto real world structures is dependent on the eye’s ability to sharply and clearly recognise the extent or shape of features within an image. The spatial resolution of the human eye defines how close two features can be
within an image and still be resolved as unique. Human vision achieves its highest spatial resolution in just a small area at the centre of the field of view called the fovea [97]. As distance from the fovea increases, the spatial resolving ability of the retina decreases [110], hence when focusing on an object, the head and eyes move so that the target light needs to be processed falls on this area.

The generally accepted spatial resolution (or acuity) of the human eye is between 50 to 75 cycles per degree, or 1.2 arc minutes to 0.8 arc minutes per degree [24, 97, 110-113]. At arms length, the width of a little finger represents about one degree. If a virtual object being rendered at arms length had a registration error of around one degree, it would be highly noticeable. Hence, for a virtual object to appear to be precisely aligned, the error in registration must be less than around 1/75\textsuperscript{th} the thickness of a little finger, or about 0.2mm at arms length.

The psychophysical perception of just noticeable differences (JNDs) [114, 115] describes the minimum change in a stimulus that an observer could detect. There has been considerable effort in defining JNDs to help bound the tolerable limits of both processing time and alignment precision for virtual to real-world registration [24, 116-118]. Some of these are explored in the next section.

### 6.6.2. Online Error Types

Several common types of error have been identified for both virtual environments [116, 118] and augmented environments [117, 119-121]. The most common and noticeable differences arise from system latency and geometric alignment errors.

Latency is a key issue when presenting virtual elements to a user, which should appear to synchronise to the motion of a user or object [117, 118]. Pose determination is by far the most significant factor in
the processing time a system requires to analyse visual data from a scene and return an augmented view. In the case of object recognition with SFM data, the largest source of time inefficiency is establishing correspondences between incoming features from the scene and the object representations in the SFM database.

As mentioned in Section 4.3.3, matching high dimensional features is a computationally intensive process. Binary trees and approximate nearest neighbour search methods can run significantly faster over linear methods, especially for high dimensional data. Implementing these strategies into an online recognition system can help reduce system latency.

One other important strategy focuses on the reduction of the cardinality of a database. As discussed in Section 2.4 and in Chapter 5, having more features to represent an object does not necessarily increase accuracy and can be detrimental in terms of efficiency [9]. In a similar notion to Baheti et al. [8], the methods of Chapter 5 aim to refine the amount of features representing an object. Storing only strong features that are persistent across several viewpoints (Section 5.2) is an ideal method of reducing the cardinality of a database, whilst retaining its discernibility.

Condensing multiple 3D points and descriptors that represent one object feature into a representative point (Section 5.6) and feature vector (Section 5.7) also significantly reduces the amount of elements in an SFM database. Refined-SFM data helps reduce latency in feature matching at its core, by reducing the amount of features that need to be matched.

Alignment errors in the registration of virtual and real world entities are another prominent issue. Inaccuracies in position and orientation estimates can cause noticeable misalignments of the virtual and real.
Thus, confidence in the a priori data used to describe objects in the real world must be high. A priori data built using the methods of Chapters 4 and 5 has high location accuracy due to the calibrated short-baseline stereo system. As the precise displacement of both cameras relative to each other is known, triangulating matched features is very accurate.

Inconsistent pose estimates in each frame during a motion sequence can also cause jitter. With unreliable registration, a virtual object could appear to float or wobble around its intended location. Jitter can occur when multiple points are used to describe one object feature. If a slightly different point in the SFM database is used to locate one matched object feature in each subsequent frame in a sequence, the small geometric differences can introduce inconsistent pose estimates. Building a representative point for a cluster of points that describe one object feature helps alleviate this problem. Also, keeping features that are consistent across multiple perspectives is also important in reducing jitter. Temporal consistency checks across multiple streaming frames are another technique to reduce jitter.

The dynamic errors of latency and registration for online AR systems are large areas of research on their own. The scope of this thesis has been to address the offline generation and refinement of a priori data for potential use in online pose estimation. Hence, the following results present the behaviour of a simple online pose estimation system when using both raw- and refined-SFM data.

### 6.7. Online System Setup

A Fujifilm FinePix REAL 3D W3 attached to an Epson robot with an RC520 controller (Figure 6.12) was used to image the test objects from known positions in space. The precise pose of the object relative to the camera was derived from the absolute position of the robot’s end
effector in relation to the calibrated position of the object. This benchmark pose was then compared to the pose estimation results of a vision based recognition system using both raw and refined sparse feature model a priori data. The robots accuracy of pose and repeatability of pose were considered to have minimal error compared to the vision system, and were hence assumed to be negligible.

Figure 6.12: The robotic system used to validate registration accuracy of both raw- and refined-SFM a priori data.
6.7.1. System Calibration

In order to relate the position of the object to an SFM in the camera reference frame on the robot, a transformation that maps that coordinate system of the object to the coordinate system of the camera must be established. The different frames of reference are listed below and are shown in Figure 6.13.

- The world frame (Figure 6.13, light blue)
- The Camera frame (Figure 6.13, red)
- The Object frame (Figure 6.13, dark blue)
- The robot’s end effector (EEF) frame (Figure 6.13, yellow)
- The robot base frame (Figure 6.13, orange)

Due to careful installation, the robot’s base frame has a fixed and known position in the world frame. The object was also placed in a precise configuration, defined in the world frame, to establish the
relationship between the robot and the object. The robot’s forward
kinematics provide the relation between the robot’s EEF frame and the
robot’s base frame, however the relationship between the robot’s end
effector and the camera is unknown (as shown in Figure 6.13, coloured
green). As the camera is fixed to the robot’s EEF, this missing relation
can be accurately obtained.

The camera’s orientation with respect to the EEF can be tightly
controlled through the proper alignment of the camera mount to the
EEF in roll, pitch and yaw. Simplified engineering drawings from
Fujifilm were used to determine the position of the left camera sensor
to the camera mount screw on the bottom. Similarly, the specifications
of the Manfrotto camera mount were used to establish the precise
distance of camera mounting base to the EEF attachment. From this
information, the relationship between the EFF and the left camera
sensor was established. (Note: SFM data is defined in the left camera
coordinate space).

6.7.2. Extracting Pose Online

Identifying the position and orientation or ‘pose’ of an object is one of
the most prominent issues in online augmented reality. As overviewed
in Chapter 2, the recognition of specific objects in visual data is
possible with a priori data, allowing the position and orientation of
multiple entities with respect to the camera to be calculated
independently. To do this, correspondence must be established
between features in the input data and features in the a priori
database. From this correspondence, the transformation that maps the
appearance of a 3D object from a single perspective can be
established.

Under the assumptions in Section 1.3, the capturing device restricts
the form of the input data in this thesis to passive digital images. Thus,
single images, spatially displaced stereo images or temporally displaced stereo images can be used. Often, online pose estimation is restricted to monocular imaging [35, 87, 122, 123], where temporal constraints between neighbouring frames can be used to smooth out a pose estimate over time. This problem is known as the pose from n points or PnP problem. When the internal parameters of a camera are known, this knowledge can be used explicitly to determine pose.

POSIT [122] is one technique to estimate pose for $n \geq 4$ 2D-to-3D point correspondences. Vachetti et al. [35] used POSIT to establish pose between single input images and 3D a priori information. Another popular solution for the PnP problem comes from Lepetit et al. [123]. EPnP proposes a non-iterative solution for the PnP problem of $n \geq 4$, whose computational complexity grows linearly with $n$. It handles both planar and non-planar cases. These techniques are restricted by single image input data. When 3D input data is available, pose estimation can become simpler and more accurate.

As discussed throughout this thesis, short-baseline stereo images allow the estimation of the depth of points of interest from one camera snapshot. Thus, the 3D location of 2D image features can be extracted from a stereo pair and mapped to an a priori database in a 3D-to-3D registration fashion. More so, the 3D distribution of features in one image can help segment features into clusters, based on their depth arrangement with respect to one another. This greatly assists the rejection of outliers in the correspondence set and also with the recognition of multiple objects in one scene.

To initiate 3D-to-3D registration, the correspondence between 3D features must be established. The majority of work in 3D-to-3D registration is restricted by purely 3D point data. Thus the shape and contours of input 3D data is used to find similar structures in a
database of 3D objects. Quite often published methods are under the assumption that the 3D input data is dense enough to form these structures [124, 125]. However, determining the 3D pose of objects within sparse range data is significantly more difficult without dense volumetric information.

The bounded Hough proposed by Greenspan et al. [126] is quite fast at detecting and tracking an object from sparse range images, but only provides a coarse estimate of its pose. Shang [127] proposed a hybrid method of the bounded Hough transform to get an initial pose estimate before running a few ICP [128, 129] iterations to determine more accurate pose. Park et al. [130] proposed a method similar to Greenspan [126] for pose estimation from a single range image, by using a brute force tactic implemented using a GPU for massive parallelism. These methods are computational intensive to run, and are restricted to performing 3D shape detection only.

The methods in this thesis retain the unique and descriptive 2D feature information from which each sparse feature model has been constructed. This 2D feature information is localised in the 3D geometric structure of an SFM. Hence, correspondence between 2D image features in the input images to their counterparts in an SFM will develop a list of possible points from which to generate a pose estimate. These sparse features in the input images can then be triangulated and filtered based on their 3D arrangement, which can then seed registration via 3D-to-3D techniques. In the case of this thesis, the SVD method of [93] as described in Program 4.1 is used to determine the pose of the test objects.

The pseudo code in Program 6.1 outlines this procedure used to establish pose in this Chapter.
Program 6.1: Pose estimation procedure

**Inputs:** sampled stereo pair, SFM a priori database

**Output:** optimal rigid transformation parameters $\begin{bmatrix} \hat{R} & \hat{t} \end{bmatrix}$ of object with respect to the camera.

1: Extract all triangulable features in the stereo pair

2: Match the extracted 2D feature descriptors to the 2D descriptive feature information in the SFM database

3: Triangulate only the positively matched features between the input data and the SFM database

4: Filter for geometric consistency

5: Seed and execute SVD

6: Refine with ICP (if required / if time permits)

7: Return $\begin{bmatrix} \hat{R} & \hat{t} \end{bmatrix}$ (object pose)

As discussed in section 6.6.2, feature matching efficiency is the main cause of latency in such a system. Choosing a fast method of matching high dimensional features can optimise the efficiency of pose estimation.

6.8. Online Pose Estimation Performance

The methods in this thesis detail the offline generation of raw and refined sparse feature model a priori data. Though complete online AR implementation is beyond the scope of this thesis, it is important to assess the behaviour of the SFM data in its intended environment: online pose estimation. Hence, the robotic system described in Section 6.7 was used to generate experimental data to assess the performance of the refined-SFM data over the raw-SFM data in several online pose estimation scenarios. The SFM a priori data generated in Section 6.3 and Section 6.4 was used during the following experiments.
To assess the performance improvements of refined-SFM data over raw-SFM data, two experiments were implemented. The first experiment quantified the efficiency improvements of raw- vs. refined-SFM a priori data. As the cardinality of an SFM decreases feature matching efficiency improves, thus reducing latency during pose estimation. The second experiment quantified the amount of error encountered during a single frame pose estimate. The quality of the registration of a virtual object coordinate system and the real object coordinate system is entirely dependent on the quality of the a priori data.

A program was developed to move the robot into 5 different points around 3 chosen objects. To analyse the repeatability of the method, the 5 different points were consecutively visited 10 times each, giving a total of 50 frames of experimental data per test object. At each location, a stereo image was captured. From the robot’s kinematics, the precise transformation that maps the real object’s coordinate system to the camera frame was derived (as described in Section 6.7.1). Each stereo pair was fed into the pose estimation routine described in Section 6.7.2 to estimate the pose of the object using both raw- and refined-SFM a priori data.

For each of the three test objects, 20 pose estimates were run for each of the 5 programmed positions: 10 using raw-SFM data and 10 using refined-SFM data. The mean matching efficiency and standard deviation of the 10 raw-SFM data samples and 10 refined-SFM data samples were taken for each programmed position. These are explored in Section 6.8.1. Similarly, for each of the 5 programmed positions, the 20 pose estimates were compared to the test object’s ‘actual’ pose as derived from the robot’s kinematics. These are explored in Section 6.8.2.
6.8.1. Feature Matching Efficiency

Once again, efficiency is entirely dependent on the implementation of the feature matching routine used and the hardware it’s deployed on. For simplicity and for the sake of the registration accuracy experiment in Section 6.8.2, a linear search was used to match features between the input stereo pair and the SFM a priori database. This returned the best possible matches for registration accuracy, but at a high time cost. In this experiment, the main aim was to demonstrate the net benefit of refining a priori data during an example search strategy. Similar benefits can translate to other search strategies; therefore the total elapsed time is unimportant in this investigation. A more useful measure is the net improvement in efficiency.

Table 6.5 lists the element size of the raw- and refined-SFM datasets. During feature matching, the subset of input feature vectors must be compared to all of the feature vectors in the raw- and refined-SFMs. Considerable increases in efficiency can be made when traversing a search space of lower cardinality.

<table>
<thead>
<tr>
<th>SFM Footprints</th>
<th>GingerBeer</th>
<th>Nuts</th>
<th>Coffee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw-SFM</td>
<td>11803</td>
<td>65233</td>
<td>13276</td>
</tr>
<tr>
<td>Refined-SFM</td>
<td>1350</td>
<td>11601</td>
<td>650</td>
</tr>
</tbody>
</table>

*Table 6.5: The raw- and refined-SFM element size per dataset*

In Table 6.6, the average improvement in feature matching efficiency is listed as a percentage improvement of refined-SFM data over raw-SFM data. The average number of feature vectors for each of the programed pose locations is given in column 2. Given the similarity of the images taken at each programmed pose test location, the number of features varied little. However, slight differences in each image may have returned slightly different 3D localisation results. This set of input
feature vectors had to then be compared to the feature vectors in the raw-and refined-SFM datasets (who’s sizes are listed in Table 6.4/6.5). Local feature matching of this form is computationally intensive, due to the number of comparisons between all 64-dimentional feature vectors.

<table>
<thead>
<tr>
<th>Pose</th>
<th>Average number of input features per pose</th>
<th>Average % improvement in feature matching efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>GingerBeer</td>
</tr>
<tr>
<td>1</td>
<td>746</td>
<td>85.40%</td>
</tr>
<tr>
<td>2</td>
<td>722</td>
<td>83.56%</td>
</tr>
<tr>
<td>3</td>
<td>563</td>
<td>88.53%</td>
</tr>
<tr>
<td>4</td>
<td>644</td>
<td>87.63%</td>
</tr>
<tr>
<td>5</td>
<td>687</td>
<td>84.57%</td>
</tr>
</tbody>
</table>

*Table 6.6: The average percentage improvement in efficiency when matching features from each programmed test pose location*

Overall, the feature matching to the GingerBeer refined-SFM was 85.94% more efficient than the raw-SFM. Matching input features to the refined Nuts SFM was 82.41% more efficient than the raw-SFM data. Similarly, the refinement of the coffee SFM found an improvement of 94.47% when matching to refined- vs. raw-SFM data.

### 6.8.2. Registration Accuracy

With the best possible correspondence returned by a linear matching strategy, a measure of accuracy becomes dependent on the quality of the 3D localisation of the feature points in the SFM a priori data. With poorly localised, or inconsistent points in an SFM, the pose estimates used for registration can inherently incur error. This experiment assessed the amount of error difference between pose estimates generated from raw-SFMs and refined-SFMs when compared to the baseline pose values from the robot.
In the following pose evaluation tables (Tables 6.7-6.12), the performance for both the position and orientation of the pose estimates generated for each SFM are assessed. Both tables have the following accuracy performance evaluations, respective to either the position \((x, y, z)\) or orientation (roll \((\gamma)\), pitch \((\beta)\), yaw\((\alpha)\)):

1. **Pose:** The 5 programmed positions that the robot visited
2. **Actual:** The mean actual pose of the object for each of the 5 positions, derived from the robot’s kinematics (camera frame).

**Raw-SFM accuracy**

3. **Mean:** the mean position/orientation calculated from the pose estimate using the raw-SFM data (in the camera frame).
4. **SSE:** the sum of squared errors between the actual pose and each position/orientation estimate using the raw-SFM data.
5. **STD:** the standard deviation (variation) from the mean position/orientation estimates

**Refined-SFM accuracy**

6. **Mean:** the mean position/orientation calculated from the pose estimate using the refined-SFM data (in the camera frame).
7. **SSE:** the sum of squared errors between the actual pose and each position/orientation estimate using the refined-SFM data.
8. **STD:** the standard deviation (variation) from the mean position/orientation estimates

**Percentage Improvement**

9. **SSE:** The improvement in the sum of the squared errors between position/orientation estimates from raw- and refined-SFM data.
10. **STD:** the improvement in the standard deviation error between position/orientation estimates from raw- and refined-SFM data.
Position evaluation: GingerBeer

<table>
<thead>
<tr>
<th>Pose</th>
<th>Actual</th>
<th>Raw-SFM Position Accuracy</th>
<th>Refined-SFM Position Accuracy</th>
<th>% Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Mean</td>
<td>SSE</td>
<td>STD</td>
</tr>
<tr>
<td>1</td>
<td>x</td>
<td>-0.4</td>
<td>-2.6</td>
<td>147.3</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>-5.5</td>
<td>-5.8</td>
<td>75.3</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>-344.1</td>
<td>-344.2</td>
<td>101.9</td>
</tr>
<tr>
<td>2</td>
<td>x</td>
<td>-23.7</td>
<td>-23.9</td>
<td>94.6</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>-2.5</td>
<td>-3.3</td>
<td>55.0</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>-466.2</td>
<td>-466.1</td>
<td>58.5</td>
</tr>
<tr>
<td>3</td>
<td>x</td>
<td>23.5</td>
<td>24.6</td>
<td>96.5</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>-18.8</td>
<td>-18.8</td>
<td>34.3</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>-455.9</td>
<td>-456.9</td>
<td>73.3</td>
</tr>
<tr>
<td>4</td>
<td>x</td>
<td>3.2</td>
<td>4.4</td>
<td>47.1</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>-2.3</td>
<td>-1.2</td>
<td>43.7</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>-442.1</td>
<td>-442.5</td>
<td>87.5</td>
</tr>
<tr>
<td>5</td>
<td>x</td>
<td>3.6</td>
<td>4.2</td>
<td>86.1</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>11.8</td>
<td>13.0</td>
<td>66.3</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>-364.3</td>
<td>-365.5</td>
<td>76.4</td>
</tr>
</tbody>
</table>

Table 6.7: GingerBeer Position Accuracy
Orientation evaluation: GingerBeer

<table>
<thead>
<tr>
<th>Pose</th>
<th>Actual</th>
<th>Raw-SFM Orientation Accuracy</th>
<th>Refined-SFM Orientation Accuracy</th>
<th>% Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>γ</td>
<td>Mean</td>
<td>SSE</td>
<td>STD</td>
</tr>
<tr>
<td>1</td>
<td>180.0</td>
<td>180.3</td>
<td>58.8</td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>125.3</td>
<td>125.1</td>
<td>28.8</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-45.0</td>
<td>-43.0</td>
<td>59.2</td>
</tr>
<tr>
<td>2</td>
<td>γ</td>
<td>159.6</td>
<td>160.0</td>
<td>87.7</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>137.9</td>
<td>138.5</td>
<td>141.4</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-71.6</td>
<td>-70.3</td>
<td>80.8</td>
</tr>
<tr>
<td>3</td>
<td>γ</td>
<td>155.9</td>
<td>156.7</td>
<td>23.1</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>116.6</td>
<td>116.5</td>
<td>86.0</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-90.0</td>
<td>-89.5</td>
<td>67.3</td>
</tr>
<tr>
<td>4</td>
<td>γ</td>
<td>90.0</td>
<td>88.8</td>
<td>119.3</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>141.1</td>
<td>142.5</td>
<td>198.5</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-135.0</td>
<td>-134.6</td>
<td>92.1</td>
</tr>
<tr>
<td>5</td>
<td>γ</td>
<td>37.6</td>
<td>38.4</td>
<td>72.8</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>123.3</td>
<td>123.9</td>
<td>120.7</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-170.5</td>
<td>-171.6</td>
<td>73.2</td>
</tr>
</tbody>
</table>

Table 6.8: GingerBeer Orientation Accuracy
<table>
<thead>
<tr>
<th>Pose</th>
<th>Actual</th>
<th>Raw-SFM Position Accuracy</th>
<th>Refined-SFM Position Accuracy</th>
<th>% Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SSE</td>
<td>STD</td>
<td>SSE</td>
<td>STD</td>
</tr>
<tr>
<td>1</td>
<td>0.3</td>
<td>3.1</td>
<td>-0.2</td>
<td>2.7</td>
</tr>
<tr>
<td>2</td>
<td>-5.5</td>
<td>2.8</td>
<td>2.8</td>
<td>52.7</td>
</tr>
<tr>
<td>3</td>
<td>-297.7</td>
<td>2.3</td>
<td>-297.9</td>
<td>39.6</td>
</tr>
<tr>
<td>4</td>
<td>-224</td>
<td>4.3</td>
<td>-224</td>
<td>119.3</td>
</tr>
<tr>
<td>5</td>
<td>126</td>
<td>7.3</td>
<td>10.1</td>
<td>48.8</td>
</tr>
</tbody>
</table>

Table 6.9: Nuts Position Accuracy
Orientation evaluation: Nuts

<table>
<thead>
<tr>
<th>Pose</th>
<th>Actual</th>
<th>Raw-SFM Orientation Accuracy</th>
<th>Refined-SFM Orientation Accuracy</th>
<th>% Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>γ</td>
<td>Mean</td>
<td>SSE</td>
<td>STD</td>
</tr>
<tr>
<td>1</td>
<td>γ</td>
<td>180.0</td>
<td>178.5</td>
<td>179.0</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>109.5</td>
<td>108.5</td>
<td>107.8</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-45.0</td>
<td>-44.9</td>
<td>106.5</td>
</tr>
<tr>
<td>2</td>
<td>γ</td>
<td>162.8</td>
<td>164.9</td>
<td>131.5</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>128.3</td>
<td>127.8</td>
<td>74.7</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-71.6</td>
<td>-70.7</td>
<td>159.7</td>
</tr>
<tr>
<td>3</td>
<td>γ</td>
<td>166.4</td>
<td>166.6</td>
<td>87.8</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>104.0</td>
<td>104.6</td>
<td>88.0</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-90.0</td>
<td>-87.3</td>
<td>89.9</td>
</tr>
<tr>
<td>4</td>
<td>γ</td>
<td>90.0</td>
<td>89.7</td>
<td>43.0</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>131.5</td>
<td>131.3</td>
<td>47.9</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-135.0</td>
<td>-133.1</td>
<td>119.3</td>
</tr>
<tr>
<td>5</td>
<td>γ</td>
<td>23.6</td>
<td>23.1</td>
<td>84.3</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>108.2</td>
<td>109.0</td>
<td>62.4</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-170.5</td>
<td>-171.1</td>
<td>53.2</td>
</tr>
</tbody>
</table>

Table 6.10: Nuts Orientation Accuracy
Position evaluation: Coffee

<table>
<thead>
<tr>
<th>Pose</th>
<th>Actual</th>
<th>Raw-SFM Position Accuracy</th>
<th>Refined-SFM Position Accuracy</th>
<th>% Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Mean</td>
<td>SSE</td>
<td>STD</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>-0.6</td>
<td>-1.4</td>
<td>39.1</td>
<td>1.9</td>
</tr>
<tr>
<td>y</td>
<td>-5.5</td>
<td>-7.0</td>
<td>69.2</td>
<td>2.3</td>
</tr>
<tr>
<td>z</td>
<td>-375.3</td>
<td>-375.4</td>
<td>80.3</td>
<td>3.0</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>-24.1</td>
<td>-22.9</td>
<td>65.9</td>
<td>2.4</td>
</tr>
<tr>
<td>y</td>
<td>-2.0</td>
<td>-2.9</td>
<td>18.4</td>
<td>1.0</td>
</tr>
<tr>
<td>z</td>
<td>-505.9</td>
<td>-505.2</td>
<td>77.5</td>
<td>2.9</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>22.8</td>
<td>21.6</td>
<td>112.3</td>
<td>3.3</td>
</tr>
<tr>
<td>y</td>
<td>-20.7</td>
<td>-20.3</td>
<td>20.2</td>
<td>1.4</td>
</tr>
<tr>
<td>z</td>
<td>-477.5</td>
<td>-476.8</td>
<td>25.1</td>
<td>1.5</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>3.2</td>
<td>2.4</td>
<td>64.3</td>
<td>2.5</td>
</tr>
<tr>
<td>y</td>
<td>-1.8</td>
<td>-2.6</td>
<td>33.2</td>
<td>1.7</td>
</tr>
<tr>
<td>z</td>
<td>-481.9</td>
<td>-481.1</td>
<td>57.3</td>
<td>2.4</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>2.7</td>
<td>2.8</td>
<td>81.3</td>
<td>3.0</td>
</tr>
<tr>
<td>y</td>
<td>12.0</td>
<td>11.4</td>
<td>30.4</td>
<td>1.7</td>
</tr>
<tr>
<td>z</td>
<td>-395.1</td>
<td>-396.3</td>
<td>25.8</td>
<td>1.1</td>
</tr>
</tbody>
</table>

Table 6.11: Coffee Position Accuracy
**Orientation evaluation: Coffee**

<table>
<thead>
<tr>
<th>Pose</th>
<th>Actual</th>
<th>Raw-SFM Orientation Accuracy</th>
<th>Refined-SFM Orientation Accuracy</th>
<th>% Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>γ</td>
<td>Mean</td>
<td>SSE</td>
<td>STD</td>
</tr>
<tr>
<td>1</td>
<td>180.0</td>
<td>180.2</td>
<td>63.4</td>
<td>2.6</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>131.5</td>
<td>131.5</td>
<td>81.3</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-45.0</td>
<td>-44.6</td>
<td>20.5</td>
</tr>
<tr>
<td>2</td>
<td>γ</td>
<td>158.6</td>
<td>157.8</td>
<td>32.0</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>141.7</td>
<td>141.3</td>
<td>28.0</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-71.6</td>
<td>-72.6</td>
<td>61.0</td>
</tr>
<tr>
<td>3</td>
<td>γ</td>
<td>152.1</td>
<td>151.9</td>
<td>13.4</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>122.0</td>
<td>121.2</td>
<td>100.6</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-90.0</td>
<td>-89.3</td>
<td>63.0</td>
</tr>
<tr>
<td>4</td>
<td>γ</td>
<td>90.0</td>
<td>90.4</td>
<td>49.2</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>144.7</td>
<td>145.3</td>
<td>98.3</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-135.0</td>
<td>-133.9</td>
<td>69.0</td>
</tr>
<tr>
<td>5</td>
<td>γ</td>
<td>41.6</td>
<td>41.0</td>
<td>75.5</td>
</tr>
<tr>
<td></td>
<td>β</td>
<td>129.4</td>
<td>128.9</td>
<td>26.2</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>-170.5</td>
<td>-169.6</td>
<td>54.6</td>
</tr>
</tbody>
</table>

*Table 6.12: Coffee Orientation Accuracy*
From these results, the following remarks can be made:

1. Refined-SFM a priori data on average returned 14.78% better position accuracy with 7.70% lower variance in comparison to raw-SFM a priori data.

2. Refined-SFM a priori data on average returned 12.22% better orientation accuracy with 4.30% lower variance in comparison to raw-SFM a priori data.

These results clearly show the advantage of refining a priori data for use in online pose estimation. A lower error in pose estimation will result in more precise registration. A lower variance in error will also reduce the amount of jitter imparted on the registration routine, and hence limit the amount of temporal smoothing that may be required.

6.9. Conclusion

In the Part A of this chapter, the methods of Chapters 4 and 5 were applied to 5 real objects. Raw-SFM a priori data was generated and subsequently refined to produce refined-SFM data. In Part B, a subset of this experimental data was used in an online pose estimation scenario to assess the efficiency and accuracy improvements of using raw- vs. refined-SFM data. A robot was used to validate the accuracy of the pose estimation. The results for three of these objects are listed below.

The refinement of the GingerBeer dataset resulted in:

- An 88.56% reduction in cardinality, resulting in an average matching efficiency improvement of 85.94% when estimating pose online.
- An accuracy improvement of 10.79%, with a 5.56% reduction in variance in position estimation
• An accuracy improvement of 3.45%, with a 1.17% reduction in variance in orientation estimation

The refinement of the Nuts dataset resulted in:

• An 82.22% reduction in cardinality, resulting in an average matching efficiency improvement of 82.41% when estimating pose online.
• An accuracy improvement of 22.76%, with a 12.12% reduction in variance in position estimation
• An accuracy improvement of 17.04%, with a 8.93% reduction in variance in orientation estimation

The refinement of the Coffee dataset resulted in:

• A 95.10% reduction in cardinality, resulting in an average matching efficiency improvement of 94.47% when estimating pose online.
• An accuracy improvement of 10.78%, with a 5.43% reduction in variance in position estimation
• An accuracy improvement of 5.38%, with a 2.79% reduction in variance in orientation estimation

These promising results highlight the benefit of refining a priori data before it is used in online augmented reality systems. Refining data with the methods proposed by this thesis results in more efficient and accurate pose estimation and hence registration for AR applications.
Conclusions and Beyond

To maintain illusion of augmented reality, the precise pose of real objects in freespace on which to render virtual information must be resolved in a quick and precise manner. The quality of the a priori data used for recognition based pose estimation is a major proponent in the misalignment of virtual and real world entities. A priori data is considered absolute truth without the need for validation. The upmost confidence in the comprehensiveness, accuracy and size efficiency of this data is paramount.

Most recognition-based augmented reality methods today do not devote enough attention to the generation of their a priori data. Without
a strong and efficient foundation, the quality of the registration and subsequently the user experience is compromised. This thesis contributes a novel a priori knowledge generation and refinement methodology to increase the efficiency and accuracy of recognition based pose estimation for augmented reality applications.

The principal methods in this thesis detailed the generation of Sparse Feature Model (SFM) a priori data. The main focus of this research was to exhaustively analyse the creation of a priori knowledge and propose an optimal solution to construct and refine SFM data. Building on the extensive knowledge and expertise of both the computer vision and image processing communities, the following methods were developed to achieve this goal.

Chapter 4 introduced a new methodology for the generation of sparse feature model data from a passive, short-baseline stereo system. The creation of a priori knowledge from many perspectives is critical to accurately and comprehensively characterise an object. Multiple short-baseline stereo images were captured from different perspectives around an object. For each stereo pair, a 2.5D point cloud was generated by the triangulation of corresponding, highly descriptive object features. A unique, raw-SFM was reconstructed from these multiple views by merging each 2.5D point cloud together using 3D-to-3D shape registration.

The main contribution of Chapter 4 was the integration of the methodology as a whole. Careful consideration and experimentation was employed for each component of the method to generate the final results, given the nature of the input data under the assumptions outlined in Chapter 1. This unique approach hybridised the best methods of 2D-to-3D and 3D-to-3D registration to further advance a priori generation from multi-view data. With the increase in cardinality
of the final SFM data over typical single camera methodologies, refinement was needed to boost matching efficiency and accuracy during online pose estimation.

Chapter 5 introduced a three-tiered approach to refine raw sparse feature model a priori data following the methods of Chapter 4. Statistical analysis was used to measure the strength and persistence of a set of triangulable features extracted from multiple short-baseline images of an object. The occurrence and variance of persistent feature vectors across multiple views were used to condense the initial raw sets. The geometric structure and descriptive feature components of a refined-SFM were analysed to reduce the cardinality of the a priori data further. Representative points were introduced as a Gaussian weighted mean of multiple point locations for individual object feature groups. Representative feature vectors were introduced to condense multiple descriptors representing the same object feature into a single vector.

The contributions of the refinement methods in Chapter 5 were demonstrated using the test dataset. There was a 62% reduction in the cardinality of the refined-SFM over raw-SFM data generated for the test dataset. The cardinality of the refined data was then further reduced by approximately 70%, by condensing multiple entities describing single object features into representative points and representative feature vectors. It was shown that the creation of a representative feature vector does not generalise its discernibility. In the test dataset, feature vectors built from the Gaussian weighted mean of a group of matched features retained a high similarity and optimal match against all group members. These results demonstrate the advantage of condensing information, rather than randomly culling points out of an a priori dataset.
Chapter 6 evaluated the a priori data generation methods of Chapters 4 and 5 for real world objects. Image datasets of five different objects were generated. These image datasets were processed by the methods of Chapter 4 and then refined by the methods of Chapter 5 to generate a raw- and refined-SFM a priori knowledge for each object. The refinement process reduced the cardinality and data footprint of raw-SFMs by approximately $85\% \pm 10\%$. This demonstrates a significant element wise reduction, resulting in faster search times when used in time critical, online applications.

Though complete online AR implementation was beyond the scope of this thesis, it was important to assess the benefits of refining a priori data in online pose estimation. Two experiments were implemented. The first in Chapter 6 evaluated the improvements in feature matching efficiency when using refined- vs. raw-SFM data. For three test cases, it was found that matching efficiency improved in roughly the same order as the reduction in cardinality. This is an expected result, as with less data to traverse, feature matching would inherently take less time. This result validates one of the main aims of this thesis: to increase the efficiency of online pose estimation via recognition by refining the a priori object representational data.

A priori data built using the methods of Chapters 4 and 5 also has high location accuracy due to the calibrated short-baseline stereo system. However, having a large number of points can sometimes be detrimental to precise registration. The second experiment of Chapter 6 quantified the amount of error in pose estimates generated off both raw- and refined-SFMs in comparison to known locations in freespace. A robot was used to establish reference poses that were compared to actual pose estimates at each location by an online recognition system. Refined-SFM data on average returned 14.78% better position accuracy with 7.70% lower variance than raw-SFM data. Refined-SFM
data on average returned 12.22% better orientation accuracy with 4.30% lower variance that raw-SFM data.

These results validate another main aim of this thesis: to improve accuracy via refinement. A lower error in pose estimation will result in more precise registration. A lower variance in error will also reduce the amount of jitter imparted on the registration routine during online operation.

The research in this thesis represents a synergy of 2D and 3D computer vision and image processing techniques, which together produce comprehensive and computationally efficient a priori data. The proposed methods to reduce the cardinality of a priori data and yet retain distinctive and persistent features are novel within the augmented reality community. This research will help reduce latency and increase accuracy in recognition based pose estimation systems, thus improving the user experience for augmented reality applications.

7.1 Future Research

The methods in this thesis have been designed to be adaptable for various a priori data types. The premise has been quite simple: generate comprehensive a priori data from many views of an object, and then retain the strongest and most persistent features across multiple perspectives. This thesis analysed the methodology’s application to textured objects, and hence the choice of the SURF feature detector and descriptor. In this configuration, this technique is not robust to poorly textured objects. To increase the robustness of the a priori datasets, additional feature detection schemes that sample not only object texture, but also object geometry could be included. This would generate hybrid a priori data that would be robust to textured and non-textured objects.
The structure of a priori datasets could also be manipulated to be robust to various unique conditions. For example, segmenting a single sparse feature model by creating node graphs of interconnected clusters of features could make the data more robust to deformable object detection and pose estimation. This approach would require some novel online recognition research, tightly coupled the offline generation process. Within the augmented reality community today, this interesting future research approach could push AR to the next level as a disruptive technology.

Further research into the online performance and efficacy of sparse feature model a priori knowledge would also be a future research direction worth exploring. The original intended application for the methods in this thesis was the recognition of known ordnance during the teleoperation of mobile reconnaissance robots. However, addressing the complexities of simply generating stable, robust and concise a priori data was a great challenge. There are many more problems without solutions in present day online AR applications, but with the methods of this thesis delivering a strong foundation of offline a priori knowledge to these online procedures, it is hoped that the issues of online recognition and registration are diminished.

The representation of objects by a priori data is presently the only way for augmented reality to contextually augment real world objects with virtual information. Generating these representations offline is the only way to develop object level recognition today. Until machines become advanced enough to learn and classify individual structures from posterior knowledge, true recognition based AR will always require some level of prior knowledge. Sparse feature model a priori data is the most comprehensive solution for recognition based augmented reality today.
References


[85] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis


“End Of Line”
The MCP, *TRON*