Multi-Image Matching using Invariant Features

by

Matthew Alun Brown

B.A., Cambridge University, 2000
M.Eng., Cambridge University, 2000

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE STUDIES

(Computer Science)

The University of British Columbia

July 26, 2005

Abstract

This thesis concerns the problems of automatic image stitching and 3D modelling from multiple views. These are basic problems of computer vision, with applications in robotics, architecture, industrial inspection, surveillance, computer graphics and film. Recent work has brought increasing automation to these tasks, but despite a large amount of progress, state-of-the-art algorithms still require some form of user input or assumptions about the image sequence. For example, the best image stitchers currently require an ordered set of input images, or user input to identify the matching images, before automatic registration can proceed. In this work we show how such tasks can be performed automatically and without any user input at all.

We formulate the multi-image matching problem as one of finding all matching images, subject to the constraint that they are consistent views from a perspective camera. We use invariant features as a mechanism for finding correspondences, and indexing techniques to efficiently find matches between multiple views. We then find all sets of geometrically consistent feature matches, using a probabilistic model for verification. This allows us to identify each object or scene in the dataset using only the structure already present in the data. The major contributions of this thesis are the development of a system that can automatically recognise and stitch 2D panoramas in unordered image datasets, and a new class of invariant features for this purpose.

Contents

List of Tables

List of Figures

Acknowledgements

Firstly, I would like to thank David Lowe for sharing his considerable wisdom in this field, and providing invaluable guidance in the supervision of this thesis. I would also like to thank the other members of my supervisory committee: Nando de Freitas, Jim Little and Wolfgang Heidrich for their insights and inspiration.

I am most grateful to my colleagues and friends at UBC for many interesting discussions: Bob Woodham, Dinesh Pai, Jesse Hoey, Don Murray, Adrian Secord, Arthur Louie, Alex Stevenson, Chris Majewski, Mark McCann, Mike Cline, Doug James, Paul Kry, Dave Burke, Ben Forsyth, Peter Carbonetto, Jacek Kisynski, Hendrik Kueck and Matthew Trentacoste. I also had the opportunity to work with a great many imaginative and inspirational people during internships at Microsoft Research. It was a pleasure to work with Andrew Blake, Phil Torr and Patrick Perez at Cambridge, and with Richard Szeliski, Matt Uyttendaele and Simon Winder in Redmond.

For extra-curricular distractions, I am deeply indebted to Steph Durocher and Reid Holmes, with whom I have spent the best of times. I would also like to thank Zosia Bornik for her patient encouragement, support and many Friday night dinners.

Finally, thanks to Mum and Dad, for encouraging me in the pursuit of my dreams, and providing a loving family in which to do so.

Chapter 1

Introduction

1.1 Motivation

The ability to deduce information about the world from multiple views is a fundamental capability of both human and machine vision systems. Such systems typically employ sensors (eyes or cameras) that lose information in the projection from 3D to 2D, and rely on a processing unit (the brain or computer) that combines information from multiple views to create the impression of a visual world.

An example of this in humans is foveal vision [Ray98]. Most of us are not immediately aware that the high-resolution portion of the retina (the fovea) has an angular range of only about 2 degrees – approximately the size of a thumb held at arms length. Yet, we perceive a much larger field of view of up to 135 $\times$ 200 degrees. This is enabled by rapid eye movements called saccades, during which the eyes move with angular velocities up to 500 degrees per second. In between the saccades are fixations, where we focus our foveal vision on a point. Though a typical task such as reading would involve fixations of 200-300 ms between saccades of 30 ms or so, the brain is able to assimilate all this information and create the illusion of a single, immersive, high-resolution scene. Another basic capability of human vision is the ability to perceive depth. Since the eyes are separated in space, each receives a slightly different image, and the difference in position (disparity) of corresponding points in these images can be used to judge depth[1].

Both of these processes have been mimicked in machine vision systems where they are known as panoramic [Mil75, BK01] and stereo [MP79, SS02] vision. The basic ideas can also be extended to multiple views and were first implemented in computers by the photogrammetry community for the purposes of aerial cartography [Bro58, Sla80]. A distinction is made between problems where the imagery is essentially 2-dimensional, and may be combined into a larger composite image (a process known as image align-

[1]This was first recognised by Euclid in 280 A.D.

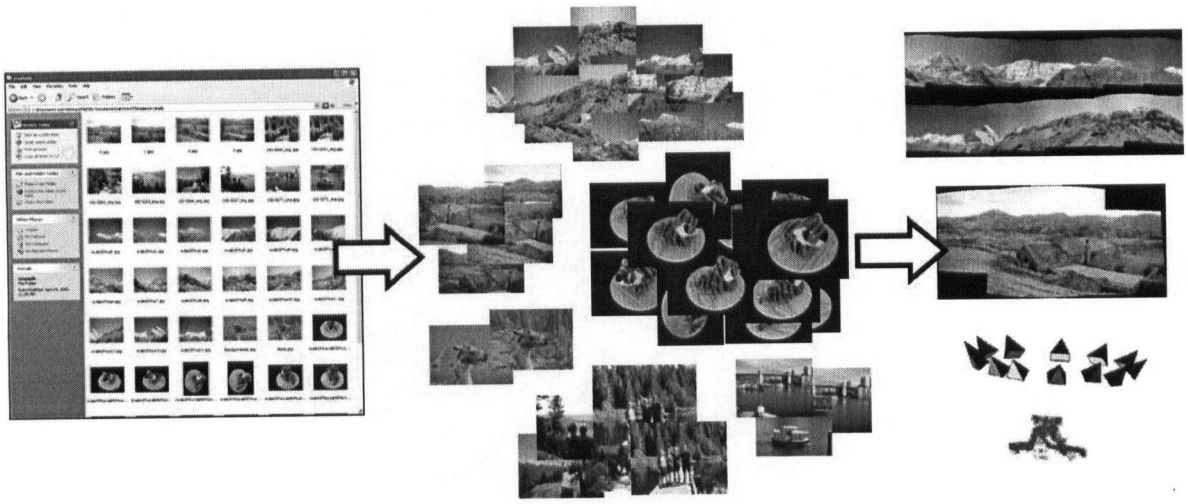


Figure 1.1: The multi-image matching framework. The objective is to operate on an unordered database of images, and find all the matching images. Subsets of matching images can be combined into 3D models or panoramic views as appropriate.

ment or stitching [Sze04]), and cases when the imagery is truly 3-dimensional. The latter case is distinguished by the fact that the images exhibit parallax (depth dependent motion), which can be used to deduce the camera motion and structure of the scene (known as structure and motion estimation [HZ04]).

Both problems also have many compelling applications. Image stitching can be used to create beautiful panoramic mosaics, which are often viewed interactively for applications such as virtual tourism, or to provide backdrops in films and video games. Camera tracking and 3D structure estimation are used extensively in the visual effects industry, for video shot stabilisation, and for modelling and visualisation in areas as diverse as archaeological digs and crime scenes.

Computer vision has brought increasing automation to such problems, with several commercial offerings [Che95, REA, 2D3] in addition to an extensive research literature [Sze04, HZ04, BTZ96, Pol00] devoted to automatic image stitching and 3D modelling from multiple views. Despite much recent progress, state-the-art systems for image registration [REA] and camera tracking [2D3] still require assumptions about the image sequence or user input to define matching images. The main theme of thesis is that such models can be discovered automatically in a database of images using the structure of the data only. No prior information is required about the camera parameters or image sequence other than mild assumptions about the nature of projection.

1.2 Approach

A basic task in vision is the correspondence problem. This is the task of finding points in different images that are corresponding in the sense that they are projections of the same point in the world. While humans perform this activity without conscious thought, it has proven to be a difficult problem in computer vision. Several approaches to image correspondence have been proposed. Some are iterative, for example tracking of object contours [BI98], and some are non-iterative, such as indexing using invariants [MZ92]. Another distinction is between techniques that are direct [IA99] (using all image information) and feature-based [TZ99] (using a sparse representation of the image). In this work we adopt a non-iterative, feature based approach to image matching.

It is well known that the first stages in the visual cortex involve the detection of low-level features such as edges [HW62]. However, more recently Tanaka [Tan97] and others have shown that certain cells in the inferotemporal cortex respond only to more complex (and irreducible) features. Feature detection in computer vision has followed a parallel path, with early work in edge [Can86] and corner [Har92] detection being augmented with more complex and distinctive image features [Low99, MCUP02, Lin98, KZB04]. Our choice of image features follows this trend, and will be discussed in more detail in chapter 2.

Our approach to image matching also differs from more traditional methods in that we see matching as an operation on *multiple images* and not just an image pair. The problem can be stated as follows:

Given an unordered set of images
and a geometric matching constraint,
find all subsets of matching images

We will mainly be interested in the cases of a) stationary but rotating cameras, and b) and moving cameras viewing rigid scenes, for which it is possible to a) stitch panoramas and b) generate 3D models (see figure 1.1). There are several advantages of multi-image matching over the traditional pairwise approach:

Complexity. By representing a collection of images as a database of features, we can pose the image matching problem as an all nearest neighbours problem [GM00], which can be solved in $O(n \log n)$ time (*cf.* $O(n^2)$ for naive pairwise matching).

Geometry. Multi-view constraints are stronger than pairwise constraints. This allows for more accurate solution of the image geometry, and more incorrect matches to be rejected.

Probability. Using a large database of images provides a background distribution of incorrect feature matches, which can be used to help verify correct matches. Furthermore, any feature matching errors that do occur are distributed across all of the images, and not just the pair being matched.

Another key advantage is the ability to automatically recognise consistent objects or scenes in an image database.

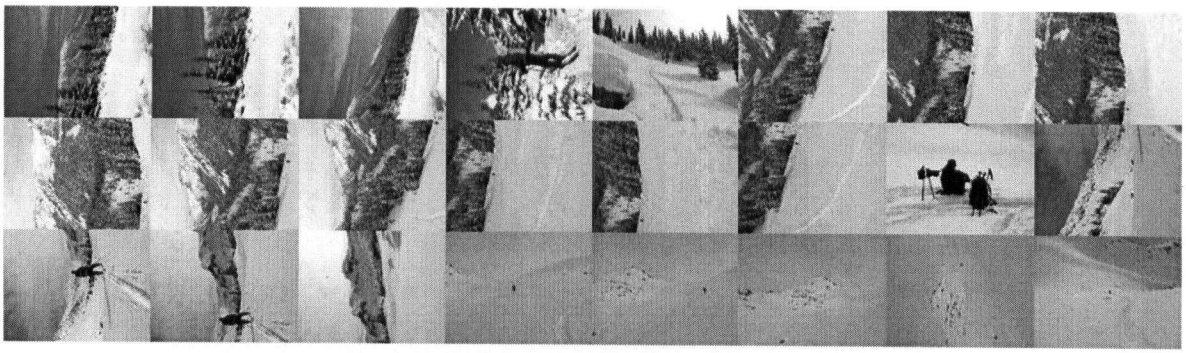
An algorithmic overview of our approach is shown in algorithm 1. We begin by extracting invariant features from all of the input images (step I). The features we have used are Scale Invariant Feature Transform (SIFT) features [Low99] and Multi-Scale Oriented Patches (MOPS) (chapter 2). The goal of this stage is to generate a set of descriptors for locations in each image that are (as far as possible) *invariant* to the imaging process. Each feature is then matched to the features from all other images using an efficient indexing technique. We have used k-d trees [BL97] and wavelet indexing (section 2.8) for this purpose. Step III consists of outlier rejection by robust estimation of multi-view geometric constraints. For this we have used pairwise constraints based on the panoramic and epipolar geometry [HZ04]. Finally we use bundle adjustment to compute optimal estimates of the camera parameters and reject further outliers [TMHF99].

1.3 Contributions

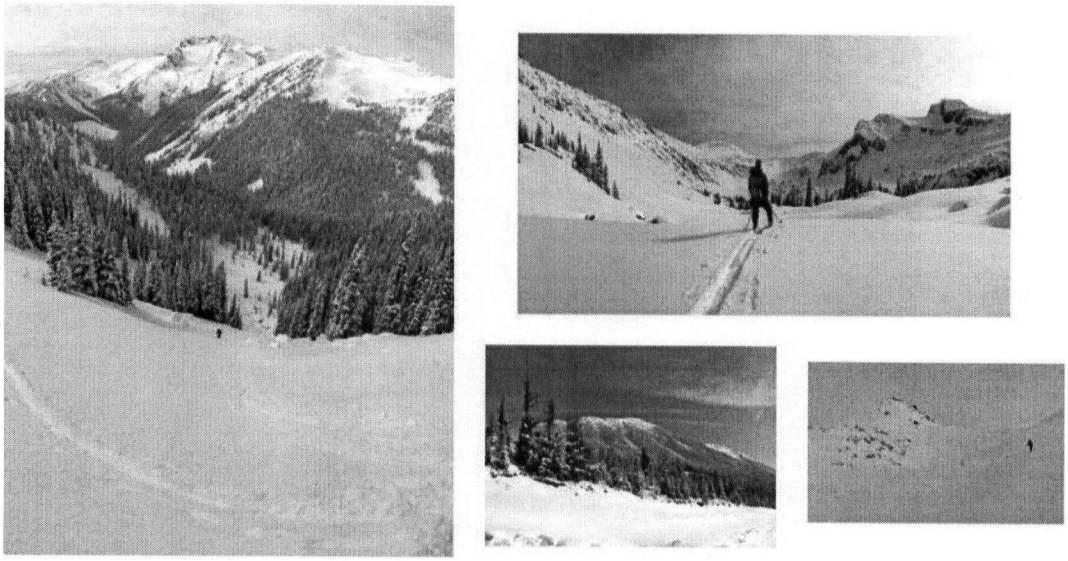
The main contribution of this thesis is the development and evaluation of a system capable of recognising and stitching 2D panoramas without any user input. We also develop a new class of invariant features (MOPS) designed specially for this purpose. Finally, we show how the multi-image matching framework can be applied to 3D object recognition and reconstruction.

Automatic Panoramic Image Stitching

We develop a novel, fully automated 2D panorama stitcher. This has the following advantages over previous image stitching approaches



(a) Input images



(b) Output panoramas

Figure 1.2: Panorama recognition: (a) an image set containing multiple panoramas and distractor images is input, and (b) panoramic sequences are recognised and rendered as output.

Algorithm: Multi-Image Matching using Invariant Features

Input: n unordered images

I. Extract invariant features from all n images

II. Find k nearest-neighbours for each feature using an efficient indexing scheme

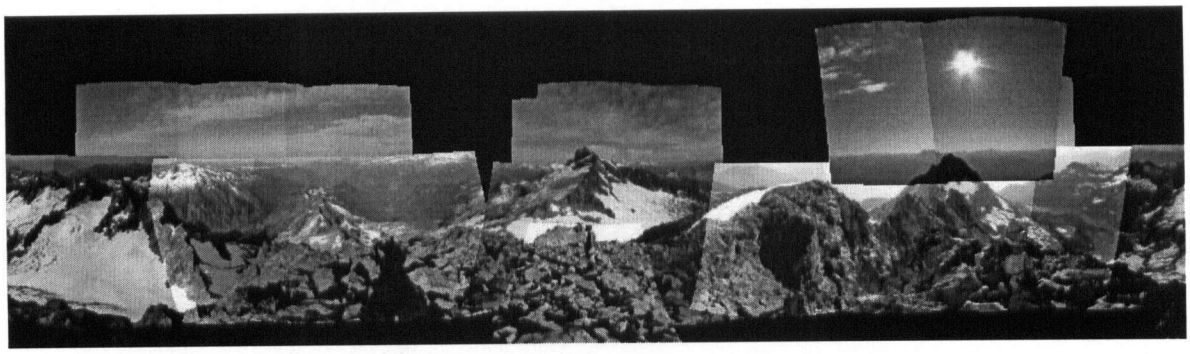
III. For each image:
   (i) Select m candidate matching images
   (ii) Find geometrically consistent feature matches by robustly estimating the pairwise image geometry
   (iii) Verify image matches using a probabilistic model

IV. Find connected components of image matches

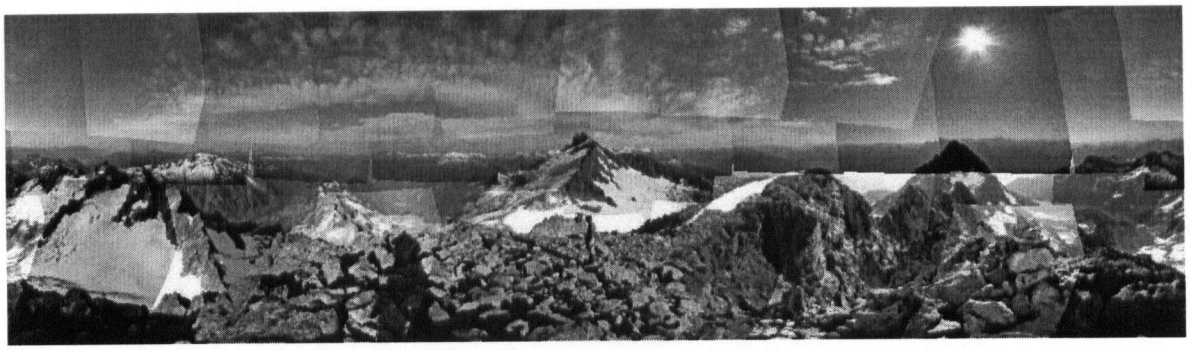
V. For each connected component:
   (i) Perform bundle adjustment to estimate the global geometry and reject further outliers.
   (ii) Compute 3D models/render panoramas as appropriate

Output: Matched images and 3D model(s)/panorama(s)

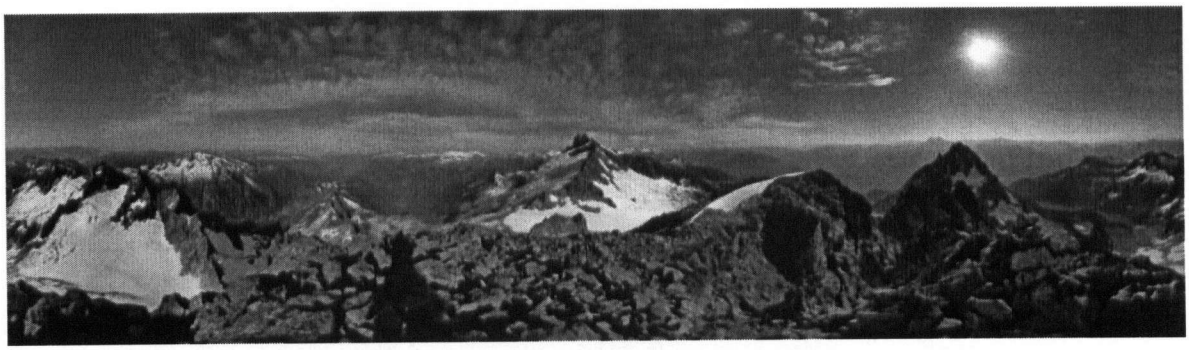
Algorithm 1: Multi-image matching



(a) 25 of 57 images registered



(b) All 57 images registered



(c) Panorama rendered with multi-band blending

Figure 1.3: Fully automatic 2D image stitching. All 57 images are registered automatically without any user input, and stitched into a seamless panoramic image.

- Robustness to image zoom, rotation and exposure change, due to the use of invariant features.
- $O(n \log n)$ running time, due to tree based matching (*cf.* $O(n^2)$ for naive pairwise matching).
- Automatic detection of matching images, using a probabilistic model for image match verification. This also allows us to *recognise panoramas* [BL03] (figure 1.2) in unordered datasets.
- High quality rendering, even in the presence of misregistration, motion and exposure differences, due to gain compensation and multi-band blending (figure 1.3).
- Automatic panorama straightening, using a heuristic based on the way users typically shoot panoramic images.

Evaluation of Image Stitching Algorithms

We introduce a framework for evaluation of multi-image registration techniques. This uses a novel error function based on the projection errors of a test alignment compared to ground truth.

Multi-Scale Oriented Patches (MOPS)

- We show that a direct patch based sampling of an oriented image patch can serve as a useful invariant feature descriptor.
- We propose an novel adaptive non-maximal suppression algorithm that distributes features more evenly over the image than previous approaches.
- We show that data driven classifiers that make use of known incorrect matches can provide superior performance to the basic Gaussian noise model.
- We show that indexing based on Haar wavelet coefficients speeds up the search for matching features, and is superior to indexing on other dimensions of the feature descriptor.

3D Object Recognition and Reconstruction

We show how the previous techniques can be extended to the case of a moving camera and enable recognition and reconstruction of 3D objects from unordered datasets. We present a graphical model formulation for the reconstruction problem, and implement an efficient solution using sparse bundle adjustment.

1.4 Outline of Thesis

The remainder of the thesis is organised as follows. Chapter 2 reviews the literature on invariant features and introduces Multi-Scale Oriented Patches (MOPS) – a new class of invariant feature that use a direct patch based sampling of the local image region. We perform a detailed analysis of the properties of these image features and include comparisons to the current state-of-the-art in feature matching. In chapter 3 we describe the design of an automatic panoramic image stitcher, capable of recognising and stitching panoramas from unordered datasets. We develop methods for evaluation and tuning the parameters of our automatic panorama stitcher in chapter 4. Chapter 5 extends the multi-view matching framework to the case of moving cameras and 3D model acquisition. In chapter 6 we present conclusions and ideas for future work.

Chapter 2

Multi-Scale Oriented Patches

2.1 Introduction

A quantity is *invariant* under a group of transformations if it is conserved (unchanged) by any transformation in that group. In the context of computer vision, the transformations of interest are those induced by the perspective projection of the world onto the image plane (see appendix A). Invariance provides a mechanism for finding correspondences between images for which the transformation parameters are unknown [MZ92, RZFM92].

An alternative approach to correspondence is to start with some initial guess of the transformation parameters and iteratively refine this whilst minimising an error function based on the quality of registration. Such error functions typically use all of the image data, for example, the sum squared error of overlapping pixels, and these are called *direct* methods [Ana89, LK81]. In contrast, *feature-based* methods attempt to extract salient features such as edges and corners, and then to use a small amount of local information, for example, correlation of a small image patch, to establish matches [Har92, ST94].

Invariant features can be seen as a hybrid of earlier matching methods. As with traditional image features, they use a distributed representation which makes matching methods robust (in that failed feature matches do not adversely affect the solution). In common with direct methods, each feature descriptor typically uses a large amount of local image data, making the features more distinctive than traditional image features such as edges and corners. Finally, the use of invariants makes it possible to find matches over a wide range of image transformations using indexing instead of iterative search.

The first work in the area was by Schmid and Mohr [SM97] who used a jet of Gaussian derivatives to form a rotationally invariant descriptor around a Harris corner. Lowe extended this approach to incorporate scale invariance [Low99, Low04]. Other

researchers have developed features that are invariant under affine transformations [Bau00, TG00, BL02]. Interest point detectors vary from standard feature detectors such as Harris corners or DOG maxima to more elaborate methods such as maximally stable regions [MCUP02] and stable local phase structures [CJ03].

Generally, interest point extraction and descriptor matching are considered as two basic steps, and there has been some progress in evaluating the various techniques with respect to interest point repeatability [SMB98] and descriptor performance [MS03]. Other researchers have suggested that interest points should be located such that the solutions for matching position [ST94], orientation and scale [Tri04] are stable. There have been several compelling applications of invariant feature based matching in the context of object recognition [Low99], structure from motion [SZ02], panoramic imaging [BL03] and searching for objects in videos [SZ03].

In this chapter, we describe the implementation of a patch-based invariant feature called Multi-Scale Oriented Patches (MOPS). MOPS have been designed with the task of panoramic image stitching in mind. They have a number of advantages over previous approaches in that regard. First, we use a novel adaptive non-maximal suppression algorithm that better distributes features across the image than previous techniques (section 2.3.1). This facilitates improved matching for panoramic sequences with low overlap. Second, we develop a feature space outlier rejection strategy that uses all of the images in an n-image matching problem to give a background distribution for incorrect matches (section 2.7). Finally, we develop an indexing scheme based on low-frequency Haar wavelet coefficients that greatly speeds up the search for feature correspondences with minimal impact on matching performance (section 2.8). We close the chapter with a discussion of our results and ideas for future work in this area.

2.2 Multi-Scale Oriented Patches

In general the transformation between corresponding regions in a pair of images is a complex function of the geometric and photometric properties of the scene and the cameras. For the purposes of this work we reduce this to a simple 6 parameter model

$$I'(\mathbf{x}') = \alpha I(\mathbf{x}) + \beta + n(\mathbf{x}) \tag{2.1}$$

and

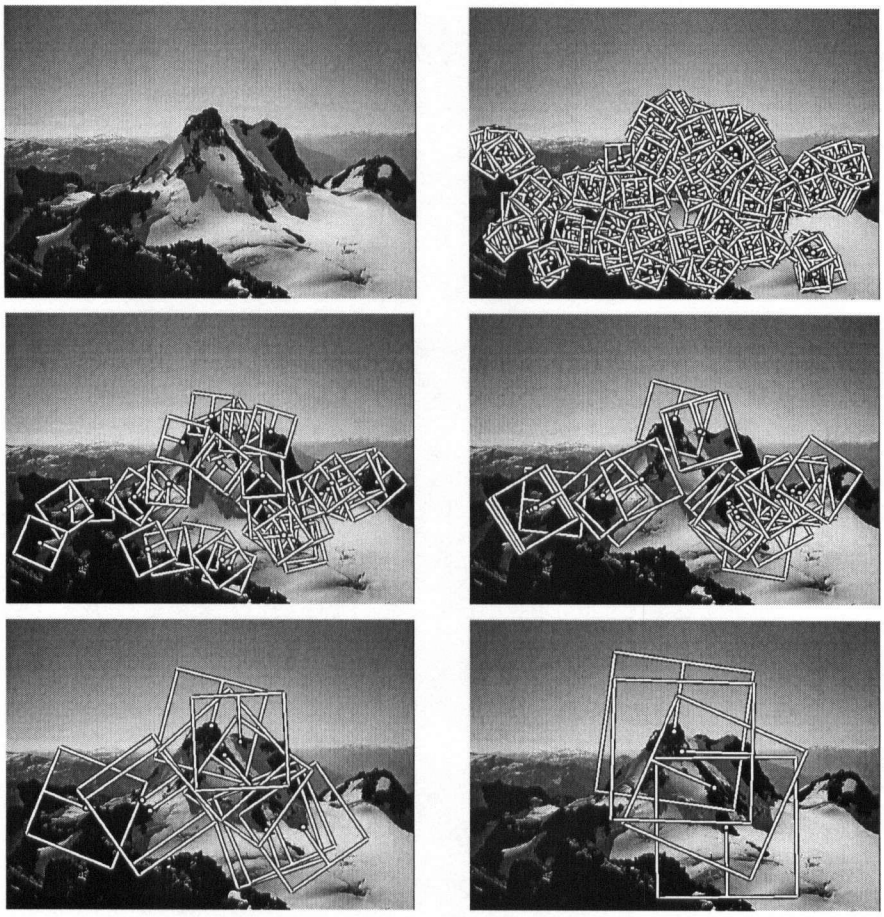


Figure 2.1: Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{t} \tag{2.2}$$

$$\mathbf{A} = s \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{2.3}$$

where $I(\mathbf{x})$ and $I'(\mathbf{x}')$ are the corresponding image patches. There are four geometric parameters t_1, t_2, θ, s (position, orientation and scale) and two photometric parameters are α, β (gain and bias). The error $n(\mathbf{x})$ represents imaging noise and modelling error. Features are located at points where this transformation is well defined i.e. the autocorrelation of $I(\mathbf{x})$ is peaked [ST94]. To compare features, one could in principle compute the maximum likelihood estimates for the transformation parameters between a pair of image locations. Assuming Gaussian noise, this can be done iteratively by solving a non-linear least squares problem [BM04]. However, this would require an iterative registration step to match any pair of features. Instead, we establish a canonical frame (see appendix A.4) for each feature, and sample invariant descriptor vectors relative to that frame. This allows us to efficiently compute an approximation to the minimum matching error $n(\mathbf{x})$ between any pair of features using indexing techniques (see section 2.7.1). We then use the statistics of the matching error $n(\mathbf{x})$ to verify whether a match is correct or incorrect.

2.3 Interest Points

The interest points we use are multi-scale Harris [Har92] corners. For efficiency, we work with greyscale images $I(x, y)$. For each input image $I(x, y)$, we form an image pyramid with the lowest level $P_0(x, y) = I(x, y)$ and higher levels related by smoothing and subsampling operations

$$P'_l(x, y) = P_l(x, y) * g_{\sigma_p}(x, y) \tag{2.4}$$

$$P_{l+1}(x, y) = P'_l(sx, sy) \tag{2.5}$$

l denotes the pyramid level, and $g_\sigma(x, y)$ denotes a Gaussian kernel of standard deviation σ. We use a subsampling rate $r = 2$ and pyramid smoothing $\sigma_p = 1.0$. Interest

points are extracted from each level of the pyramid. Other authors use sub-octave pyramids, for example Lowe [Low04]. This gives improved matching for images at different scales. Since we are mostly concerned with matching images that have the same scale, this is left for future work.

The Harris matrix at level l and position (x, y) is the smoothed outer product of the gradients

$$\mathbf{H}_l(x,y) = \nabla_{\sigma_d} P_l(x,y) \nabla_{\sigma_d} P_l(x,y)^T * g_{\sigma_i}(x,y) \tag{2.6}$$

∇_σ represents the spatial derivative at scale σ i.e.

$$\nabla_\sigma f(x,y) \triangleq \nabla f(x,y) * g_\sigma(x,y) \tag{2.7}$$

We set the integration scale $\sigma_i = 1.5$ and the derivative scale $\sigma_d = 1.0$ and use the corner detection function

$$f_{HM}(x,y) = \frac{\det \mathbf{H}_l(x,y)}{\operatorname{tr} \mathbf{H}_l(x,y)} = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2} \tag{2.8}$$

which is the harmonic mean of the eigenvalues (λ_1, λ_2) of $\mathbf{H}$. Interest points are located where the corner strength $f_{HM}(x,y)$ is a local maximum of a 3×3 neighbourhood, and above a threshold $t = 10.0$.

The reason for this choice of interest point detection function can be understood in terms of the relationship between $\mathbf{H}$ and the local autocorrelation function. For an image $I(\mathbf{x})$, the first order Taylor expansion gives an expression for the local autocorrelation

$$e(\mathbf{x}) = |I(\mathbf{x}) - I_0|^2 \approx \mathbf{x}^T \frac{\partial I}{\partial \mathbf{x}} \frac{\partial I}{\partial \mathbf{x}}^T \mathbf{x} = \mathbf{x}^T \mathbf{H} \mathbf{x} \tag{2.9}$$

Interest points are located at peaks in the autocorrelation function. This means that $e(\mathbf{u})$ is large for all unit vectors $\mathbf{u}$, which is equivalent to requiring that both eigenvalues of $\mathbf{H}$ are large[1]. Figure 2.2 compares isocontours of our interest point detection function (Harmonic mean) with the common Harris [Har92] and Shi-Tomasi [ST94] detectors. Note that all the detectors require both eigenvalues to be large. Harmonic mean and Shi-Tomasi detectors have the slight advantage that they are parameter free, whereas

[1]Note that in practice $\mathbf{H}$ is integrated over a range as in equation 2.6 (otherwise it would be rank 1).

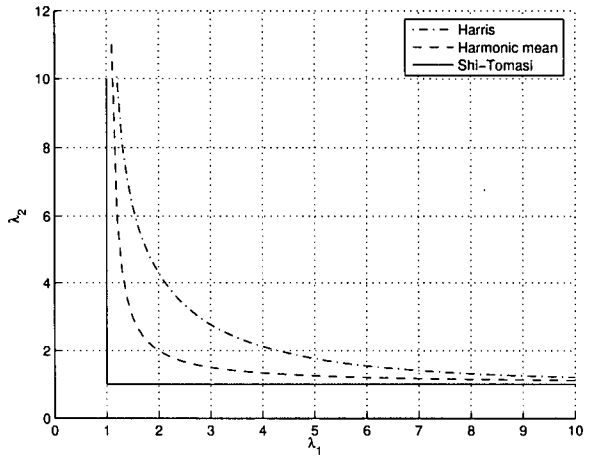


Figure 2.2: Isocontours of popular interest point detection functions. Each detector looks for points where the eigenvalues λ_1, λ_2 of $\mathbf{H} = \int_{\mathcal{R}} \nabla I \nabla I^T d\mathbf{x}$ are both large.

the Harris detector has a single parameter to be tuned.

$$\begin{array}{rrcccc} \text{Harris} & f_H & = & \lambda_1\lambda_2 - 0.04(\lambda_1+\lambda_2)^2 & = & \det \mathbf{H} - 0.04(\text{tr } \mathbf{H})^2 \\ \text{Harmonic mean} & f_{HM} & = & \frac{\lambda_1\lambda_2}{\lambda_1+\lambda_2} & = & \frac{\det \mathbf{H}}{\text{tr } \mathbf{H}} \\ \text{Shi-Tomasi} & f_{ST} & = & \min(\lambda_1, \lambda_2) & & \end{array}$$

Preliminary experiments suggest each of these detectors give roughly the same performance, although one could compute repeatability statistics to confirm this (see section 2.3.3).

2.3.1 Adaptive Non-Maximal Suppression

Since the computational cost of matching is superlinear in the number of interest points, it is desirable to restrict the maximum number of interest points that are extracted from each image. At the same time it is important that the interest points that are generated are well spatially distributed over the image, since the area of overlap between a pair of images may be small. To satisfy these requirements, we use an adaptive non-maximal suppression (ANMS) strategy to select a fixed number of interest points from each image.

Interest points are suppressed based on the corner strength f_{HM} and only those that are a maximum in a neighbourhood of radius r pixels are retained. Conceptually, we initialise the suppression radius $r = 0$ and then increase it, removing interest points by non-maximal suppression, until the desired number of interest points n_{ip} is obtained. In practice, we can perform this operation without search as the set of interest points which are generated in this way form an ordered list.

The first entry in the list is the global maximum, which is not suppressed at any radius (however large). As the suppression radius decreases from infinity, interest points are added to the list. However, once an interest point appears, it will always remain in the list. This is true because if an interest point is a maximum in radius r then it is also a maximum in radius $r' < r$. In practice we robustify the non-maximal suppression by requiring that a neighbour has a sufficiently larger strength. Thus the minimum suppression radius r_i is given by

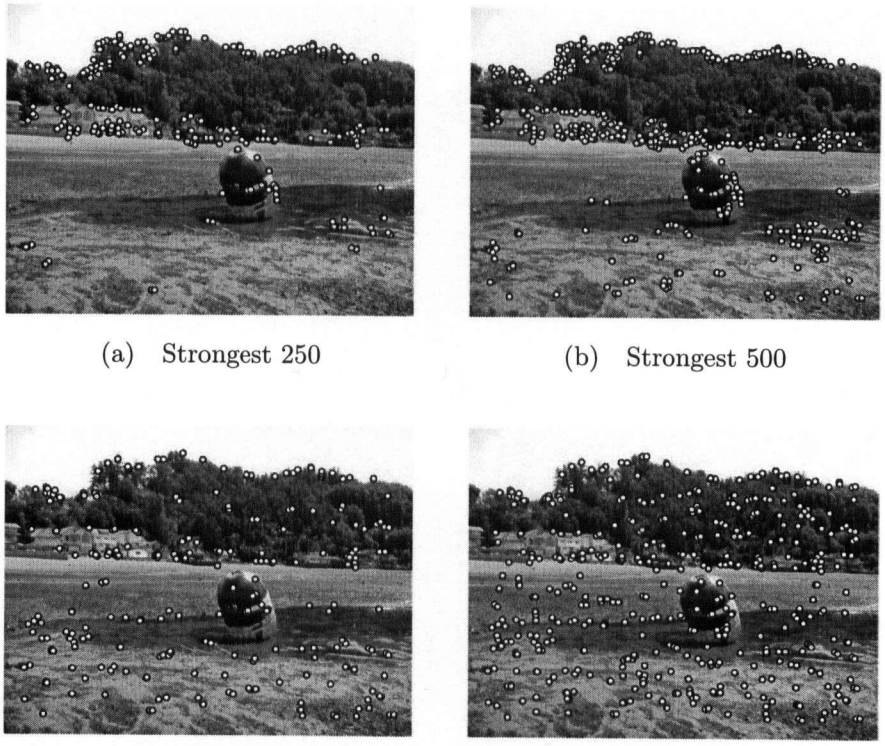
$$r_i = \min_j |\mathbf{x}_i - \mathbf{x}_j|, \text{ s.t. } f(\mathbf{x}_i) < cf(\mathbf{x}_j), \ \mathbf{x}_j \epsilon \mathcal{I} \tag{2.10}$$

where $\mathbf{x}_i$ is a 2D interest point image location, and $\mathcal{I}$ is the set of all interest point locations. We use a value $c = 0.9$, which ensures that a neighbour must have significantly higher strength for suppression to take place. We select the $n_{ip} = 500$ interest points with the largest values of r_i. Experiments on a large database of panoramic images suggest that distributing interest points spatially in this way, as opposed to selecting based on max corner strength, results in fewer dropped image matches (we found improved matching on 5 sequences from our 200 sequence dataset). Another interesting experiment would be to test if ANMS also improves registration *accuracy* for a fixed number of features. This test could be performed using the apparatus developed in chapter 4.

2.3.2 Sub-Pixel Accuracy

Interest points are located to sub-pixel accuracy by fitting a 2D quadratic to the corner strength function in a local 3×3 neighbourhood (at the detection scale) and finding its maximum.

$$f(\mathbf{x}) = f + \frac{\partial f}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 f}{\partial \mathbf{x}^2}\mathbf{x} \tag{2.11}$$



(a) Strongest 250 (b) Strongest 500

(c) ANMS 250, $r = 24$ (d) ANMS 500, $r = 16$

Figure 2.3: Adaptive non-maximal suppression (ANMS). The two upper images show interest points with the highest corner strength, while the lower two images show interest points selected with adaptive non-maximal suppression (along with the corresponding suppression radius r). Note how the latter features have a much more uniform spatial distribution across the image.

$f_{-1,1}$	$f_{0,1}$	$f_{1,1}$
$f_{-1,0}$	$f_{0,0}$	$f_{1,0}$
$f_{-1,-1}$	$f_{0,-1}$	$f_{1,-1}$

Figure 2.4: For subpixel accuracy, derivatives are computed from pixel difference in a 3×3 neighbourhood according to the equation 2.17.

where $\mathbf{x}$ denotes position (x, y), and $f(\mathbf{x}) = f_{HM}(\mathbf{x})$ is the corner strength measure. Derivatives are computed from the 3×3 neighbourhood using pixel differences i.e.

$$\frac{\partial f}{\partial x} = (f_{1,0} - f_{-1,0})/2 \tag{2.12}$$

$$\frac{\partial f}{\partial y} = (f_{0,1} - f_{0,-1})/2 \tag{2.13}$$

$$\frac{\partial^2 f}{\partial x^2} = f_{1,0} - 2f_{0,0} + f_{-1,0} \tag{2.14}$$

$$\frac{\partial^2 f}{\partial y^2} = f_{0,1} - 2f_{0,0} + f_{0,-1} \tag{2.15}$$

$$\frac{\partial^2 f}{\partial x \partial y} = (f_{-1,-1} - f_{-1,1} - f_{1,-1} - f_{1,1})/4 \tag{2.16}$$

See figure 2.4. The subpixel location is given by

$$\mathbf{x}_m = \mathbf{x}_0 - \frac{\partial^2 f}{\partial \mathbf{x}^2}^{-1} \frac{\partial f}{\partial \mathbf{x}} \tag{2.17}$$

2.3.3 Repeatability

The fraction of interest points whose transformed position is correct[2] up to some tolerance epsilon is known as repeatability [SMB98]. We use a slightly different definition

[2]Here we assume that images are related by a homography (see appendix A.2.1). Hence interest point detections in a pair of images are 'correct' if they are consistent with the homography between that pair of images. Later chapters will discuss alternate motion models between images.

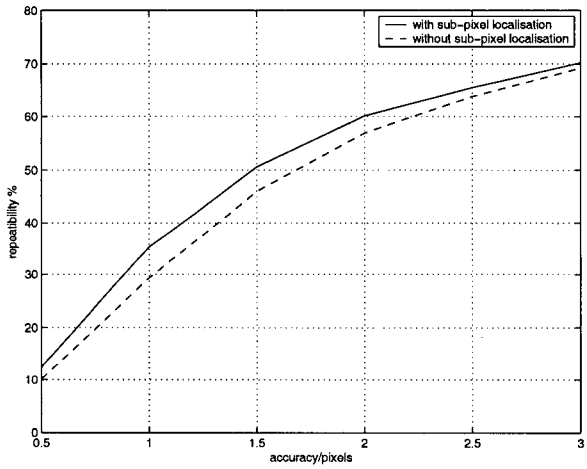


Figure 2.5: Repeatability of interest points with and without sub-pixel correction. These results were computed from the Matier dataset.

of repeatability to that defined in [SMB98] (which is not symmetric) as follows. Let I_M denote the set of all points belonging to image M, and $\mathcal{I}_M$ denote the set of interest points in image M. The set of points from image M that project to image N is given by $\mathcal{P}_{MN}$

$$\mathcal{P}_{MN} = \{\mathbf{x}_i : \mathbf{H}_{NM}\mathbf{x}_i \, \epsilon \, I_N\} \tag{2.18}$$

where $\mathbf{H}_{NM}$ is the homography between images M and N. The set of points from image M that are repeated in image N (within tolerance ϵ) is given by $\mathcal{R}_{MN}(\epsilon)$

$$\mathcal{R}_{MN}(\epsilon) = \{\mathbf{x}_i : \exists j : |\mathbf{x}_i - \mathbf{H}_{MN}\mathbf{x}_j| < \epsilon, \; \mathbf{x}_i \, \epsilon \, \mathcal{I}_M, \; \mathbf{x}_j \, \epsilon \, \mathcal{I}_N\} \tag{2.19}$$

The repeatability is the number of interest points that *are* repeated as a fraction of the total number of interest points that *could* be repeated. It is useful to adopt a symmetrical definition

$$r(\epsilon) = \min\left(\frac{|\mathcal{R}_{MN}|}{|\mathcal{P}_{MN}|}, \frac{|\mathcal{R}_{NM}|}{|\mathcal{P}_{NM}|}\right) \tag{2.20}$$

The repeatability of our interest points with and without sub pixel localisation is shown in figure 2.5. Note that sub-pixel localisation gives approximately 5% improvement in repeatability.

2.4 Orientation

Each interest point has an orientation θ, where the orientation vector $[\cos\theta, \sin\theta] = \mathbf{u}/|\mathbf{u}|$ comes from the smoothed local gradient

$$\mathbf{u}_l(x, y) = \nabla_{\sigma_o} P_l(x, y) \tag{2.21}$$

Note that the image gradient is *covariant* under similarity transforms, and hence can be used to compute a canonical frame (see appendix A.4). The integration scale for orientation is $\sigma_o = 4.5$. A large derivative scale is desirable so that the vector field $\mathbf{u}_l(x, y)$ varies smoothly across the image, making orientation estimation robust to errors in interest point location. The orientation estimate is poorly conditioned if the first derivative is close to zero, in which case it may be favourable to look at higher order derivatives [SF95]. This is left for future work.

2.5 Analysis of Interest Point Extraction

Figure 2.6 compares the errors introduced in four stages of feature extraction: position, scale and orientation measurement, and descriptor matching[3]. These experiments were conducted using the Matier dataset (see appendix B). Features were extracted and matched between all 7 images, and the top 2 image matches for each image were selected. The maximum number of matches per feature was 5. For each of these (14) image matches, the number of features in the area of overlap was found, and the number of features with consistent position, scale and orientation measurements computed. Consistent position means that the interest point was detected within ϵ pixels of the projected position using the homographies computed from bundle adjustment. Consistent scale means that the interest point was detected at the same scale in the two images. Consistent orientation means that the transformed orientations differ by less than 3 standard deviations (= 3 $\times$ 18.5 degrees). To an accuracy of 3 pixels, 72% of interest points are repeated (have correct position), 66% have the correct position and scale, 64% also have correct orientation, and in total 59% of interest points are correctly matched (meaning they are one of the top 5 matches in terms of Euclidean distance in feature space). That is, given that an interest point overlaps another image,

[3]Note that the extraction and matching of descriptor vectors is discussed in sections 2.6 and 2.7. The results are included here for completeness.

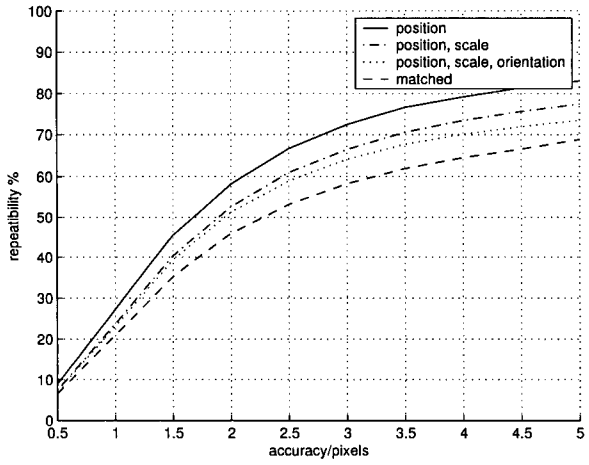


Figure 2.6: Repeatability vs accuracy for Multi-Scale Oriented Patches. To an accuracy of 3 pixels, 72% of interest points in the overlap region have consistent position, 66% have correct position and scale, 64% also have correct orientation, and in total 59% of interest points in the overlap region are correctly matched.

the probability that it will be correctly matched is 59%.

Whilst figure 2.6 shows combined results for all levels, figure 2.7 shows separate results for interest points extracted at each level of the pyramid. All measurements are relative to the base image. Note that contrary to the popular perception that Harris corners are sub-pixel accurate, the majority of interest points have location errors in the 0-3 pixel range, even at the finest scale of detection. Also note that interest points at higher levels of the pyramid are less accurately localised relative to the base image than those at a lower level, due to the larger sample spacing. Although less useful for accurate localisation, these higher level features are still useful in verifying an image match or a coarse RANSAC hypothesis. Also, the orientation estimate improves as the level increases. As expected, features at levels 4 and 5 generally have poor accuracy, and their distributions show many features have accuracy worse than 3 pixels. However, it is slightly counter intuitive that features at levels 2 and 3 tend to have accuracies of 3 pixels or better.

Figure 2.8 show the same results as computed for the Van Gogh sequence. This is a pure rotation sequence, with no projective distortion. As compared to the Matier sequence, which does have perspective distortion, matching is improved. Note in particular that the orientation repeatability curves and the matched curves are very close, indicating that if feature orientation is correctly estimated, then it is very likely that

the feature will also be correctly matched. This is not the case for the Matier dataset due to perspective distortion.

2.6 Feature Descriptor

Once we have determined where to place our interest points, we need to extract a description of the local image structure that will support reliable and efficient matching of features across images. A wide range of such local feature vectors have been developed, including local intensity patches [For86, Har92], Gaussian derivatives [SM97], scale invariant feature transforms [Low04], and affine-invariant descriptors [Bau00, TG00, BL02]. In their comparative survey, Mikolajczyk and Schmid [MS03] evaluated a variety of these descriptors and found that SIFT features generally perform the best. Local patches oriented to the dominant local orientation were also evaluated, but found not to perform as well. In this section, we show how such patches can be made less sensitive to the exact feature location by sampling the pixels at a *lower frequency* than the one at which the interest points are located.

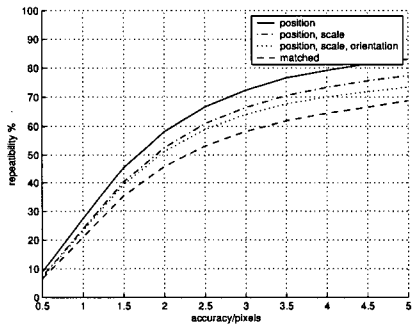
Given an interest point (x, y, l, θ), the descriptor is formed by sampling from a patch centred at (x, y) and oriented at angle θ from pyramid level l. We sample an 8×8 patch of pixels around the sub-pixel location of the interest point, using a spacing of $s = 5$ pixels between samples (figure 2.9). Figure 2.10 shows how varying the sample spacing s affects the reliability of feature matching. We have found that performance increases up to a value $s = 5$, with negligible gains thereafter.

To avoid aliasing, the sampling is performed at a higher pyramid level, such that the sampling rate is approximately once per pixel (the Nyquist frequency). This means sampling the descriptor from a level l_s levels above the detection scale, where

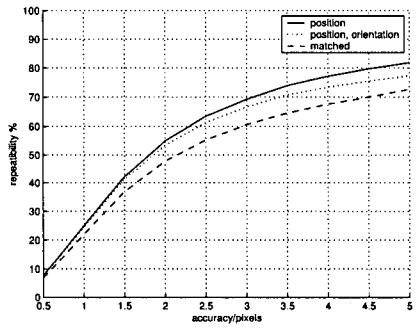
$$l_s = \text{floor}\left(\frac{\log s}{\log 2} + 0.5\right) \tag{2.22}$$

The descriptor vector is sampled using bilinear interpolation. In practice, $s = 5$ so the descriptor vectors are sampled at $l_s = 2$ levels above the detection scale.

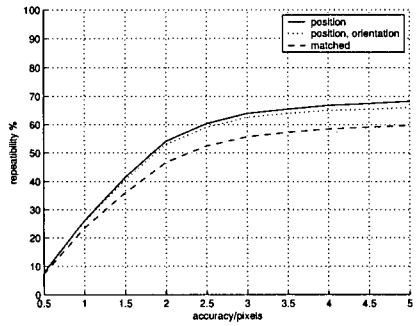
Suppose the interest point was detected at level l. This suggests sampling the descriptor from $P_{l+l_s}(x, y) = P_{l+2}(x, y)$. However, we have found better results by instead sampling the descriptor from $P'_{l+1}(x, y)$, where $P'_{l+1}(x, y) = P_{l+1}(x, y) * g_{\sigma_p}(x, y)$, i.e. blurring but not downsampling. Further (smaller) gains are made by sampling from



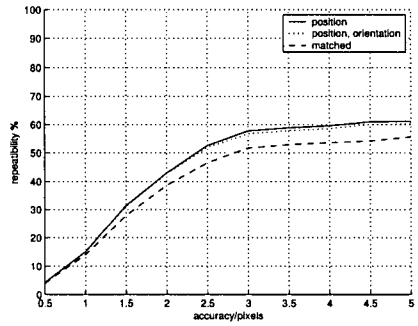
(a) All levels. 6649 features extracted, 6610 correct matches



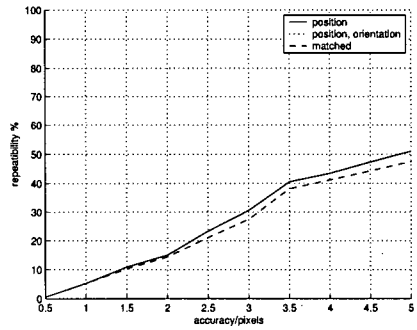
(b) Level 1. 4997 features extracted, 5318 correct matches



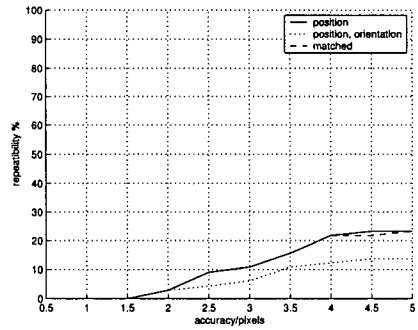
(c) Level 2. 1044 features extracted, 860 correct matches



(d) Level 3. 372 features extracted, 295 correct matches

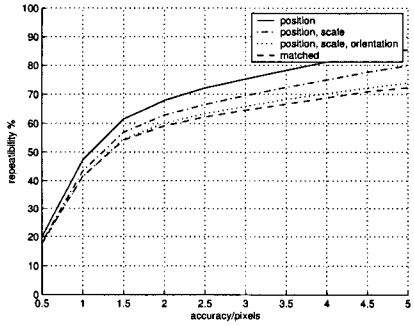


(e) Level 4. 180 features extracted, 120 correct matches

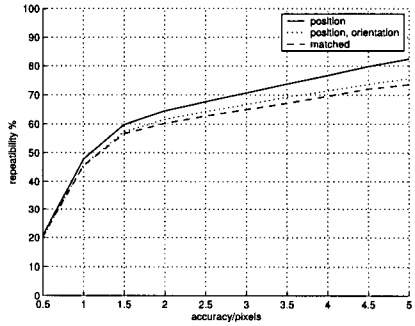


(f) Level 5. 56 features extracted, 17 correct matches

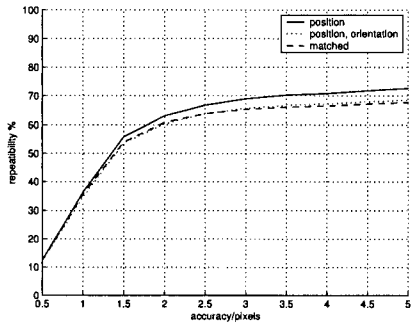
Figure 2.7: Repeatability of interest points, orientation and matching for Multi-Scale Oriented Patches at 5 pyramid levels (Matier dataset). The top left figure is a combined result for all levels. All measurements are relative to the base image in the pyramid.



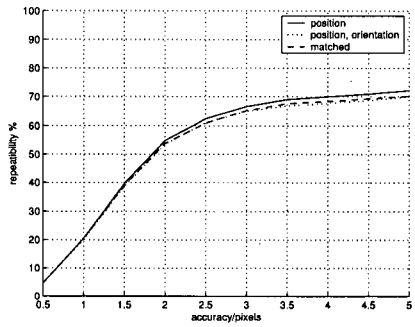
(a) All levels. 6557 features extracted, 9880 correct matches



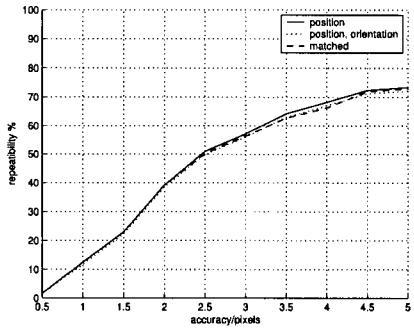
(b) Level 1. 4925 features extracted, 7559 correct matches



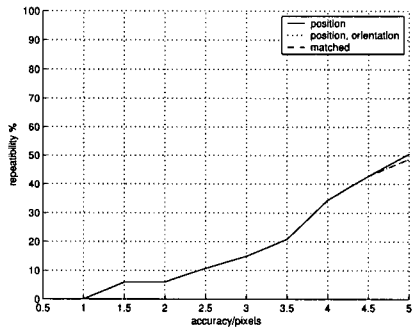
(c) Level 2. 1041 features extracted, 1512 correct matches



(d) Level 3. 392 features extracted, 542 correct matches



(e) Level 4. 158 features extracted, 212 correct matches



(f) Level 5. 41 features extracted, 55 correct matches

Figure 2.8: Repeatability of interest points, orientation and matching for Multi-Scale Oriented Patches at 5 pyramid levels (Van Gogh dataset). This dataset consists of pure rotations with no perspective distortion. The top left figure is a combined result for all levels. All measurements are relative to the base image in the pyramid.

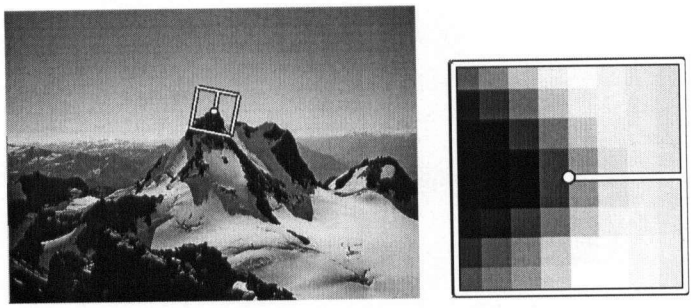


Figure 2.9: Descriptors are formed using an 8×8 sampling of bias/gain normalised intensity values, with a sample spacing of 5 pixels relative to the detection scale. This low frequency sampling gives the features some robustness to interest point location error, and is achieved by sampling at a higher pyramid level than the detection scale.

$P_l''(x,y) = P_l(x,y) * g_{2\times\sigma_p}(x,y)$. Whilst theoretically one can interpolate a function exactly given a sampling of that function at the Nyquist frequency, in practice it is better to maintain a denser sampling if using a bilinear resampling kernel. This is discussed in more detail with quantative results in section 2.9.2.

2.6.1 Illumination Invariance

In the previous sections we have described some of the *geometrical* transformations under which we wish our image features to achieve invariance. However, there is also a potentially complex transformation of the illumination between corresponding image regions, which depends upon the surface reflectance properties and lighting conditions. Digital cameras also perform several non-linear transformations on the image intensities such as white-balancing and gamma correction. In this work we use a simple affine model for illumination change

$$I' = \alpha I + \beta \tag{2.23}$$

Whilst it would be desirable to define a more accurate model of illumination change and represent features using illumination invariants, this is left for future work. In practice, we normalise the descriptor vector so that the mean is 0 and the standard deviation is 1, i.e.

$$d_i = (d_i' - \mu)/\sigma \tag{2.24}$$

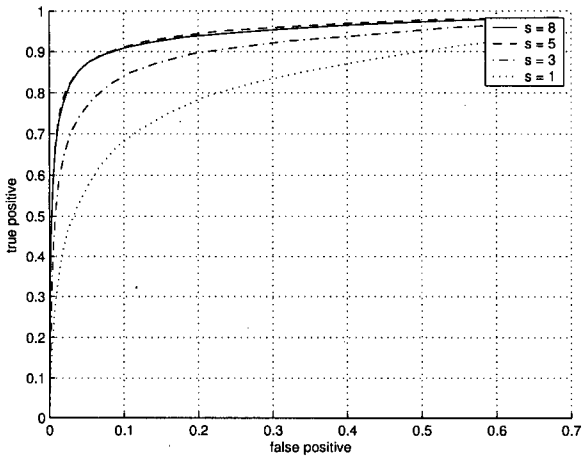


Figure 2.10: Effect of changing the descriptor sample spacing on performance. These ROC curves show the results of thresholding feature matches based on normalised match distance as in section 2.7.1. Performance improves as the sample spacing increases (larger patches), but gains are minimal above a sample spacing of 5 pixels.

where d'_i, $i\epsilon\{1..d^2\}$ are the elements of the descriptor vector, with $\mu = \frac{1}{d^2}\sum_{i=1}^{d^2} d'_i$ and $\sigma = \sqrt{\frac{1}{d^2}\sum_{i=1}^{d^2}(d'_i - \mu)^2}$. This makes the features invariant to affine changes in intensity (bias and gain).

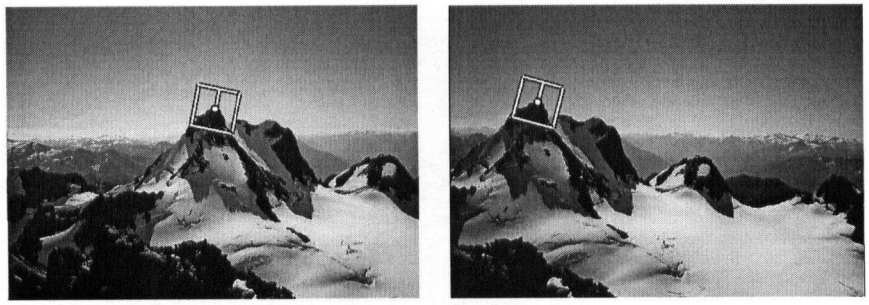
2.6.2 Haar Wavelet Transform

Finally, we perform the Haar wavelet transform on the 8×8 descriptor patch d_i to form a 64 dimensional descriptor vector containing the wavelet coefficients c_i. Due to the orthogonality property of Haar wavelets, distances are preserved

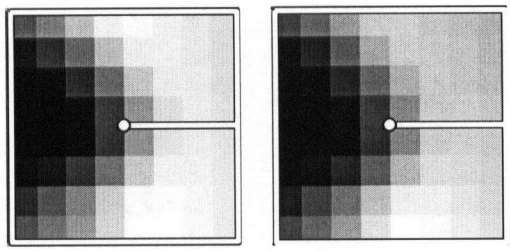
$$\sum_i (d_i^1 - d_i^2)^2 = \sum_i (c_i^1 - c_i^2)^2 \tag{2.25}$$

So nearest neighbours in a sum-squared difference sense are unchanged.

Our motivation for using wavelet coefficients to parameterise descriptors was the intuition that some dimensions of the feature descriptor would be more noisy than others (in fact we show that this is true in section 2.7.2). We exploit this in an indexing strategy which uses the first 3 non-zero wavelet coefficients c_1, c_2, c_3 (see section 2.8). Note that the mean c_0 is equal to 0.

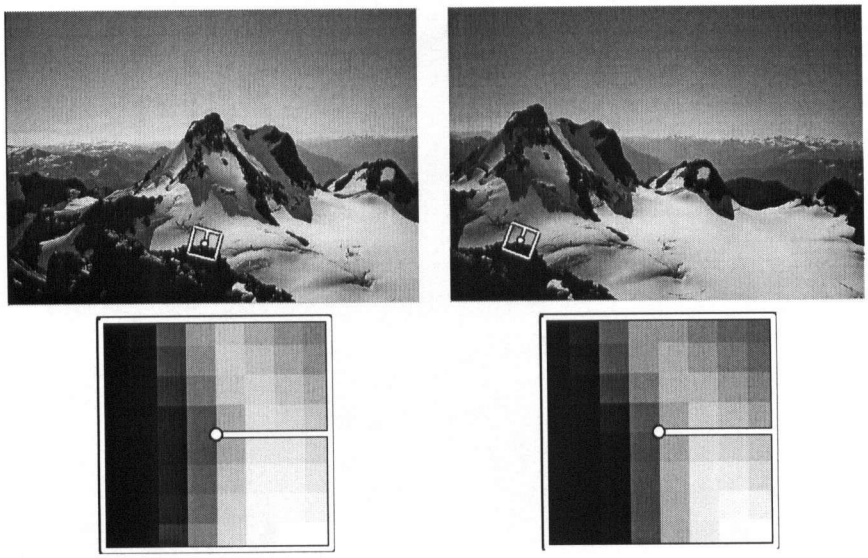


(a) Feature locations

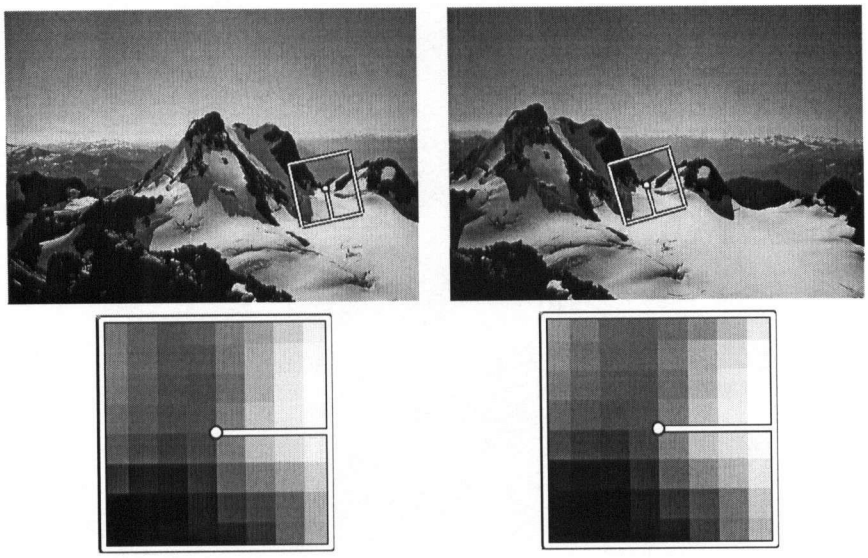


(b) Feature descriptors

Figure 2.11: Corresponding features in a pair of images. For each feature a characteristic scale and orientation is established and an 8×8 patch of pixels sampled for the feature descriptor. Since the reference frame and the image undergo the same transformation between the images, the descriptor vector is the same in both cases (up to noise and modelling error).



(a)



(b)

Figure 2.12: Examples of corresponding features from different images in the Matier dataset. For each image, MOPS are extracted and descriptor vectors are stored in an indexing structure. Feature descriptors are indexed and matched as described in section 2.7.

2.7 Feature Matching

Given Multi-Scale Oriented Patches extracted from all n images, the goal of the matching stage is to find geometrically consistent feature matches between all images. This proceeds as follows. First, we find a set of candidate feature matches using an approximate nearest neighbour algorithm (section 2.8). Then we refine matches using an outlier rejection procedure based on the noise statistics of correct/incorrect matches. Finally we use RANSAC to apply geometric constraints and reject remaining outliers.

2.7.1 Feature-Space Outlier Rejection

Our basic noise model assumes that a patch in one image, when correctly oriented, located and scaled, corresponds to a patch in the other image modulo additive Gaussian noise:

$$I'(\mathbf{x}') = \alpha I(\mathbf{x}) + \beta + n(\mathbf{x}) \tag{2.26}$$

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{t} \tag{2.27}$$

$$\mathbf{A} = s \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{2.28}$$

$$n(\mathbf{x}) \sim \mathcal{N}(0, \sigma_n^2) \tag{2.29}$$

where $I(\mathbf{x})$ and $I'(\mathbf{x})$ are the corresponding patches, and $n(\mathbf{x})$ is independent Gaussian noise at each pixel. To compare two features, we estimate the geometrical parameters from the translation, rotation and scale of the canonical frames (figure A.3), and the photometric parameters using the approximations

$$\alpha = \frac{\sigma'}{\sigma} \tag{2.30}$$

$$\beta = \mu' - \alpha\mu \tag{2.31}$$

These expressions are obtained by taking the mean and variance of equation 2.26, and assuming that n is small. The parameters $\mu, \sigma, \mu', \sigma'$ are the means and variances of patches $I(\mathbf{x})$ and $I'(\mathbf{x}')$ respectively. We then compute the matching error (Euclidean distance) between the two patches $e = \sqrt{\sum_{\mathbf{x}} n(\mathbf{x})^2}$.

Unfortunately, we have found this model to be inadequate for classification, as the error distributions for correctly and incorrectly matching patches overlap significantly (see figure 2.14(a)). Hence, it is not possible to set a global threshold on the matching error to distinguish between correct and incorrect matches.

Note that the above results apply to errors in the image plane, after correcting for the brightness changes α, β between patches. We have also repeated this experiment using a Gaussian noise model in (bias-gain normalised) feature space

$$n(\mathbf{x}) = \left| \frac{I_1(\mathbf{x}_1) - m_1}{\sigma_1} - \frac{I_2(\mathbf{x}_2) - m_2}{\sigma_2} \right| \tag{2.32}$$

and found similar results.

This behaviour has also been observed by Lowe [Low04], who suggested thresholding instead on the ratio e_{1-NN}/e_{2-NN}. Here e_{1-NN} denotes the error for the best match (first nearest neighbour) and e_{2-NN} denotes the error for the second best match (second nearest neighbour). As in Lowe's work, we have also found that the distributions of e_{1-NN}/e_{2-NN} for correct and incorrect matches are better separated than the distributions of e_{1-NN} alone (figure 2.14(b)).

The intuition for why this works is as follows. For a given feature, correct matches always have substantially lower error than incorrect matches. However, the overall scale of errors varies greatly, depending upon the appearance of that feature (location in feature space). See figures 2.19 and 2.20. For this reason it is better to use a discriminative classifier that compares correct and incorrect matches for a particular feature, than it is to use a uniform Gaussian noise model in feature space.

Lowe's technique works by assuming that the 1-NN in some image is a potential correct match, whilst the 2-NN in the same image is an incorrect match. In fact, we have observed that the distance in feature space of the 2-NN and subsequent matches is almost constant[4]. We call this the *outlier distance* $e_{outlier}$, as it gives an estimate of the matching distance (error) for an incorrect match (figure 2.15).

We have found that in the n image matching context we can improve outlier rejection by using information from all of the images (rather than just the two being matched). Using the same argument as Lowe, the 2-NN from each image will almost certainly be an incorrect match. Hence we *average* the 2-NN distances from all n

[4]This is known as the shell property ([Bis95] exercise 1.4). The distances of a set of uniformly distributed points from a query point in high dimensions are almost equal.

images, to give an improved estimate for the outlier distance[5]. This separates the distributions for correct and incorrect matches still further, resulting in improved outlier rejection (figure 2.14(d)).

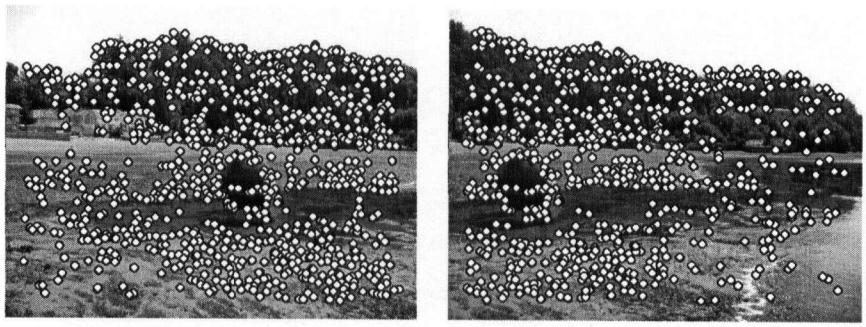
Hence we accept a feature match with error e iff

$$e < f \times e_{outlier} \tag{2.33}$$

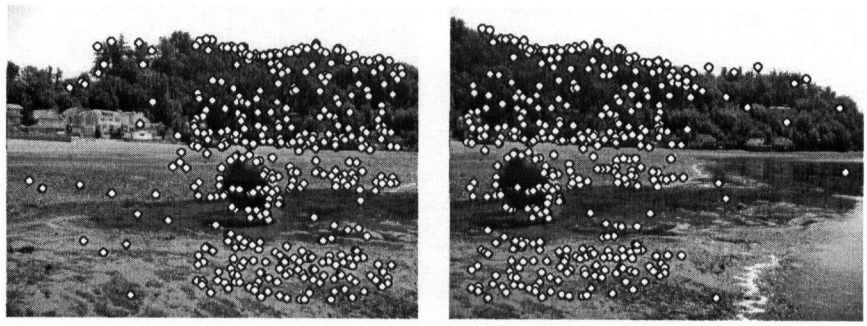
Where the threshold $f = 0.65$. In the general case it is prudent to attempt to match a given feature to the features from *all* other images in the dataset, using the above criterion (this assumes that every pair of images may have a match). However, in many applications e.g. panoramic stitching (chapter 3) and 3D modelling (chapter 5) the number of images that view a ray or point in the world $n_{overlap}$ is small, and hence it is only necessary to find a small number of candidate matches for each feature. Let us assume that we know the maximum number of images that may overlap a given ray $n_{overlap}$. In an ordered list of nearest-neighbour matches, we assume that the first $n_{overlap} - 1$ elements are potential correct matches, and that the $n_{overlap}$ and subsequent elements are incorrect matches. Typically we use a value $n_{overlap} = 5$. In addition to speeding up the search for nearest neighbour matches, finding k matches per feature in a collection of n images (where $k << n$) has the advantage that any incorrect feature matches that do occur are distributed over a large number of images, and thus less likely to cause an incorrect image match to be declared.

In general the feature-space outlier rejection test is very powerful. For example, we can eliminate 80% of the false matches for a loss of less than 10% correct matches. This allows for a significant reduction in the number of RANSAC iterations required in subsequent geometry estimation steps (see figure 2.13). These results are computed for Matier (7 images, 6649 features, 5970 correct matches), Van Gogh (7 images, 6557 features, 9260 correct matches) and Abbey (20 images, 18567 features, 15558 correct matches).

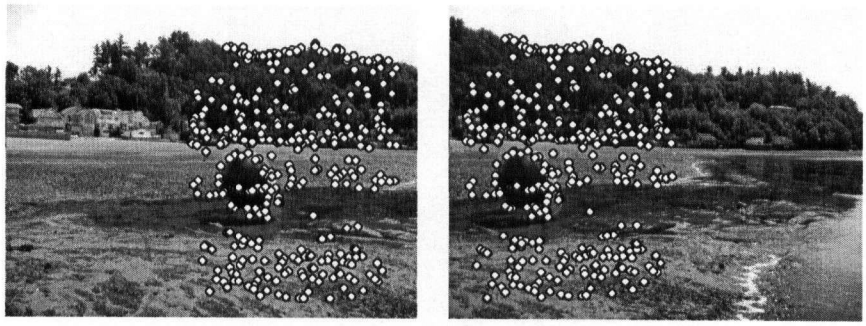
[5]We also tried using other statistics of the distribution of outliers e.g. max, min, median, but found that using the average gave the best results. Future work might try to model the whole outlier distribution for these tests.



(a) All 1313 feature matches



(b) 839 outliers rejected using feature space outlier rejection



(c) A further 96 matches rejected using geometrical constraints

Figure 2.13: Outlier rejection using b) feature space outlier rejection c) geometric constraints. The raw matches are a). There were 1313 initial matches, of which 839 were rejected without geometric constraints by thresholding based on the outlier distance, and a further 96 were rejected using geometric constraints. The input images are 385×512 and there were 378 matches in the final solution.

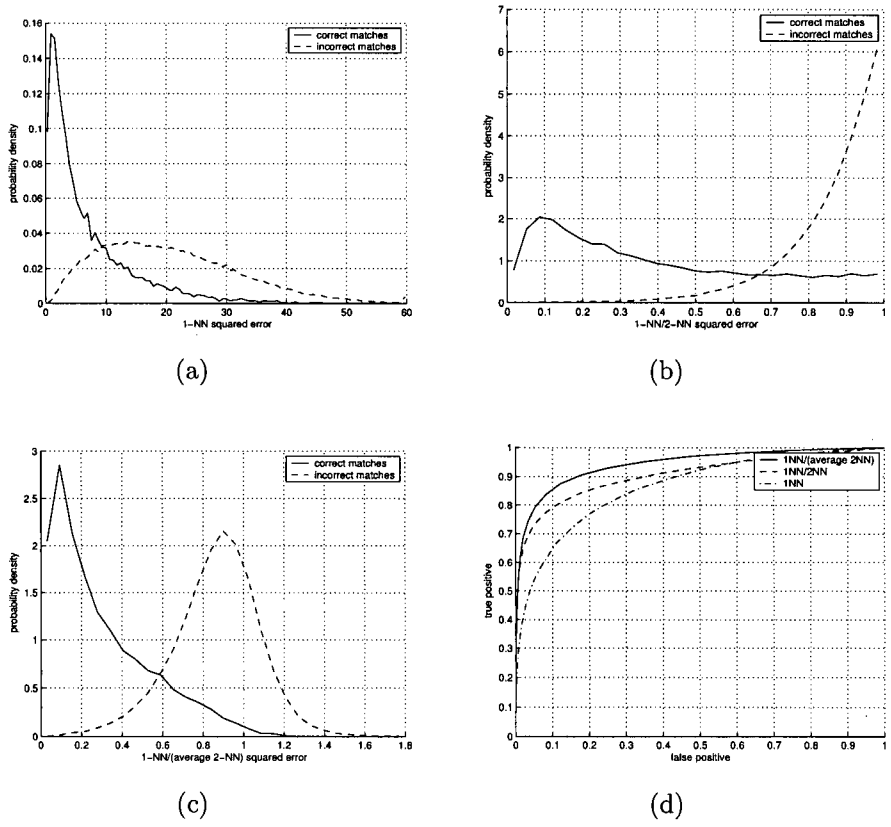


Figure 2.14: Distributions of matching error for correct and incorrect matches. Note that the distance of the closest match (the 1-NN) is a poor metric for distinguishing whether a match is correct or not (figure (a)), but the ratio of the closest to the second closest (1-NN/2-NN) is a good metric (figure (b)). We have found that using an average of 2-NN distances from multiple images (1NN/(average 2-NN)) is an even better metric (figures (c)-(d)). These results were computed from 18567 features in 20 images of the Abbey dataset (see appendix B), and have been verified for several other datasets.

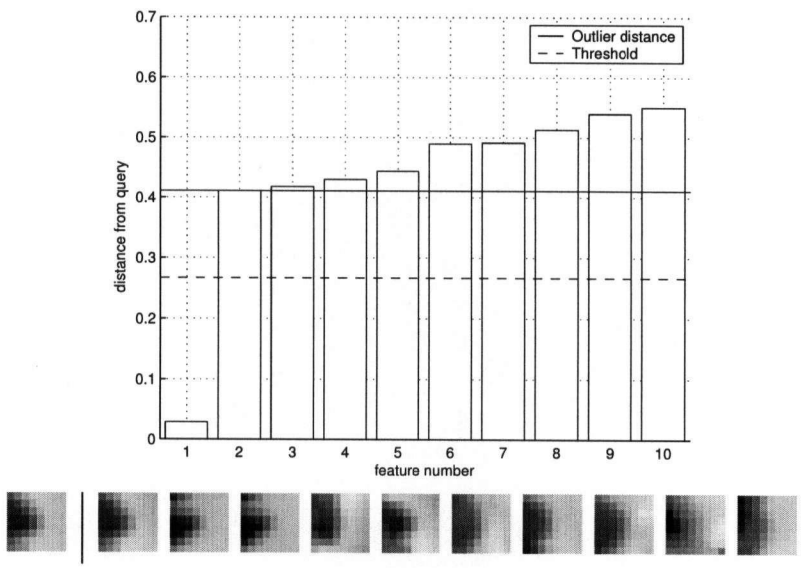


Figure 2.15: Thresholding based on outlier distance. This figure shows the best 10 matches for a sample feature. The first is a correct match, and the rest are incorrect matches. Thresholding based purely on matching error gives poor results, since matching errors vary greatly depending upon the position in feature space. However, thresholding at a fraction of the outlier distance gives better results.

2.7.2 Spatial Variation of Errors

In actual fact the errors between corresponding patches are not uniform across the patch as suggested in equation 2.29. We have also computed the error variance assuming a diagonal covariance model

$$\mathbf{n}(\mathbf{x}) \sim \mathcal{N}(0, \mathbf{\Sigma}_n) \tag{2.34}$$

where

$$\mathbf{\Sigma}_n = \begin{bmatrix} \sigma_{11}^2 & 0 & 0 & \\ 0 & \sigma_{22}^2 & 0 & \dots \\ 0 & 0 & \sigma_{33}^2 & \dots \\ & \vdots & & \ddots \end{bmatrix} \tag{2.35}$$

If we assume that the error variance is constant across the patch ($\mathbf{\Sigma}_n = \sigma_n^2 \mathbf{I}$) we find that the standard deviation of intensity errors for correct matches is $\sigma_n = 0.0334$ (for brightness values in the range $0 < I < 1$). That is, the error for correct matches is around 3%. However, with the diagonal covariance model of equation 2.35 we find that

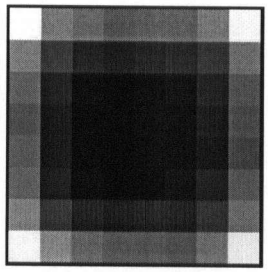
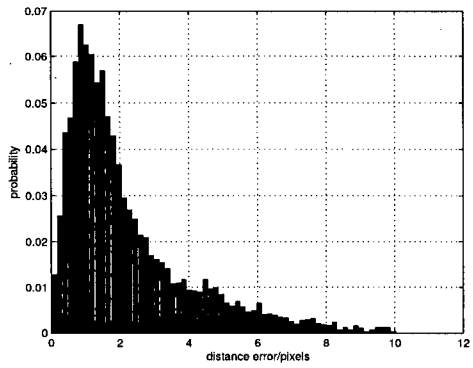


Figure 2.16: Spatial variation of errors across the patch (for correct feature matches). Lighter tones indicate larger values of variance. The variance of the errors at the edge of the patches are larger than those in the centre. This is consistent with making small errors in scale / orientation selection.

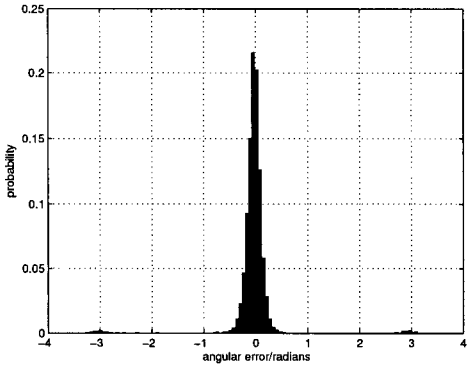
the standard deviation of errors at the edge of the patch is approximately 2 times that in the centre. This is shown in figure 2.16. Note that this is consistent with small errors in scale / orientation estimation for each patch, as these would generate larger errors at the edge of the patch. Preliminary experiments have shown that weighting the errors by their inverse standard deviation i.e. minimising $|\Sigma_n^{-\frac{1}{2}}\mathbf{n}(\mathbf{x})|^2$ does not give much improvement over simply minimising $|\mathbf{n}(\mathbf{x})|^2$. These results were computed using 7572 correctly matching features (RANSAC inliers with $\epsilon = 10$ pixels) from the Matier dataset.

2.7.3 Position and Orientation Errors for Correct Matches

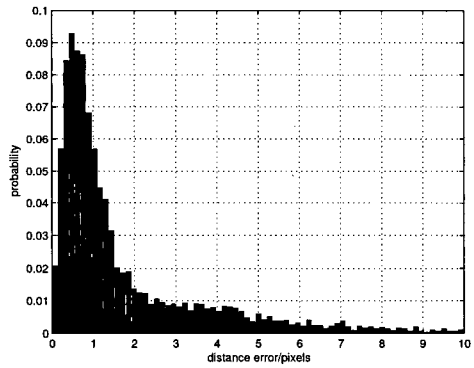
Figure 2.17 shows the residual image position errors and errors in rotation estimates for correctly matching features. Note that the features from the (rotation only) Van Gogh dataset are more accurately located than those in the Matier dataset (see appendix B). For the pure rotation dataset, features are typically accurate in the 0-2 pixel range, whilst those from the Matier dataset are typically in the 0-4 pixel range. This discrepancy could be due to perspective distortion (which could adversely affect the feature location), and also unmodelled parameters such as radial distortion. An interesting future experiment would be to correct for radial distortion to check if the accuracy of features is still poorer when perspective distortion is present. This could also be simulated synthetically as in chapter 4. Another interesting observation is that there are a significant number of features that match correctly when the rotation estimate is 180° out. This suggests that some the of the features might be rotationally symmetric e.g. the 2×2 checkerboard pattern has the properties of ambiguous gradient



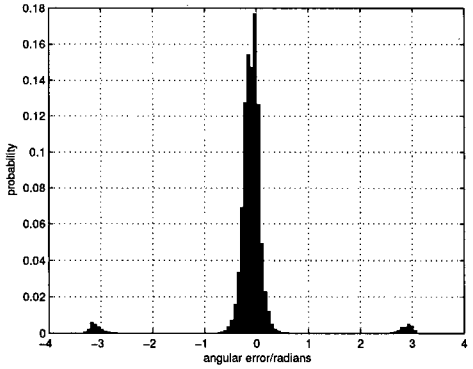
(a) Image position error (Matier)



(b) Orientation error (Matier)



(c) Image position error (Van Gogh)



(d) Orientation error (Van Gogh)

Figure 2.17: Distributions of image location error and feature orientation error for correctly matching features (RANSAC inliers). Note that features from the Van Gogh dataset are more accurately located. Also, there are a significant number of features that match correctly when the rotation estimate is 180° out. This could occur for rotationally symmetric features where the gradient is ambiguous e.g. a 2×2 checkerboard pattern.

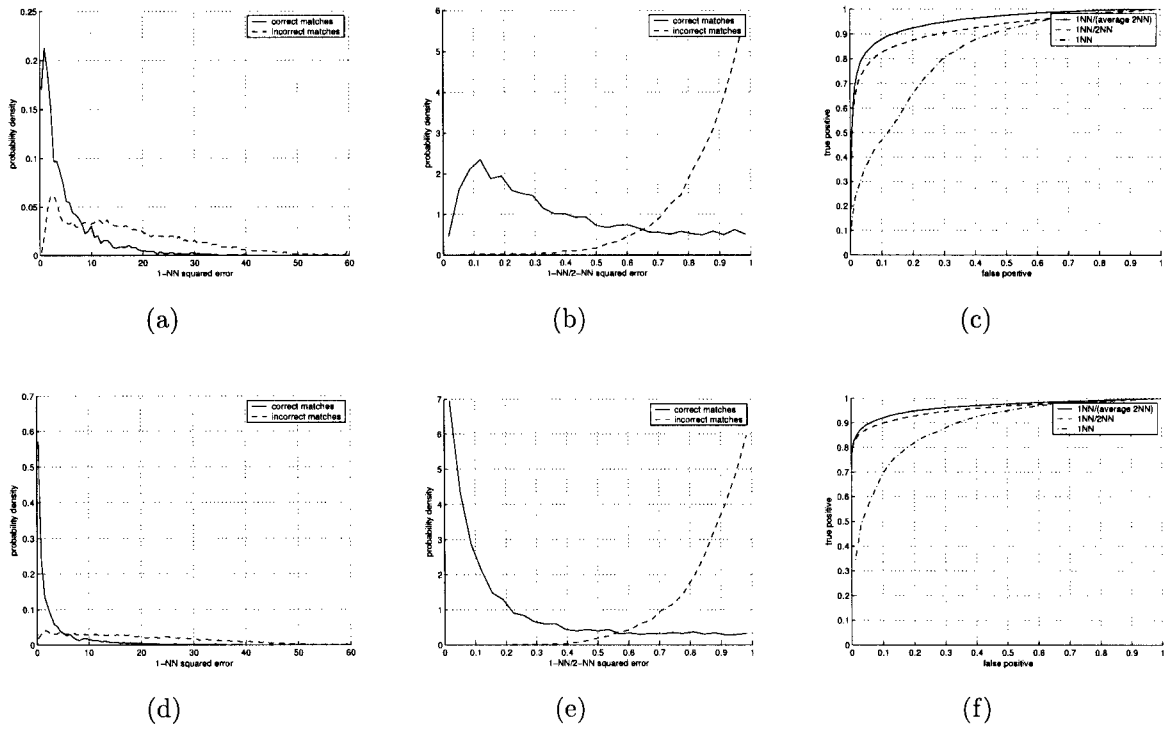


Figure 2.18: Distributions of matching error for correct and incorrect matches. The results were computed for the Matier (a, b, c) and Van Gogh (d, e, f) datasets. Again the distance of the closest match (the 1-NN) is a poor metric for distinguishing whether a match is correct or not (figures (a), (d)). The ratio of the closest to the second closest (1-NN/2-NN) is a better metric (figures (b), (e)), but using an average of 2-NN distances from multiple images (1NN/(average 2-NN)) is better still (figures (c), (f)).

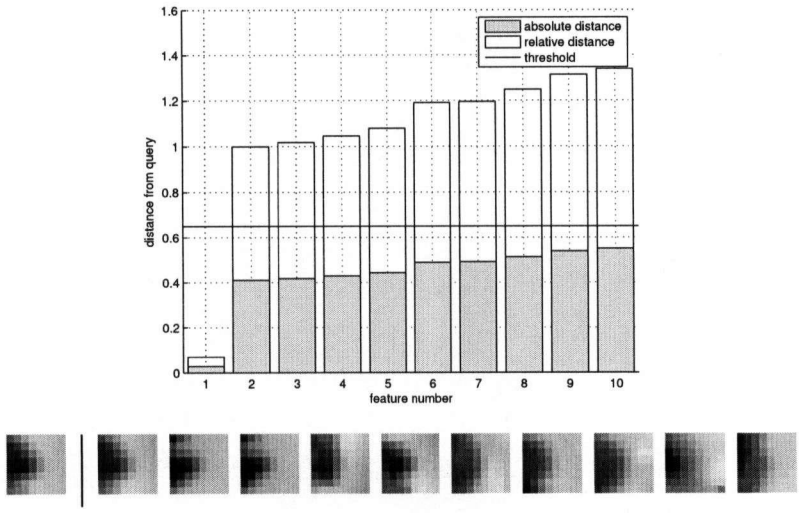
and rotational symmetry. To compute position and orientation errors, features were projected between images using the homographies obtained from bundle adjustment over all images.

2.8 Feature Indexing

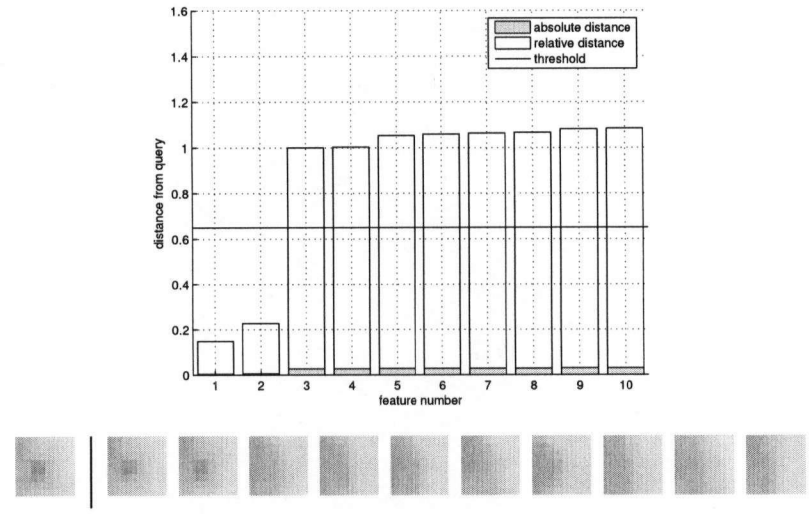
At this stage we wish to find nearest neighbours for all features from all images. This is the well known all nearest neighbours problem

$$\forall j \quad NN(j) = \arg\min_i ||\mathbf{x}_i - \mathbf{x}_j||, \; i \neq j \tag{2.36}$$

This is naively $O(n^2)$ but several more efficient (approximate) solutions have been pro-

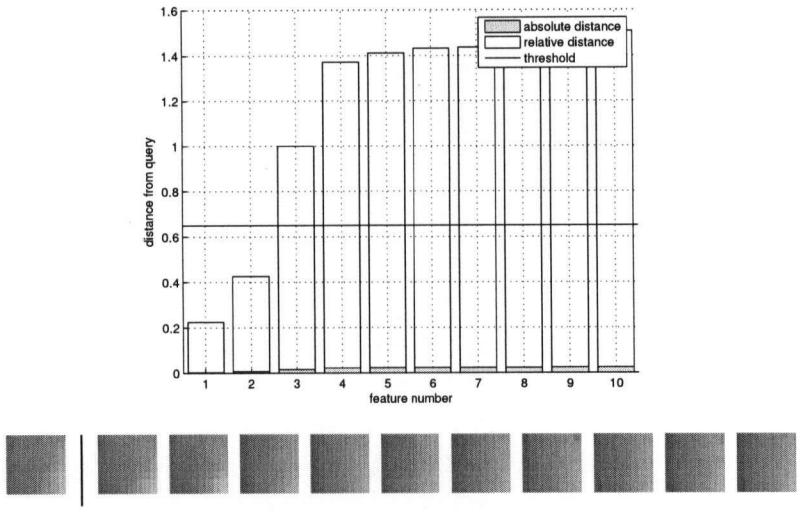


(a)

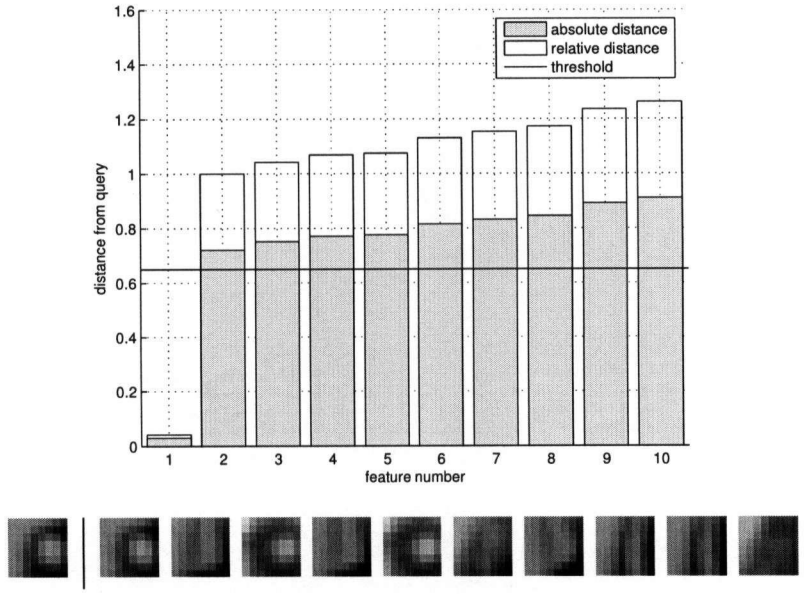


(b)

Figure 2.19: Distances of correct and incorrect matches for high and low contrast features. We plot *absolute* and *relative* distances. Absolute distance is the Euclidean distance between brightness normalised patches. Relative distance is the distance relative to the outlier distance (the outlier distance is take as the distance of the closest 2-NN match from all other images). Note that for the high contrast features, the absolute distances are all much larger than for the low contrast features.



(a)



(b)

Figure 2.20: Distances of correct and incorrect matches for high and low contrast features. Absolute distances are larger for high contrast features than low contrast features. Hence, thresholding based on absolute match distances is a poor test, but thresholding on relative distances is better.

Figure 2.21: Indexing is performed on the first 3 non-zero wavelet coefficients (the mean is 0). These represent the first derivatives in x and y and the second order cross derivative.

posed [BL97, NN97, GM00, SVD03]. In this section, we describe a nearest-neighbour algorithm that exploits the properties of our descriptor vectors. In particular, it makes use of the stability of the low-frequency components of our descriptor vectors, by indexing on the first 3 non-zero wavelet coefficients.

Features are indexed in a three-dimensional lookup table with dimensions corresponding to the first 3 non-zero wavelet coefficients c_1, c_2, c_3 (estimates of $\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}, \frac{\partial^2 I}{\partial x \partial y}$ over the patch) (see figure 2.21). The lookup table has $b = 10$ bins per dimension, which cover $\pm n_\sigma = 3$ standard deviations from the mean of that dimension. Note that the means are typically around zero except for the first derivative that is aligned with the feature orientation, which is significantly positive.

The bins are overlapped so that data within half a bin width, i.e. $\frac{2n_\sigma}{b-1}\frac{1}{2} = \frac{\sigma}{3}$, are guaranteed to be matched against the query. These are approximate nearest neighbours as it is possible (but unlikely) that the true nearest neighbour lies outside $\frac{\sigma}{3}$ in one of the 3 dimensions. The query is exhaustively matched to all features in the query bin, and k approximate nearest neighbours are selected. We then apply the outlier distance constraint as described in section 2.7.1 to verify correct matches and eliminate outliers. Indexing with b bins on 3 dimensions gives a speedup of $b^3/2^3$ (assuming features are evenly distributed in the histogram) at the expense of some potential for lost feature matches.

Table 2.1 shows the percent recall for 3 indexing methods:

Wavelet low freq Indexing uses the first 3 non-zero wavelet coefficients

Pixel random Indexing uses 3 random grey values from the descriptor

Wavelet random Indexing uses 3 random wavelet coefficients from the descriptor

It is clear from the results that using the low frequency wavelet coefficients for indexing is most effective. Choosing 10 bins per dimension gives a speedup of $10^3/2^3 =$

125 compared to exhaustive nearest neighbour matching, with the loss of less than 10% of the matches. Indexing using low frequency wavelet coefficients is 10-20% better than the other methods at this operating point. In later chapters (section 3.2) we also describe an alternative, k-d tree based nearest neighbour algorithm.

Indexing Method	*Dataset*	*Number of bins / dimension*			
		1	*5*	*10*	*15*
Wavelet low freq	Matier	100	99.6	91.4	72.4
	Dash point	100	99.8	93.2	76.8
	Abbey	100	99.9	95.1	80.2
Pixel random	Matier	100	97.9	79.8	57.8
	Dash point	100	96.4	74.0	52.9
	Abbey	100	96.7	77.8	56.3
Wavelet random	Matier	100	84.4	49.2	28.1
	Dash point	100	81.5	42.8	25.6
	Abbey	100	83.0	45.4	24.6

Table 2.1: Indexing on wavelet coefficients vs pixel values - percent recall in database matching. Using 10 bins per dimension, indexing on the 3 non-zero low frequency Haar wavelet coefficients (x and y derivative and the cross derivative) gives about 10% better recall than indexing on random dimensions (pixels) of the descriptor.

2.9 Analysis of Descriptor Sampling

In this section we describe experiments to test the properties of our patch based descriptors. Firstly, we attempt an iterative refinement strategy for patches using a a direct method for registration (section 2.9.1). Next we discuss the effects of different image sampling procedures (section 2.9.2). Finally, we compare our descriptors with the commonly used SIFT descriptors (section 2.9.3).

2.9.1 Patch Refinement

In [MS03], Mikolajczyk and Schmid note that "It would be interesting to include correlation with patch alignment which corrects for these errors and to measure the gain obtained by such an alignment." Since sensitivity to localization errors has been touted as one of the weaknesses of pixel-based descriptors, we decided to implement this suggestion to see how much it would help. Rather than computing sum-squared error on

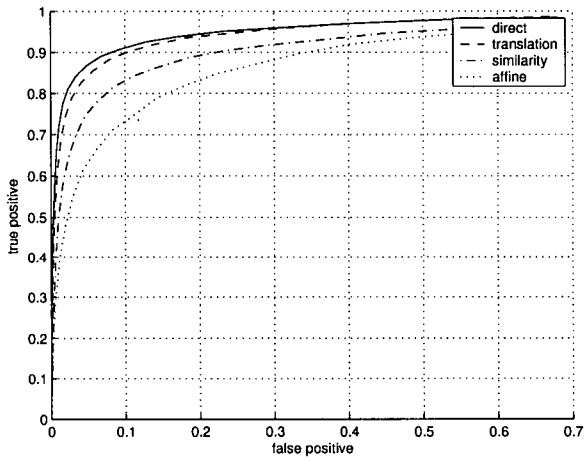


Figure 2.22: ROC curves for patch refinement with different alignment models (Matier dataset). Each additional free parameter degrades the matching performance.

pixel patches (or wavelet coefficients) directly, we included a stage of Lucas-Kanade [LK81, BM04] refinement to bring the patches more closely into spatial alignment before computing the pairwise descriptor distance. Since this has elements in common with the use of tangent distances [SLDV96] we expected that there might be an improvement in the separation of good and bad matches. Instead we found the opposite to be true.

We used four motion models (direct, translation, similarity and affine) with 0, 2, 4 and 6 parameters respectively. The results are shown in figure 2.22. Note that matching performance is degraded for each new parameter that is added to the model. Since correct matches are already fairly well aligned, but bad matches typically have large errors, refinement tends to overfit the incorrect matches, whilst making only small improvements to the correct matches. This means that Lucas-Kanade refinement actually makes it more difficult to distinguish between correct and incorrect matches than before.

This is a somewhat unexpected result. Future work would incorporate priors on the transformation parameters to prevent overfitting of the incorrect matches.

2.9.2 Image Downsampling and Bilinear Interpolation

A typical approach to image sampling is to downsample the original signal to the sampling frequency before interpolation, so that the image is sampled at the Nyquist

Relative Sampling Level	*Extra Smoothing*	*Number of feature matches*		
		Matier	*Dash Point*	*Abbey*
0	0	2620	2323	7467
-1	σ	3017	2988	9078
-2	2σ	3139	3058	9268

Table 2.2: Effect of pyramid downsampling on feature matching. We found better results by sampling the feature descriptor at a smoothed version of a finer pyramid level, when using bilinear resampling.

frequency (once per pixel). In theory this results in a correct sampling of the (bandlimited) original signal, assuming the correct decimation/interpolation kernel is used (a sinc function). However, in practice bilinear interpolation is often used for speed. In this case, I have found that better results are obtained by maintaining a denser signal sampling during interpolation than the usual 1 pixel/sample.

Table 2.2 compares feature matching performance when bilinear resampling is performed at successively finer pyramid levels corresponding to 1, 2 and 4 pixels of the original signal per sample. 'Relative level' means the level relative to the Nyquist sampling level, where one would sample exactly once per pixel in the pyramid. 'Extra smoothing' is the standard deviation of the extra Gaussian smoothing applied (to prevent aliasing) i.e. instead of sampling at some level l, we sample at level $l-1$, but introduce extra smoothing with a Gaussian kernel standard deviation σ. In each case, exactly the same interest points were extracted, so the total number of feature matches that result is a good indication of how well the descriptors are working.

The results show that better results are obtained (about 15%-20% more matches) by performing bilinear resampling at the next finer pyramid level (where the signal sampling is approximately twice the sampling frequency). Smaller gains are obtained by moving to the next finer pyramid level again. These gains must be balanced against the cost of the extra convolution operations required in Gaussian smoothing.

2.9.3 Comparison to SIFT features

To compare Multi-Scale Oriented Patches (MOPS) and SIFT features, I used 3 datasets of panoramic images. For each method, I extracted approximately the same number of interest points from each image, and then exhaustively matched them to find $k = 4$ exact nearest neighbour matches for each feature. I then used identical RANSAC

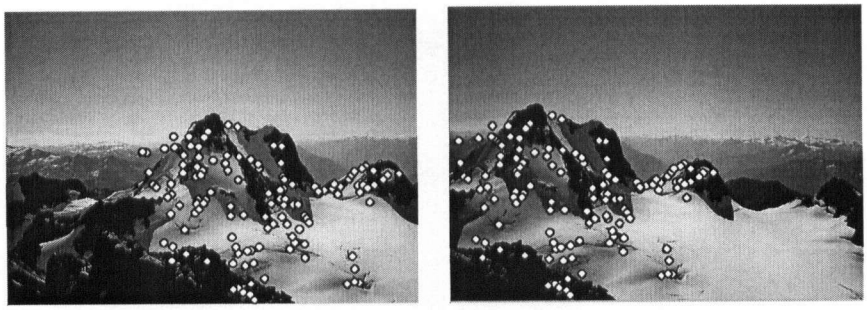
Dataset		*MOPS*	*SIFT*
Matier	#interest points	3610	3678
	#matches	3776	4344
	#matches/interest point	1.05	1.18
Dash point	#interest points	32689	32517
	#matches	11750	22928
	#matches/interest point	0.36	0.71
Abbey	#interest points	15494	15710
	#matches	18659	21718
	#matches/interest point	1.20	1.38

Table 2.3: Comparison of Multi-Scale Oriented Patches and SIFT feature matching. Note that SIFT features have a larger number of matches per interest point for each of the 3 datasets.

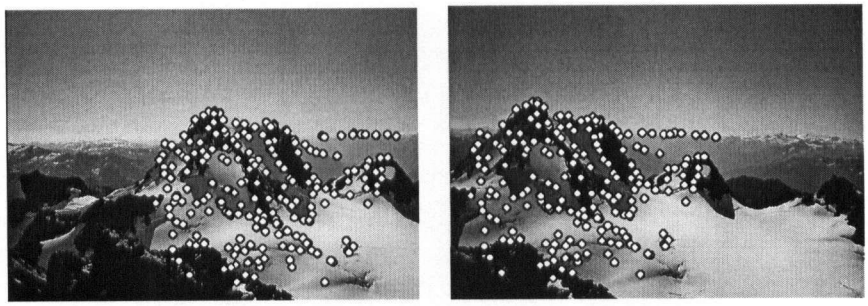
algorithms to find the number of correct feature matches. In each case I have tabulated the number of correct matches per feature. The results are given in table 2.3. Note that both methods could potentially find more matches by using more scales / adjusting interest point strength thresholds etc.

From the results of table 2.3 it seems that in terms of number of matches per interest point SIFT features outperform MOPS. Why is this? Lowe [Low04] reports higher repeatability for his difference of Gaussian (DOG) interest points (around 90% compared to 70% for our Harris corners), although this is highly dataset dependent. The rotation estimate used in SIFT features (maxima of a histogram of local gradients) is more robust since multiple orientations can be assigned if the histogram peaks are close. Lowe reports repeatability of around 80% for position (to an accuracy of 3 pixels), orientation and scale compared to our value of 58%. Another issue is that SIFT features are found to be located very close to image edges, but since MOPS use relatively large image patches they are constrained to be at least 20 pixels from the image edge (this is the main reason SIFT performs much better on the Dash Point dataset). This is shown in figure 2.24. Finally, the SIFT descriptor is more robust to affine change and small shifts in interest point position than patch correlation, due to accumulating measurements in spatial histograms.

Note however, that MOPS and SIFT features tend to concentrate on different areas in the images. In particular, the DOG detector used for SIFT makes it more likely to find 'blobs' – bright areas surrounded by dark pixels or vice versa, whereas the autocorrelation detector used for MOPS makes it more likely to find edge or corner



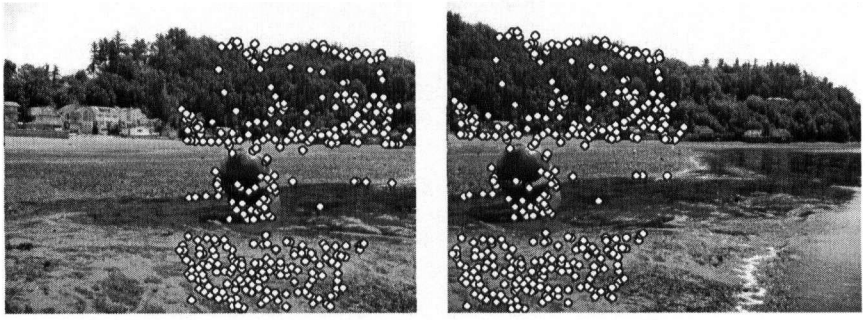
(a) SIFT feature matches (167)



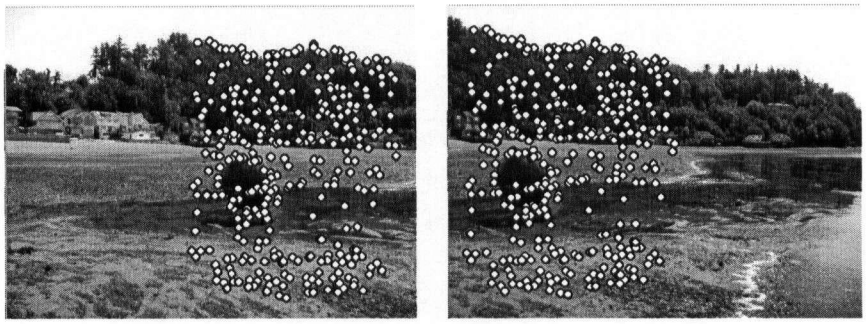
(b) MOPS feature matches (238)

Figure 2.23: Comparison of SIFT features and MOPS features. Note that more MOPS features are found at edges/corners, whereas SIFT features concentrate on blobs. Though in this case there were more MOPS feature matches than SIFT feature matches, in general SIFT features gave better overall performance on our test datasets (see table 2.3).

like features. This suggests that a combination of the two feature types might be effective. Also, it is important to note that MOPS features matches are by design well spatially distributed in the image. Sometimes SIFT feature matches are very close together. Are feature matches equally useful if they are very close together? We think not. It seems that some other criterion, such as registration accuracy, would be a better criterion for evaluating features, than simply counting the number of matches per feature (see chapter 4). Another approach would be to compare the number of matched/dropped images in a panorama dataset.

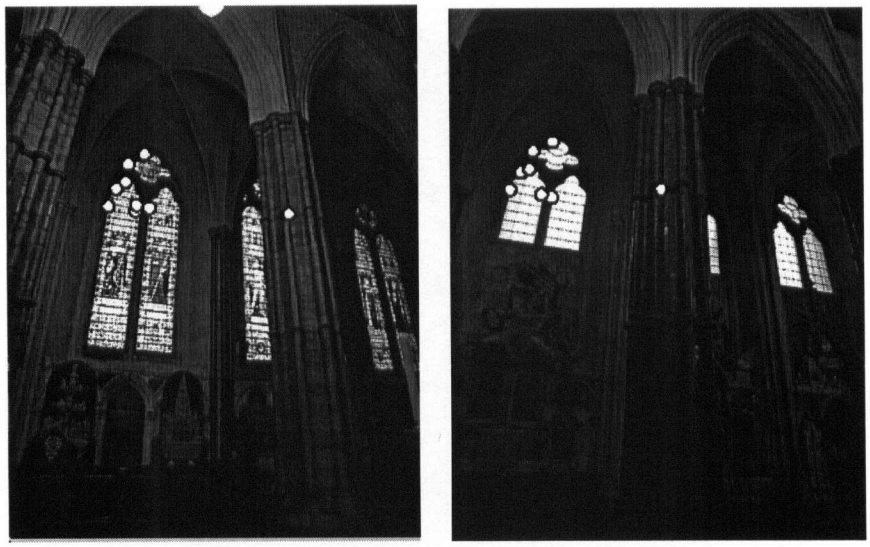


(a) SIFT feature matches (421)

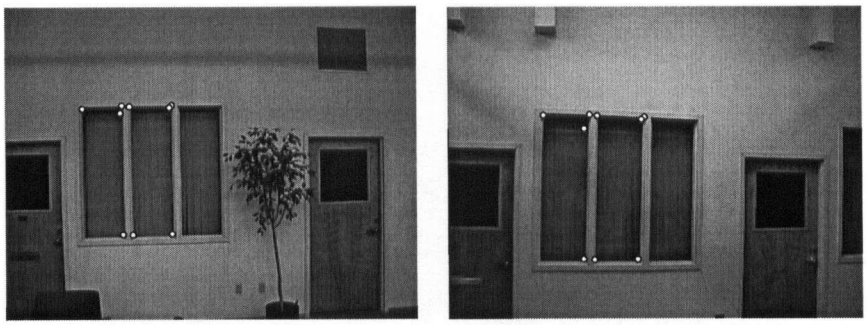


(b) MOPS feature matches (372)

Figure 2.24: Comparison of SIFT features and MOPS features. For the Dash Point dataset, the SIFT feature detector performed better in terms of number of matches. However, note that the MOPS feature matches are better spatially distributed e.g. there are many more matches in the sea/sand. Also note that the SIFT features are located right up to the edge of the image, but due to the large patches used in MOPS, the features are constrained to be at least 20 pixels away from an image edge.



(a) Abbey. Though several feature matches have been found between the two images, the stained glass windows are in fact different. Though generalisation capability is desirable in many object recognition applications, in image stitching we are generally only interested in matching to the *same* object or scene.



(b) Office. Though the windows look strikingly similar, the presence of the tree in the first image gives away the fact that they are in fact different. In this case it would be a difficult to distinguish whether motion of the tree or the observer caused the error.

Figure 2.25: Matching mistakes are are often caused by repeating structures or similar looking objects that appear in multiple views. Often, the set of feature matches may appear consistent, but the images will differ substantially in other areas (a). This suggests the need for robust image matching metrics based on *all* the image data, and not just feature positions. In some cases however, it will still be difficult to tell if the source of error is object motion, or if the images are really different (b).

2.10 Summary

We have presented a new type of invariant feature, which we call Multi-Scale Oriented Patches (MOPS). These features utilise a novel adaptive non-maximal suppression algorithm for interest point location, and a simple sampling of the (oriented) local image intensity for the feature descriptor. We have also introduced two innovations in multi-image matching. First, we have demonstrated an improved test for verification of pairwise image matches that uses matching results from all n images. Second, we have shown that an indexing scheme based on low frequency wavelet coefficients yields a fast approximate nearest neighbour algorithm that is superior to indexing using the raw data values.

Future Work

We conclude by noting some possible areas for future development of MOPS. We discuss more general possibilities for invariant features in the final conclusions (chapter 6).

Orientation Estimation Orientation estimation is currently problematic if the gradient is not well defined (i.e. small) or varies rapidly around the interest point (i.e. small changes in interest point location lead to large changes in the orientation). Alternative methods could include: using the largest eigenvector of $\mathbf{H}$ (although this also has a degeneracy for rotational symmetry), steerable filters, or peaks in a histogram of local orientations. The latter option is attractive as it can be made robust by considering multiple peaks in the histogram in ambiguous cases.

Colour We could use RGB features instead of greyscale, or just add a few dimensions of colour information (e.g. the average [R, G, B]/(R + G + B) for the patch) to the descriptors with an appropriate weighting.

Interest Operators In addition to using autocorrelation maxima, we could form features using other interest operators e.g. difference of Gaussian maxima, watershed regions or edge based features.

Multi-scale/3D Matching In order to better cope with multi-scale matching we could use a true scale-space interest operator. For example, we could use an

image pyramid with a finer scale sampling, and interpolate interest points to sub-scale accuracy. To cope better with 3D matching problems we should introduce more robustness to affine change and relative shifting of edge positions.

Learning Feature Matching The multi-image matching systems that are described in subsequent chapters make it easy to generate large datasets of known correct and incorrect matches. This could be used to learn data driven classifiers to distinguish between correct and incorrect matches.

Chapter 3

Automatic Panoramic Image Stitching

3.1 Introduction

This chapter extends the ideas of invariant feature matching developed in the previous chapter to the problem of panoramic image stitching. Automatic panoramic image stitching has an extensive research literature [Sze04] and several commercial applications [Che95, REA, MSF]. The basic geometry of the problem is well understood, and consists of estimating a 3×3 camera matrix or homography for each image [HZ04, SS97]. This estimation process needs an initialisation, which is typically provided by user input to approximately align the images, or a fixed image ordering. For example, the PhotoStitch software bundled with Canon digital cameras requires a horizontal or vertical sweep, or a square matrix of images. The REALVIZ Stitcher [REA] has a user interface to roughly position the images with a mouse, before automatic registration proceeds. Our work is novel in that we require no such initialisation to be provided.

In the research literature methods for automatic image alignment and stitching fall broadly into two categories – direct [SK95, IA99, SK99, SS00] and feature based [ZFD97, CZ98, MJ02]. Direct methods have the advantage that they use all of the available image data and hence can provide very accurate registration, but they require a close initialisation. Feature based registration does not require initialisation, but traditional feature matching methods (e.g. correlation of image patches around Harris corners [Har92, ST94]) have lacked the invariance properties needed to enable reliable matching of arbitrary panoramic image sequences.

In this chapter we describe an invariant feature based approach to fully automatic panoramic image stitching. This has several advantages over previous approaches. Firstly, our use of invariant features enables reliable matching of panoramic image sequences despite rotation, zoom and illumination change in the input images. Secondly,

by viewing image stitching as a multi-image matching problem, we can automatically discover the matching relationships between the images, and even recognise panoramas in unordered datasets. Thirdly, we generate high-quality results using automatic straightening, gain compensation, and multi-band blending to render seamless output panoramas. The results shown in this chapter are from a system implemented using SIFT features [Low99]. We have also implemented a similar system using MOPS (chapter 2).

The remainder of this chapter is structured as follows. Section 3.2 develops the geometry of the problem and motivates our choice of invariant features. Section 3.3 describes our image matching methodology (RANSAC) and a probabilistic model for image match verification. In section 3.4 we describe our image alignment algorithm (bundle adjustment) which jointly optimises the parameters of each camera. Sections 3.5 - 3.7 describe the rendering pipeline including automatic straightening, gain compensation and multi-band blending. In section 3.9 we present conclusions and ideas for future work.

3.2 Feature Matching

The first step in the panoramic recognition algorithm is to extract and match SIFT features between all of the images. SIFT features are located at scale-space maxima/minima of a difference of Gaussian function. At each feature location, a characteristic scale and orientation is established. This gives a similarity-invariant frame in which to make measurements. Although simply sampling intensity values in this frame would be similarity invariant, the invariant descriptor is actually computed by accumulating local gradients in orientation histograms. This allows edges to shift slightly without altering the descriptor vector, giving some robustness to affine change. This spatial accumulation is also important for shift invariance, since the interest point locations are typically accurate in the 0-3 pixel range (see figure 2.17 and also [BSW05, SZ03]). Illumination invariance is achieved by using gradients (which eliminates bias) and normalising the descriptor vector (which eliminates gain).

Assuming that the camera rotates about its optical centre, the group of transformations the images may undergo is a special group of homographies. We parameterise each camera by a rotation vector $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ and focal length f. This gives pairwise

homographies $\tilde{\mathbf{u}}_i = \mathbf{H}_{ij}\tilde{\mathbf{u}}_j$ where

$$\mathbf{H}_{ij} = \mathbf{K}_i\mathbf{R}_i\mathbf{R}_j^T\mathbf{K}_j^{-1} \tag{3.1}$$

and $\tilde{\mathbf{u}}_i$, $\tilde{\mathbf{u}}_j$ are the homogeneous image positions ($\tilde{\mathbf{u}}_i = s_i[\mathbf{u}_i, 1]$, where $\mathbf{u}_i$ is the 2-dimensional image position). The 4 parameter camera model is defined by

$$\mathbf{K}_i = \begin{bmatrix} f_i & 0 & 0 \\ 0 & f_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

and (using the exponential representation for rotations)

$$\mathbf{R}_i = e^{[\boldsymbol{\theta}_i]_\times}, \quad [\boldsymbol{\theta}_i]_\times = \begin{bmatrix} 0 & -\theta_{i3} & \theta_{i2} \\ \theta_{i3} & 0 & -\theta_{i1} \\ -\theta_{i2} & \theta_{i1} & 0 \end{bmatrix} \tag{3.3}$$

Ideally one would use image features that are invariant under this group of transformations. However, for small changes in image position

$$\mathbf{u}_i = \mathbf{u}_{i0} + \left.\frac{\partial \mathbf{u}_i}{\partial \mathbf{u}_j}\right|_{\mathbf{u}_{i0}} \Delta\mathbf{u}_j \tag{3.4}$$

or equivalently $\tilde{\mathbf{u}}_i = \mathbf{A}_{ij}\tilde{\mathbf{u}}_j$, where

$$\mathbf{A}_{ij} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

is an affine transformation obtained by linearising the homography about $\mathbf{u}_{i0}$. This implies that each small image patch undergoes an affine transformation, and justifies the use of SIFT features which are partially invariant under affine change.

Once features have been extracted from all n images (linear time), they must be matched. Since multiple images may overlap a single ray, each feature is matched to its k nearest neighbours (we use $k = 4$). This can be done in $O(n \log n)$ time by using a k-d tree to find approximate nearest neighbours [BL97]. A k-d tree is an axis aligned binary space partition, which recursively partitions the feature space at the mean in the dimension with highest variance. The k-d tree is ultimately (for large n) more

efficient than the indexing scheme described in section 2.8 as each feature lookup is $O(\log n)$ (and not $O(n)$).

3.3 Image Matching

At this stage the objective is to find all matching (i.e. overlapping) images. Connected sets of image matches will later become panoramas. Since each image could potentially match every other one, this problem appears at first to be quadratic in the number of images. However, it is only necessary to match each image to a small number of neighbouring images in order to get a good solution for the image geometry.

From the feature matching step, we have identified images that have a large number of matches between them. We consider a constant number m images, that have the greatest number of feature matches to the current image, as potential image matches (we use $m = 6$). First, we use RANSAC to select a set of inliers that are compatible with a homography between the images. Next we apply a probabilistic model to verify the match.

3.3.1 Robust Homography Estimation using RANSAC

RANSAC (random sample consensus) [FB81] is a robust estimation procedure that uses a minimal set of randomly sampled correspondences to estimate image transformation parameters, and finds a solution that has the best consensus with the data. In the case of panoramas we select sets of $r = 4$ feature correspondences and compute the homography $\mathbf{H}$ between them using the direct linear transformation (DLT) method [HZ04]. We repeat this with $n = 500$ trials and select the solution that has the maximum number of inliers (whose projections are consistent with $\mathbf{H}$ within a tolerance ϵ pixels). Given the probability that a feature match is correct between a pair of matching images (the inlier probability) is p_i, the probability of finding the correct transformation after n trials is

$$p(\mathbf{H} \textit{ is correct}) = 1 - (1 - (p_i)^r)^n \tag{3.6}$$

After a large number of trials the probability of finding the correct homography is very high. For example, for an inlier probability $p_i = 0.5$, the probability that the correct homography is *not* found after 500 trials is approximately 1×10^{-14}.

RANSAC is essentially a sampling approach to estimating **H**. If instead of maximising the number of inliers one maximises the sum of the log likelihoods, the result is maximum likelihood estimation (MLE). Furthermore, if priors on the transformation parameters are available, one can compute a maximum a posteriori estimate (MAP). These algorithms are known as MLESAC and MAPSAC respectively [Tor02].

3.3.2 Probabilistic Model for Image Match Verification

For each pair of potentially matching images we have a set of feature matches that are geometrically consistent (RANSAC inliers) and a set of features that are inside the area of overlap but not consistent (RANSAC outliers). The idea of our verification model is to compare the probabilities that this set of inliers/outliers was generated by a correct image match or by a false image match.

For a given image we denote the total number of features in the area of overlap n_f and the number of inliers n_i. The event that this image matches correctly/incorrectly is represented by the binary variable $m \in \{0, 1\}$. The event that the i^{th} feature match $f^{(i)} \in \{0, 1\}$ is an inlier/outlier is assumed to be independent Bernoulli, so that the total number of inliers is Binomial

$$p(f^{(1:n_f)}|m = 1) = B(n_i; n_f, p_1) \tag{3.7}$$

$$p(f^{(1:n_f)}|m = 0) = B(n_i; n_f, p_0) \tag{3.8}$$

where p_1 is the probability a feature is an inlier given a correct image match, and p_0 is the probability a feature is an inlier given a false image match. The number of inliers $n_i = \sum_{i=1}^{n_f} f^{(i)}$ and $B(.)$ is the Binomial distribution

$$B(x; n, p) = \frac{n!}{x!(n-x)!} p^x (1-p)^{n-x} \tag{3.9}$$

We choose values $p_1 = 0.6$ and $p_0 = 0.1$. We can now evaluate the posterior probability that an image match is correct using Bayes' Rule

$$p(m=1|f^{(1:n_f)}) = \frac{p(f^{(1:n_f)}|m=1)p(m=1)}{p(f^{(1:n_f)})} \quad (3.10)$$

$$= \frac{1}{1+\frac{p(f^{(1:n_f)}|m=0)p(m=0)}{p(f^{(1:n_f)}|m=1)p(m=1)}} \quad (3.11)$$

We accept an image match if $p(m=1|f^{(1:n_f)}) > p_{min}$

$$\frac{B(n_i;n_f,p_1)p(m=1)}{B(n_i;n_f,p_0)p(m=0)} \underset{reject}{\overset{accept}{\gtrless}} \frac{1}{\frac{1}{p_{min}}-1} \quad (3.12)$$

choosing values $p(m=1)=10^{-6}$ and $p_{min}=0.999$ gives the condition

$$n_i > \alpha + \beta n_f \quad (3.13)$$

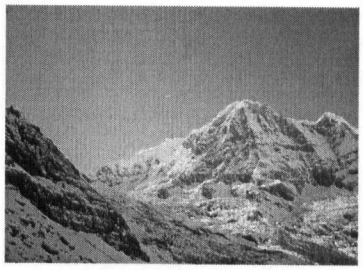
for a correct image match, where $\alpha = 8.0$ and $\beta = 0.3$. Though in practice we have chosen values for p_0, p_1, $p(m=0)$, $p(m=1)$ and p_{min}, they could in principle be learnt from the data. For example, p_1 could be estimated by computing the fraction of matches consistent with correct homographies over a large dataset.

Once pairwise matches have been established between images, we can find panoramic sequences as connected sets of matching images. This allows us to recognise multiple panoramas in a set of images, and reject noise images which match to no other images (see figure (3.2)).

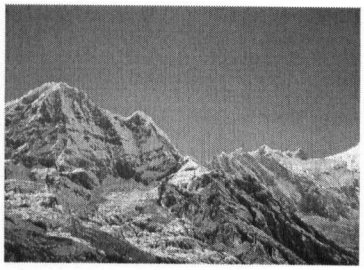
3.4 Bundle Adjustment

Given a set of geometrically consistent matches between the images, we use bundle adjustment [TMHF99] to solve for all of the camera parameters jointly. This is an essential step as concatenation of pairwise homographies would cause accumulated errors and disregard multiple constraints between images e.g. that the ends of a panorama should join up. Images are added to the bundle adjuster one by one, with the best matching image (maximum number of matches) being added at each step. The new image is initialised with the same rotation and focal length as the image to which it best matches. Then the parameters are updated using Levenberg-Marquardt.

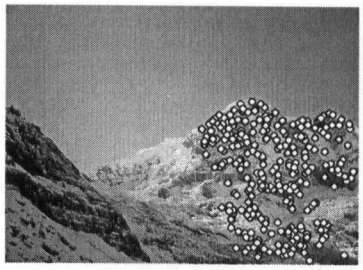
The objective function we use is a robustified sum squared projection error. That



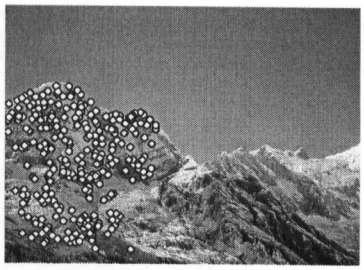
(a) Image 1



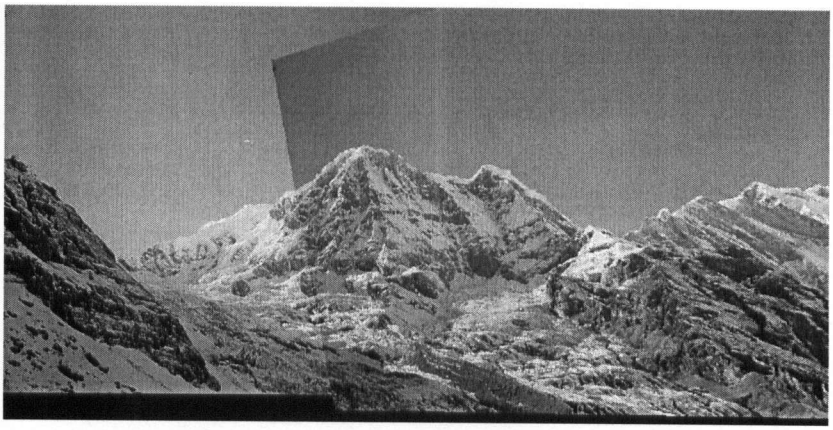
(b) Image 2



(c) SIFT matches 1

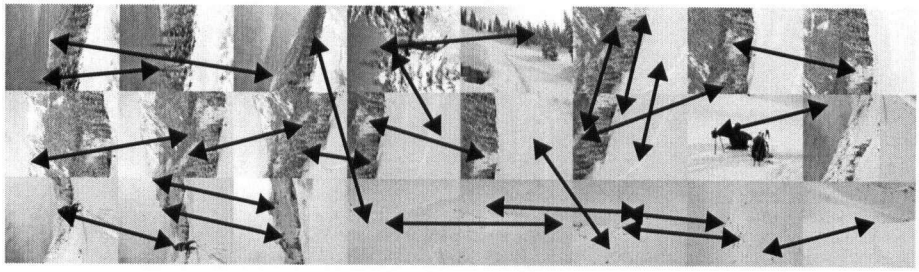


(d) SIFT matches 2

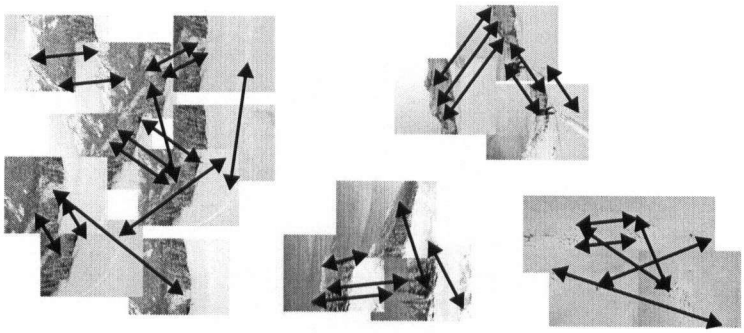


(e) Images aligned according to a homography

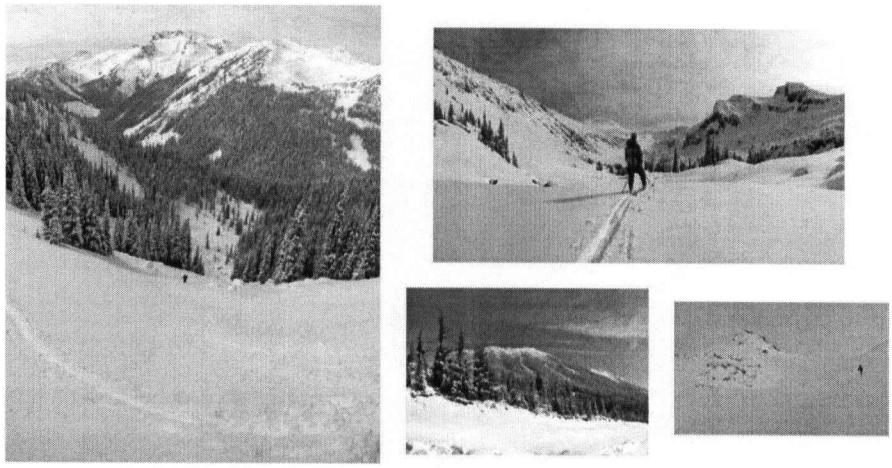
Figure 3.1: SIFT features are extracted from all of the images. After matching all of the features using a k-d tree, the m images with the greatest number of feature matches to a given image are checked for an image match. First RANSAC is performed to compute the homography, then a probabilistic model is invoked to verify the image match based on the number of inliers. In this example the input images are 517×374 pixels and there are 247 correct feature matches.



(a) All feature matches



(b) Geometrically consistent feature matches



(c) Output panoramas

Figure 3.2: Recognising panoramas. Given a noisy set of feature matches (a), we use RANSAC and a probabilistic verification procedure to find consistent image matches (b). Connected components of image matches are stitched into panoramas (c).

is, each feature is projected into all the images in which it matches, and the sum of squared image distances is minimised with respect to the camera parameters[1]. Given a correspondence $\mathbf{u}_i^k \leftrightarrow \mathbf{u}_j^l$ ($\mathbf{u}_i^k$ denotes the position of the kth feature in image i), the residual is

$$\mathbf{r}_{ij}^k = \mathbf{u}_i^k - \mathbf{p}_{ij}^k \tag{3.14}$$

where $\mathbf{p}_{ij}^k$ is the projection from image j to image i of the point corresponding to $\mathbf{u}_i^k$

$$\tilde{\mathbf{p}}_{ij}^k = \mathbf{K}_i \mathbf{R}_i \mathbf{R}_j^T \mathbf{K}_j^{-1} \tilde{\mathbf{u}}_j^l \tag{3.15}$$

The error function is the sum over all images of the robustified residual errors

$$e = \sum_{i=1}^{n} \sum_{j \epsilon \mathcal{I}(i)} \sum_{k \epsilon \mathcal{F}(i,j)} f(\mathbf{r}_{ij}^k) \tag{3.16}$$

where n is the number of images, $\mathcal{I}(i)$ is the set of images matching to image i, $\mathcal{F}(i,j)$ is the set of feature matches between images i and j. We use a Huber robust error function

$$f(\mathbf{x}) = \begin{cases} |\mathbf{x}|^2, & \text{if } |\mathbf{x}| < \sigma \\ 2\sigma|\mathbf{x}| - \sigma^2, & \text{if } |\mathbf{x}| \geq \sigma \end{cases} \tag{3.17}$$

This error function combines the fast convergence properties of an L_2 norm optimisation scheme for inliers (distance less than σ), with the robustness of an L_1 norm scheme for outliers (distance greater than σ). We use an outlier distance $\sigma = \infty$ during initialisation and $\sigma = 2$ pixels for the final solution.

This is a non-linear least squares problem which we solve using the Levenberg-Marquardt algorithm. Each iteration step is of the form

$$\mathbf{\Theta} = (\mathbf{J}^T\mathbf{J} + \lambda \mathbf{C}_p^{-1})^{-1} \mathbf{J}^T \mathbf{r} \tag{3.18}$$

where $\mathbf{\Theta}$ are all the parameters, $\mathbf{r}$ the residuals and $\mathbf{J} = \partial \mathbf{r} / \partial \mathbf{\Theta}$. We encode our prior

[1]Note that it would also be possible (and in fact statistically optimal) to represent the unknown ray directions $\mathbf{X}$ explicitly, and to estimate them jointly with the camera parameters. This would not increase the complexity of the algorithm if a sparse bundle adjustment method was used (as in section 5.4.1). This is left for future work.

belief about the parameter changes in the (diagonal) covariance matrix $\mathbf{C}_p$

$$\mathbf{C}_p = \begin{bmatrix} \sigma_\theta^2 & 0 & 0 & 0 & 0 & \dots \\ 0 & \sigma_\theta^2 & 0 & 0 & 0 & \dots \\ 0 & 0 & \sigma_\theta^2 & 0 & 0 & \dots \\ 0 & 0 & 0 & \sigma_f^2 & 0 & \dots \\ 0 & 0 & 0 & 0 & \sigma_\theta^2 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \end{bmatrix} \tag{3.19}$$

This is set such that the standard deviation of angles is $\sigma_\theta = \pi/16$ and focal lengths $\sigma_f = \bar{f}/10$ (where $\bar{f}$ is the mean of the focal lengths estimated so far). This helps in choosing suitable step sizes, and hence speeding up convergence. For example, if a spherical covariance matrix were used, a change of 1 radian in rotation would be equally penalised as a change of 1 pixel in focal length. Finally, the λ parameter is varied at each iteration to ensure that the objective function of equation 3.16 does in fact decrease.

The derivatives are computed analytically via the chain rule, for example

$$\frac{\partial \mathbf{p}_{ij}^k}{\partial \theta_{i1}} = \frac{\partial \mathbf{p}_{ij}^k}{\partial \tilde{\mathbf{p}}_{ij}^k} \frac{\partial \tilde{\mathbf{p}}_{ij}^k}{\partial \theta_{i1}} \tag{3.20}$$

where

$$\frac{\partial \mathbf{p}_{ij}^k}{\partial \tilde{\mathbf{p}}_{ij}^k} = \frac{\partial \begin{bmatrix} x/z & y/z \end{bmatrix}}{\partial \begin{bmatrix} x & y & z \end{bmatrix}} = \begin{bmatrix} 1/z & 0 & -x/z^2 \\ 0 & 1/z & -y/z^2 \end{bmatrix} \tag{3.21}$$

and

$$\frac{\partial \tilde{\mathbf{p}}_{ij}^k}{\partial \theta_{i1}} = \mathbf{K}_i \frac{\partial \mathbf{R}_i}{\partial \theta_{i1}} \mathbf{R}_j \mathbf{K}_j^{-1} \tilde{\mathbf{u}}_j^l \tag{3.22}$$

$$\frac{\partial \mathbf{R}_i}{\partial \theta_{i1}} = \frac{\partial}{\partial \theta_{i1}} e^{[\boldsymbol{\theta}_i]_\times} = e^{[\boldsymbol{\theta}_i]_\times} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \tag{3.23}$$

3.4.1 Fast Solution by Direct Computation of the Linear System

Since the matrix $\mathbf{J}$ is sparse, forming $\mathbf{J}^T\mathbf{J}$ by explicitly multiplying $\mathbf{J}$ by its transpose is inefficient. In fact, this would be the most expensive step in bundle adjustment, costing $O(MN^2)$ for an $M \times N$ matrix $\mathbf{J}$. The sparseness arises because each image typically only matches to a small subset of the other images. This means that in practice each element of $\mathbf{J}^T\mathbf{J}$ can be computed in much fewer than M multiplications

$$(\mathbf{J}^T\mathbf{J})_{ij} = \sum_{k\epsilon\mathcal{F}(i,j)} \frac{\partial \mathbf{r}_{ij}^k}{\partial \Theta_i}^T \frac{\partial \mathbf{r}_{ij}^k}{\partial \Theta_j} = \mathbf{C}_{\boldsymbol{\Theta}}^{-1} \tag{3.24}$$

i.e. the inverse covariance between cameras i and j depends only on the residuals of feature matches between i and j.

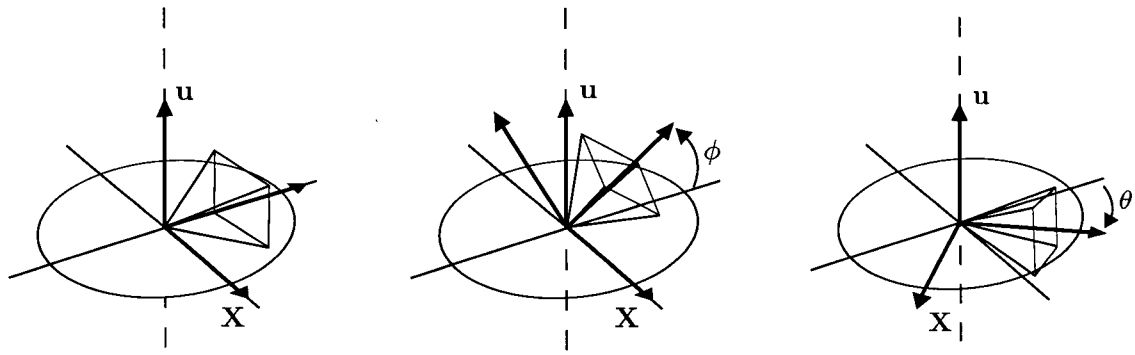
Similarly, $\mathbf{J}^T\mathbf{r}$ need not be computed explicitly, but can be computed via

$$(\mathbf{J}^T\mathbf{r})_i = \sum_{i=1}^{n} \sum_{j\epsilon\mathcal{I}(i)} \sum_{k\epsilon\mathcal{F}(i,j)} \frac{\partial \mathbf{r}_{ij}^k}{\partial \Theta_i}^T \mathbf{r}_{ij}^k \tag{3.25}$$

In both cases each summation would require M multiplications if each feature matched to every single image, but in practice the number of feature matches for a given image is much less than this.

3.5 Automatic Panorama Straightening

Image registration using the steps of sections 3.2 - 3.4 gives the relative rotations between the cameras, but there remains an unknown 3D rotation to a chosen world coordinate frame. If we simply assume that $\mathbf{R} = \mathbf{I}$ for one of the images, we typically find a wavy effect in the output panorama. This is because the real camera was unlikely to be perfectly level and un-tilted. We can correct this wavy output and automatically straighten the panorama by making use of a heuristic about the way people typically shoot panoramic images. The idea is that it is rare for people to *twist* the camera relative to the horizon, so the camera $\mathbf{X}$ vectors typically lie in a plane. By finding the null vector of the covariance matrix of the camera $\mathbf{X}$ vectors, we can find the



(a) (b) (c)

Figure 3.3: Finding the up-vector **u**. A good heuristic to align wavy panoramas is to note that people rarely *twist* the camera relative to the horizon. Hence despite tilt (b) and rotation (c), the camera **X** vectors typically lie in a plane. The up-vector **u** (opposite to the direction of gravity) is the vector normal to this plane.

"up-vector" **u** (normal to the plane containing the camera centre and the horizon)

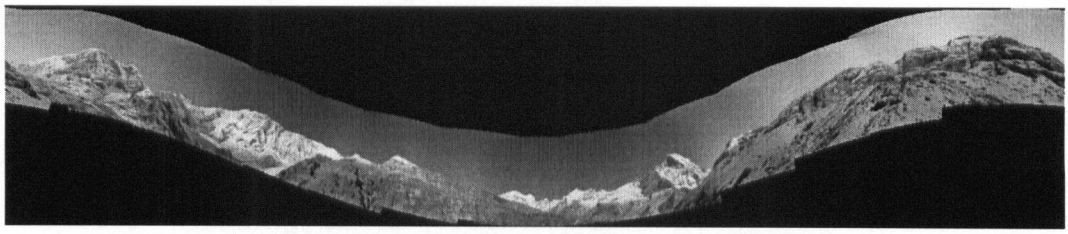
$$\left(\sum_{i=1}^{n} \mathbf{X}_i \mathbf{X}_i^T\right) \mathbf{u} = \mathbf{0} \tag{3.26}$$

Applying a global rotation such that up-vector **u** is vertical (in the rendering frame) effectively removes the wavy effect from output panoramas as shown in figure 3.4.

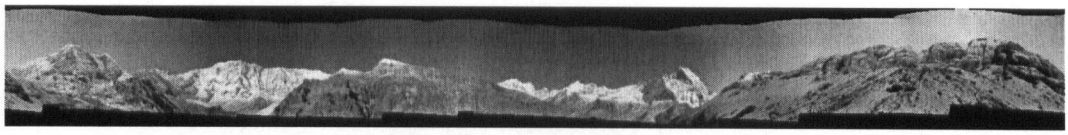
3.6 Gain Compensation

In previous sections, we described a method for computing the geometric parameters (orientation and focal length) of each camera. In this section, we show how to solve for a photometric parameter, namely the overall gain between images. This is set up in a similar manner, with an error function defined over all images. The error function is the sum of gain normalised intensity errors for all overlapping pixels

$$e = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\substack{\mathbf{u}_i \epsilon \mathcal{R}(i,j) \\ \tilde{\mathbf{u}}_i \sim \mathbf{H}_{ij} \tilde{\mathbf{u}}_j}} (g_i I_i(\mathbf{u}_i) - g_j I_j(\mathbf{u}_j))^2 \tag{3.27}$$



(a) Without automatic straightening



(b) With automatic straightening

Figure 3.4: Automatic panorama straightening. Using the heuristic that users rarely twist the camera relative to the horizon allows us to straighten wavy panoramas by computing the up-vector (perpendicular to the plane containing the horizon and the camera centre).

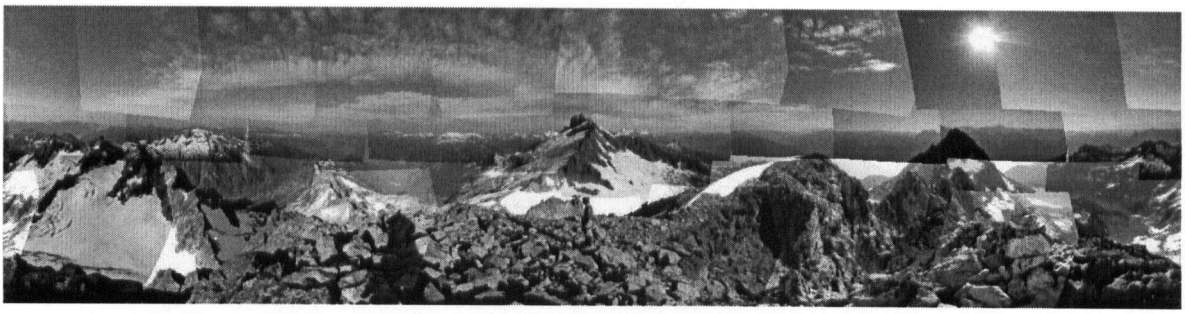
where g_i, g_j are the gains, and $\mathcal{R}(i,j)$ is the region of overlap between images i and j. In practice we approximate $I(\mathbf{u}_i)$ by the mean in each overlapping region $\bar{I}_{ij}$

$$\bar{I}_{ij} = \frac{\sum_{\mathbf{u}_i \epsilon \mathcal{R}(i,j)} I_i(\mathbf{u}_i)}{\sum_{\mathbf{u}_i \epsilon \mathcal{R}(i,j)} 1} \tag{3.28}$$

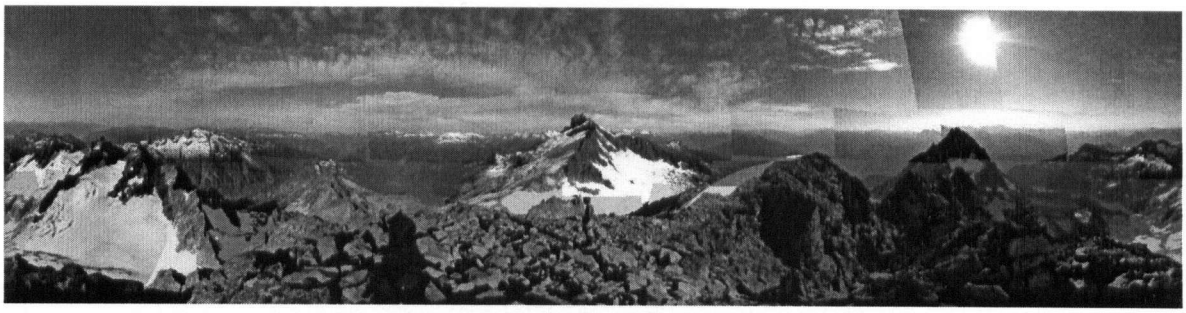
This simplifies the computation and gives some robustness to outliers, which might arise due to small misregistrations between the images. Also, since $\mathbf{g} = 0$ is an optimal solution to the problem, we add a prior term to keep the gains close to unity. Hence the error function becomes

$$e = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} N_{ij} \left((g_i \bar{I}_{ij} - g_j \bar{I}_{ji})^2 / \sigma_N^2 + (1 - g_i)^2 / \sigma_g^2 \right) \tag{3.29}$$

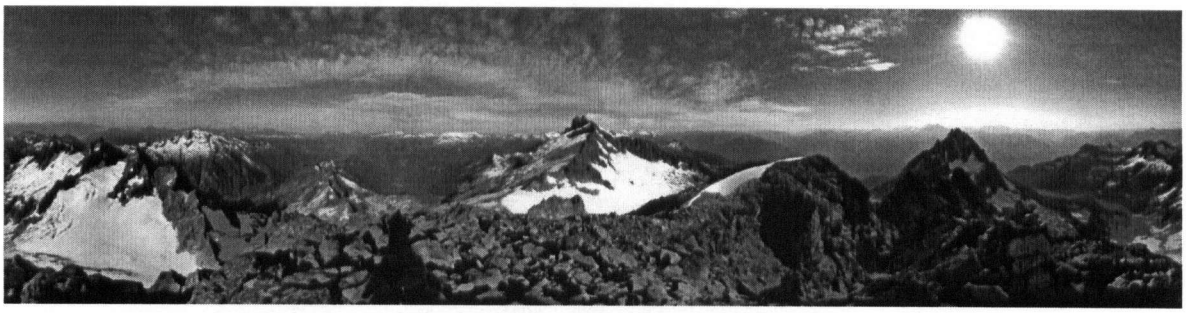
where $N_{ij} = |\mathcal{R}(i,j)|$ equals the number of pixels in image i that overlap in image j. The parameters σ_N and σ_g are the standard deviations of the normalised intensity error and gain respectively. We choose values $\sigma_N = 10.0$, $(I \epsilon \{0..255\})$ and $\sigma_g = 0.1$. This is a quadratic objective function in the gain parameters $\mathbf{g}$ which can be solved in closed form by setting the derivative to 0 (see figure 3.5).



(a) Without gain compensation



(b) With gain compensation



(c) With gain compensation and multi-band blending

Figure 3.5: Gain compensation. Note that large changes in brightness between the images are visible if gain compensation is not applied (a). After gain compensation, some image edges are still visible due to unmodelled effects such as vignetting (b). These can be effectively smoothed out using multi-band blending (c).

3.7 Multi-Band Blending

Ideally each sample (pixel) along a ray would have the same intensity in every image that it intersects, but in reality this is not the case. Even after gain compensation some image edges are still visible due to a number of unmodelled effects, such as vignetting (intensity decreases towards the edge of the image), parallax effects due to unwanted motion of the optical centre, mis-registration errors due to mis-modelling of the camera, radial distortion and so on. Because of this a good blending strategy is important.

From the previous steps we have n images $I^i(x,y)$ $(i \in \{1..n\})$ which, given the known registration, may be expressed in a common (spherical) coordinate system as $I^i(\theta,\phi)$. In order to combine information from multiple images we assign a weight function to each image $W(x,y) = w(x)w(y)$ where $w(x)$ varies linearly from 1 at the centre of the image to 0 at the edge. The weight functions are also resampled in spherical coordinates $W^i(\theta,\phi)$. A simple approach to blending is to perform a weighted sum of the image intensities along each ray using these weight functions

$$I^{linear}(\theta,\phi) = \frac{\sum_{i=1}^{n} I^i(\theta,\phi)W^i(\theta,\phi)}{\sum_{i=1}^{n} W^i(\theta,\phi)} \tag{3.30}$$

where $I^{linear}(\theta,\phi)$ is a composite spherical image formed using linear blending. However, this approach can cause blurring of high frequency detail if there are small registration errors (see figure 3.9). To prevent this we use the multi-band blending algorithm of Burt and Adelson [BA83]. The idea behind multi-band blending is to blend low frequencies over a large spatial range, and high frequencies over a short range.

We initialise blending weights for each image by finding the set of points for which image i is most responsible

$$W^i_{max}(\theta,\phi) = \begin{cases} 1 & \text{if } W^i(\theta,\phi) = \arg\max_j W^j(\theta,\phi) \\ 0 & \text{otherwise} \end{cases} \tag{3.31}$$

i.e. $W^i_{max}(\theta,\phi)$ is 1 for (θ,ϕ) values where image i has maximum weight, and 0 where some other image has a higher weight (see figure 3.6). These max-weight maps are successively blurred to form the blending weights for each band.

A high pass version of the rendered image is formed

$$I^i_\sigma(\theta,\phi) = I^i(\theta,\phi) - I^i(\theta,\phi) * g_\sigma(\theta,\phi) \tag{3.32}$$

where and $g_\sigma(\theta,\phi)$ is a Gaussian of standard deviation σ, and $I_\sigma(\theta,\phi)$ represents spatial frequencies in the range of wavelengths $\lambda = 0 \rightarrow \sigma$. We blend this band between images using a blending weight formed by blurring the max-weight map for this image

$$B^i_\sigma(\theta,\phi) = W^i_{max}(\theta,\phi) * g_\sigma(\theta,\phi) \tag{3.33}$$

where $B^i_\sigma(\theta,\phi)$ is the blend weight for the wavelength $0 \rightarrow \sigma$ band. Subsequent frequency bands are blended by forming lower frequency bandpass images and further blurring the blend weights, i.e. for $k \geq 1$

$$I^i_{(k+1)\sigma} = I^i_{k\sigma} - I^i_{k\sigma} * g_\sigma \tag{3.34}$$

$$B^i_{(k+1)\sigma} = B^i_{k\sigma} * g_\sigma \tag{3.35}$$

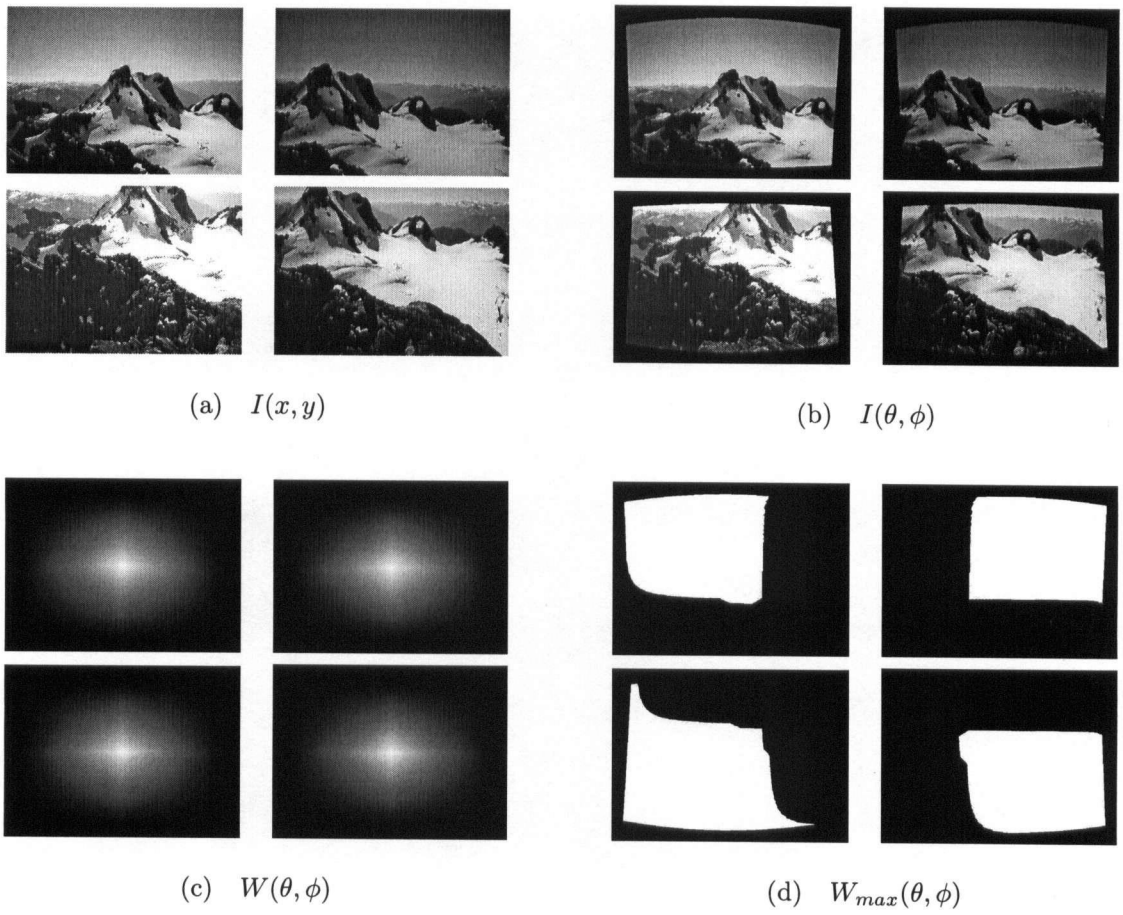
For each band, overlapping images are linearly combined using the corresponding blend weights

$$I^{multi}_{k\sigma}(\theta,\phi) = \frac{\sum_{i=1}^n I^i_{k\sigma}(\theta,\phi) B^i_{k\sigma}(\theta,\phi)}{\sum_{i=1}^n B^i_{k\sigma}(\theta,\phi)} \tag{3.36}$$

This causes high frequency bands (small $k\sigma$) to be blended over short ranges whilst low frequency bands (large $k\sigma$) are blended over larger ranges (see figure (3.7)).

Note that we have chosen to render the panorama in spherical coordinates θ, ϕ. In principle one could choose any 2-dimensional parameterisation of a surface around the viewpoint for rendering. One good choice would be to render to a triangulated sphere, constructing the blending weights in the image plane. This would have the advantage of uniform treatment of all images, and it would also allow easy resampling to other surfaces (in graphics hardware).

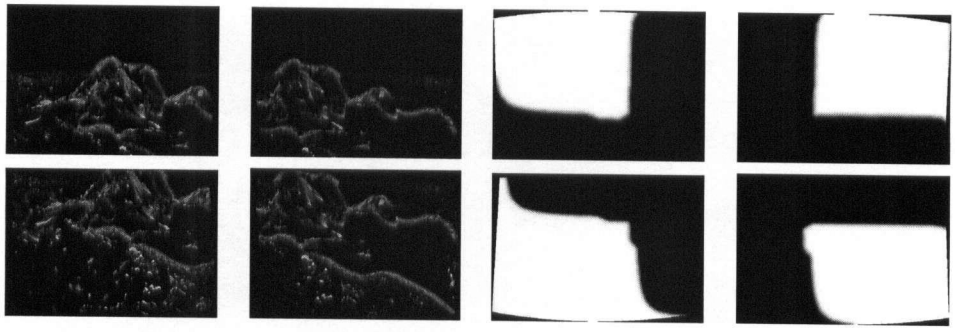
An assumption of our blending strategy is that the image whose centre is closest to a given pixel in the rendering has the 'best' information about that pixel, and therefore the highest blending weight. This might not always be the case, for example, if the image set contained defocussed or otherwise degraded images. In this scenario one might want to select blending weights based on some other criterion e.g., sharpness of the image.



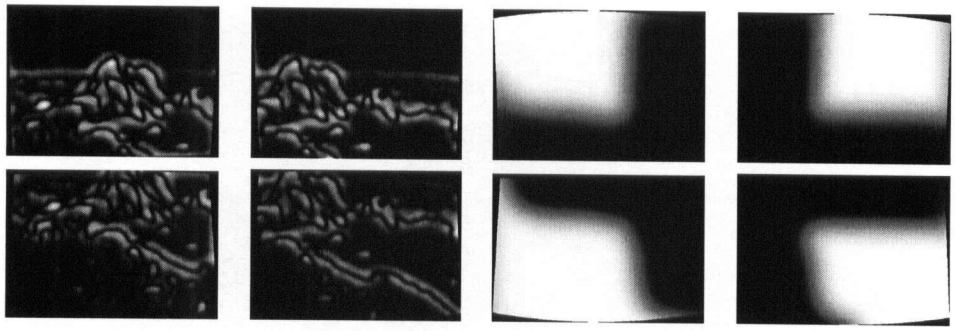
(a) $I(x, y)$ (b) $I(\theta, \phi)$

(c) $W(\theta, \phi)$ (d) $W_{max}(\theta, \phi)$

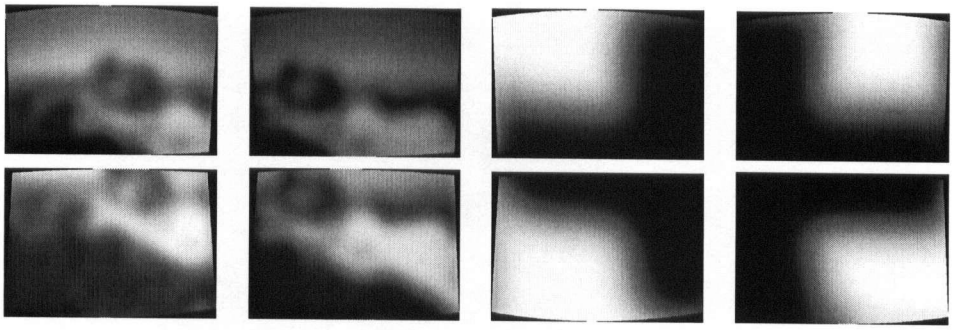
Figure 3.6: Images $I(x, y)$ and weights $W(x, y)$ are resampled in spherical coordinates $I(\theta, \phi)$, $W(\theta, \phi)$ and max weight maps $W_{max}(\theta, \phi)$ are computed. In linear blending images are combined as linear sums using the weights $W(\theta, \phi)$. In multi-band blending blurred versions of the max weight map are used to form separate blending functions for each frequency band.



(a) Band 1 (scale 0 to σ)

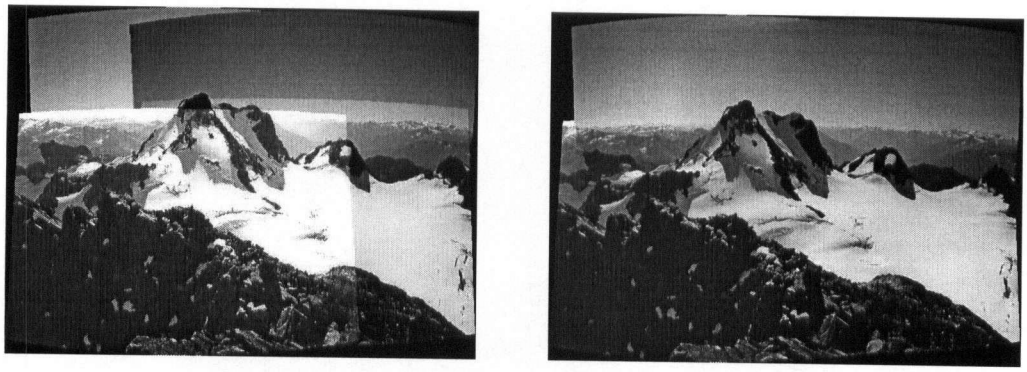


(b) Band 2 (scale σ to 2σ)



(c) Band 3 (scale lower than 2σ)

Figure 3.7: Multi-band blending. Bandpass images $I_{k\sigma}(\theta, \phi)$ for $k = 1, 2, 3$ are shown on the left, with the corresponding blending weights $B_{k\sigma}(\theta, \phi)$ shown on the right. Initial blending weights are assigned to 1 where each image has maximum weight. To obtain each blending function, the weights are blurred at spatial frequency σ and bandpass images of the same spatial frequency are formed. The bandpass images are blended together using weighted sums based on the blending weights (Note: the contrast has been enhanced and blending widths exaggerated for clarity in these figures).



(a) No blending (b) Multi-band blending

Figure 3.8: Results for panorama rendering with and without multi-band blending. See figure 3.7 for the blending functions used for these images. In this case multi-band blending smooths out exposure differences and vignetting between the images. For an example of dealing with with misregistrations, see figure 3.9.

(a) Linear blending (b) Multi-band blending

Figure 3.9: Comparison of linear and multi-band blending. The image on the right was blended using multi-band blending using 5 bands and $\sigma = 5$ pixels. The image on the left was linearly blended. In this case matches on the moving person have caused small misregistrations between the images, which cause blurring in the linearly blended result, but multi-band blended image is clear.

Algorithm: Panoramic Recognition

Input: n unordered images

I. Extract SIFT features from all n images

II. Find k nearest-neighbours for each feature using a k-d tree

III. For each image:
   (i) Select m candidate matching images that have the most feature matches to this image
   (ii) Find geometrically consistent feature matches using RANSAC to solve for the homography between pairs of images
   (iii) Verify image matches using a probabilistic model

IV. Find connected components of image matches

V. For each connected component:
   (i) Perform bundle adjustment to solve for the rotation $\theta_1, \theta_2, \theta_3$ and focal length f of all cameras
   (ii) Render panorama using multi-band blending

Output: Panoramic image(s)

3.8 Results

Figure 3.10 shows typical operation of the panoramic recognition algorithm. A set of images containing 2 panoramas and 5 noise images was input. The algorithm detected 2 connected components of image matches and 5 unmatched images, and output the 2 blended panoramas. The complete algorithm ran in 83 seconds on a 2GHz PC, with input images 525 × 375 pixels ($7'' \times 5''$ prints scanned at 75 dpi), and rendering the larger output panorama as a 300 × 3000 pixel image. The majority of computation time is spent in extracting the SIFT features from the images.

Figure 3.11 shows a larger example where 80 images were used to create a 360° × 90° panorama. No user input is required: the object recognition system decides which images match, and the bundle adjustment algorithm optimises jointly for the

$4 \times 80 = 320$ parameters of all the cameras. Finally, the multi-band blending scheme effectively hides the seams despite the illumination changes (camera flash[2] and change in aperture/exposure). We have tested the system on many other image sets, for example full $360° \times 180°$ panoramas, and sequences where different cameras are used in the same panorama.

3.9 Summary

This chapter has presented a novel system for fully automatic panorama stitching. Our use of invariant local features and a probabilistic model to verify image matches allows us recognise multiple panoramas in unordered image sets, and stitch them fully automatically without user input. The system is robust to camera zoom, orientation of the input images, and changes in illumination due to flash and exposure/aperture settings. A multi-band blending scheme ensures smooth transitions between images despite illumination differences, whilst preserving high frequency details.

Future Work

Possible areas for future work include compensation for motion in the camera and scene, and more advanced modelling of the geometric and photometric properties of the camera:

Camera Motion Panoramas often suffer from parallax errors due to small motions of the optical centre. These could be removed by solving for camera translations and depths in the scene, before re-rendering from a central point. A good representation to use might plane at infinity plus parallax [RC02] (appendix A.6). Whilst gross camera motions cause parallax artifacts, small motions during shooting result in motion blur. Motion blurred images could be deblurred using nearby in-focus images as in [BBZ96]. Similar techniques can also be used to generate super-resolution images [CZ98].

Scene Motion Though our multi-band blending strategy works well in many cases, large motions of objects in the scene cause visible artifacts when blending between

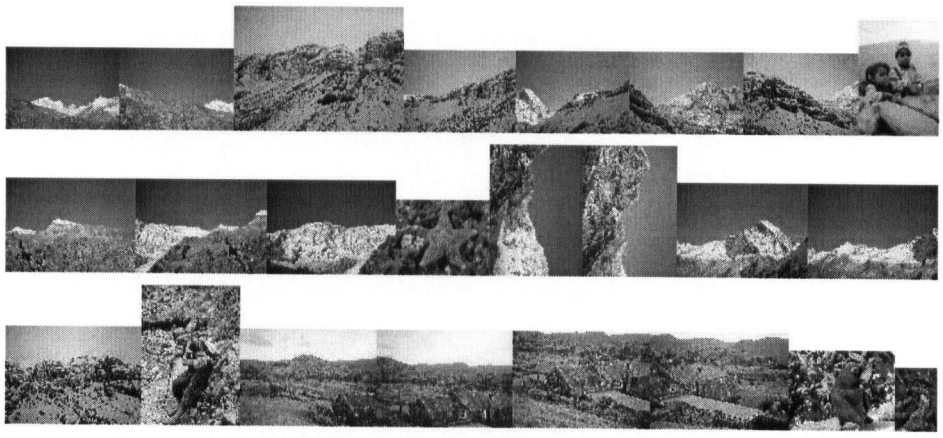
[2]Note that in this case the camera flash caused significant illumination changes between images of the grass in the lower part of the panorama. In some of the grass images the flash fired whilst in others it did not.

multiple images. Another approach would be to automatically find optimal seam lines based on regions of difference between the images [Dav98, UES01, ADA+04].

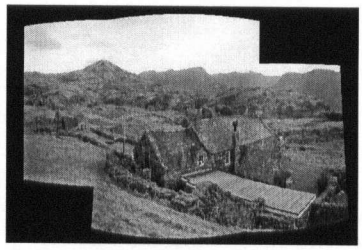
Advanced Camera Modelling An important characteristic of most cameras that is not modelled by the projective camera model (which preserves straight lines) is radial distortion [Bro71]. This can be accurately modelled by low order polynomials, the parameters of which could be included in the bundle adjustment framework. The ideal image stitcher would also support multiple motion models, for example, rotation about a point (e.g. panoramas), viewing a plane (e.g. whiteboards) and Euclidean transforms (e.g. aligning scanned images). One could also render to multiple surface types e.g. spherical, cylindrical, planar. Further geometric corrections could also be applied using local warping as in [SS00].

Photometric Modelling In principle it should also be possible to estimate many of the photometric parameters of the camera. Vignetting (decrease in intensity towards image edges) is a common source of artifacts, particularly in uniform colour regions such as sky [LS05]. One could also acquire high-dynamic range [DM97, SHS+] information from the overlapping image regions, and render tone mapped or synthetic exposure images.

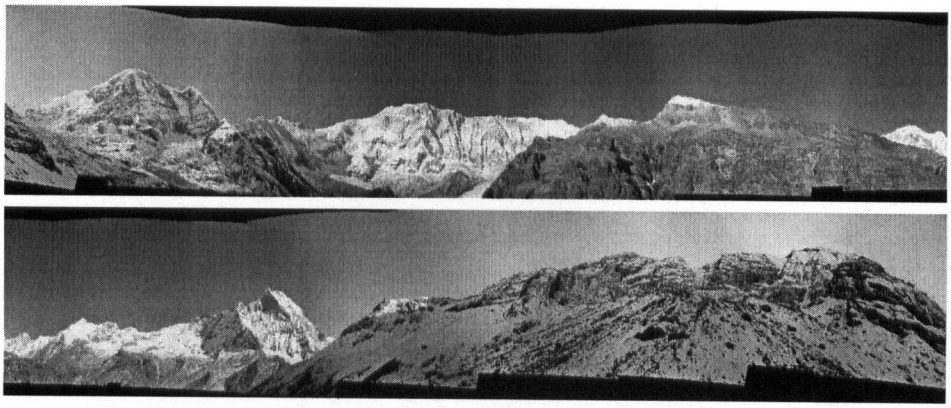
We have developed a C++ implementation of the algorithm described in this chapter, called AutoStitch. A demo of this program can be downloaded from the website at http://www.autostitch.net.



(a) Input images



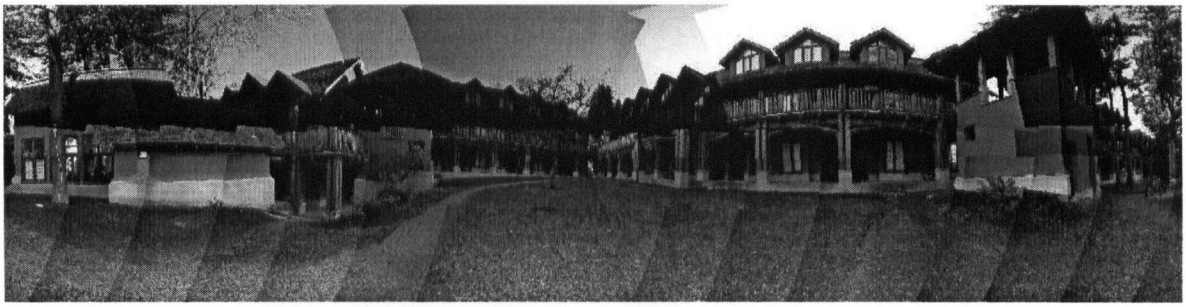
(b) Output panorama 1



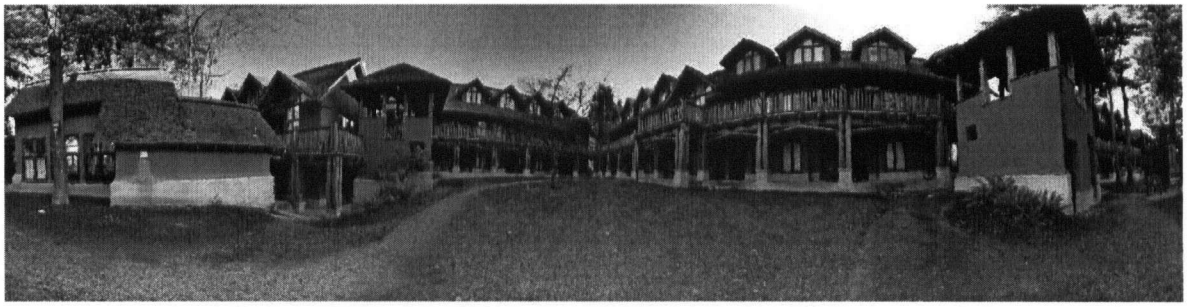
(c) Output panorama 2

Figure 3.10: Typical operation of the panoramic recognition algorithm: an image set containing multiple panoramas and noise images is input, and panoramic sequences are recognised and rendered as output. The algorithm is insensitive to the ordering, scale and orientation of the images. It is also insensitive to noise images which are not part of a panorama.

(a) 40 of 80 images registered



(b) All 80 images registered



(c) Rendered with multi-band blending

Figure 3.11: Green College. This sequence was shot using the camera's automatic mode, which allowed the aperture and exposure time to vary, and the flash to fire on some images. Despite these changes in illumination, the SIFT features match robustly and the multi-band blending strategy yields a seamless panorama. These $360° \times 90°$ images have been rendered in spherical coordinates (θ, ϕ). The sequence consisted of 80 images all of which were matched fully automatically with no user input, and a $4 \times 80 = 320$ parameter optimisation problem was solved for the final registration. With 400×300 pixel input images and a 500×2000 pixel output panorama, the whole process ran in 197 seconds on a 2GHz PC.

Chapter 4

Evaluation of Image Stitching Algorithms

4.1 Introduction

In the previous chapters we developed apparatus for fully automatic multi-image registration. This chapter develops a framework for evaluation and performance tuning of such multi-image matching methods. Algorithms for stitching multiple images into seamless photomosaics have been used in satellite photography and digital mapping for decades [Mil75]. Early approaches involved much user input and specialised hardware [Sla80, Mee90]. Computer vision techniques have brought increasing automation to the problem, from automatic pairwise alignment [BAHH92, IA99] to bundle adjustment [SK99, SS00], and culminating in systems that recognise matches and stitch images with no user input whatsoever [BL03, BSW05]. As more approaches are proposed [ZFD97, CZ98, SK99, SS00, BL03] it becomes increasingly important to form comparisons and elicit best practices.

There has been excellent comparison work in the related areas of stereo [SS02] and feature matching [MS03]. However, knowing the performance of individual feature matching techniques is not sufficient to predict overall stitching performance, which involves global decisions as to which image pairs truly match and global optimization for registration. Relying on visual inspection of stitching results to assess algorithm performance is tedious and becomes unworkable for large test sets. Instead we develop a database of image sequences for which the ground truth registration is known, and a metric for quantative evaluation of stitching results. This allows us to form comparisons between different image stitching methods, and to tune the parameters of these increasingly complex algorithms on real world data.

4.1.1 Image Stitching Algorithms

In this chapter we focus on multi-image matching problems in which there is a one-to-one correspondence between the images. In particular we are interested in problems where images are related by a linear transformation in 2D projective space, so that each camera can be characterised by a 3×3 projection matrix $\mathbf{P}$. This generalises most image stitching problems, for example panoramas (rotating camera), whiteboards (moving camera, planar scene) and flatbed scanning. It does not include cases where the camera is free to move in 3-dimensions (and the scene is non-planar), or cases where the objects in the scene are in motion.

4.2 Evaluation of Image Stitching Algorithms

For the purposes of this chapter, we concentrate on datasets with only a single image sequence to be stitched, although outlier images which are not part of the sequence may be present. Given a set of gold standard projection matrices

$$\mathbf{P}^* = \{\mathbf{P}_1^*, \mathbf{P}_2^*, ...\mathbf{P}_n^*\} \tag{4.1}$$

and a test set of image stitching parameters

$$\mathbf{P} = \{\mathbf{P}_1, \mathbf{P}_2, ...\mathbf{P}_n\} \tag{4.2}$$

we wish to write an evaluation function

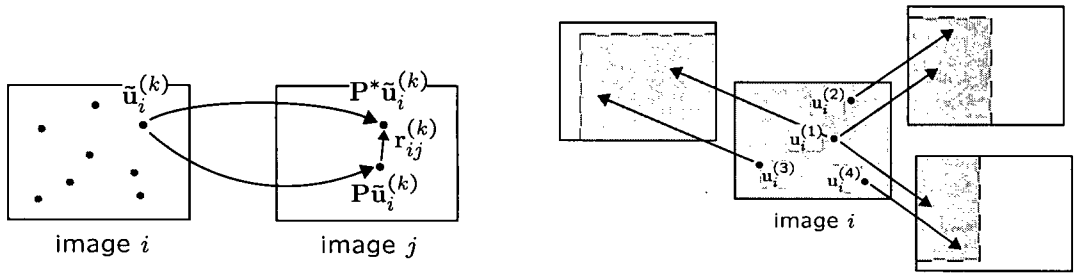
$$e = e(\mathbf{P}^*, \mathbf{P}) \tag{4.3}$$

to evaluate the test set against the gold standard.

Since we are only interested in the point mapping between images

$$\mathbf{P}_{ij} = \mathbf{P}_i \mathbf{P}_j^{-1} \tag{4.4}$$

the $\mathbf{P}$ matrices may be post-multiplied by an arbitrary (invertible) 3×3 matrix, and still yield the same stitching results. One could attempt to recover this matrix and compute distances in the space of projection matrices. However, we have found that different sets of camera parameters can often give quite similar stitching results (see



(a) (b)

Figure 4.1: Evaluation function for multi-image matching. Uniformly distributed points $\mathbf{u}_i^{(k)}$ are projected from image i to each matching image j. The residual $\mathbf{r}_{ij}^{(k)}$ between the projections of the point under the gold standard and test homographies is computed (a). The error function is the root mean square of all such residuals. Random points are generated for all images i, and residuals are summed for all images that overlap image i (b).

figures 4.3 and 4.4). Also, for stitching problems, we are generally most concerned with errors in the image plane, since these lead to visible artifacts in the stitching results. Hence, we propose an evaluation function based on the sum squared projection error of random image points, relative to the gold standard

$$e(\mathbf{P}^*, \mathbf{P}) = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{\substack{k=1 \\ \mathbf{u}_i^{(k)} \epsilon \mathcal{O}(i,j)}}^{N} |\mathbf{r}_{ij}^{(k)}|^2 \tag{4.5}$$

where the residual $\mathbf{r}_{ij}^{(k)}$ is the difference between the projections of the kth random point from image i to image j under the gold standard and test homographies

$$\mathbf{r}_{ij}^{(k)} = \mathbf{u}_j^{(k)} - \mathbf{u}_j^{*(k)} \tag{4.6}$$

$$\tilde{\mathbf{u}}_j^{(k)} = \mathbf{P}_{ij}\tilde{\mathbf{u}}_i^{(k)} \tag{4.7}$$

$$\tilde{\mathbf{u}}_j^{*(k)} = \mathbf{P}_{ij}^*\tilde{\mathbf{u}}_i^{(k)} \tag{4.8}$$

$\mathbf{u}_i^{(k)}$ is a random point uniformly distributed in image i, and $\mathbf{P}_{ij}^*$ and $\mathbf{P}_{ij}$ are the gold standard and test homographies from image i to image j. See figure 4.1. The third summation in equation 4.5 is over all randomly generated points $\mathbf{u}_i^{(k)}$ that success-

fully project inside image j under either the gold standard or test homographies i.e. $\mathbf{u}_i^{(k)} \epsilon \mathcal{O}(i,j)$ iff $\mathbf{u}_j^{(k)} \epsilon \mathcal{I}(j)$ or $\mathbf{u}_j^{*(k)} \epsilon \mathcal{I}(j)$ where $\mathcal{I}(j)$ denotes the set of points in image j.

In practice we divide by the number of residuals and take the square root to give the RMS projection error over all images

$$e_{RMS}(\mathbf{P}, \mathbf{P}^*) = \left(\frac{e(\mathbf{P}, \mathbf{P}^*)}{\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{\mathbf{u}_i^{(k)} \epsilon \mathcal{O}(i,j)} 1} \right)^{0.5} \tag{4.9}$$

We generate a fixed number of random samples $N = 100$ in each image.

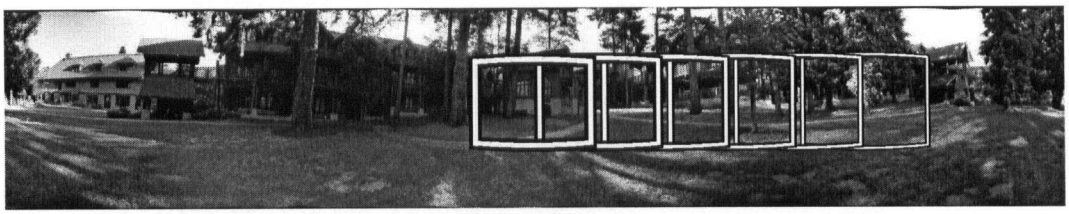
4.2.1 Dealing with Failed Matches

Suppose that the image dataset contains some outlier images which do not belong to the panorama. In this case it will be possible for an image stitcher to generate false positives (images that are incorrectly matched, and do not match in the gold standard) and false negatives (images that fail to match, and are matched in the gold standard). We would also like to label gross registration errors as failed matches.

In order to differentiate minor registration errors from failed matches we compute the summation of equation 4.5 only over image pairs whose RMS errors are less than a threshold

$$e_{RMS}^{ij} = \frac{\sum_{k=1, \mathbf{u}_i^{(k)} \epsilon \mathcal{O}(i,j)}^{N} |\mathbf{r}_{ij}^{(k)}|^2}{\sum_{k=1, \mathbf{u}_i^{(k)} \epsilon \mathcal{O}(i,j)}^{N} 1} < r_{max}^2 \tag{4.10}$$

Furthermore, we label an image i as a failed match if it is (a) a false positive, (b) a false negative, or (c) belongs to a pair whose RMS error is above the threshold r_{max}. Hence, the overall evaluation of an image stitching result should be based on the RMS projection error of all correctly matching images e_{RMS} (pixels), and the number of images which fail to match n_F. Note that we count the number of *images* that fail to match correctly, instead of counting the number of failed pairwise matches, so the number of failed matches $0 < n_F < n$ (where n is the number of images in the dataset).

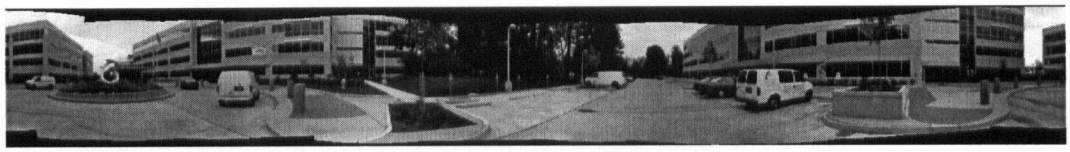


(a) Initial panorama

(b) Resampled camera views

Figure 4.2: Generating synthetic ground truth. To generate synthetic ground truth data we first use our existing stitchers to generate a panorama (a). Next, we resample virtual camera views from this panorama, for which the camera matrices are known exactly (b).

(a) Without 360° wrap-around



(b) With 360° wrap-around

Figure 4.3: Panorama stitching with and without 360° wrap-around. In both cases an optimal algorithm (bundle adjustment) has been used to solve for the camera parameters. However, in (a) matching between the ends of the panorama has been suppressed, whereas in (b) matching between the ends is allowed. Note that although the stitching is accurate (RMS pixel errors are small) in both cases, the focal length estimate in (a) is out by almost 20% (520 pixels in (a) compared to 630 pixels in (b)). These results demonstrate that it is difficult to solve for focal length (even using an optimal bundle adjustment algorithm) unless 360 wrap-around is achieved (Cedar Court sequence, real image dataset).

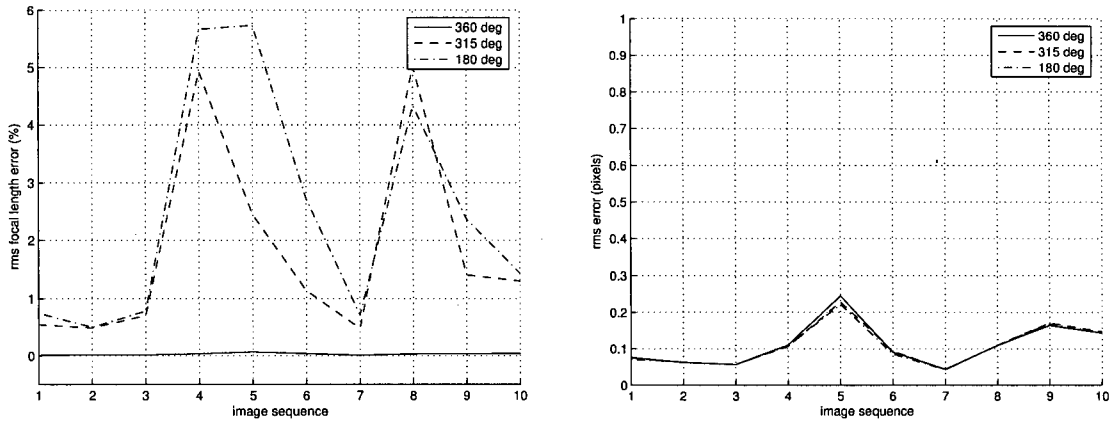


Figure 4.4: Solution of focal length for panoramas of 360°, 315° and 180°. These results were computed for the synthetic database. Note that even though the RMS pixel errors are approximately the same in each case (figure (b)), the focal length is commonly off by up to 5% (even with synthetic data and an optimal algorithm for the solution) when the panorama is less than 360°. These results indicate that 360° wrap-around is essential for accurate solution of focal length. It also justifies our use of projection error and not distance in parameter space for our evaluation function.

4.3 Creating Gold Standard Stitching Results

We have used two methods for generating gold standard stitching results, using both synthetic and real world data.

Synthetic Virtual camera views are generated from previously stitched panoramas. The camera matrices are known exactly.

Real World Real camera views are registered using a state of the art algorithm (different from those being tested) and manual intervention as necessary.

See appendix B for examples from these datasets. Note that only the first case gives actual "ground truth" camera matrices. The second approach is limited by the current best algorithms and human error, and is best described as "gold standard".

To create our synthetic data, we first use an image stitching algorithm to generate large panoramas. We then resample these panoramas to form virtual camera views. The statistics of the resulting images will differ slightly from natural images due to errors in the stitching process, but the camera matrices will be exact.

To create our real world dataset, we have used a direct (intensity-based) algorithm similar to [SS97]. Manual intervention was used for challenging situations such as low overlap or low texture areas, and to correct any visually unsatisfactory results. While the resulting camera matrices may be subject to small errors, real world images are used as inputs, which makes the results more representative of expected performance.

4.4 Experimental Setup

In this section, we describe the experimental methodology used to compare the performance of two automatic stitching algorithms, as well as showing how the same apparatus is used to tune algorithm parameters.

4.4.1 Comparison of Automatic Stitching Algorithms

We use the error metrics defined in section 4.2 to perform a comparison between two well known automatic image stitchers: AutoStitch[1], based on SIFT features [BL03]

[1] http://www.autostitch.net

from UBC, and MSRStitch, based on MOPS [BSW05] from Microsoft Research. We compare performance on synthetic and real datasets. The synthetic database consists of 10 sequences, each containing 16 images of 600 × 800 pixels, with 50% overlap between the images. The real database consists of 40 sequences of intentionally difficult stitching problems. These contain significant amounts of radial distortion, featureless regions (e.g. sky), motion (e.g. water) and parallax errors. We have created gold standard stitching data for 8 of these sequences using VideoMosaic [SS97]. We also use synthetically generated sequences to characterise performance with variable overlap and scale.

4.4.2 Parameter Tuning for Automatic Stitchers

Our evaluation function can be used to tune the parameters of automatic image stitching algorithms. For example, AutoStitch uses a Huber robust error function in the bundle adjustment stage. This function takes a parameter σ which corresponds to the distance (in pixels) of outliers to be suppressed. By plotting the RMS projection error e_{RMS} against σ, we can find an optimal value of σ. We perform similar experiments for the α parameter in AutoStitch (which controls the probability of declaring a valid image match given a set of feature matches) and also for the number of features extracted from each image.

4.5 Results

Figure 4.4 shows RMS errors in focal length, and RMS pixel errors for the synthetic dataset. We found (as in [KW99]) that when the panoramas wrap around 360°, the solution for focal length is very good (average RMS error in focal length = 0.029%). However, if the panorama is less than 360°, the focal length estimates are often up to 5% off (figure 4.4(a)), yet the RMS pixel errors are unchanged (figure 4.4(b)).

Figure 4.7 shows results for testing AutoStitch and MSRStitch on the synthetic database. Both stitchers perform very well, with RMS projection errors typically around 0.1 pixel. MSRStitch has a larger RMS error for sequence #5. This was caused by the stitcher failing to notice a match between a pair of images of fairly featureless snow (see figure 4.5). In these examples we used an outlier threshold of $r_{max} = 2$ pixels.

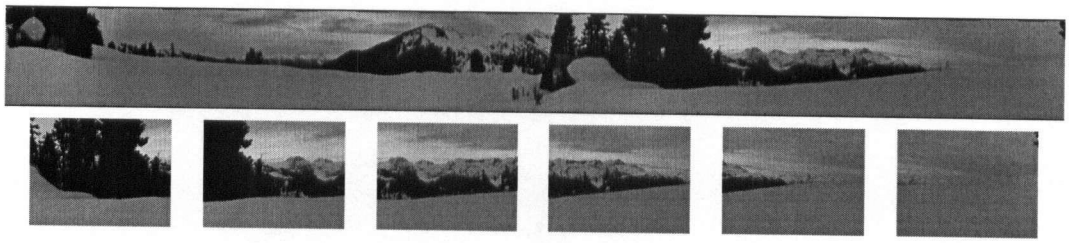
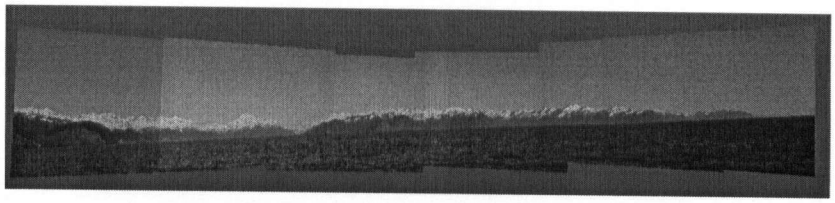
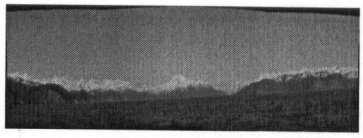


Figure 4.5: Elfin sequence from the synthetic dataset. MSRStitch failed to match the rightmost pair of images, in which the area of overlap is mainly featureless snow.



(a) Ground truth



(b) AutoStitch

(c) MSRStitch

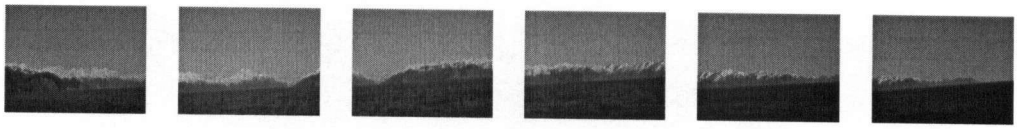


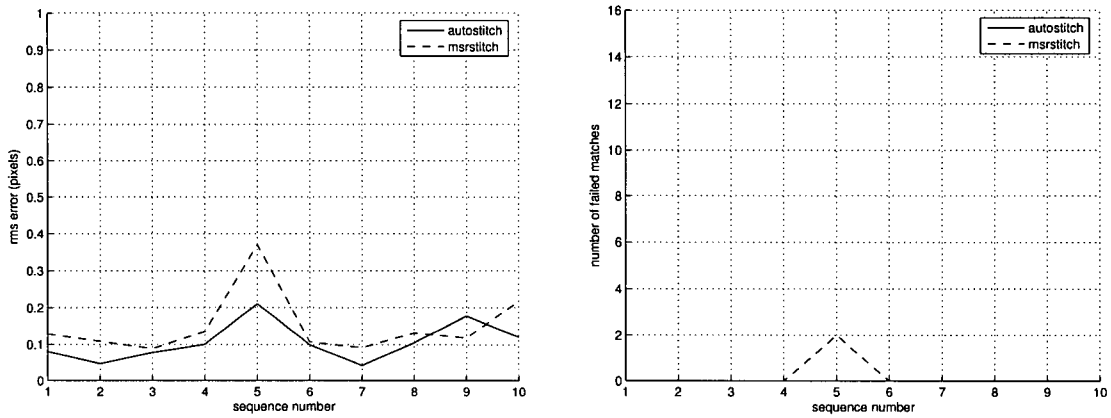
Figure 4.6: Comparison of AutoStitch and MSRStitch for the Alaska sequence from the real dataset. MSRStitch finds 6 matching images, and AutoStitch only 3. However, MSRStitch has actually stitched some images out of order, and has larger registration errors. The ground truth was stitched manually.

Figure 4.8 shows results for the (harder) real database. In this case we have set the outlier threshold r_{max} to 50 pixels as there is significant radial distortion in many of these images. The radial distortion is the main cause of the relatively high RMS errors in figure 4.8(a) . For the more difficult sequences there are also many match failures due to dropped or misregistered images, see figure 4.6 for an example.

We also tested AutoStitch and MSRStitch with variable image overlap and scale (figure 4.9). For the image overlap test we used images from the Green dataset (figure 4.2). The image size (600×800) and focal length were kept constant whilst the number of equally spaced images was varied to generate a range of overlap from 10% to 90%. The performance of MSRStitch dropped off for low image overlap ($< 20\%$). This is probably due to the larger footprint of MOPS relative to the detection scale when compared to SIFT, which means that fewer features can be extracted towards image edges.

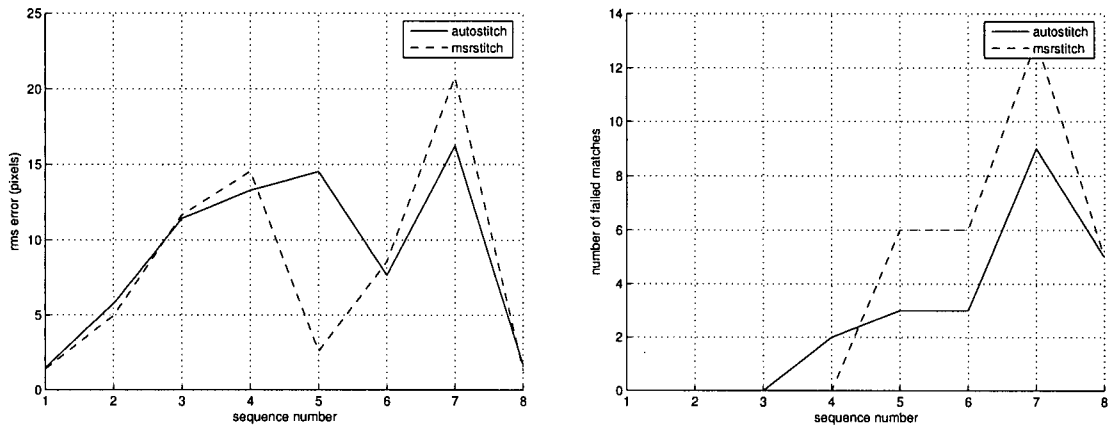
For the variable image scale test, we again used the Green dataset, but with images at scales from 10% to 100%. The RMS error is shown relative to the full size images. AutoStitch gave superior performance for the smaller image scales. This may again be due to the larger footprint of MOPS compared to SIFT at a given detection scale.

Figures 4.10 and 4.11 show stitching performance for AutoStitch whilst the parameters α, σ and the number of features per image were varied. The parameter α is the minimum number of matches for a correct image match to be declared in the probabilistic matching model of equation 3.13. The parameter σ is the outlier distance in the Huber robust error function 3.17. The plot for Huber σ in figure 4.10(b) shows a clear minimum at 0.25 pixels which represents to the optimal setting for this parameter on this dataset. It would be instructive to repeat this experiment on other datasets e.g. real images with moving objects, where the optimal setting would likely be higher. The plot for α in figure 4.10(a) shows a wide basin ($20 < \alpha < 120$) in which the optimal solution is achieved. Note that as α is increased, the RMS error increases as the quality of image registration falls, until finally some images fail to match. Once images fail to match, the RMS error often dips down again, as those remaining images can be registered adequately. Finally, figure 4.11 shows how stitching performance varies with the number of features extracted from each image. With small numbers of features (< 200) failed matches and misregistrations are common. This improves as the number of features increases, with negligible gains after about 400 features per image.



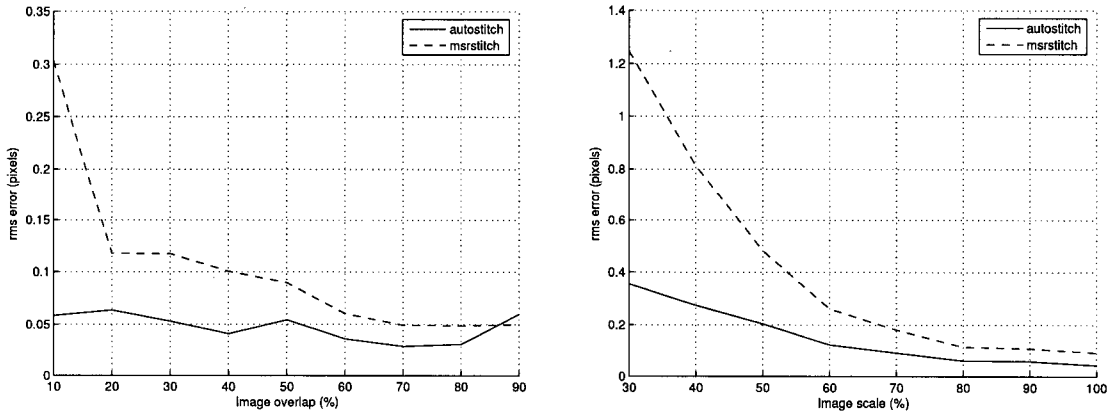
(a) (b)

Figure 4.7: Comparison of AutoStitch and MSRStitch using the synthetic image dataset. This dataset contains 10 synthetic image sequences. Note that both stitchers give very accurate solutions (0.1 pixel error) compared to the ground truth. MSRStitch has a glitch in sequence 5 (figure 4.5) which contains a pair of images of snow with few features.



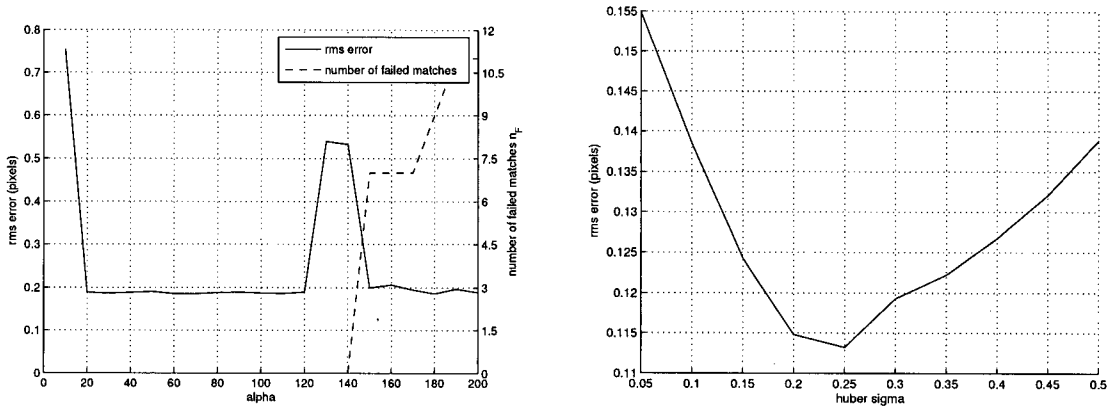
(a) (b)

Figure 4.8: Comparison of AutoStitch and MSRStitch using the real image dataset. This dataset contains 8 real image sequences. This dataset contains difficult sequences with radial distortion, featureless regions, motion and parallax. The radial distortion is the main cause of RMS errors around 10 pixels shown in (a).



(a) (b)

Figure 4.9: Comparison of AutoStitch and MSRStitch for panoramas with variable image overlap and scale. The first dataset (a) contains sequences with overlap from 10% to 90%. The second dataset (b) contains sequences with images with scale from 30% to 100%. Note that the performance of MSRStitch drops off at low image scales and small overlaps. This may be due to the larger footprint of MOPS compared to SIFT at a given detection scale, meaning that fewer features can be extracted towards image edges.



(a) (b)

Figure 4.10: Tuning of α and σ parameters. The results in (a) were generated by stitching a dataset of 16 synthetic images whilst varying α in increments of 10 from 0 to 200. The results are compared to ground truth. Here, values of alpha in the range 20-120 give the best stitching results. The results in (b) show similar results for tuning the σ parameter in the Huber robust error function. It can be seen that the optimal value of Huber σ is 0.25 pixels in this case.

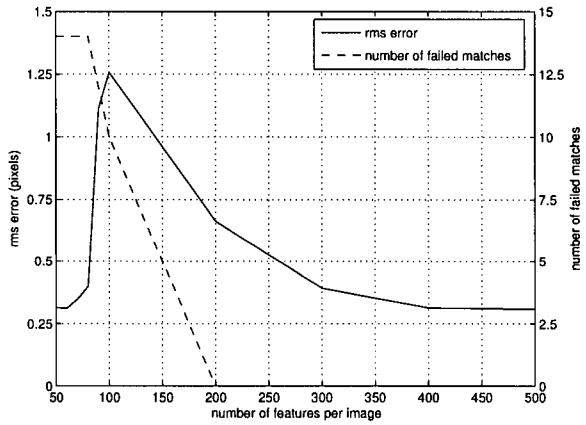


Figure 4.11: Effect of number of features extracted per image on RMS error, and number of failed matches. These results were computed using the Elfin (figure 4.5) dataset. Note that if the number of features extracted per image is small (< 200), then the number of failed matches is high. The RMS errors can still be low if the few images that remain are registered well. As more features are added, the RMS errors can actually increase as new images are matched but are poorly registered. In this case all images are correctly matched using 200 features per image but gains are minimal after more than about 400 features per image are used.

4.6 Summary

We have proposed a framework for evaluation of image stitching algorithms and used it to compare two automatic image stitchers, AutoStitch and MSRStitch. Both algorithms performed very well on the synthetic ground truth data, with registration errors typically around 0.1 pixels. Generally, AutoStitch performed better on the real dataset, but both algorithms showed room for improvement, with radial distortion a major cause of large RMS projection errors.

Future Work

Advanced Camera/Scene Models In this work we sampled virtual camera views from panoramas using a simple 4 parameter camera model. An interesting direction for future work would be to generate high quality computer graphic renderings of virtual scenes with a large number of camera and scene parameters e.g. motion of the camera (parallax), radial distortion, illumination changes etc. One would then attempt to solve for all of these parameters and evaluate the results using the image based error function of equation 4.5.

Feature Based vs Direct Methods The relative merits of direct and feature based methods have been the subject of much discussion in the computer vision community [TZ99, IA99]. An interesting area for future work would be to compare these two methods based on their registration accuracy, using the evaluation framework presented here.

Chapter 5

3D Object Recognition and Reconstruction

5.1 Introduction

Object recognition and structure and motion recovery are two long standing problems in computer vision. The structure and motion (SAM) problem[1] has reached a degree of maturity, with several commercial offerings [2D3, REA], in addition to an extensive research literature [SK93, BTZ96, Pol00, HZ04]. Object recognition is also well studied but remains an extremely active research area, with recent advances in image features and probabilistic modelling inspiring previously unexplored areas such as object class recognition [FPZ03]. Invariant local features have emerged as an invaluable tool in tackling the ubiquitous image correspondence problem. By using descriptors that are invariant not just to translation, but also to rotation [SM97], scale [Low99] and affine warping [Bau00, MS02, MCUP02], invariant features provide much more robust matching than previous correlation based methods.

Until recently, the majority of object recognition algorithms have depended upon some form of training phase [Low99, VJ01]. However, algorithms have been developed recently that operate in an unsupervised manner on an image dataset [BL03, SZ02, RLSP03]. In this chapter we develop such an algorithm. We operate in an unsupervised setting on an unordered image dataset, and pose the object recognition problem as one of finding matches that are consistent views of some 3D scene. Our algorithm recognises objects in the sense that it finds all images that view a given object or scene. However, it makes no attempt to attach a label to the object [DBdFF02] or generalise object classes [FPZ03].

The remainder of this chapter is structured as follows. In section 5.2 we describe our

[1] Also known as structure *from* motion (SFM). We prefer to use the term structure *and* motion (SAM), to emphasise the fact that the estimation of 3D structure and camera motion parameters are fundamentally linked.

invariant feature extraction and matching scheme. Section 5.3 describes the geometric constraints used to find correct image matches. Section 5.4 describes the sparse bundle adjustment algorithm used to solve jointly for the cameras and structure. Section 5.5 demonstrates results of object recognition and reconstruction on a test dataset, and section 5.6 presents conclusions and ideas for future work.

5.2 Feature Matching

We extract and match SIFT features from all images in exactly the same way as for automatic panorama stitching (described in section 3.2), using a k-d tree to find nearest neighbour matches. We then apply a feature space outlier rejection test as described in section 2.7.1.

5.3 Image Matching

During this stage, the objective is to find all matching images, that is, those that view a common subset of 3D points. Connected sets of image matches will later become 3D models. From the feature matching step, we have identified images with a large number of matches between them. As in section 3.3, we consider a constant number m images as potential image matches (we use $m = 6$).

We parameterise each camera using 7 parameters. These are a rotation vector $\boldsymbol{\theta}_i = [\theta_{i1}, \theta_{i2}, \theta_{i3}]$, translation $\mathbf{t}_i = [t_{i1}, t_{i2}, t_{i3}]$ and focal length f_i (see section 3.2). Each pairwise image match adds four constraints on the camera parameters whilst adding three unknown structure parameters $\mathbf{X} = [X_1, X_2, X_3]$

$$\tilde{\mathbf{u}}_i = \mathbf{K}_i \mathbf{X}_{ci} \tag{5.1}$$

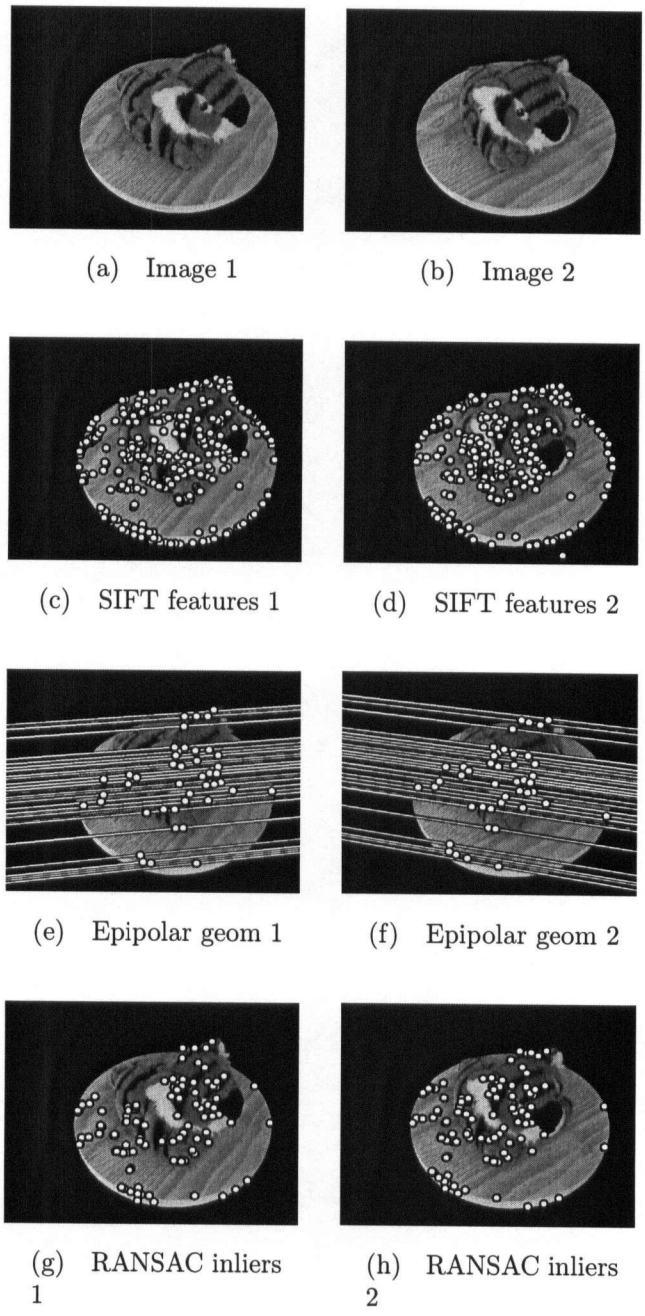
$$\tilde{\mathbf{u}}_j = \mathbf{K}_j \mathbf{X}_{cj} \tag{5.2}$$

$$\mathbf{X}_{ci} = \mathbf{R}_i \mathbf{X} + \mathbf{t}_i \tag{5.3}$$

$$\mathbf{X}_{cj} = \mathbf{R}_j \mathbf{X} + \mathbf{t}_j \tag{5.4}$$

where $\tilde{\mathbf{u}}_i, \tilde{\mathbf{u}}_j$ are the homogeneous image positions in camera i and j respectively.

The single remaining constraint (4 equations minus 3 unknowns = 1 constraint) expresses the fact that the two camera rays $\tilde{\mathbf{p}}_i$, $\tilde{\mathbf{p}}_j$ and the translation vector between



(a) Image 1 (b) Image 2

(c) SIFT features 1 (d) SIFT features 2

(e) Epipolar geom 1 (f) Epipolar geom 2

(g) RANSAC inliers 1 (h) RANSAC inliers 2

Figure 5.1: Finding sets of consistent matches using SIFT and RANSAC. SIFT features are extracted from all input images, and each feature is matched to k = 4 nearest neighbours. Outliers are first rejected by thresholding against the distance of an incorrect match (section (2.7.1)), before RANSAC is used to find a final set of inliers that are consistent with the fundamental matrix. For this pair of 1024 × 768 input images, there were 365 SIFT features in image 1 and 379 in image 2. Of the initial feature matches, 133 matches remained after feature space outlier rejection, and there were 103 matches in the final solution after using RANSAC.

camera centres $\mathbf{t}_{ij}$ are coplanar, and hence their scalar triple product is equal to zero

$$\tilde{\mathbf{p}}_i^T [\mathbf{t}_{ij}]_\times \tilde{\mathbf{p}}_j = 0 \tag{5.5}$$

Writing $\tilde{\mathbf{p}}_i, \tilde{\mathbf{p}}_j$ and $\mathbf{t}_{ij}$ in terms of camera parameters

$$\begin{aligned} \tilde{\mathbf{p}}_i &= \mathbf{R}_i^T \mathbf{K}_i^{-1} \tilde{\mathbf{u}}_i & (5.6) \\ \tilde{\mathbf{p}}_j &= \mathbf{R}_j^T \mathbf{K}_j^{-1} \tilde{\mathbf{u}}_j & (5.7) \\ \mathbf{t}_{ij} &= \mathbf{R}_j^T \mathbf{t}_j - \mathbf{R}_i^T \mathbf{t}_i & (5.8) \end{aligned}$$

and substituting in equation 5.5 gives

$$\tilde{\mathbf{u}}_i^T \mathbf{F}_{ij} \tilde{\mathbf{u}}_j = 0 \tag{5.9}$$

where

$$\mathbf{F}_{ij} = \mathbf{K}_i^{-T} \mathbf{R}_i [\mathbf{R}_j^T \mathbf{t}_j - \mathbf{R}_i^T \mathbf{t}_i]_\times \mathbf{R}_j^T \mathbf{K}_j^{-1} \tag{5.10}$$

This is the well known epipolar constraint. Image matching entails robust estimation of the fundamental matrix $\mathbf{F}_{ij}$ [TM97]. Since equation 5.9 is non-linear in the camera parameters, it is commonplace to relax the non-linear constraints and estimate a general 3×3 matrix $\mathbf{F}_{ij}$. This enables a closed form solution via SVD [HZ04].

We use RANSAC to robustly estimate $\mathbf{F}$ and hence find a set of inliers that have consistent epipolar geometry. An image match is declared if the number of RANSAC inliers $n_{inliers} > n_{match}$, where the minimum number of matches n_{match} is a constant (typically around 20). 3D objects/scenes are identified as connected components of image matches. This simple approach typically requires hand tuning of the threshold n_{match}. An improved verification procedure might extend the probabilistic model of section 3.3.2 to include parallax. This is left for future work.

5.4 Bundle Adjustment

Given a set of geometrically consistent matches, we use bundle adjustment to solve for the camera and structure parameters jointly. In contrast to other approaches

[HZ04, Pol00] that begin with a projective reconstruction and later refine to a metric reconstruction, we solve directly for the metric structure and camera parameters. The cameras are added one by one, starting with the best matching pair. We have found that initialising each new camera with the rotation, translation and focal length of the best matching image works well, even if the images have different rotation and scale (see example in figure 5.4). In practice we peturb the camera positions slightly by adding Gaussian noise (of standard deviation unity) to the translation vector of each new camera that is added. To cope with Necker reversal, we first run bundle adjustment on the initial image pair, noting the final value of the error function (section 5.4.1). We then swap the camera positions, and flip the 3D point depths, before repeating bundle adjustment (as in [SK93]). This normally converges to a different local minimum. We retain the solution that minimises the error function.

5.4.1 Sparse Bundle Adjustment

Each connected component of feature matches defines a 3D point $\mathbf{X}_j$, and our error function is the sum squared error between the projected 3D point and the measured feature position

$$e = \sum_{i \epsilon \mathcal{I}} \sum_{j \epsilon \mathcal{X}(i)} f(\mathbf{r}_{ij}) \tag{5.11}$$

where $\mathcal{I}$ is the set of all images, $\mathcal{X}(i)$ is the set of 3D points projecting to image i, and $\mathbf{r}_{ij}$ is the residual error in image i for 3D point j. The residual $\mathbf{r}_{ij}$ is the difference between the measured feature position and projected 3D point

$$\mathbf{r}_{ij} = \mathbf{m}_{ij} - \mathbf{u}_{ij} \tag{5.12}$$

where $\mathbf{m}_{ij}$ is the measured feature position, and $\mathbf{u}_{ij}$ is the projection of point $\mathbf{X}_j$ in image i

$$\tilde{\mathbf{u}}_{ij} = \mathbf{K}_i(\mathbf{R}_i\mathbf{X}_j + \mathbf{t}_i) \tag{5.13}$$

We use a Huber robust error function as in equation 3.17. The outlier distance σ is set at 3 standard deviations of the current (un-normalised) residual error. We use the Levenberg-Marquardt algorithm to solve this non-linear least squares problem. Each

iteration step is of the form

$$\mathbf{\Phi} = (\mathbf{J}^T\mathbf{J} + \lambda\mathbf{C}_p^{-1})^{-1}\mathbf{J}^T\mathbf{r} \tag{5.14}$$

where $\mathbf{\Phi} = [\mathbf{\Theta}, \mathbf{X}]$ is the vector of camera ($\mathbf{\Theta}$) and structure ($\mathbf{X}$) parameters, $\mathbf{r}$ is the vector of residuals and $\mathbf{J} = \partial\mathbf{r}/\partial\mathbf{\Phi}$. The Jacobean $\mathbf{J}$ is an $M \times N$ matrix, where M is the number of measurements (twice the number of features), and $N = n_\Theta + n_X$ is the number of camera (n_Θ) and structure (n_X) parameters (7 for each camera plus 3 for each 3D point). The prior covariance matrix $\mathbf{C}_p$ is set such that the standard deviation of angles is $\sigma_\theta = \pi/16$, translations $\sigma_t = 0.01$, focal lengths $\sigma_f = \bar{f}/100$ and 3D points $\sigma_X = 0.1$. Note that new cameras are initialsed with the same translation, rotation and focal length as the best matching camera (largest number of matches), but Gaussian noise of standard deviation 1 is added to the new camera to prevent the camera centres being coincident. Although one could in principle solve equation 5.14 directly (by solving an $N \times N$ linear system), to do so would ignore the sparse structure of the problem, and be very inefficient.

Firstly, the matrix $\mathbf{J}$ is mostly zeros (since the derivatives of residuals for image i are zero except with respect to the parameters of image i), so the elements of $\mathbf{J}^T\mathbf{J}$ should be computed directly, instead of computing $\mathbf{J}$ first. Examining the structure of $\mathbf{J}^T\mathbf{J}$

$$\mathbf{J}^T\mathbf{J} = \left[\begin{array}{c|c} \frac{\partial\mathbf{r}}{\partial\mathbf{\Theta}}^T\frac{\partial\mathbf{r}}{\partial\mathbf{\Theta}} & \frac{\partial\mathbf{r}}{\partial\mathbf{\Theta}}^T\frac{\partial\mathbf{r}}{\partial\mathbf{X}} \\ \hline \frac{\partial\mathbf{r}}{\partial\mathbf{X}}^T\frac{\partial\mathbf{r}}{\partial\mathbf{\Theta}} & \frac{\partial\mathbf{r}}{\partial\mathbf{X}}^T\frac{\partial\mathbf{r}}{\partial\mathbf{X}} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{C}_\Theta^{-1} & \mathbf{C}_{\Theta\mathbf{X}}^{-1} \\ \hline \mathbf{C}_{\Theta\mathbf{X}}^{-T} & \mathbf{C}_\mathbf{X}^{-1} \end{array}\right] \tag{5.15}$$

where the camera parameter inverse covariance matrix

$$\mathbf{C}_\Theta^{-1} = \begin{bmatrix} \sum_j \frac{\partial\mathbf{r}_{1j}}{\partial\mathbf{\Theta}_1}^T\frac{\partial\mathbf{r}_{1j}}{\partial\mathbf{\Theta}_1} & 0 & 0 & \dots \\ 0 & \sum_j \frac{\partial\mathbf{r}_{2j}}{\partial\mathbf{\Theta}_2}^T\frac{\partial\mathbf{r}_{2j}}{\partial\mathbf{\Theta}_2} & 0 & \dots \\ 0 & 0 & \sum_j \frac{\partial\mathbf{r}_{3j}}{\partial\mathbf{\Theta}_3}^T\frac{\partial\mathbf{r}_{3j}}{\partial\mathbf{\Theta}_3} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{5.16}$$

is block diagonal, consisting of 7×7 blocks, and the structure parameter inverse co-

variance matrix

$$\mathbf{C}_{\mathbf{X}}^{-1} = \begin{bmatrix} \sum_i \frac{\partial \mathbf{r}_{i1}}{\partial \mathbf{X}_1}^T \frac{\partial \mathbf{r}_{i1}}{\partial \mathbf{X}_1} & 0 & 0 & \cdots \\ 0 & \sum_i \frac{\partial \mathbf{r}_{i2}}{\partial \mathbf{X}_2}^T \frac{\partial \mathbf{r}_{i2}}{\partial \mathbf{X}_2} & 0 & \cdots \\ 0 & 0 & \sum_i \frac{\partial \mathbf{r}_{i3}}{\partial \mathbf{X}_3}^T \frac{\partial \mathbf{r}_{i3}}{\partial \mathbf{X}_3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{5.17}$$

is also block diagonal, consisting of 3×3 blocks. The camera/structure cross covariance is a full matrix

$$\mathbf{C}_{\mathbf{\Theta X}}^{-1} = \begin{bmatrix} \frac{\partial \mathbf{r}_{11}}{\partial \mathbf{\Theta}_1}^T \frac{\partial \mathbf{r}_{11}}{\partial \mathbf{X}_1} & \frac{\partial \mathbf{r}_{12}}{\partial \mathbf{\Theta}_1}^T \frac{\partial \mathbf{r}_{12}}{\partial \mathbf{X}_2} & \frac{\partial \mathbf{r}_{13}}{\partial \mathbf{\Theta}_1}^T \frac{\partial \mathbf{r}_{13}}{\partial \mathbf{X}_3} & \cdots \\ \frac{\partial \mathbf{r}_{21}}{\partial \mathbf{\Theta}_2}^T \frac{\partial \mathbf{r}_{21}}{\partial \mathbf{X}_1} & \frac{\partial \mathbf{r}_{22}}{\partial \mathbf{\Theta}_2}^T \frac{\partial \mathbf{r}_{22}}{\partial \mathbf{X}_2} & \frac{\partial \mathbf{r}_{23}}{\partial \mathbf{\Theta}_2}^T \frac{\partial \mathbf{r}_{23}}{\partial \mathbf{X}_3} & \cdots \\ \frac{\partial \mathbf{r}_{31}}{\partial \mathbf{\Theta}_3}^T \frac{\partial \mathbf{r}_{31}}{\partial \mathbf{X}_1} & \frac{\partial \mathbf{r}_{32}}{\partial \mathbf{\Theta}_3}^T \frac{\partial \mathbf{r}_{32}}{\partial \mathbf{X}_2} & \frac{\partial \mathbf{r}_{33}}{\partial \mathbf{\Theta}_3}^T \frac{\partial \mathbf{r}_{33}}{\partial \mathbf{X}_3} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{5.18}$$

but consists of a single multiplication for each element (the covariance between image i and point j depends only on the residual of point j in image i). Computing $\mathbf{J}^T\mathbf{J}$ by explicit multiplication of $\mathbf{J}$ would take $O(MN^2)$ operations. However, $\mathbf{J}^T\mathbf{J}$ can in fact be computed in $O(n_\Theta n_X)$ operations (the cost of computing $\mathbf{C}_{\mathbf{\Theta X}}^{-1}$).

Secondly, the matrix inversion involving $\mathbf{J}^T\mathbf{J}$ need not be computed explicitly ($O(N^3)$) due to the sparse structure of $\mathbf{J}^T\mathbf{J}$. This sparseness reflects the loose coupling inbetween cameras, and inbetween 3D points, in the error function of equation 5.11. The cameras are independent given the 3D structure parameters, and the 3D points are independent given the cameras. Equation 5.14 may be rewritten

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{\Theta} \\ \mathbf{X} \end{bmatrix} = \begin{bmatrix} e_{\mathbf{\Theta}} \\ e_{\mathbf{X}} \end{bmatrix} \tag{5.19}$$

where

$$\mathbf{A} = \mathbf{C}_{\Theta}^{-1} + \sigma^2 \mathbf{C}_{p\Theta}^{-1} \tag{5.20}$$

$$\mathbf{C} = \mathbf{C}_{\mathbf{X}}^{-1} + \sigma^2 \mathbf{C}_{p\mathbf{X}}^{-1} \tag{5.21}$$

$$\mathbf{B} = \mathbf{C}_{\Theta\mathbf{X}}^{-1} \tag{5.22}$$

$$\mathbf{e}_{\Theta} = \frac{\partial \mathbf{r}}{\partial \Theta}^T \mathbf{r} \tag{5.23}$$

$$\mathbf{e}_{\mathbf{X}} = \frac{\partial \mathbf{r}}{\partial \mathbf{X}}^T \mathbf{r} \tag{5.24}$$

and

$$\mathbf{C}_p^{-1} = \begin{bmatrix} \mathbf{C}_{p\Theta}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{p\mathbf{X}}^{-1} \end{bmatrix} \tag{5.25}$$

Multiplying both sides of equation 5.19 by $\begin{bmatrix} \mathbf{I} & -\mathbf{B}\mathbf{C}^{-1} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}$ gives

$$\begin{bmatrix} \mathbf{A} - \mathbf{B}\mathbf{C}^{-1}\mathbf{B}^T & \mathbf{0} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix} \begin{bmatrix} \Theta \\ \mathbf{X} \end{bmatrix} = \begin{bmatrix} e_{\Theta} - \mathbf{B}\mathbf{C}^{-1} e_{\mathbf{X}} \\ e_{\mathbf{X}} \end{bmatrix} \tag{5.26}$$

which eliminates $\mathbf{X}$ from the solution for Θ. This is known as a reduced camera subsystem [TMHF99].

Note that again none of the matrix products need be computed directly. For example each element of $\mathbf{e}_{\Theta} = [\mathbf{e}_{\Theta 1}, \mathbf{e}_{\Theta 2}, \ldots]$ is computed as

$$\mathbf{e}_{\Theta i} = \sum_k \frac{\partial \mathbf{r}_{ik}}{\partial \Theta_i}^T \mathbf{r}_{ik} \tag{5.27}$$

which involves only one term for each residual in image i. Similarly, each element of $\mathbf{e}_{\mathbf{X}} = [\mathbf{e}_{\mathbf{X}1}, \mathbf{e}_{\mathbf{X}2}, \ldots]$ is given by

$$\mathbf{e}_{\mathbf{X}i} = \sum_k \frac{\partial \mathbf{r}_{ki}}{\partial \mathbf{X}_i}^T \mathbf{r}_{ki} \tag{5.28}$$

and the right-hand side elimination product $\mathbf{B}\mathbf{C}^{-1}\mathbf{e}_{\mathbf{X}}$

$$(\mathbf{B}\mathbf{C}^{-1}\mathbf{e}_{\mathbf{X}})_i = \sum_k \mathbf{B}_{ik}\mathbf{C}_{kk}\mathbf{e}_{\mathbf{X}k} \tag{5.29}$$

where

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \mathbf{B}_{13} & \dots \\ \mathbf{B}_{21} & \mathbf{B}_{22} & \mathbf{B}_{23} & \dots \\ \mathbf{B}_{31} & \mathbf{B}_{32} & \mathbf{B}_{33} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{5.30}$$

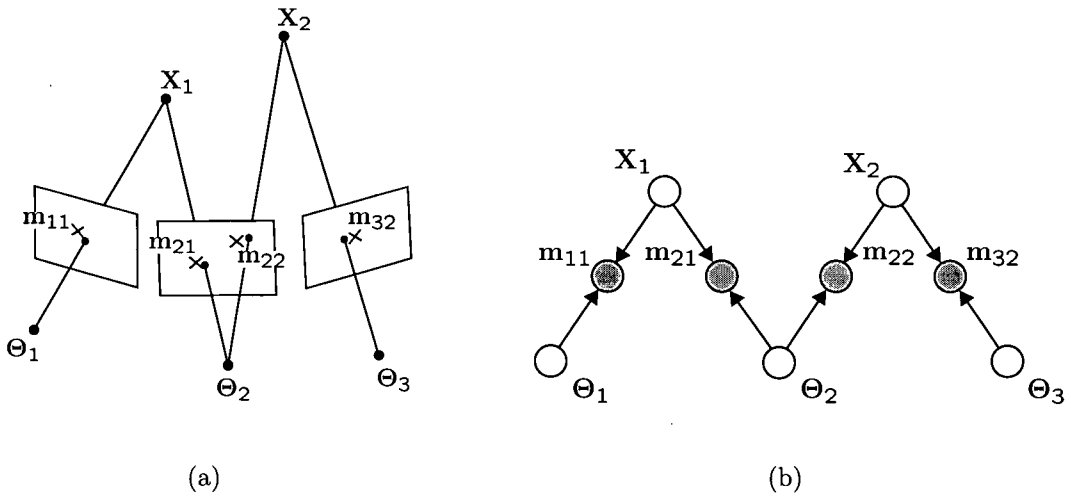
$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{11} & 0 & 0 & \dots \\ 0 & \mathbf{C}_{22} & 0 & \dots \\ 0 & 0 & \mathbf{C}_{33} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{5.31}$$

Where each element $\mathbf{B}_{ij} = \frac{\partial \mathbf{r}_{ij}}{\partial \mathbf{\Theta}_i}^T \frac{\partial \mathbf{r}_{ij}}{\partial \mathbf{X}_j}$ is a 7×3 matrix (number of parameters per camera $\times$ number of parameters per 3D point).

This gives an $n_\Theta \times n_\Theta$ linear system to solve for the camera parameters $\mathbf{\Theta}$. The resulting value of $\mathbf{\Theta}$ can be substituted into the linear system for $\mathbf{X}$, which reduces to independent 3×3 linear systems for each 3D point. The most expensive stage in this process is (potentially) in computation of the left-hand side elimination product $\mathbf{B}\mathbf{C}^{-1}\mathbf{B}^T$. The ijth element is given by

$$(\mathbf{B}\mathbf{C}^{-1}\mathbf{B}^T)_{ij} = \sum_k \mathbf{B}_{ik}\mathbf{C}_{kk}\mathbf{B}_{kj} \tag{5.32}$$

where $\mathbf{B}_{ij} = \frac{\partial \mathbf{r}_{ij}}{\partial \mathbf{\Theta}_i}^T \frac{\partial \mathbf{r}_{ij}}{\partial \mathbf{X}_j}$ is a 7×3 matrix (number of parameters per camera $\times$ number of parameters per 3D point) and $\mathbf{C}_{kk} = \sum_i \frac{\partial \mathbf{r}_{ik}}{\partial \mathbf{X}_k}^T \frac{\partial \mathbf{r}_{ik}}{\partial \mathbf{X}_k}$ is a 3×3 matrix. This summation may involve in the worst case M terms (if every 3D point is imaged in every camera). Hence the worst case complexity of sparse bundle adjustment is $O(Mn_\Theta^2)$. Note that this is still much cheaper than it would be were $\mathbf{C}$ not block diagonal. If $\mathbf{C}$ were a general $n_X \times n_X$ matrix the cost of this elimination step would be $O(n_X^3)$. However, the cost of bundle adjustment is usually much less than $O(Mn_\Theta^2)$. This is because the terms $\mathbf{B}_{ij}$ are zero unless point j is viewed in camera i. This means that each summation above involves only a constant number of 3D points for each camera. In this case, the complexity of sparse bundle adjustment is $O(mn_\Theta^2)$, where m is the number of residuals in each image. The best case complexity (given small m) is $O(n_\Theta^3)$, which is the cost



(a) (b)

Figure 5.2: A simple bundle adjustment problem and the corresponding probabilistic graphical model. The unknown quantities to be inferred are the camera parameters $\mathbf{\Theta}_j$ and the 3D points $\mathbf{X}_i$. The measured quantities are the projected feature positions $\mathbf{m}_{ij}$.

of solving the linear system for the camera parameters.

Hence the total computational cost for one step of sparse bundle adjustment is now $O(mn_\Theta^2)$, reduced from $O(MN^2)$ for naive solution of the normal equations of equation 5.14. Note that $n_\Theta \ll N$ since the number of camera parameters n_Θ is very much less than the number of structure parameters n_X ($N = n_\Theta + n_X$). This is a very significant reduction in practice. For example, with 10 cameras ($n_\Theta = 70$), and 1000 3D points ($n_X = 3000$), sparse bundle adjustment would be about $(N/n_\Theta)^2 = (3070/70)^2 \approx 2000$ times faster than naive bundle adjustment. Furthermore, if a constant number of 3D points are imaged by each camera, the cost would be further reduced.

5.4.2 Graphical Model Interpretation

The bundle adjustment process (joint estimation of camera and structure parameters from noisy image based measurements) can be described conveniently using a probabilistic graphical model (see figure 5.2). Each 3D point $\mathbf{X}_i$ gives rise to a measurement $\mathbf{m}_{ij}$ in each camera j (with parameters $\mathbf{\Theta}_j$) that it is imaged. The joint probability of the measurements, cameras and structure (assuming uniform priors on $\mathbf{\Theta}_i$, $\mathbf{X}_j$) is

given by

$$p(\mathbf{m}, \mathbf{X}, \Theta) = \prod_{i \epsilon \mathcal{I}} \prod_{j \epsilon \mathcal{X}(i)} p(\mathbf{m}_{ij} | \Theta_i, \mathbf{X}_j) \tag{5.33}$$

If the measurement density is assumed to be Gaussian then

$$p(\mathbf{m}_{ij} | \Theta_i, \mathbf{X}_j) = N(\mathbf{m}_{ij} - \mathbf{u}_{ij}; \mathbf{0}, \sigma_n^2 \mathbf{I}) \tag{5.34}$$

where σ_n is the standard deviation of measurement noise in the image plane and $\mathbf{u}_{ij}$ is the projection of 3D point $\mathbf{X}_j$ to image i

$$\tilde{\mathbf{u}}_{ij} = \mathbf{K}_i(\mathbf{R}_i \mathbf{X}_j - \mathbf{t}_i) \tag{5.35}$$

Taking the negative logarithm of equation 5.33 gives

$$-\log p(\mathbf{m}|\Theta, \mathbf{X}) \propto \sum_{i \epsilon \mathcal{I}} \sum_{j \epsilon \mathcal{X}(i)} -\log N(\mathbf{m}_{ij} - \mathbf{u}_{ij}; \mathbf{0}, \sigma_n^2 \mathbf{I}) \tag{5.36}$$

$$\propto \sum_{i \epsilon \mathcal{I}} \sum_{j \epsilon \mathcal{X}(i)} |\mathbf{m}_{ij} - \mathbf{u}_{ij}|^2 / \sigma_n^2 \tag{5.37}$$

The maximum likelihood solution is

$$\Theta, \mathbf{X} = \arg\min_{\Theta, \mathbf{X}} e(\Theta, \mathbf{X}) = \sum_{i \epsilon \mathcal{I}} \sum_{j \epsilon \mathcal{X}(i)} |\mathbf{m}_{ij} - \mathbf{u}_{ij}|^2 / \sigma_n^2 \tag{5.38}$$

Hence, the optimal solution for camera and structure parameters (under the assumption of Gaussian noise) is obtained by solving a non-linear least squares problem (equation 5.38). In practice robustness is achieved by using Gaussian priors on the parameters Θ, $\mathbf{X}$ and a robust Huber kernel is used instead of a Gaussian kernel in the measurement density of equation 5.34.

5.4.3 Sparsity in the Graphical Model

Sparsity manifests itself in two ways in bundle adjustment problems. These can be seen clearly from the graphical model structure of the problem (figure 5.3). Firstly, the structure of the observation process implies fundamental independence relationships

between cameras and 3D points. More precisely, the camera parameters are independent given the 3D points, and the 3D points are independent given the cameras. Secondly, many bundle adjustment problems are sparse in the sense that the unknown camera and structure parameters are loosely coupled. This leads to a reduction in the complexity of sparse bundle adjustment algorithms.

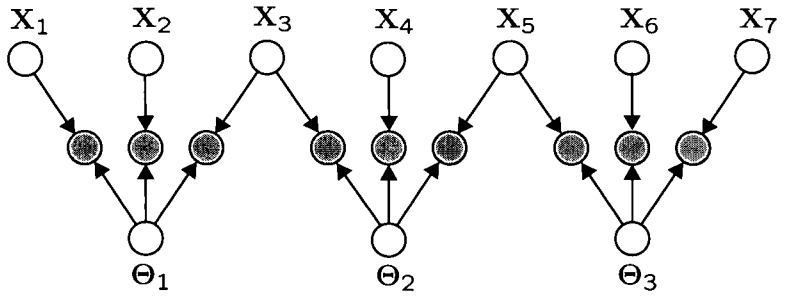
The first level of sparseness arises because camera parameters are only coupled by the 3D points that they observe, and vice versa. A pair of cameras $\mathbf{\Theta}_i$, $\mathbf{\Theta}_j$ are dependent if they view a common 3D point $\mathbf{X}_k$, or if there is sequence of cameras (viewing common 3D points), that connects them. For example in figure 5.3(a) , cameras $\mathbf{\Theta}_1$ and $\mathbf{\Theta}_2$ are coupled because they share a common 3D point. However, cameras $\mathbf{\Theta}_1$ and $\mathbf{\Theta}_3$ are also coupled because camera $\mathbf{\Theta}_2$ shares common 3D points with both. In other words, there is a path for a 'Bayes Ball' [Sha98] between $\mathbf{\Theta}_1$ and $\mathbf{\Theta}_3$, so they are not independent. The cameras can be made to be independent if all of the structure parameters $\mathbf{X}$ are known (observed). Likewise, the 3D points can be made to be independent if all the camera parameters $\mathbf{\Theta}$ are known. This is the basis for the sparse bundle adjustment algorithm of section 5.4.1.

The second level of sparseness arises in certain kinds of bundle adjustment problems. Consider a sequence consisting of a few images of an object taken from similar viewpoints. In this case (almost) all of the 3D points will be visible in each of the cameras, and the graph has a fully connected structure (figure 5.3(b)). This might occur in a short turntable sequence with a small range of viewpoints. Conversely, if the sequence consists of widely separated views which share few common points, the graph has a sparse structure (figure 5.3(a)). The first case leads to the worst case complexity of sparse bundle adjustment of $O(Mn_{\mathbf{\Theta}}^2)$ (where M is the number of measurements and $n_{\mathbf{\Theta}}$ is the number of camera parameters). The second case leads to the best case complexity of sparse bundle adjustment $O(n_{\mathbf{\Theta}}^3)$.

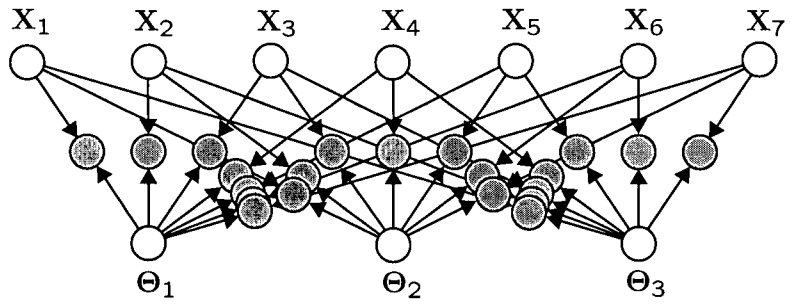
Note that in the sparsely connected case (figure 5.3) more general sparse matrix techniques [GL81] may be used in addition to the reduced camera/structure systems described in section 5.4.1.

5.5 Results

Figure 5.4 shows typical operation of the object recognition algorithm. A set of images containing 2 objects and 6 distractor images was input. The algorithm detected 2

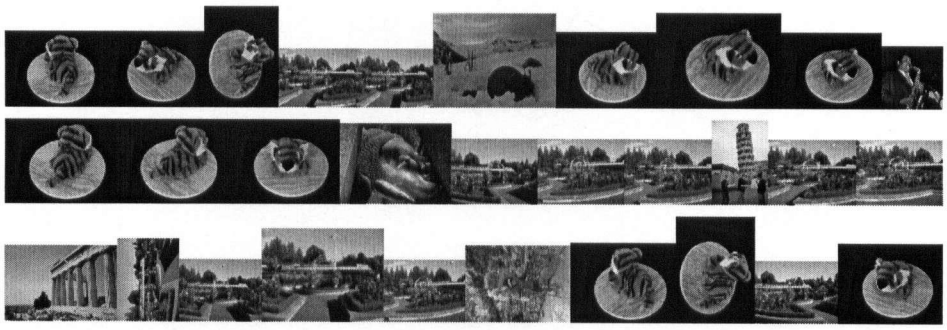


(a)

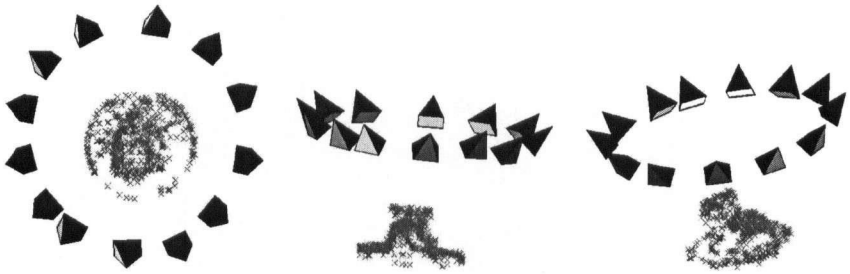


(b)

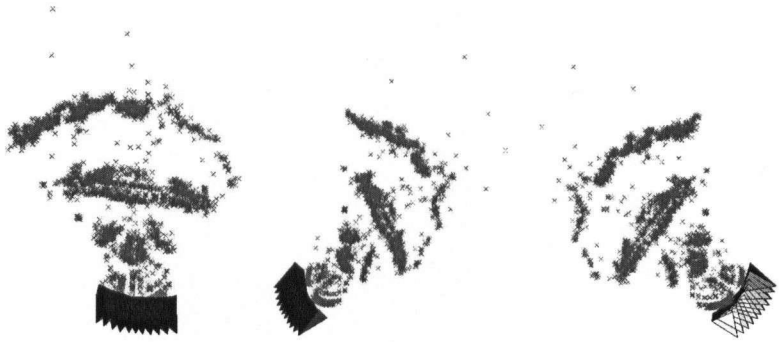
Figure 5.3: Graphical models for (a) sparsely and (b) fully connected bundle adjustment problems. Note that in both cases the value of each unknown parameters $\mathbf{X}_i$, $\mathbf{\Theta}_j$ is dependent on *every other* unknown parameter. However, if all the camera parameters $\mathbf{\Theta}_i$ are known (observed), the 3D structure parameters $\mathbf{X}_i$ are independent, and vice versa. This is the basis for the sparse bundle adjustment method described in section 5.4.1. The best case complexity of this algorithm is $O(n_\Theta^3)$ i.e. cubic in the number of camera parameters (in the sparsely connected case (a)), and the worst case complexity is $O(Mn_\Theta^2)$ (in the fully connected case (b)). The number of camera parameters is n_Θ and the number of measurements is $\frac{1}{2}M$.



(a) Input images



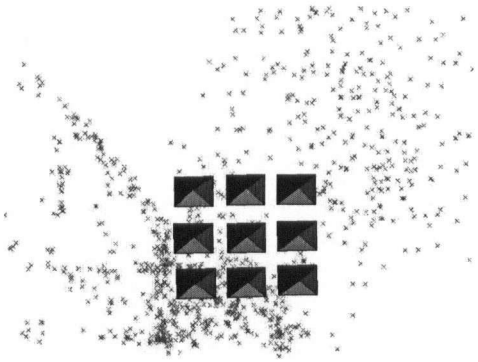
(b) Output 3D model 1 – Tiger



(c) Output 3D model 2 – Rose Garden

Figure 5.4: Fully automatic object recognition and 3D reconstruction. Note that despite the incorrect ordering, rotation and scale changes, and distractor images in the input, the system is able to successfully recognise the two consistent objects and perform 3D reconstruction. The Tiger sequence consisted of 13 images and yielded a 3D model with 675 points. The Rose Garden sequence consisted of 11 images and the 3D model contained 1351 points. The whole process of feature matching, image matching and bundle adjustment took a total of 556 seconds, of which 230 seconds were spent during bundle adjustment. The tests were run using a MATLAB implementation on a 2.8GHz Pentium processor.

(a) Input images



(b) Output 3D model

Figure 5.5: Fully automatic structure and motion (SAM) estimation for a 3×3 array of cameras. Note that the relative camera positions have been recovered correctly. In this example the input images were 800×600, and a 3D model of 1684 points was computed. The total computation time for feature extraction and matching, image matching and bundle adjustment was 293 seconds.

Algorithm: 3D Object Recognition/Reconstruction

Input: n unordered images

I. Extract SIFT features from all n images

II. Find k nearest-neighbours for each feature using a k-d tree

III. For each image:
   (i) Select m candidate matching images (with the maximum number of feature matches to this image)
   (ii) Find geometrically consistent feature matches using RANSAC to solve for the fundamental matrix between pairs of images
   (iii) (Future work) Verify image matches using a probabilistic model

IV. Find connected components of image matches

V. For each connected component:
   (i) Perform sparse bundle adjustment to solve for the rotation $\theta_1, \theta_2, \theta_3$, translation t_1, t_2, t_3 and focal length f of all cameras, and pointwise 3D geometry
   (ii) (Future Work) Compute dense depth estimates, triangulate, texture map etc.

Output: 3D model(s)

connected components of image matches and 6 unmatched images, and output the 2 reconstructed 3D models. The complete algorithm ran in 556 seconds on a 2.8GHz PC. About half of the computation time was spent in bundle adjustment. Another example of fully automatic structure and motion estimation is given in figure 5.5.

5.6 Summary

We have presented a fully automatic 3D object recognition and reconstruction system. Our system starts by extracting SIFT features from a collection of images, and recognises 3D objects/scenes as geometrically consistent sets of feature matches. We perform bundle adjustment for metric structure directly, without the initial projective

reconstruction common to other approaches. We have found that initialising each new camera with the same parameters as the best matching camera gives no problems with convergence.

Future Work

We close the chapter by noting some possible areas for future work development of the rigid object recognition/reconstruction system. We leave discussion of more general image matching and object class recognition for the final conclusions (chapter 6).

Probabilistic Model for Image Match Verification An important part of the object recognition system is the ability to identify whether a given set of feature matches represents a correct or incorrect image match. Currently this has been implemented using the feature space test of 2.7.1 and simply thresholding on the number of inliers to the fundamental matrix $\mathbf{F}$ between each image pair. An important area for future work would be to develop are more principled model for verifying correct image matches. One possibility would be to extend the model of 3.3.2 to include parallax.

Multi-View Tensors In this work we have used pairwise geometric constraints to reject outliers. These constraints are fairly weak because corresponding points may lie anywhere along an epipolar line. This gives the potential for introducing outliers with unrealistic (e.g. negative) depths. Such outliers could be identified by using 3 view matching constraints. For example, one could robustly estimate the trifocal tensor [Har96] using RANSAC, and reject matches which are inconsistent between the 3 views.

Multiple Rigid Motions A straightforward extension of the ideas presented in this chapter would be to associate more than one set of camera parameters with each image to enable multiple independently moving rigid objects to be discovered in the images. This would require a model selection/cross validation stage to determine how many objects are present in each image.

Photorealistic 3D Modelling The algorithm presented in this chapter gives a method for automatic camera calibration from unordered image datasets. Given calibrated cameras, various methods could be applied to generate photorealistic 3D

models. A traditional approach would be to perform dense stereo and extract a triangle mesh [CSG03]. Another method would be to represent the scene as a voxel grid which is coloured using an algorithm such as [SD99]. These representations have depth ambiguities for areas which are viewed in only one camera, or areas that have constant intensity [BSK01]. One way to deal with such ambiguities is to adopt an image based rendering approach, using priors based on the image statistics [FWZ03].

Chapter 6

Conclusions

This thesis has presented a practical approach to multi-image matching for the automatic discovery and reconstruction of image panoramas and 3D models from image datasets. We have proposed an invariant feature based approach, using geometric constraints to find structure in an unordered dataset. There are several advantages to such an approach. Firstly, by organising the feature database as a tree structure one can reduce the complexity of matching n images from $O(n^2)$ for naive pairwise matching to $O(n \log n)$. Secondly, the geometric constraints for multiple views are stronger than their pairwise counterparts, which allows more incorrect matches to be rejected. Finally, we can exploit the probabilistic nature of an n image matching problem by using known incorrect matches in data driven classifiers.

The major contribution of this work has been the development of a novel image stitching algorithm, that can automatically recognise and stitch high quality image panoramas from unsorted image datasets. We have also described a new type of invariant feature – Multi-Scale Oriented Patches (MOPS) that are especially suited for this purpose. We have developed a framework for evaluation of multi-image matching methods and used it to test the performance and tune the parameters of our image stitching algorithms. Finally, we have also shown how our framework can also be applied to the problem of structure and motion recovery in unordered datasets.

Future Directions

We conclude by identifying some avenues for future exploration:

Partially Invariant Features The results of section 2.9.1 demonstrate the pitfalls of adding *too much* invariance to feature descriptors. One major improvement to feature matching methods would be to use feature descriptors that are *partially* invariant under some transformations. For many parameters such as affine stretch, full invariance is not really desirable, and we would rather specify a prior

probability distribution on those parameters. For example, very large or small stretch factors would be very unlikely, and difficult to match due to sampling issues. Whilst it would be straightforward to add priors to the Lucas-Kanade refinement framework, performing indexing with partial invariance would be more of a challenge.

Computational Photography In chapters 3 and 5 we discussed automatic image stitching and 3D modelling from multiple views. These problems can be thought of within the general framework of *computational photography*, where one attempts to augment the capabilities of a digital imaging device using computational processing. In the context of stitching and 3D modelling the most obvious application is novel view synthesis – generating virtual camera views with arbitrary positions/orientations from a collection of images. However, one could also generate novel views which have virtual exposures [DM97] or digital refocussing [Ng05]. In the future, there may be no such thing as a 'bad' photograph, because the shooting conditions will be completely reconfigurable after the event has been recorded. An even bigger challenge is to capture and resynthesise *dynamic* scenes [AZP+05].

Non-Rigid Object Recognition In chapter 5 we describe a system for recognising rigid objects in an unordered dataset. A straightforward extension of this work would be to relax the global geometric constraints and instead search for sets of feature matches that are consistent with local geometric constraints. This would enable multi-image matching of non-rigid objects or scenes.

Object Class Recognition Recent results in neuroscience [IUMH00] and machine learning [FPZ03] have shown that both humans and machines can detect *classes* of objects. Progress in this area will require flexible models for correspondence with similarity metrics linked to human perception.

Image Searching and Sorting With the explosion of digital photography, our ability to understand images has not kept pace with our ability to generate them. In the future, algorithms for searching and sorting in image databases will become as fundamental as those for searching for text on the World-Wide Web. Such algorithms could borrow from the ideas of Brin and Page[PBMW98], using feature matches as the visual analogue of text hyperlinks on the web.

Bibliography

[2D3] 2d3. http://www.2d3.com.

[ADA+04] A. Agarwala, M. Dontcheva, M. Agarwala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen. Interactive digital photomontage. In *ACM Transactions on Graphics (SIGGRAPH'04)*, 2004.

[Ana89] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[AZP+05] A. Agarwala, C. Zheng, C. Pal, M. Agrawala, M. Cohen, B. Curless, D. Salesin, and R. Szeliski. Panoramic video textures. In *ACM Transactions on Graphics (SIGGRAPH'05)*, 2005.

[BA83] P. Burt and E. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics*, 2(4):217–236, 1983.

[BAHH92] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierachical model-based motion estimation. In *Proceedings of the 2nd European Conference on Computer Vision (ECCV92)*, pages 237–252. Springer-Verlag, May 1992.

[Bau00] A. Baumberg. Reliable feature matching across widely separated views. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR00)*, pages 774–781, 2000.

[BBZ96] B. Bascle, A. Blake, and A. Zisserman. Motion deblurring and super-resolution from and image sequence. In *Proceedings of the 4th European Conference on Computer Vision (ECCV96)*, pages 312–320. Springer-Verlag, 1996.

[BI98] A. Blake and M. Isard. *Active Contours.* Springer-Verlag, 1998.

[Bis95] C. Bishop. *Neural Networks for Pattern Recognition.* Clarendon, Oxford, UK, 1995.

[BK01] R. Benosman and S. Kang. *Panoramic Vision: Sensors, Theory and Applicatons.* Springer, New York, 2001.

[BL97] J. Beis and D. Lowe. Shape indexing using approximate nearest-neighbor search in high-dimensional spaces. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR97)*, pages 1000–1006, 1997.

[BL02] M. Brown and D. Lowe. Invariant features from interest point groups. In *Proceedings of the 13th British Machine Vision Conference (BMVC02)*, pages 253–262, Cardiff, 2002.

[BL03] M. Brown and D. Lowe. Recognising panoramas. In *Proceedings of the 9th International Conference on Computer Vision (ICCV03)*, volume 2, pages 1218–1225, Nice, October 2003.

[BL05] M. Brown and D. Lowe. Unsupervised 3D object recognition and reconstruction in unordered datasets. In *5th International Conference on 3D Imaging and Modelling (3DIM05)*, 2005.

[BM04] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework: Part 1: The quantity approximated, the warp update rule, and the gradient descent approximation. *International Journal of Computer Vision*, 56(3):221–255, March 2004.

[Bro58] D. Brown. A solution to the general problem of multiple station analytical stereotriangulation. Technical report, Patrick Airforce Base, 1958.

[Bro71] D. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37(8):855–866, 1971.

[BSK01] S. Baker, T. Sim, and T. Kanade. A characterization of inherent stereo ambiguities. In *Proceedings of the 8th International Conference on Computer Vision (ICCV01)*, volume 1, pages 428–435, Vancouver, July 2001.

[BSW05] M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale oriented patches. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR05)*, San Diego, June 2005.

[BTZ96] P. Beardsley, P. Torr, and A. Zisserman. 3D model acquisition from extended image sequences. In *Proceedings of the 4th European Conference on Computer Vision (ECCV96)*, volume II, pages 683–695, Cambridge, April 1996. Springer-Verlag.

[Can86] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[Che95] S. Chen. QuickTime VR – An image-based approach to virtual environment navigation. In *SIGGRAPH'95*, volume 29, pages 29–38, 1995.

[CJ03] G. Carneiro and A. Jepson. Multi-scale local phase-based features. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR03)*, 2003.

[CRB99] R. Cipolla, D. Robertson, and E. Boyer. 3D models of architectural scenes from uncalibrated images. In *IEEE International Conference on Multimedia Computing and Systems*, 1999.

[CSG03] T. Tuytelaars C. Strecha and L. Van Gool. Dense matching of multiple wide-baseline views. In *Proceedings of the 9th International Conference on Computer Vision (ICCV03)*, 2003.

[CZ98] D. Capel and A. Zisserman. Automated mosaicing with super-resolution zoom. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR98)*, pages 885–891, June 1998.

[Dav98] J. Davis. Mosaics of scenes with moving objects. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR98)*, pages 354–360, 1998.

[DBdFF02] P. Duygulu, K. Barnard, N. de Freitas, and D. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabu-

lary. In *Proceedings of the 7th European Conference on Computer Vision (ECCV02)*, pages 97–112, 2002.

[DC99] T. Drummond and R. Cipolla. Real-time tracking of complex structures with on-line camera calibration. In *Proceedings of the 10th British Machine Vision Conference (BMVC99)*, pages 574–583, Nottingham, 1999.

[DM97] P. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. *Computer Graphics*, 31:369–378, 1997.

[DSTT00] F. Dellaert, S. Seitz, C. Thorpe, and S. Thrun. Structure from motion without correspondence. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR00)*, pages 557–564, June 2000.

[Fau93] O. Faugeras. *Three-Dimensional Computer Vision – A Geometric Viewpoint.* MIT press, 1993.

[FB81] M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Communications of the ACM*, 24:381–395, 1981.

[For86] W. Forstner. A feature-based correspondence algorithm for image matching. *Intl. Arch. Photogrammetry & Remote Sensing*, 26(3):150–166, 1986.

[FPZ03] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR03)*, 2003.

[FS03] J. Flusser and T. Suk. Construction of complete and independent systems of rotation moment invariants. In *Computer Analysis of Images and Patterns: 10th International Conference (CAIP2003)*, pages 41–48, Groningen, Netherlands, 2003.

[FWZ03] A. Fitzgibbon, Y. Wexler, and A. Zisserman. Image-based rendering using image-based priors. In *Proceedings of the 9th International Conference on Computer Vision (ICCV03)*, 2003.

[FZ03] W. Freeman and H. Zhang. Shape-time photography. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR03)*, volume 2, pages 151–157, June 2003.

[GL81] J. George and J. Liu. *Computer Solution of Large Sparse Positive Definite Systems.* Prentice-Hall, 1981.

[GM00] A. Gray and A. Moore. 'N-Body' problems in statistical learning. In *NIPS00*, pages 521–527, 2000.

[Har92] C. Harris. Geometry from visual motion. In A. Blake and A. Yuille, editors, *Active Vision*, pages 263–284. MIT Press, 1992.

[Har94] R. Hartley. Self-calibration from multiple views with a rotating camera. In *Proceedings of the 3rd European Conference on Computer Vision (ECCV94)*, pages 471–478, Stockholm, 1994.

[Har96] R. Hartley. Lines and points in three views and the trifocal tensor. *International Journal of Computer Vision*, 22(2):125–140, 1996.

[HS03] A. Hertzmann and S. Seitz. Shape and materials by example: A photometric stereo approach. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR03)*, volume 1, pages 533–540, Madison, WI, June 2003.

[HW62] D. Hubel and T. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962.

[HZ04] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[IA99] M. Irani and P. Anandan. About direct methods. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, number 1883 in LNCS, pages 267–277. Springer-Verlag, Corfu, Greece, September 1999.

[IUMH00] A. Ishai, L. Ungerleider, A. Martin, and J. Haxby. The representation of objects in the human occipital and temporal cortex. *Journal of Cognitive Neuroscience*, 12(2):35–51, 2000.

[KSU03] S. Kang, R. Szeliski, and M. Uyttendaele. Seamless stitching using multi-perspective plane sweep. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR03)*, Madison, WI, June 2003.

[KW99] S. Kang and R. Weiss. Characterization of errors in compositing panoramic images. *Computer Vision and Image Understanding*, 73(2):269–280, February 1999.

[KZB04] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. In *Proceedings of the 8th European Conference on Computer Vision (ECCV04)*, pages 404–416, 2004.

[Lin98] T. Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, 1998.

[LK81] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 674–679, 1981.

[Low99] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the 7th International Conference on Computer Vision (ICCV99)*, pages 1150–1157, Corfu, Greece, September 1999.

[Low04] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[LS05] A. Litvinov and Y. Schechner. Adressing radiometric nonidealities: A unified framework. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR05)*, volume 2, pages 52–59, 2005.

[MCUP02] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the 13th British Machine Vision Conference (BMVC02)*, 2002.

[Mee90] J. Meehan. *Panoramic Photography.* Amphoto Books, September 1990.

[Mil75] D. Milgram. Computer methods for creating photomosaics. *IEEE Transactions on Computers*, C-24(11):1113–1119, November 1975.

[MJ02] P. McLauchlan and A. Jaenicke. Image mosaicing using sequential bundle adjustment. *Image and Vision Computing*, 20(9-10):751–759, August 2002.

[MP79] D. Marr and T. Poggio. A computational theory of human stereo vision. *Proceedings of the Royal Society of London*, B204:301–328, 1979.

[MS02] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proceedings of the 7th European Conference on Computer Vision (ECCV02)*, 2002.

[MS03] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR03)*, 2003.

[MSF] Microsoft Digital Image Pro. http://www.microsoft.com/products/imaging.

[MZ92] J. Mundy and A. Zisserman, editors. *Geometrical Invariance in Computer Vision.* MIT Press, 1992.

[Ng05] R. Ng. Fourier slice photography. In *ACM Transactions on Graphics (SIGGRAPH'05)*, 2005. To appear.

[NN97] S. Nene and S. Nayar. A simple algorithm for nearest neighbour search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):989–1003, September 1997.

[OTdF+04] K. Okuma, A. Taleghani, N. de Freitas, J. Little, and D. Lowe. A boosted particle filter: Multitarget detection and tracking. In *Proceedings of the 8th European Conference on Computer Vision (ECCV04)*, number 3021 in LNCS, pages 28–39. Springer-Verlag, 2004.

[PBMW98] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[PKG99] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. *International Journal of Computer Vision*, 1999.

[Pol00] M. Pollefeys. 3D modelling from images. In *Tutorial organised in conjunction with ECCV 2000*, Dublin, June 2000. Tutorial notes.

[Ray98] K. Rayner. Eye movements in reading and information processing: 20 years of research. *Pyschological Bulletin*, 124(3):372–422, 1998.

[RC02] C. Rother and S. Carlsson. Linear multi view reconstruction and camera recovery using a reference plane. *International Journal of Computer Vision*, 49(2/3):117–141, 2002.

[REA] Realviz. http://www.realviz.com.

[RLSP03] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3D object modelling and recognition using affine-invariant patches and multi-view spatial constraints. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR03)*, pages 272–277, Madison, WI, June 2003.

[RZFM92] C. Rothwell, A. Zisserman, D. Forsyth, and J. Mundy. Canonical frames for planar object recognition. In *Proceedings of the 2nd European Conference on Computer Vision (ECCV92)*, pages 757–772, 1992.

[RZFM95] C. Rothwell, A. Zisserman, D. Forsyth, and J. Mundy. Planar object recognition using projective shape representation. In *International Journal of Computer Vision*, number 16, pages 57–99, 1995.

[Saw94] H. Sawhney. 3D geometry from planar parallax. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR94)*, pages 929–934, Seattle, WA, 1994.

[SD99] S. Seitz and C. Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35(2), 1999.

[SF95] E. Simoncelli and W. Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *International Conference*

on Image Processing, volume 3, pages 444–447, 23-26 Oct. 1995, Washington, DC, USA, 1995.

[Sha98] R. Shachter. Bayes-ball: The rational pastime for determining irrelevance and requisite information in belief networks and influence diagrams. In G. Cooper and S. Moral, editors, *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pages 480–487, San Francisco, CA, 1998. Morgan Kaufmann.

[SHS+] H. Seetzen, W. Heidrich, W. Stuerzlinger, G. Ward, L. Whitehead, M. Trentacoste, A. Ghosh, and A. Vorozcovs. High dynamic range display systems. In *ACM Transactions on Graphics (SIGGRAPH'04)*.

[SK93] R. Szeliski and S. Kang. Recovering 3D shape and motion from image streams using non-linear least squares. Technical Report CRL 93/3, Digital Equipment Corporation, Cambridge Research Laboratory, March 1993.

[SK95] R. Szeliski and S. Kang. Direct methods for visual scene reconstruction. In *IEEE Workshop on Representations of Visual Scenes*, pages 26–33, Cambridge, MA, 1995.

[SK99] H. Sawhney and R. Kumar. True multi-image alignment and its application to mosaicing and lens distortion correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(3):235–243, 1999.

[Sla80] C. Slama, editor. *Manual of Photogrammetry.* American Society of Photogrammetry, Falls Church, Virginia, fourth edition edition, 1980.

[SLDV96] P. Simard, Y. LeCun, J. Denker, and B. Victorri. Transformation invariance in pattern recognition-tangent distance and tangent propagation. In *Neural Networks: Tricks of the Trade*, pages 239–27, 1996.

[SM97] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, May 1997.

[SMB98] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. In *Proceedings of the 6th International Conference on Computer Vision (ICCV98)*, pages 230–235, Bombay, 1998.

[SS97] R. Szeliski and H. Shum. Creating full view panoramic image mosaics and environment maps. *Computer Graphics (SIGGRAPH'97)*, 31(Annual Conference Series):251–258, 1997.

[SS00] H. Shum and R. Szeliski. Construction of panoramic mosaics with global and local alignment. *International Journal of Computer Vision*, 36(2):101–130, February 2000.

[SS02] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, May 2002.

[ST94] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR94)*, Seattle, June 1994.

[Ste95] G. Stein. Accurate internal camera calibration using rotation, with analysis of sources of error. In *Proceedings of the 5th International Conference on Computer Vision (ICCV95)*, pages 230–236, 1995.

[SVD03] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter sensitive hashing. In *Proceedings of the 9th International Conference on Computer Vision (ICCV03)*, pages 750–757, Nice, October 2003.

[SZ02] F. Schaffalitzky and A. Zisserman. Multi-view matching for unordered image sets, or "How do I organise my holiday snaps?". In *Proceedings of the 7th European Conference on Computer Vision (ECCV02)*, pages 414–431, 2002.

[SZ03] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the 9th International Conference on Computer Vision (ICCV03)*, October 2003.

[Sze04] R. Szeliski. Image alignment and stitching: A tutorial. Technical Report MSR-TR-2004-92, Microsoft Research, December 2004.

[Tan97] K. Tanaka. Mecahnisms of visual object recognition: Monkey and human studies. *Current Opinion in Neurobiology*, (7):523–529, 1997.

[TB01] K. Toyama and A. Blake. Probabilistic tracking in a metric space. In *Proceedings of the 8th International Conference on Computer Vision (ICCV01)*, volume 2, pages 50–57, 2001.

[TG00] T. Tuytelaars and L. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *Proceedings of the 11th British Machine Vision Conference (BMVC00)*, pages 412–422, Bristol, UK, 2000.

[TM97] P. Torr and D. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal of Computer Vision*, 24(3):271–300, 1997.

[TMHF99] W. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment: A modern synthesis. In *Vision Algorithms: Theory and Practice*, number 1883 in LNCS, pages 298–373. Springer-Verlag, Corfu, Greece, September 1999.

[Tor02] P. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *International Journal of Computer Vision*, 50(1):35–61, 2002.

[Tri04] B. Triggs. Detecting keypoints with stable position, orientation, and scale under illumination changes. In *Proceedings of the 8th European Conference on Computer Vision (ECCV04)*, volume 4, pages 100–113, Prague, May 2004.

[TZ99] P. Torr and A. Zisserman. Feature based methods for structure and motion estimation. In B. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, number 1883 in LNCS, pages 278–294. Springer-Verlag, Corfu, Greece, September 1999.

[UES01] M. Uyttendaele, A. Eden, and R. Szeliski. Eliminating ghosting and exposure artifacts in image mosaics. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR01)*, volume 2, pages 509–516, Kauai, Hawaii, December 2001.

[VJ01] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the Interational Conference on Computer Vision and Pattern Recognition (CVPR01)*, December 2001.

[ZFD97] I. Zoghlami, O. Faugeras, and R. Deriche. Using geometric corners to build a 2D mosaic from a set of images. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition, Puerto Rico.* IEEE, June 1997.

Appendix A

Multiple View Geometry

A.1 Camera Models

A camera induces a mapping between the 3-dimensional world and the 2-dimensional image. In order to form invariants that describe the images of points in the world, we must understand this mapping. In the next sections we describe pinhole, orthographic, projective and affine camera models, and characterise their properties when viewing a plane. For a more detailed presentation of these results, see [HZ04].

A.1.1 Pinhole Camera

In its simplest form, a camera is a device which interprets 3-dimensional points in the world ($\mathbb{R}^3$) as points of a 2-dimensional projective space ($\mathbb{P}^2$).

$$\tilde{\mathbf{x}} = \mathbf{X}_c \tag{A.1}$$

where $\mathbf{X}_c = [X, Y, Z]$ is the position in camera coordinates, and $\tilde{\mathbf{x}} = s[x, y, 1]$ is the homogeneous image position. Hence, the actual image coordinates $\mathbf{x} = [x, y]$ are given by

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix} \tag{A.2}$$

See figure A.1. This is the basic pinhole camera model. Assuming a rigid body transformation $\mathbf{X}_c = \mathbf{R}\mathbf{X} + \mathbf{t}$ between world coordinates $\mathbf{X}$ and camera coordinates $\mathbf{X}_c$, and a linear transformation in the image plane $\tilde{\mathbf{u}} = \mathbf{K}\tilde{\mathbf{x}}$, we have

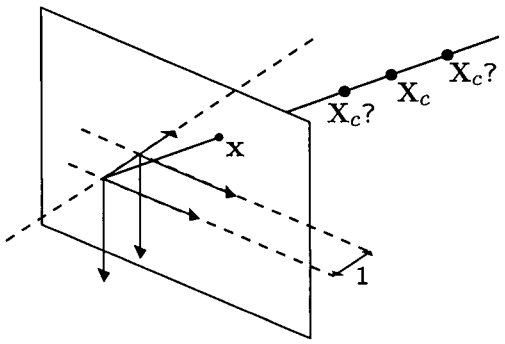


Figure A.1: The pinhole camera interprets points in $\mathbb{R}^3$ as points of $\mathbb{P}^2$. Points $\mathbf{X}_c$ that lie on a line through the optical centre are equivalent in $\mathbb{P}^2$ as they map to the same point $\mathbf{x}$ on the plane $Z = 1$.

$$\tilde{\mathbf{u}} = \mathbf{K}(\mathbf{R}\mathbf{X} + \mathbf{t}) \tag{A.3}$$

$$s\begin{bmatrix} u_1 \\ u_2 \\ 1 \end{bmatrix} = \begin{bmatrix} f_1 & 0 & u_{1_0} \\ 0 & f_2 & u_{2_0} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \tag{A.4}$$

where $\mathbf{R}$, $\mathbf{t}$ are the rotation and translation of the camera (extrinsic parameters) and $\mathbf{K}$ is the calibration matrix (intrinsic parameters). The actual image position is $\mathbf{u} = [u_1, u_2]$ and the world position is $\mathbf{X} = [X_1, X_2, X_3]$. Here, we parameterise $\mathbf{K}$ by focal lengths f_1, f_2, and the position of the principal point u_{1_0}, u_{2_0}. Parameterisation of $\mathbf{R}$ is discussed in appendix A.7.

A.1.2 Projective Camera

Sometimes it is convenient to use a linear model for the mapping between homogeneous world and image coordinates. This can be accomplished by relaxing the non-linear constraints in equation A.3 (e.g. that $\mathbf{R}^T\mathbf{R} = \mathbf{I}$)

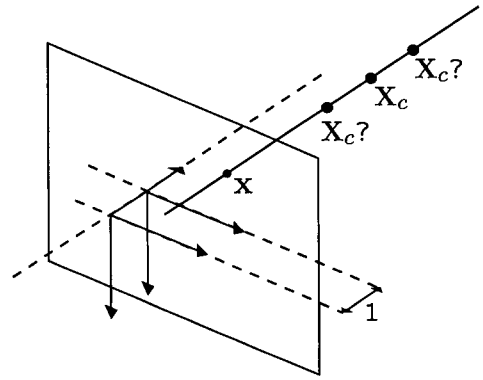


Figure A.2: The orthographic camera projects points $\mathbf{X}_c$ parallel to the Z axis (with scaling in the image plane based on the average depth). Points $\mathbf{X}_c$ that lie on a line perpendicular to the plane $Z = 1$ project to the same point $\mathbf{x}$.

$$\tilde{\mathbf{u}} = \mathbf{P}\tilde{\mathbf{X}} \tag{A.5}$$

$$s\begin{bmatrix} u_1 \\ u_2 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \tag{A.6}$$

where $\mathbf{P}$ is an arbitrary 3×4 matrix. This is known as a *projective camera*. This enables closed form linear solutions for the camera parameters given image and 3D correspondences [HZ04]. Projective cameras have the property that straight lines in the world map to straight lines in the image. This is also known as *rectilinear* projection in photography.

A.1.3 Orthographic Camera

A simplified camera model assumes that the camera depth $Z = Z_0 + \triangle Z$ and the depth variation $\triangle Z$ is small. Hence

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} X/Z_0 \\ Y/Z_0 \end{bmatrix} \tag{A.7}$$

This is the basic orthographic camera model (figure A.2). In this model world points

are projected perpendicular to the image plane, and then scaled based on their average depth.

A.1.4 Affine Camera

Substituting for the rigid body and linear transforms, and linearising the result gives

$$\tilde{\mathbf{u}} = \mathbf{P}_a \tilde{\mathbf{X}} \tag{A.8}$$

$$s \begin{bmatrix} u_1 \\ u_2 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ 1 \end{bmatrix} \tag{A.9}$$

which gives a linear relationship between the ordinary (non-homogeneous) world and image coordinates. This is known as an *affine camera.*

A.2 Viewing a Plane

Given the above camera models, we now wish to characterise the geometric invariants of the imaging process. Unfortunately, the geometric relationships between corresponding image regions is complex, depending on the unknown 3D structure. One way to proceed is to linearise this structure, which is equivalent to viewing a planar region. In this section we describe the mappings between images of planes given the camera models developed above.

A.2.1 Homography

Consider viewing a plane with a projective camera. Without loss of generality, assume that this is the plane $X_3 = 0$. Then

$$s \begin{bmatrix} u_1 \\ u_2 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ 1 \end{bmatrix} \tag{A.10}$$

or

$$\tilde{\mathbf{u}} = \mathbf{H}\tilde{\mathbf{X}}_p \tag{A.11}$$

If two cameras view the same plane

$$\tilde{\mathbf{u}}_1 = \mathbf{H}_1\mathbf{X}_p \tag{A.12}$$
$$\tilde{\mathbf{u}}_2 = \mathbf{H}_2\mathbf{X}_p \tag{A.13}$$

so

$$\tilde{\mathbf{u}}_1 = \mathbf{H}_{12}\tilde{\mathbf{u}}_2 \tag{A.14}$$

where $\mathbf{H}_{12} = \mathbf{H}_1\mathbf{H}_2^{-1}$. Writing this in terms of image coordinates

$$s\begin{bmatrix} u_{11} \\ u_{12} \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}\begin{bmatrix} u_{21} \\ u_{22} \\ 1 \end{bmatrix} \tag{A.15}$$

So views of a plane with projective cameras are related by a matrix multiplication in homogeneous coordinates, known as a homography. Since the homography has the property that straight lines map to straight lines, it is also known as a collineation [Fau93]. A homography has 8 degrees of freedom corresponding to translation (2), rotation, scale, shear (2), and perspective (2) (characterised by the horizon line).

A.2.2 Affine Transformation

Similarly, when viewing the plane $X_3 = 0$ with an affine camera

$$s\begin{bmatrix} u_1 \\ u_2 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ 1 \end{bmatrix} \tag{A.16}$$

So views are related by

$$\tilde{\mathbf{u}}_1 = \mathbf{A}_{12}\tilde{\mathbf{u}}_2 \tag{A.17}$$

where $\mathbf{A}_{12} = \mathbf{A}_1\mathbf{A}_2^{-1}$. Writing in terms of image coordinates

$$s\begin{bmatrix} u_{11} \\ u_{12} \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u_{21} \\ u_{22} \\ 1 \end{bmatrix} \tag{A.18}$$

Hence views of a plane with affine cameras are related by a linear transformation in (non-homogeneous) image coordinates, called a 2D affine transformation. An affine transformation differs from a homography by the fact that parallel lines are preserved. This is easily seen from the fact that points at infinity $\tilde{\mathbf{u}} = (u_1, u_2, 0)$ (where parallel lines intersect) remain at infinity under an affinity, but not a homography. An affine transform has 6 degrees of freedom corresponding to translation (2), rotation, scale and shear (2).

A.2.3 Similarity Transformation

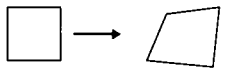
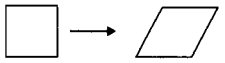
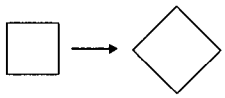
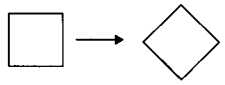
A further simplification can be made if there is no rotation in depth of the plane relative to the cameras

$$s\begin{bmatrix} u_{11} \\ u_{12} \\ 1 \end{bmatrix} = \begin{bmatrix} a\cos\theta & -a\sin\theta & t_1 \\ a\sin\theta & a\cos\theta & t_2 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u_{21} \\ u_{22} \\ 1 \end{bmatrix} \tag{A.19}$$

This is known as a similarity transform. The similarity transform has four degrees of freedom corresponding to translation (2), rotation and scale.

A.3 Hierarchy of 2D Transformations

The formulas for homographies, affinities and similarities derived above no longer depend on the 3D geometry, so planar regions differ only by a planar projective transformation. Some examples of these invariants under these transformations are given below

Group	Example	Projection matrix	Invariants
Projective		$\begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix}$	cross ratio, intersection
Affine		$\begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ 0 & 0 & 1 \end{bmatrix}$	+ parallel lines, ratio of areas
Similarity		$\begin{bmatrix} a\cos\theta & -a\sin\theta & t_1 \\ a\sin\theta & a\cos\theta & t_2 \\ 0 & 0 & 1 \end{bmatrix}$	+ ratio of lengths, angles
Euclidean		$\begin{bmatrix} \cos\theta & -\sin\theta & t_1 \\ \sin\theta & \cos\theta & t_2 \\ 0 & 0 & 1 \end{bmatrix}$	+ length, area

One approach to matching and object recognition is to characterise objects by geometric invariants. For example, one might compute the cross ratios of colinear points and use this for indexing. One can also compute invariants from the local brightness structure. For example, the integral of intensity around a circle would be invariant under rotations about that point.

A.4 Canonical Frames

A desirable characteristic for the set of invariants chosen is that they should be complete and independent [FS03]. In other words, one should be able to exactly reproduce the region of interest given the invariant descriptor and the unknown transformation parameters (completeness). Also, the number of elements of the descriptor vector should be no greater than necessary to reproduce the original data (independence). One way to do this is to warp corresponding image regions to a canonical frame (figure A.3). Consider a reference frame with coordinates $\mathbf{u}_{ref}$ defined by a homography $\mathbf{H}_{ref}$

$$\tilde{\mathbf{u}}_{ref} = \mathbf{H}_{ref}\tilde{\mathbf{u}} \tag{A.20}$$

Suppose that the coordinates $\tilde{\mathbf{u}}$ undergo a planar projective transformation $\tilde{\mathbf{u}}' = \mathbf{H}\tilde{\mathbf{u}}$.

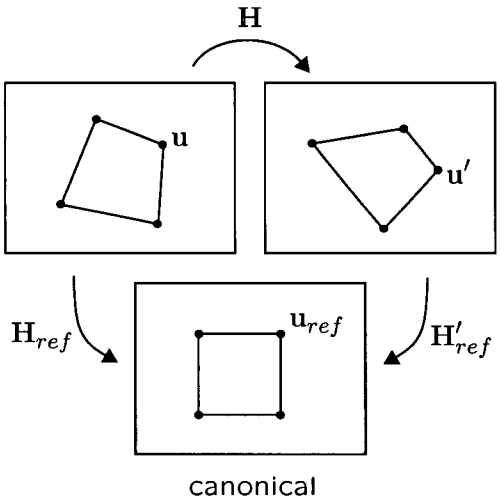


Figure A.3: A canonical reference frame transforms according to the same homography as the images i.e. $\tilde{\mathbf{u}}' = \mathbf{H}\tilde{\mathbf{u}} \implies \mathbf{H}'_{ref} = \mathbf{H}_{ref}\mathbf{H}^{-1}$

If the new reference frame $\mathbf{H}'_{ref}$ also transforms according to $\mathbf{H}$

$$\mathbf{H}'_{ref} = \mathbf{H}_{ref}\mathbf{H}^{-1} \tag{A.21}$$

then

$$\tilde{\mathbf{u}}_{ref} = \mathbf{H}_{ref}\tilde{\mathbf{u}} = \mathbf{H}'_{ref}\tilde{\mathbf{u}}' \tag{A.22}$$

Then the corresponding feature points $\mathbf{u}$, $\mathbf{u}'$ map to the same reference point $\mathbf{u}_{ref}$. The reference frame defined by $\mathbf{H}_{ref}$ is called a *canonical frame*. Note that though corresponding feature points are *covariant* (transform under $\mathbf{H}$), the reference homography is *contravariant* (transforms under $\mathbf{H}^{-1}$).

A.5 Multi-View Constraints

In the next sections we describe the pairwise constraints that apply to panoramic image geometry and epipolar geometry. Information about 3 and 4 view constraints can be found in [HZ04].

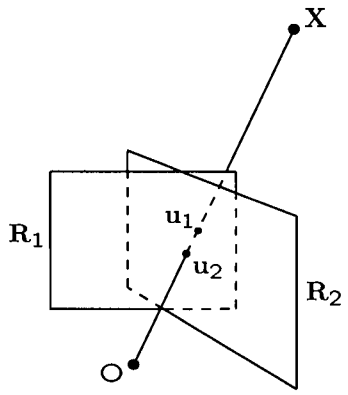


Figure A.4: Panoramic Geometry. For a camera that rotates about it's optical centre, there is a one-to-one mapping between the points in the two images.

A.5.1 Panoramic Geometry

For images related by a rotation about the camera centre there is a one-to-one mapping between corresponding points. Consider two cameras centred at $\mathbf{0}$ viewing the point $\mathbf{X}$

$$\tilde{\mathbf{x}}_1 = [\,\mathbf{R}_1 \,|\, \mathbf{0}\,]\,\tilde{\mathbf{X}} = \mathbf{R}_1\mathbf{X} \tag{A.23}$$

$$\tilde{\mathbf{x}}_2 = [\,\mathbf{R}_2 \,|\, \mathbf{0}\,]\,\tilde{\mathbf{X}} = \mathbf{R}_2\mathbf{X} \tag{A.24}$$

Hence

$$\tilde{\mathbf{x}}_1 = \mathbf{R}_1\mathbf{R}_2^{-1}\tilde{\mathbf{x}}_2 \tag{A.25}$$

Substituting $\tilde{\mathbf{u}} = \mathbf{K}\tilde{\mathbf{x}}$ gives

$$\tilde{\mathbf{u}}_1 = \mathbf{H}_{12}\tilde{\mathbf{u}}_2 \tag{A.26}$$

where

$$\mathbf{H}_{12} = \mathbf{K}_1\mathbf{R}_1\mathbf{R}_2^{-1}\mathbf{K}_2^{-1} \tag{A.27}$$

is a special homography known as the homography of the plane at infinity. In the general case when the camera centres are distinct, the homography of the plane at infinity

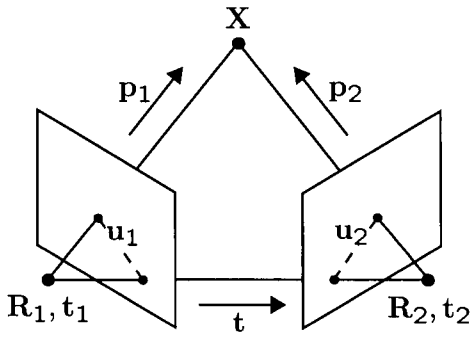


Figure A.5: The epipolar constraint expresses the fact that the two camera centres and the 3D point lie in the same plane (the epipolar plane). Hence the scalar triple product $\mathbf{p}_1^T(\mathbf{t} \times \mathbf{p}_2)$ is equal to zero.

describes the motion between points that are (infinitely) far from both camera centres (see appendix A.6). If the camera centres are coincident, this homography describes the motion of all points. The elements of $\mathbf{H}$ are typically linearised to facilitate linear solution by SVD.

A.5.2 Epipolar geometry

In the general case of moving cameras a point viewed in one image defines a line in the second, corresponding to a continuous range of possible depths for that point. Hence there is a point to line mapping between images. Consider a point X viewed in a pair of images

$$\tilde{\mathbf{x}}_1 = \mathbf{X}_{c_1} \qquad \mathbf{X} = \mathbf{R}_1\mathbf{X}_{c_1} + \mathbf{t}_1 \tag{A.28}$$

$$\tilde{\mathbf{x}}_2 = \mathbf{X}_{c_2} \qquad \mathbf{X} = \mathbf{R}_2\mathbf{X}_{c_2} + \mathbf{t}_2 \tag{A.29}$$

The homogeneous coordinates $\tilde{\mathbf{x}}$ are related to the normalised ray directions $\hat{\mathbf{x}}$ via the unknown depths s

$$\tilde{\mathbf{x}}_1 = s_1 \begin{bmatrix} \mathbf{x}_1 \\ 1 \end{bmatrix} = s_1 \hat{\mathbf{x}}_1 \tag{A.30}$$

$$\tilde{\mathbf{x}}_2 = s_2 \begin{bmatrix} \mathbf{x}_2 \\ 1 \end{bmatrix} = s_2 \hat{\mathbf{x}}_2 \tag{A.31}$$

For consistency of the 3D point position between the cameras

$$\mathbf{R}_1 \mathbf{X}_{c_1} + \mathbf{t}_1 = \mathbf{R}_2 \mathbf{X}_{c_2} + \mathbf{t}_2 \tag{A.32}$$

Substituting for the observed ray directions

$$\mathbf{R}_1 s_1 \hat{\mathbf{x}}_1 + \mathbf{t}_1 = s_2 \mathbf{R}_2 \hat{\mathbf{x}}_2 + \mathbf{t}_2 \tag{A.33}$$

this is of the form

$$s_1 \mathbf{p}_1 + s_2 \mathbf{p}_2 = \mathbf{t} \tag{A.34}$$

where $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{t}$ are 3-vectors as follows

$$\mathbf{p}_1 = \mathbf{R}_1 \hat{\mathbf{x}}_1 \tag{A.35}$$

$$\mathbf{p}_2 = \mathbf{R}_2 \hat{\mathbf{x}}_2 \tag{A.36}$$

$$\mathbf{t} = \mathbf{t}_2 - \mathbf{t}_1 \tag{A.37}$$

Equation A.34 gives 3 (scalar) equations in 2 unknowns. Eliminating the unknown depths s_1, s_2 gives

$$\mathbf{p}_1^T (\mathbf{t} \times \mathbf{p}_2) = 0 \tag{A.38}$$

This expresses the fact that $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{t}$ all lie in the same plane, and hence their scalar triple product is equal to zero. Writing this out in terms of the camera extrinsic parameters

$$\hat{\mathbf{x}}_1^T \mathbf{R}_1^T [\mathbf{t}_2 - \mathbf{t}_1]_\times \mathbf{R}_2 \hat{\mathbf{x}}_2 = 0 \tag{A.39}$$

or $\hat{\mathbf{x}}_1^T \mathbf{E} \hat{\mathbf{x}}_2 = 0$ where

$$\mathbf{E} = \mathbf{R}_1^T [\mathbf{t}_2 - \mathbf{t}_1]_\times \mathbf{R}_2 \tag{A.40}$$

is a 3×3 matrix known as the essential matrix. Substituting $\tilde{\mathbf{u}} = \mathbf{K}\tilde{\mathbf{x}}$ gives

$$\tilde{\mathbf{u}}_1^T \mathbf{K}_1^{-1} \mathbf{R}_1^T [\mathbf{t}_2 - \mathbf{t}_1]_\times \mathbf{R}_2 \mathbf{K}_2^{-1} \tilde{\mathbf{u}}_2 = 0 \tag{A.41}$$

or $\tilde{\mathbf{u}}_1^T \mathbf{F} \tilde{\mathbf{u}}_2 = 0$ where

$$\mathbf{F} = \mathbf{K}_1^{-1} \mathbf{R}_1^T [\mathbf{t}_2 - \mathbf{t}_1]_\times \mathbf{R}_2 \mathbf{K}_2^{-1} \tag{A.42}$$

is known as the fundamental matrix. Again, the fundamental matrix is often linearised to facilitate linear solution via SVD.

A.6 Plane (at Infinity) Plus Parallax

All image motion can be decomposed as a homography of the plane at infinity (rotation between the cameras) and depth dependent motion towards the epipole (parallax). Adopting similar camera models as used previously

$$\tilde{\mathbf{u}}_1 = s_1 \hat{\mathbf{u}}_1 = \mathbf{K}_1 \mathbf{X}_{c_1} \tag{A.43}$$

$$\tilde{\mathbf{u}}_2 = s_2 \hat{\mathbf{u}}_2 = \mathbf{K}_2 \mathbf{X}_{c_2} \tag{A.44}$$

where

$$\mathbf{X} = \mathbf{R}_1 \mathbf{X}_{c_1} + \mathbf{t}_1 = \mathbf{R}_2 \mathbf{X}_{c_2} + \mathbf{t}_2 \tag{A.45}$$

Assuming that $\mathbf{K}$ is upper triangular and scaled such that $k_{33} = 1$, $s = Z_c$, the unknown point depth. Substituting for $\mathbf{u}$

$$\mathbf{R}_1 \mathbf{K}_1^{-1} s_1 \hat{\mathbf{u}}_1 + \mathbf{t}_1 = \mathbf{R}_2 \mathbf{K}_2^{-1} s_2 \hat{\mathbf{u}}_2 + \mathbf{t}_2 \tag{A.46}$$

Hence

$$\frac{s_2}{s_1}\hat{\mathbf{u}}_2 = \mathbf{K}_2\mathbf{R}_2^T\mathbf{R}_1\mathbf{K}_1^{-1}\hat{\mathbf{u}}_1 + \frac{1}{s_1}\mathbf{K}_2\mathbf{R}_2^T(\mathbf{t}_1 - \mathbf{t}_2) \tag{A.47}$$

For points far from both cameras, s_1 and s_2 are both large, and the image positions are related by a homography

$$\tilde{\mathbf{u}}_2 = \mathbf{H}_\infty\tilde{\mathbf{u}}_1 \tag{A.48}$$

where

$$\mathbf{H}_\infty = \mathbf{K}_2\mathbf{R}_2^T\mathbf{R}_1\mathbf{K}_1^{-1} \tag{A.49}$$

is known as the homography of the plane at infinity. As the depth of the point decreases, it moves in a line towards $\tilde{\mathbf{e}} = \mathbf{K}_2\mathbf{R}_2^T(\mathbf{t}_1 - \mathbf{t}_2)$, which is the projection of the centre of camera 1 in camera 2, otherwise known as the *epipole*. Hence the motion of points may be written

$$\tilde{\mathbf{u}}_2 = \mathbf{H}_\infty\tilde{\mathbf{u}}_1 + \lambda\tilde{\mathbf{e}} \tag{A.50}$$

i.e. points transform under the homography of the plane at infinity (rotation between the cameras), plus depth dependent motion towards the epipole. Note that a similar result can be shown for an arbitrary plane $\mathbf{H}$. This is known as plane plus parallax representation [Saw94].

A.7 Axis-Angle Rotations

The axis-angle representation provides a representation for rotations that is a) minimal b) easily differentiable. The first point is important because some other rotation representations e.g. a 3×3 rotation matrix, can become non-Euclidean due to round off errors after manipulation.

Consider a rotation of the vector $\mathbf{x}$ by an angle θ about axis $\mathbf{u}$, resulting in the vector $\mathbf{x}'$. We begin with the identity

$$\mathbf{x} = \mathbf{u}\mathbf{u}^T\mathbf{x} - \mathbf{u}\times(\mathbf{u}\times\mathbf{x}) \tag{A.51}$$

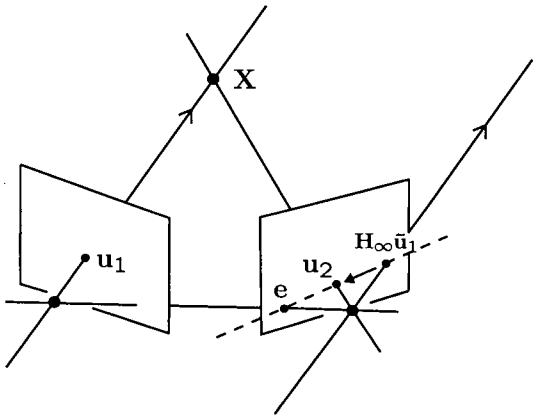


Figure A.6: The motion of point $\mathbf{X}$ between cameras can be expressed as a homography plus motion towards the epipole. If the point is far from the cameras, it's position is given by the homography of the plane at infinity. As the depth of the point Z_c decreases, it moves along a line in the image towards the epipole $\mathbf{e}$.

See figure A.7(a) . Note that the vectors

$$\begin{matrix} \mathbf{u} \\ \mathbf{u}\times\mathbf{x} \\ \mathbf{u}\times(\mathbf{u}\times\mathbf{x}) \end{matrix} \tag{A.52}$$

are orthogonal and $-\mathbf{u}\times(\mathbf{u}\times\mathbf{x})$ is the component of $\mathbf{x}$ in the plane perpendicular to $\mathbf{u}$. Under a rotation about an angle θ about $\mathbf{u}$, the component $\mathbf{u}\mathbf{u}^T\mathbf{x}$ is unchanged, but the component $-\mathbf{u}\times(\mathbf{u}\times\mathbf{x})$ becomes

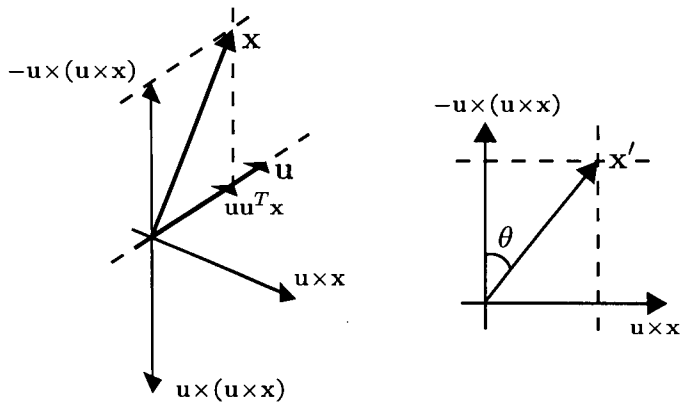
$$-\mathbf{u}\times(\mathbf{u}\times\mathbf{x})\cos\theta + (\mathbf{u}\times\mathbf{x})\sin\theta \tag{A.53}$$

Figure A.7(b) . Therefore the rotation of $\mathbf{x}$ by angle θ about axis $\mathbf{u}$ is given by

$$\mathbf{x}' = \mathbf{u}\mathbf{u}^T\mathbf{x} - \mathbf{u}\times(\mathbf{u}\times\mathbf{x})\cos\theta + (\mathbf{u}\times\mathbf{x})\sin\theta \tag{A.54}$$

Equivalently

$$\mathbf{x}' = \mathbf{R}\mathbf{x} \tag{A.55}$$



(a) (b)

Figure A.7: Axis-angle rotations. The vector $\mathbf{x}$ is expressed as components parallel to $\mathbf{u}$ and in the plane perpendicular to $\mathbf{u}$ (a). Only the component perpendicular to $\mathbf{u}$ is affected by rotation about $\mathbf{u}$ (b).

where

$$\mathbf{R} = \mathbf{u}\mathbf{u}^T + [\mathbf{u}]_\times \sin\theta + (\mathbf{I} - \mathbf{u}\mathbf{u}^T)\cos\theta \tag{A.56}$$

and $[\mathbf{u}]_\times$ is the cross product matrix defined by

$$[\mathbf{u}]_\times = \begin{bmatrix} 0 & -\theta_3 & \theta_2 \\ \theta_3 & 0 & -\theta_1 \\ -\theta_2 & \theta_1 & 0 \end{bmatrix} \tag{A.57}$$

Rodriguez Equation

It can be shown (by polynomial expansion of $e^{[\boldsymbol{\theta}]_\times}$) that

$$e^{[\boldsymbol{\theta}]_\times} = \hat{\boldsymbol{\theta}}\hat{\boldsymbol{\theta}}^T + [\hat{\boldsymbol{\theta}}]_\times \sin|\boldsymbol{\theta}| + (\mathbf{I} - \hat{\boldsymbol{\theta}}\hat{\boldsymbol{\theta}}^T)\cos|\boldsymbol{\theta}| \tag{A.58}$$

Hence

$$\mathbf{R} = e^{[\boldsymbol{\theta}]_\times} \tag{A.59}$$

represents a rotation by an angle $|\boldsymbol{\theta}|$ around the axis $\hat{\boldsymbol{\theta}}$, where $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}/|\boldsymbol{\theta}|$.

Inverting Rodriguez Equation

Consider the trace of $\mathbf{R}$

$$tr(\mathbf{R}) = 3 - 2(\hat{\theta}_1^2 + \hat{\theta}_2^2 + \hat{\theta}_3^2)(1 - \cos|\boldsymbol{\theta}|) \tag{A.60}$$

also

$$\mathbf{v} = \begin{bmatrix} \mathbf{R}_{32} - \mathbf{R}_{23} \\ \mathbf{R}_{13} - \mathbf{R}_{31} \\ \mathbf{R}_{21} - \mathbf{R}_{12} \end{bmatrix} = \begin{bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \hat{\theta}_3 \end{bmatrix} \sin|\boldsymbol{\theta}| \tag{A.61}$$

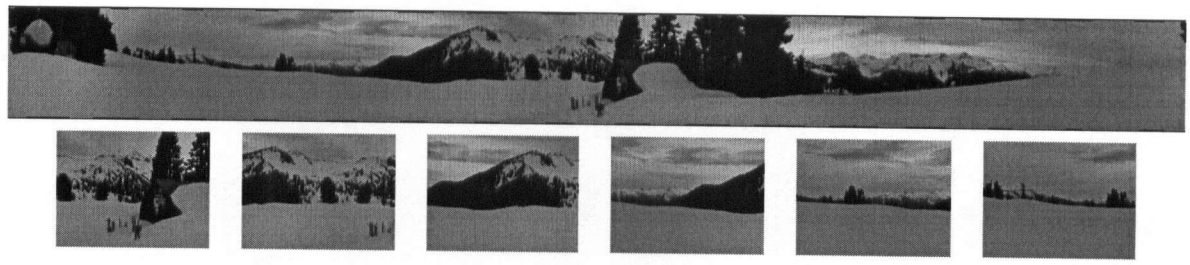
and therefore

$$|\boldsymbol{\theta}| = \cos^{-1}\frac{tr(\mathbf{R}) - 1}{2} \tag{A.62}$$

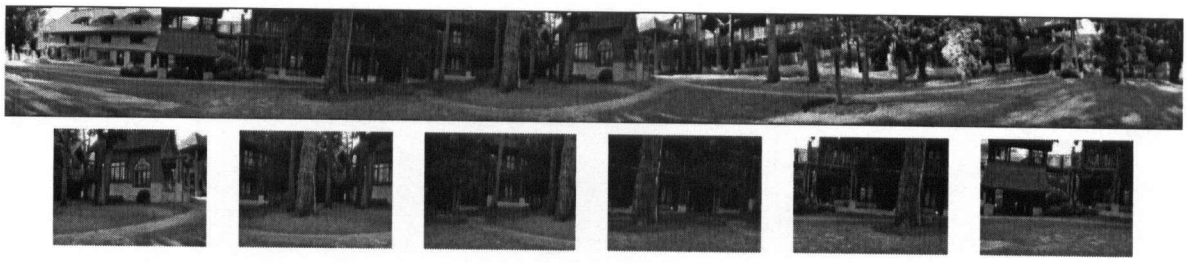
$$\hat{\boldsymbol{\theta}} = \mathbf{v}/|\mathbf{v}| \tag{A.63}$$

Appendix B

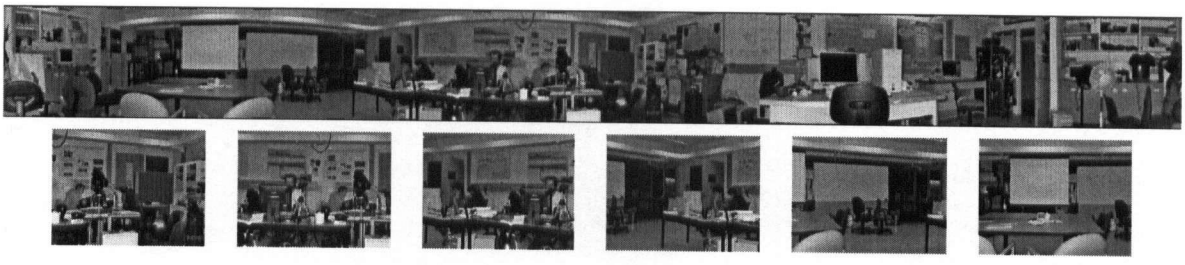
Image Datasets



(a) Elfin

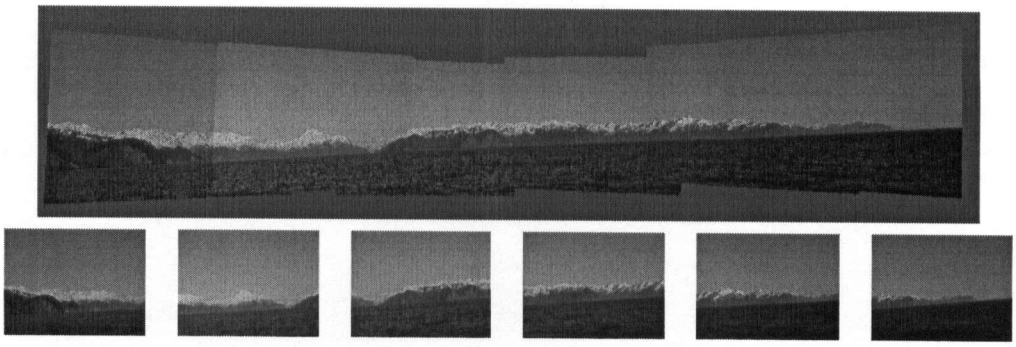


(b) Green

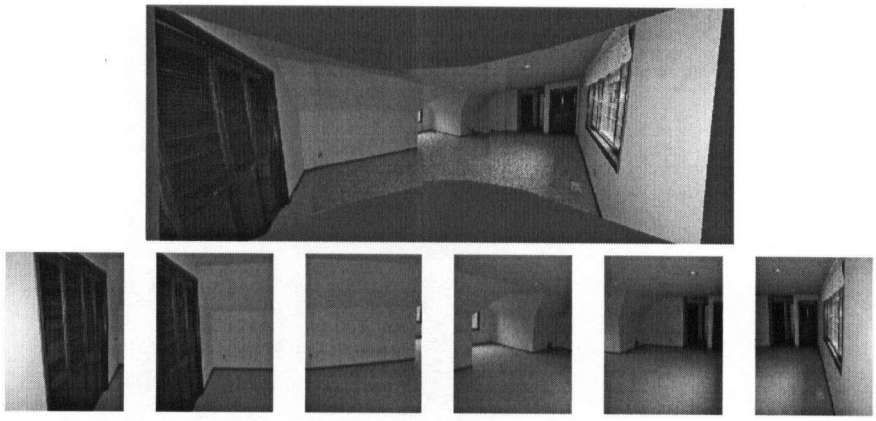


(c) LCI

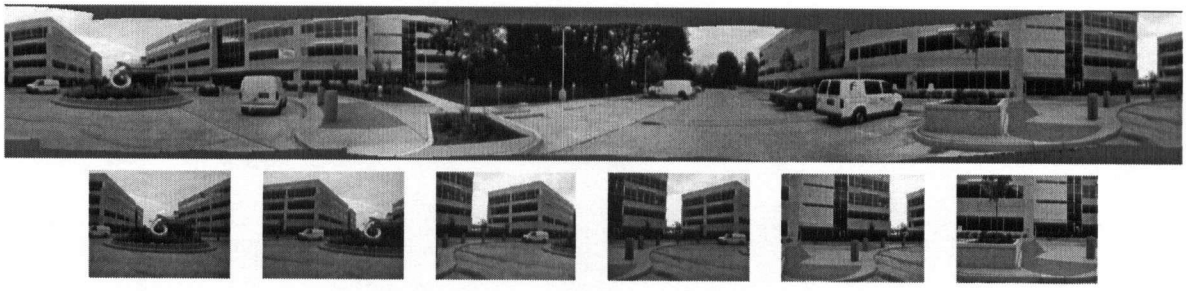
Figure B.1: Example panorama sequences from the synthetic dataset. Each camera view is synthetically sampled from a previously stitched panorama. Synthetic sampling allows us to generate arbitrary image sequences whose camera matrices are known exactly.



(a) Alaska



(b) Room



(c) Cedar

Figure B.2: Example panorama sequences from the real dataset. The ground truth registrations shown here have been generated by manual stitching. Our complete test dataset consists of over 200 real image sequences containing 1D (single row) and 2D (multi row) panoramas.

Figure B.3: Van Gogh dataset (pure rotation).

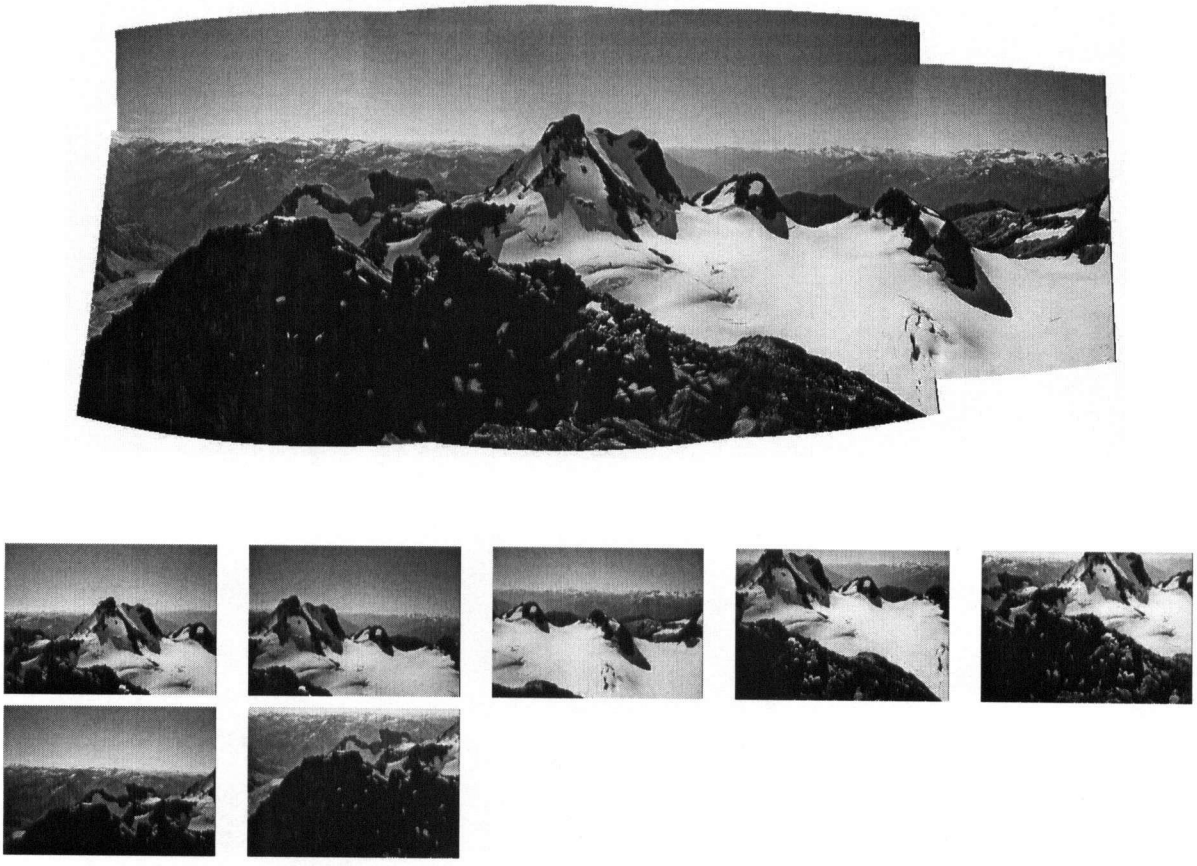


Figure B.4: Matier dataset.

Figure B.5: Abbey dataset.

Figure B.6: Dash point dataset.

Figure B.7: Times Square. This difficult stitching problem contains many moving objects and large changes in brightness between the images. Future automatic image stitchers could detect the moving objects, and compute high dynamic range radiance maps of the scene. This would enable the user to 're-photograph' the scene with different exposure settings and moving objects selected.