Self-Calibration from Image Sequences

Martin Neil Armstrong

Department of Engineering Science
University of Oxford

Michaelmas Term, 1996

This thesis is submitted to the Department of Engineering Science, University of Oxford, in partial fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work, and except where otherwise stated, describes my own research.

Martin Neil Armstrong, Lincoln College

Martin Neil Armstrong
Lincoln College

Doctor of Philosophy
Michaelmas Term, 1996

Self-Calibration from Image Sequences

Abstract

This thesis develops new algorithms to obtain the calibration parameters of a camera using only information contained in an image sequence, with the objective of using the camera calibration to compute a Euclidean reconstruction. This problem is known as self-calibration. The motivation for this work is to allow the Euclidean reconstruction of a scene using only a pre-recorded image sequence where no information is available on the camera or the objects in the scene.

The approach used is to utilise known motion constraints, which are common for cameras mounted on mobile vehicles or robotic arms, to simplify the algebraic complexity of the self-calibration problem. The algorithms are designed to be easily extendible to use multiple images rather than the minimum number of three required for self-calibration. The uncertainty of the parameters are also computed to give a measure of confidence in the camera calibration.

The first method uses a pure camera translation to allow the problem to be stratified by computing the intermediate step of an affine reconstruction. As a result the algorithm is linear and can be easily extended to multiple images.

The second method is used for a camera moving under planar motion, a fixed rotation axis and translation confined to the plane perpendicular to the axis, which is the motion of a mobile vehicle. The method is based on the use of fixed entities, points and lines in the image and in the world, whose position does not change while the camera motion is constrained to being planar. It is shown how the position of these fixed entities determines the camera calibration. An error analysis is computed for this method of self-calibration, and as a result the algorithm is adapted to give accurate estimates of the uncertainty of the different parameters.

Finally a common rigid object tracker RAPiD is extended to give RoRAPiD, which by utilising the complexity inherent in a rigid object is robust to condition such as partial occlusions. The tracker is designed to be able to use several camera models as well as different restricted motion models.

Extensive results are given for real image sequences taken by cameras mounted on several platforms, and these results are shown to be both stable and accurate.

Acknowledgements

There are many people I wish to thank, but foremost is Dr Andrew Zisserman, whose supervisions during my three years in Oxford have been a source of ideas, pertinent comments, and most of all an infectious enthusiasm for the subject. Thanks to Professor Mike Brady for his ceaseless encouragement and his exhortations to use real images. Thanks also to the other members of the Robotics Research Group including Paul Beardsley, Phil Torr, and Phil McLauchlan for the software, Pete Lindsey and Chris Rabson for the system administration, and in no particular order Rhys Newman, Jason Merron, John Clarke, Fuxing Li, Simon Rowe, Mike Taylor, Ian Reid, Nic Pillow, and Annette Gingell. Thanks in particular to those of the above who have proof read this thesis.

Thanks also to the people who have made Oxford an enjoyable place to live: Rhys, Rich, and Mike B. for the squash; Rich, Caz, and Jason for the drinks; Chris J. for all the cycling; Dan for the night-life; and the members of LCBC for the time spent on the river. I am also grateful to my parents for all the support over the last seven years. I acknowledge the funding from an EPSRC studentship and an IEE scholarship.

Finally and most of all, I would like to thank Sinéad and her entourage (Emma the dog, Eric the cat, and Gertrude, Georgina, and Henrietta the guinea pigs) for being there when I needed them.

Contents

Chapter 1

Introduction

1.1 The Goal: Euclidean Reconstruction

Vision systems provide a very compact method of recording or watching a three-dimensional scene, but in their very nature, the projection from three dimensions to two loses a large amount of information. As a result, the reverse projection from a two-dimensional image sequence to a full three-dimensional structure is a non-trivial problem.

The reasons for requiring a three-dimensional reconstruction of a scene are innumerable, but typical problems tackled by the computer vision community include path planning for robot navigation [10], the grasping of objects by robotic arms [89], and the recognition of three-dimensional objects [108]. However, the advent of virtual reality and virtual worlds will dramatically increase the need for scene reconstruction from recorded image sequences. Virtual worlds are often copies of the real world, and for a significant number of accurate virtual worlds to be constructed, some simple method is required to reconstruct a scene from a pre-recorded image sequence. The image sequence can either be taken deliberately with the camera motion either known or constrained, or more interestingly old film footage could be used. With the latter, the pre-recorded film is the only information available which increases the complexity of obtaining a reconstruction.

Scene reconstruction or structure from motion is one of the oldest topics of computer vision [56]. There are several levels of scene reconstruction, based on projective geometry, and these are the projective, affine, and Euclidean reconstructions (see section 2.2).

Different vision tasks require different levels of reconstruction. So that only a projective reconstruction maybe required for object recognition [108], and only an affine reconstruction maybe required for path planning [10], grasping [89], and fixation point tracking [80]. However, many tasks (especially creating virtual worlds) require a Euclidean reconstruction so that, to a human observer, the reconstruction appears to be the same as the original scene.

To compute a Euclidean reconstruction requires the camera calibration to be known. The calibration consists of the camera internal parameters which describe how the camera is set up (i.e., the focal length and position of the optical centre). Camera calibration was traditionally obtained off-line before any vision tasks were commenced, and used a 3D calibration object with a known structure. This meant that first the camera is calibrated, then the image sequence is taken, and finally the Euclidean reconstruction is computed. However, for several scenarios, this method does not work. When using pre-recorded image sequences the calibration of the camera can be unknown. Also, the camera calibration could change during normal operation, either by design or accident, such as: zooming into a scene, undergoing significant changes in operating conditions (i.e., the temperature of a camera on a satellite), or due to a collision which can occur in industrial applications. However, recent advances have shown that is is possible to construct a Euclidean reconstruction from pre-recorded image sequences where the camera calibration is unknown.

Faugeras, Luong, and Maybank [25] showed that it is possible to compute the camera calibration using only information contained in the images, and this is the definition of **self-calibration**. Since that initial work many different methods have been suggested for self-calibration, and these are reviewed in section 3.1.

1.2 Approach and Themes

The approach used for the work presented in this thesis is to understand the underlying situation (geometry and complexity) so that the problem can be simplified and a concise algebraic solution computed. This approach produces several themes which appear throughout the thesis:

- **Stratification** Often a problem can be stratified into several smaller steps, for which the solutions are much simpler than the solution obtained by solving the problem directly.
- **Utilising Constraints** Often computer vision algorithms try to solve a problem for the most general case, but common constraints (such as constrained motion) can be used to simplify the complexity of the problem.

The ideal result of applying these themes is to get a concise solution to a problem, so that the solution can be applied to an arbitrary large amount of input data (i.e., images of a scene) without significantly increasing the complexity of the calculation, and increasing the amount of input data should improve the accuracy of a calculation.

1.3 Thesis Outline

The following two chapters of this thesis introduce some of the basic concepts of computer vision, as well as reviewing the relevant literature. **Chapter 2** examines the general idea of reconstructing a scene from images taken by a standard camera. The different camera models that can be used are explained. Projective geometry is introduced and it is shown how it is possible to have different levels of structure representation. These areas are then linked to show how the level of reconstruction for a scene is dependent on knowledge about camera calibration, the camera motion, and the objects in the scene. **Chapter 3** examines

the possible methods of calibrating a camera. The idea of self-calibration is introduced and then the theory and results for the many different methods are reviewed. Finally, for comparison, an off-line traditional calibration method is explained and extensive results are given.

This thesis presents two novel methods for self-calibration, which utilise knowledge of the type of motion the camera is undergoing. Note, the methods do not require knowledge of the actual motion, only the type of motion. **Chapter 4** introduces the first novel method, which is self-calibration via affine structure [4], and extends and combines the work of other authors. The method requires the first camera motion to be pure translation followed by any number of general displacements. Extensive assessment of the method is made, and results are given for real image sequences. **Chapter 5** introduces the second novel method, which is self-calibration via fixed points [5]. Here the restrictions are that the camera is moving under planar motion[1] and at least 3 views are available. The idea of fixed entities is introduced and it is shown how the position of these fixed entities in the image and in the world allow the camera to be calibrated. The actual method used is explained and involves using the fundamental geometric relationships for two and three views, namely the fundamental matrix and the trifocal tensor. Results are given for real image sequences using cameras mounted on several different platforms, and these results are shown to be both stable and accurate. **Chapter 6** examines the error analysis of the method for self-calibration from fixed points, and several ideas emerge. A novel batch parameterisation for the fundamental matrices gives significantly better accuracy than combining the results of the individual fundamental matrices. It is also shown that error estimation using the trifocal tensor is a very difficult problem, and explains why current methods do not give accurate

[1] Planar motion is defined as translation confined to a plane, and the rotation is around a fixed axis which is perpendicular to the plane of translation. This type of motion is common for cameras mounted on AGVs or when watching vehicles in traffic scene analysis.

results.

The work presented in the preceding chapters uses multiple images taken from long image sequences. One of the problems associated with long image sequences is the accurate tracking of features, and several methods [8, 67, 100, 104] have been suggested for achieving this. The self-calibration results use the work of Beardsley *et al.* [8] to match and track features. However, another method would be to fixate actively on and track a moving object, which could supply data from arbitrarily long image sequences. To this end, **Chapter 7** moves away from self-calibration, to the area of object tracking. The well known rigid object tracker RAPiD [34] is extended in many areas to give the more robust tracker RoRAPiD [3], and this tracker can take advantage of known motion or camera constraints. Extensive results are given for sequences running at frame rate on standard hardware. In the future, RoRAPiD could be used as a test bed for self-calibration on extended image sequences. **Chapter 8** draws some conclusions and identifies the areas for future research.

1.4 Notation

Wherever possible the standard notation described below is used in this thesis. Vectors are given in bold (e.g., $\mathbf{v}$) which generally indicates a column vector, while matrices are given in courier (e.g., $\mathtt{R}$). The tensor notation follows the standard convention of summation over contravariant and covariant indices where applicable, and tensors are shown as T_i^{jk}. Points and lines in the image are shown as lower case vectors, $\mathbf{x}$ and $\mathbf{l}$, while points and lines in the world are shown in upper case, $\mathbf{X}$ and $\mathbf{L}$. Corresponding points on different images are shown as $\mathbf{x}$, $\mathbf{x}'$, $\mathbf{x}''$, etc. Points and lines are generally given in homogeneous coordinates, and for any homogeneous quantity the equality sign = signifies equality up to a non-zero scale factor. The expression $[\mathbf{u}]_\times$ is the matrix form of the cross product such that $\mathbf{u} \times \mathbf{v}$ and $[\mathbf{u}]_\times \mathbf{v}$ are identical.

Chapter 2

Scene Reconstruction

Overview

This chapter introduces and reviews some of the basic ideas of computer vision in the area of computing a 3D Euclidean reconstruction using images of a scene taken by a standard camera. The pinhole camera model is explained, and its limitations and other related models are discussed. The camera model attempts to mathematically model the imaging process of a real camera from the 3D world to the 2D image plane. The idea of structure representation is introduced, and by using projective geometry it is shown that there are several different levels of representation for the same structure, each level having less information than the Euclidean representation. The information required to move between the different levels of representation is explained in terms of both projective geometry and the pinhole camera.

Finally, these separate strands are brought together in the area of scene reconstruction using images. It is shown how different amounts of knowledge about the camera results in different levels of reconstruction. For each level of reconstruction, the theory and relevant literature is reviewed in detail, and several reconstruction methods are described. The conclusion of the chapter is that the camera calibration is required to achieve the goal of a Euclidean reconstruction.

2.1 Camera Models

In computer vision and other related subjects, there are numerous different camera models which model the imaging process by mapping points in the world to positions on the image plane. The choice of which model is appropriate depends on several factors: the accuracy required in the mapping; the actual camera used; and the relationship between the camera and the scene being viewed.

Generally this work uses the ideal pinhole camera, which is described below, and is one of the most commonly used camera models in computer vision [22, 36, 72]. In section 2.1.2 the calibrated pinhole camera is explained, and then the possible variations to weak perspective and uncalibrated camera models are briefly explained in section 2.1.3. However, the theory of camera calibration and representation of structure is based on projective geometry, and so first the basic ideas of projective geometry are introduced.

2.1.1 Projective Geometry I - Homogeneous Coordinates

Projective geometry [81] is the theoretical framework for camera calibration and the representation of structure. It is an extension of Euclidean geometry in which points, lines or planes at infinity are treated no differently from those in finite space. This results in simpler formulae, and removes the problem of exceptions resulting from infinity (i.e., two lines always intersect in projective space, even if they are parallel in Euclidean space). The following is based on [64, 105].

In n dimensional projective space $\mathcal{P}^n$, a point may be represented by an $n+1$ vector $(X_1, X_2, \cdots, X_{n+1})^\top$. For 3-space $\mathcal{P}^3$, the homogeneous vector representing a point $\mathbf{X}_p = (X_1, X_2, X_3, X_4)^\top$ is related to the corresponding point in Euclidean 3-space $\mathcal{R}^3$, $\mathbf{X}_e = (X, Y, Z)^\top$ by

$$X = X_1/X_4, \quad Y = X_2/X_4, \quad Z = X_3/X_4.$$

$\mathbf{X}_p$ is only defined up to a non-zero scaling, such that for a non-zero λ, then $\lambda\mathbf{X}_p$ defines the same point as $\mathbf{X}_p$, but conventionally it is chosen that $X_4 = 1$. Points at infinity can now be treated in the same way as finite points, except that $X_4 = 0$. However, the points at infinity still have some significance and this will be explained in section 2.2.4.

2.1.2 Pinhole Camera Model

The ideal pinhole camera is a perspective projection from the world to the image plane, which does not model any non-linear distortion introduced by the camera (see section 2.1.3). The mapping is a perspective projection from 3D projective space $\mathcal{P}^3$ to the 2D image plane $\mathcal{P}^2$ with the position of the world and image points expressed in homogeneous coordinates (see section 2.1.1). The world point $\mathbf{X}_w$ is mapped to the image point $\mathbf{x}_i$ by the 3×4 projection matrix $\mathtt{P}$

$$\mathbf{x}_i = \mathtt{P}\mathbf{X}_w. \tag{2.1}$$

As homogeneous coordinates are only defined up to a non-zero scale factor, the projection matrix $\mathtt{P}$ is also only defined up to a non-zero scale factor, and has 11 independent degrees of freedom. The projection matrix ($\mathtt{P}$) can be split up into three parts

$$\mathtt{P} = \mathtt{CTG} = \mathtt{C}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} = \mathtt{C}[\mathtt{R} \mid \mathbf{t}], \tag{2.2}$$

where: $\mathtt{G}$ is the transformation from world coordinates $\mathbf{X}_w$ to camera-centred coordinates $\mathbf{X}_c$ (see figure 2.1) and contains the external (orientation) parameters ($\mathtt{R}$ - rotation, $\mathbf{t}$ - translation); $\mathtt{T}$ is the projection of $\mathcal{P}^3$ into $\mathcal{P}^2$; while $\mathtt{C}$ is the camera calibration, which maps the camera-centred coordinates $\mathbf{X}_c$ to the image points $\mathbf{x}_i$, and contains the internal (calibration) parameters. Internal and external differentiate those parameters which depend purely on the camera calibration (internal), and those which depend purely on the position and orientation of the camera (external).

The calibration parameters which model the pinhole camera are encoded in $\mathtt{C}$ which is an upper triangular matrix with 5 degrees of freedom. The matrix $\mathtt{C}$ is defined only up to a non-zero scaling, which is removed by setting $\mathtt{C}_{33} = 1$,

$$\mathtt{C} = \begin{bmatrix} \alpha_u & -\alpha_u \cot\theta & u_0 \\ 0 & \alpha_v / \sin\theta & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.3}$$

where the five calibration parameters are

α_u,α_v	The focal length (f in figure 2.1) of the camera, measured in pixel units along the horizontal (x_i) and vertical (y_i) directions respectively.
(u_0, v_0)	The principal point of the camera, which is the intersection of the optical axis and the image plane, and is measured in pixels (see figure 2.1).
θ	The angle between the horizontal (x_i) and vertical (y_i) camera axes (see figure 2.1).

Generally the image axes are nearly perpendicular with $\theta \approx 90°$, and so $\alpha_v / \sin\theta \approx \alpha_v$. With nearly perpendicular image axes the image skew ($k = \alpha_u \cot\theta$) is often set to zero to reduce the number of internal parameters. Also, instead of expressing the focal length with two different measurements (α_u,α_v), the aspect ratio is used with just one measurement of the focal length. The aspect ratio is the ratio of the horizontal and vertical pixel sizes ($\zeta = \alpha_v / \alpha_u$). This gives an alternative expression for the calibration parameters and $\mathtt{C}$

$$\mathtt{C} = \begin{bmatrix} \alpha_u & k & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \alpha_u & k & u_0 \\ 0 & \zeta\alpha_u & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.4}$$

2.1.3 Other Camera Models

Weak Perspective Camera Model

The pinhole camera uses perspective projection, but if the perspective effects are small, the equations associated with the camera model can be numerically ill-conditioned [33]. There are several situations which result in small perspective effects, but are typically a

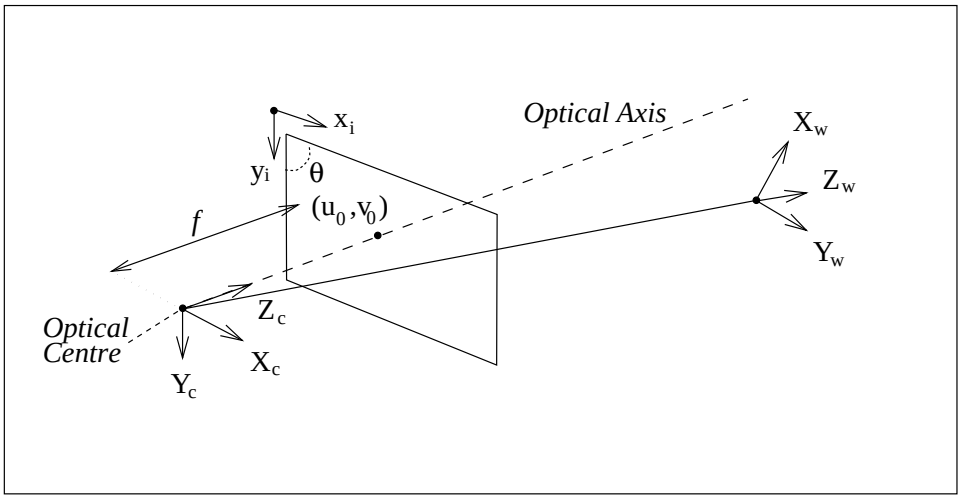


Figure 2.1: *The pinhole camera model which maps from the world coordinates* $\mathbf{X}_w$ *to the image coordinates* $\mathbf{x}_i$ *via the camera-centred coordinates* $\mathbf{X}_c$ *with a perspective projection.* f *is the focal length,* (u_0, v_0) *the principal point, and* θ *the angle between the image axes.*

combination of the following: a small field of view; the depth of the object being viewed being small compared to the distance from the object to the camera; and a long focal length. These situations are commonly known as affine imaging conditions [72, 82].

The camera model is derived from equations (2.1) and (2.2) such that

$$\mathtt{P}_{wp} = \mathtt{CTG} = \mathtt{C} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & Z_{ave} \end{bmatrix} = \mathtt{C} \begin{bmatrix} \mathbf{R}_1^\top & t_x \\ \mathbf{R}_2^\top & t_y \\ \mathbf{0}^\top & Z_{ave} \end{bmatrix}, \tag{2.5}$$

where Z_{ave} is the average depth of the points from the camera, and the camera model can be expressed in non-homogeneous coordinates

$$\mathbf{x} = \frac{\alpha_u}{Z_{ave}} \begin{bmatrix} \mathbf{R}_1^\top \\ \zeta \mathbf{R}_2^\top \end{bmatrix} \mathbf{X} + \frac{\alpha_u}{Z_{ave}} \begin{bmatrix} t_x \\ \zeta t_y \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = \mathtt{M}_{wp} \mathbf{X} + \mathbf{t}_{wp}. \tag{2.6}$$

Uncalibrated Camera Models

When the camera calibration is unknown, an uncalibrated camera model has to be used. The uncalibrated ideal pinhole camera model, termed the *projective camera* by Mundy and Zisserman [72], is a generalisation of equation (2.2) and is not decomposed in the same

a

b

Figure 2.2: *(a) The radial distortion affect with a short focal length camera. Note that lines straight in the scene are not necessarily straight in the image. (b) The same scene viewed with a longer focal length lens.*

way. Rather, it has 11 degrees of freedom in the form

$$\mathtt{P} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix}, \tag{2.7}$$

but the world-to-image mapping is still expressed by equation (2.1). Similarly, the uncalibrated weak perspective camera model, termed the *affine camera*, is a generalisation of equation (2.5) and has 8 degrees of freedom

$$\mathtt{P} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ 0 & 0 & 0 & P_{34} \end{bmatrix}. \tag{2.8}$$

Distortion

The pinhole camera model assumes that the imaging process is a perfect perspective projection from world to image coordinate frames ($\mathcal{P}^3$ to $\mathcal{P}^2$). However, real cameras are not perfect perspective projections (especially when used with a short focal length lens) and non-linear distortions are introduced into the imaging process. Figure 2.2a shows a typical image taken by a camera with a short focal length, where lines which are straight in the world are not necessarily straight in the image. There are several different forms of non-linear distortion, but usually only radial distortion is modelled, where the error is a radial displacement proportional to an even power of the distance from the centre of

the image. Several methods have been suggested which estimate and correct for radial distortion [7, 30, 84, 94]. This thesis has not considered distortion problems, and only uses cameras where the non-linear distortion can be assumed to be negligible.

2.2 Reconstructing a Scene

The aim of the work presented here is to compute a Euclidean reconstruction of a scene being viewed by a camera. Scene reconstruction from images is first discussed. Then the idea that there are several levels of representation for a structure is introduced. Finally each different level of structure representation is explained in detail, and methods of reconstructing a scene to that level are reviewed.

2.2.1 Structure Representation

For any real scene, there exists a set of points representing the structures in that scene, and the positions of those points can be measured in a Euclidean world coordinate frame. Any other representation is related to the original set of points by a transformation of the corresponding homogeneous coordinates $\mathbf{X}_o$. There are four levels of representation (or frames): projective, affine, similarity, and Euclidean [24]. Similarity[1] and Euclidean are generally grouped together as metric. The metric level of representation is the aim of this work. The level of representation a set of points is in, depends only on the transformation required to map the points to $\mathbf{X}_o$. An alternative explanation is that in any level, the set of points is only defined up to an ambiguity determined by that level (i.e., an affine representation of a structure is only defined up to affine ambiguity, and two sets of points describing the structure, which are related by an affine transformation, are equivalent and cannot be differentiated).

[1] A similarity transformation is a Euclidean transformation and an isotropic scaling.

Representation	Invariant Measurements
Euclidean	angles, distances
Similarity	angles, relative distances
Affine	parallelism, centre of mass
Projective	collinearity, cross ratio

Table 2.1: *The measurements that are invariant in each level of representation. The invariant measurements for each level also include those of lower levels (i.e., the invariant measurements lower in the table).*

Affine and metric transformations are sub-groups of projective transformations (metric transformations are sub-groups of affine transformations) [81]. Equivalently, affine and metric geometry are specialisations of projective geometry and so more details of projective geometry are given in section 2.2.4. Then the different levels of representation are described, including what knowledge is required for each.

For each level of representation, different properties are invariant (i.e., invariant property measurements give the same value in the original and in any transformed frame at the same level), and table 2.1 lists various invariant properties for each level of representation. Also, figure 2.3 gives a graphical idea of the types of representation for a set of simple objects.

2.2.2 Scene Reconstruction

When two or more images are taken of the same scene, from different positions and/or different cameras, it is possible to reconstruct the 3D structure being viewed[2]. The most commonly used approach for scene reconstruction is feature based, where the features (or data primitives) are points, lines, etc. in the image, which correspond to features in the 3D structure.

[2]Two (or more) cameras taking images from different positions at the same instant (binocular/trinocular) is the same as two images taken by the same camera (monocular) from different positions at different times, provided the scene has not changed in the intervening period. Also, a static camera viewing a moving object is the same as moving camera viewing a static object, provided that the moving object is correctly segmented from the static background [91, 100].

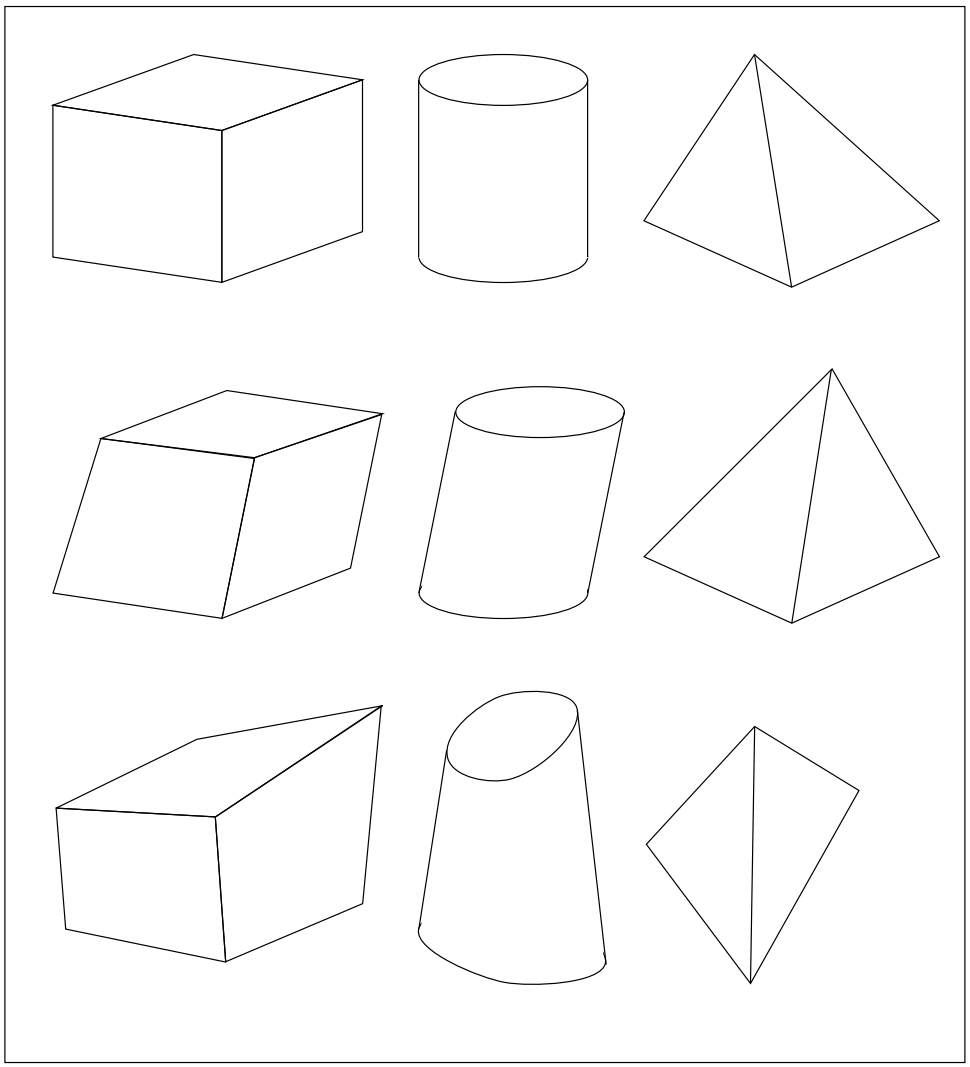


Figure 2.3: *Graphical representation of the possible structure representation: Top — a metric representation of the actual structure; Middle — an affine representation of the same structure (Note: parallel lines remain parallel); Bottom — a projective representation.*

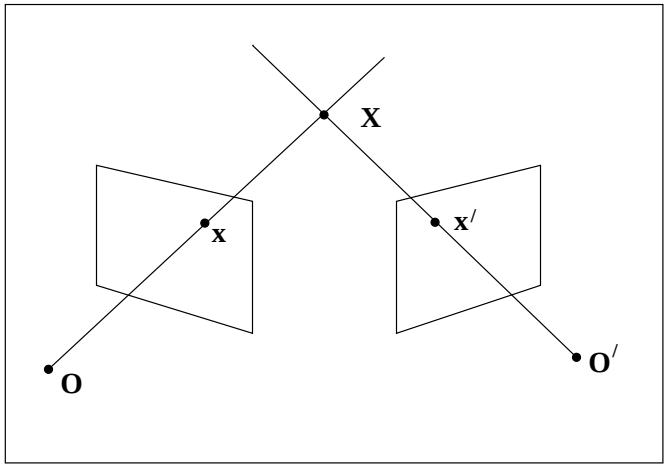


Figure 2.4: *Reconstructing in 3D space by back-projecting from 2 images taken by cameras in different positions. Rays from the camera centres ($\mathbf{O}, \mathbf{O}'$) pass through the matched points ($\mathbf{x}, \mathbf{x}'$), and intersect in 3-space to give the position of the world points* $\mathbf{X}$.

Features

Features are used to reduce the computational demands of analysing image sequences. One second's (25 frames) worth of a 512×512, eight bit grey level image sequence contains 6MB of data. However, following the observation "not all information is created equal" [14], it has been suggested that using a more sparse representation of an image (i.e., features) will facilitate the analysis of image sequences within a *reasonable* time scale. The features in the image are computed as the one or two-dimensional loci of grey levels in the image, and many methods have been suggested for computing their position [15, 18, 35, 97].

The point and line features need to be matched across the different images [8, 46, 103], and then by back-projection [23, 56], the corresponding points and lines in 3-space can be found (see figure 2.4). A problem encountered when back-projecting real image points, is that image noise means that the position of the image points will not be exact and the back-projected rays will not necessarily intersect in 3-space. Some form of error measurement will be required, so that the *intersection point* can be computed as the minimum of the error measurement [43].

Back-Projection

Several authors [23, 43] have examined the mathematics of back-projecting points from two or more images to give the corresponding point in 3D. Here the most general camera model, the projective camera (see equation (2.7)), is used, and rays are back-projected from two images. The analysis can easily be extended to include more images, or different camera models. First, a linear method is given which has a closed form solution, and then a non-linear method is presented which iteratively minimises the error in the image plane.

Consider a point $\mathbf{X} = (X, Y, Z, 1)^\top$ in 3-space which is projected onto two image planes. The projection matrices can be expressed as $\mathtt{P} = [\mathtt{M}|\mathbf{m}]$ and $\mathtt{P}' = [\mathtt{M}'|\mathbf{m}']$. From equation (2.1) two vector equations are obtained

$$\mathbf{x} = \mathtt{P}\mathbf{X} = [\mathtt{M}|\mathbf{m}]\,\mathbf{X}, \tag{2.9}$$

$$\mathbf{x}' = \mathtt{P}'\mathbf{X} = [\mathtt{M}'|\mathbf{m}']\,\mathbf{X}. \tag{2.10}$$

Eliminating the unknown scale factor between the three equations formed by the vector equations (2.9) and (2.10) gives four equations in the three unknowns (X, Y, Z)

$$\begin{aligned}
0 &= X(M_{11} - M_{31}x) + Y(M_{12} - M_{32}x) + Z(M_{13} - M_{33}x) + m_1 - m_3 x, \\
0 &= X(M_{21} - M_{31}y) + Y(M_{22} - M_{32}y) + Z(M_{23} - M_{33}y) + m_2 - m_3 y, \\
0 &= X(M'_{11} - M'_{31}x') + Y(M'_{12} - M'_{32}x') + Z(M'_{13} - M'_{33}x') + m'_1 - m'_3 x', \\
0 &= X(M'_{21} - M'_{31}y') + Y(M'_{22} - M'_{32}y') + Z(M'_{23} - M'_{33}y') + m'_2 - m'_3 y'.
\end{aligned}$$

This over-constrained system will not have an unique solution due to image noise (the rays will be skew and will not intersect), so another constraint or minimisation criterion is required. The constraint which can be used to find the intersection depends on the level of 3D reconstruction being computed (see below). For metric reconstruction, the

mid-point of the shortest line between rays can be used[3]. However, for affine or projective reconstructions the shortest line is not defined (i.e., the relative length of non-parallel lines is not an affine invariant) and so cannot be used.

Rather than using a constraint to find the intersection of the skewed lines, a non-linear method [23] can be used to minimise the image plane error

$$E = \sum_i d(\mathbf{x}_i - \hat{\mathbf{x}}_i)^2, \tag{2.11}$$

where $d(\mathbf{a}, \mathbf{b})$ is the distance between two image points given by the homogeneous vectors $\mathbf{a}$ and $\mathbf{b}$, $\mathbf{x}_i$ is the actual image point in view i, and $\hat{\mathbf{x}}_i$ is the projection of $\mathbf{X}$ by the projection matrix $\mathtt{P}_i$ (i.e., $\hat{\mathbf{x}}_i = \mathtt{P}_i\mathbf{X}$). This is a non-linear function of $\mathbf{X}$, which can be solved by standard techniques [31], but has the advantage that the error being minimised has a physical interpretation.

However, rather than compute a projective reconstruction, Hartley and Sturm [43] algebraically solve the error function (2.11). For general motion, the method reduces to solving a single variable polynomial of degree six, and for degenerate motions further simplifications are possible. This method has the advantage of not requiring a non-linear minimisation.

Accuracy

The accuracy of any reconstruction depends on several factors:

- The distance between the camera centres, known as the baseline. If the baseline is small, the angle between the back-projecting rays will be small, and image noise can produce a large error in back-projection (see figure 2.5a). However, if there is a large

[3] For three or more intersecting rays, the intersection point is defined as the point which minimises the sum of perpendicular distance to each ray.

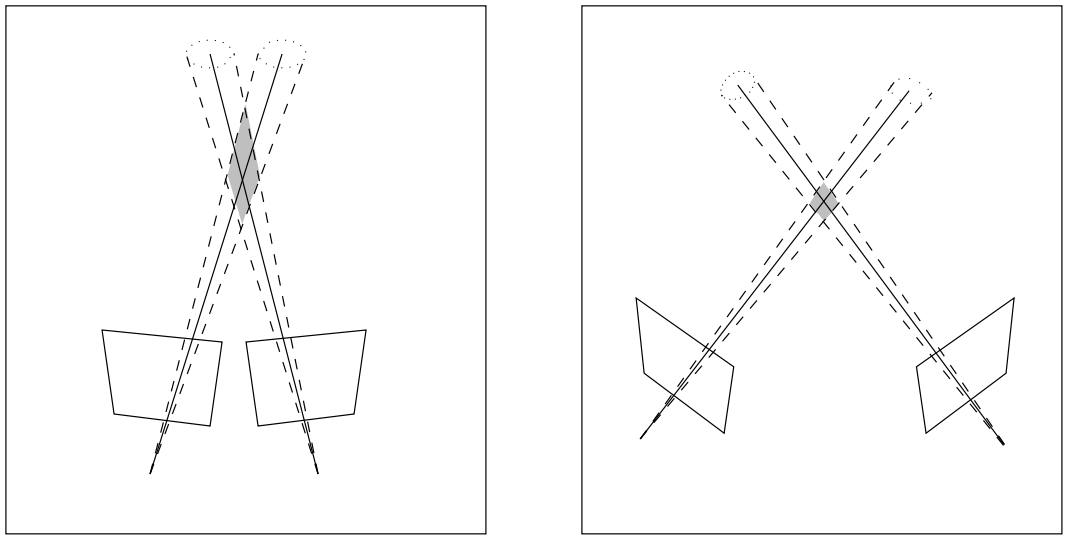
baseline, the back-projecting rays are generally well-conditioned and the image noise has a smaller effect (see figure 2.5b).

- The accuracy of the information known about the camera, the motion involved, and the objects in the scene, including:
  - The camera motion. This is not necessarily the actual displacement of the camera, but the type of motion the camera is undergoing. If for example, the camera is assumed to only translate, with no rotation, then how close is the actual motion to this assumption?
  - Assuming a set of points lies on a planar surface in the scene.
  - The camera calibration.
- The number of images containing views of the same points increases the number of rays back-projected when estimating the position of a point. This should reduce the effect of errors introduced by image noise. How to accurately track features over image sequences is a separate area of computer vision [8, 32, 67, 100].

2.2.3 Hierarchy of Scene Reconstruction

When reconstructing a scene from images, the level of representation of the 3D reconstruction depends on what is known about the camera, the motion involved, and the objects in the scene. At the lowest level, a projective reconstruction requires no knowledge of the camera, motion, or scene, and several methods are described for this case in section 2.2.5.

For the higher levels (affine or metric), two different approaches have been used to compute a reconstruction. The first uses knowledge of the camera, motion, scene, or imaging conditions to allow direct reconstruction of the affine/metric structure by back-projecting the image points directly into an affine/metric coordinate frame. The second



a b

Figure 2.5: *The effect of baseline on scene reconstruction: (a) with a small baseline, any errors in the image can produce a large error in the reconstruction; (b) with a large baseline, errors in the image have a smaller affect. The cones show the possible back-projected rays when image noise is taken into account, and the shaded area is the possible positions of the world point.*

method is a stratified approach, so that a projective (affine) reconstruction is computed, and then the structure is upgraded to an affine (metric) ambiguity. Section 2.2.4 describes the knowledge required to upgrade to affine and metric ambiguity, and the possible methods for reconstruction are reviewed in sections 2.2.6 and 2.2.7.

2.2.4 Projective Geometry II - The Role of Infinity

Section 2.1.1 introduced the idea that in projective geometry the points at infinity are treated in the same way as finite points, except that the final projective coordinate is set to zero (i.e., points at infinity in $\mathcal{P}^3$ have $X_4 = 0$). However, the points at infinity retain significance in the different levels of representation of structure. Moreover, knowledge of two different geometric structures, the plane at infinity and the absolute conic, control the level of structure representation [81]. All the points at infinity lie on the plane at infinity π_∞, and conventionally the equation for this plane is $(0, 0, 0, 1)^\top$. The absolute conic Ω_∞ lies

on this plane (π_∞), has a radius of $\sqrt{-1}$, and consists of complex points. Conventionally, the conic is defined as

$$\begin{aligned} X_1^2 + X_2^2 + X_3^2 &= 0, \\ X_4 &= 0. \end{aligned}$$

Alternatively, Ω_∞ is the intersection of the absolute quadric $\mathtt{Q}_\infty$ and π_∞, which gives the equations defining Ω_∞ as

$$\begin{aligned} \mathbf{X}^\top \mathtt{Q}_\infty \mathbf{X} &= 0, \\ X_4 &= 0, \\ \mathtt{Q}_\infty &= \mathtt{I}_4. \end{aligned}$$

Strictly speaking, to achieve an affine or metric reconstruction only requires π_∞ and Ω_∞ to be identified in the projective structure, but not necessarily placed at the conventional positions described above.

2.2.5 Projective Reconstruction

The projective representation of a structure is the most general, with the the set of points $\mathbf{X}_p$ being defined up to a real projective transformation of the original set $\mathbf{X}_o$. This projective transformation (also called a *homography* or *collineation*) is represented by the invertible 4×4 matrix $\mathtt{H}$ which has real elements,

$$\mathbf{X}_o = \mathtt{H}\mathbf{X}_p = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\top & 1 \end{bmatrix} \mathbf{X}_p, \tag{2.12}$$

where $\mathtt{A}$ is a 3×3 affine matrix, and $\mathbf{v}$ and $\mathbf{t}$ are three-vectors. For a projective representation π_∞ is not identified, but $(\mathbf{v}^\top, 1)^\top$ is identical to π_∞ in the affine and metric frames.

Direct Projective Reconstructions

It has been shown that a projective reconstruction requires only point (line) matches between two (three) views from uncalibrated cameras [22, 39, 42]. Two different methods have been used for projective reconstruction using points:

- Faugeras [22] and Mohr *et al.* [68, 70] use the minimum number of points to recover the structure up to a projective ambiguity. A basis of 5 points is chosen from all the matched points to form a projective coordinate frame in which the other points are then measured. A drawback of this approach is that the accuracy of the whole reconstruction degrades if there is an error in the image position of one or more of the 5 points.

- Beardsley *et al.* [10] and Hartley *et al.* [42] utilise all the point matches to form the projective frame for the reconstruction. This minimises the effect of inaccuracies in the position of the points in the image. The method requires that the projection matrices are found for each view, then the reconstruction is found by back-projecting the rays (see section 2.2.2). With real images containing noise, the rays will not intersect but will be skew. Different methods to find an intersection point in projective space were reviewed in section 2.2.2. However, Beardsley *et al.* [10] work in a *quasi-Euclidean* frame in which, they argue, the mid-point is usable.

For a projective reconstruction using lines:

- Hartley [39] computes a projective reconstruction using lines matched over three views. The approach is to compute the equations which transfer lines from two views to the third, and these equations give constraints on the camera projection matrices. With sufficient lines (> 13) it is possible to compute the transfer equations, and find the camera projection matrices corresponding to a projective reconstruction. More

recently, it has been found that the transfer equations correspond to the trifocal tensor which is described in section 5.4.1.

2.2.6 Affine Reconstruction

An affine representation requires that the plane at infinity is known, so that finite points cannot be mapped to or from π_∞. This gives the invariant property of affine representation: parallel lines remain parallel (see figure 2.3). From equation (2.12), $\mathbf{v}$ needs to be identified to transform from a projective to affine representation. Conventionally, π_∞ is set to $(0,0,0,1)^\top$, and this requires $\mathbf{v} = \mathbf{0}$. The representation is defined up to an affine ambiguity

$$\mathbf{X}_o = \mathtt{H}\mathbf{X}_a = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \mathbf{X}_a . \tag{2.13}$$

Lemma 2.1 *The position of the plane at infinity ($\pi_\infty = (0,0,0,1)^\top$) is invariant under affine transformations.*

Proof: Under a point transformation $\mathtt{H}$, such that $\mathbf{X}' = \mathtt{H}\mathbf{X}$, a plane ($\pi$) transforms as $\pi' = \mathtt{H}^{-\top}\pi$. So π_∞ transforms as

$$\pi'_\infty = \mathtt{H}^{-\top} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \mathtt{A}^{-1} & \mathbf{0} \\ -\mathbf{t}^\top \mathtt{A}^{-\top} & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \pi_\infty .$$

□

Direct Affine Reconstructions

Normally, from uncalibrated cameras, structure can only be recovered up to a projective ambiguity. However, for some special motions, or when some assumptions are made about the structure or imaging conditions, it is possible to recover an affine reconstruction.

- Koenderink and van Doorn [49], Quan and Mohr [79], and Tomasi and Kanade [90] assume that the depth of the object is small compared to the distance to the camera,

and that there is a small field of view. This allows the affine camera model to be used (see section 2.1.3). Using the affine camera allows an affine reconstruction to be obtained simply by back-projecting the rays. Again, either the minimum number of points can be used [49], or all the points used from several images in a batch method [90].

- If the camera motion is a pure translation, Moons *et al.* [71] shows that an affine reconstruction can be computed from two or more perspective views. The original method used the minimum number of points for the reconstruction. More details of the extended method, which uses all the points, can be found in section 4.1.1.

Upgrading Projective to Affine Structure

To upgrade from a projective to an affine reconstruction requires the position of π_∞ to be known. Several methods exist:

- If three or more points or lines in the image are known to be the images of points or lines at infinity, back-projecting these points into the reconstruction uniquely identifies π_∞.

- Pollefeys *et al.* [76] use the *modulus constraint.* From two camera projection matrices for a projective reconstruction, it is possible to obtain a fourth-order polynomial in the three elements of $\mathbf{v}$. With four views it is possible to solve the set of three simultaneous polynomial equations using non-linear methods, and hence identify π_∞. This result was independently observed [2] and is described in section 4.1.2, but no results were obtained.

- Hartley [37, 38] uses chiral inequalities to give a range of possible values for $\mathbf{v}$. From each view of a projective reconstruction it is possible to obtain a set of inequalities constraining the position of $\mathbf{v}$. Using the different views available and linear programming with the goal of maximising the margin by which the inequalities are satisfied, a convex hull of possible solutions in the 3 parameter space of $\mathbf{v}$ is obtained. This is not an exact solution for $\mathbf{v}$, but points inside the convex hull are approximate solutions for $\mathbf{v}$, which give a *quasi-affine* reconstruction.

Once a plane $(\mathbf{v}, 1)^\top$ in the projective reconstruction has been identified as corresponding to the plane at infinity, it can be transformed to the conventional position $(0, 0, 0, 1)^\top$ in the affine reconstruction using the point transformation

$$\mathbf{X}_a = \begin{bmatrix} \mathtt{I} & \mathbf{0} \\ \mathbf{v}^\top & 1 \end{bmatrix} \mathbf{X}_p. \tag{2.14}$$

2.2.7 Metric Reconstruction

A metric representation not only requires π_∞ to be known, but also the absolute conic (Ω_∞). This results in the representation being defined up to a metric ambiguity,

$$\mathbf{X}_o = \mathtt{H}\mathbf{X}_m = \begin{bmatrix} \mathtt{R} & \mathbf{t} \\ \mathbf{0}^\top & \lambda \end{bmatrix} \mathbf{X}_m. \tag{2.15}$$

In a similarity representation λ is the unknown isotropic scaling, while in a Euclidean representation $\lambda = 1$.

Lemma 2.2 *The absolute conic (Ω_∞) is invariant under metric transformations.*

Proof: Under a point transformation $\mathtt{H}$, such that $\mathbf{X}' = \mathtt{H}\mathbf{X}$, a quadric ($\mathtt{Q}$) transforms as $\mathtt{Q}' = \mathtt{H}^{-\top}\mathtt{Q}\mathtt{H}^{-1}$. The absolute quadric $\mathtt{Q}_\infty$ transforms as

$$\mathtt{Q}'_\infty = \begin{bmatrix} \mathtt{R} & \mathbf{0} \\ -\mathbf{t}^\top\mathtt{R}/\lambda & 1/\lambda \end{bmatrix} \mathtt{Q}_\infty \begin{bmatrix} \mathtt{R}^\top & -\mathtt{R}^\top\mathbf{t}/\lambda \\ \mathbf{0}^\top & 1/\lambda \end{bmatrix} = \begin{bmatrix} \mathtt{I} & -\mathbf{t}/\lambda \\ -\mathbf{t}^\top/\lambda & 1/\lambda^2 \end{bmatrix}.$$

With π_∞ invariant (metrics are a sub-group of affinities), the intersection of the transformed quadric ($\mathtt{Q}'_\infty$) with π_∞ (i.e., set $X'_4 = 0$) gives

$$\begin{aligned} X_1'^2 + X_2'^2 + X_3'^2 &= 0, \\ X'_4 &= 0, \end{aligned}$$

which defines $\mathbf{\Omega}_\infty$.

□

Direct Metric Reconstructions

When the cameras are calibrated, it is possible to compute a metric reconstruction [23, 56, 98]. Using the essential matrix [56] allows the structure to be recovered up to a similarity ambiguity. However, when the camera motion is known the full Euclidean structure can be recovered. Both methods back-project the points using the projection matrices in the form of equation (2.2), but differ in that the translation between views is known exactly for the Euclidean reconstruction and only up to scale for the similarity reconstruction.

Upgrading Affine to Metric Structure

To upgrade to a metric reconstruction requires that $\mathbf{\Omega}_\infty$ is known. However, as will be explained in section 3.1, knowing $\mathbf{\Omega}_\infty$ is equivalent to knowing the camera calibration ($\mathtt{C}$). Knowing the camera calibration allows the metric structure to be recovered from the affine structure with the transformation

$$\mathbf{X}_m = \mathtt{H}\mathbf{X}_a.$$

Both the affine and metric structures are projected to the same image points with equation (2.1), and equations (2.13) and (2.15) describe the affine and metric ambiguity. Simple

algebraic manipulation gives the projection for the metric and affine structures as

$$\begin{aligned} \mathbf{x} = \quad \mathtt{P}_m \mathbf{X}_m \quad &= [\mathtt{C} \mid \mathbf{0}] \mathbf{X}_m, \\ \mathbf{x} = \quad \mathtt{P}_a \mathbf{X}_a \quad &= [\mathtt{A} \mid \mathbf{t}] \mathbf{X}_a, \end{aligned}$$

and the transformation $\mathtt{H}$ is

$$\mathtt{H}^{-1} = \begin{bmatrix} \mathtt{C} & \mathbf{0} \\ \mathbf{0}^\top & 1 \end{bmatrix},$$

giving the upgraded metric structure as

$$\mathbf{X}_m = \mathtt{C}^{-1} \mathbf{X}_a. \tag{2.16}$$

Summary

This chapter introduced and reviewed several areas of computer vision concerned with the 3D Euclidean reconstruction of a scene using images taken by a camera. These were the camera models, the connection between structure representation and projective geometry, and the different levels of scene reconstruction. It concluded that to obtain a Euclidean reconstruction of a scene using an image sequence, the camera calibration needs to be known. The camera calibration can be obtain in several ways, and is the topic of the next chapter.

Chapter 3

Camera Calibration

Overview

The previous chapter introduced the idea of computing a 3D reconstruction using only images of the scene, and concluded that to obtain a Euclidean reconstruction requires that the camera calibration is known. This chapter discusses methods for obtaining the camera calibration.

Traditionally, the camera calibration was obtained off-line and used images of a special calibration object. High accuracy is obtainable, but the method cannot cope when the calibration of the camera changes during the normal operation (i.e., zooming in or out), or when trying to reconstruct a scene from a pre-recorded image sequence where the camera calibration cannot be known. Section 3.2 explains the theory and gives results for a standard off-line calibration method.

Faugeras et al. [25, 58, 65] introduced the idea of self-calibration, where the camera calibration can be obtained from the image sequences themselves, without requiring knowledge of the scene. This has allowed the possibility of reconstructing a scene from pre-recorded images sequences, or computing the camera calibration during the normal vision tasks. A lot of work has been done in this area, and section 3.1 reviews the different methods which have been suggested. Also explained is the knowledge required to self-calibrate a camera.

3.1 Self-Calibration

Faugeras, Luong, and Maybank [25, 58, 65] introduced the idea that a camera could be calibrated using only point matches between images, and termed the method *self-calibration.* This avoided the use of a calibration object (or known scene), or any knowledge of the camera motion. Since then several algorithms have been suggested, which differ in the permissible camera motions, and in the actual methods for finding the camera calibration. Some methods self-calibrate directly in one step, while others use a stratified approach and calibrate via a projective or affine reconstruction. The different approaches are reviewed below and each method is explained in more details in sections 3.1.2–3.1.8.

Monocular Image Sequences When the calibration remains fixed, there are two general approaches for self-calibration for a monocular camera: either only image measurements are used, or a reconstruction is computed simultaneously.

- Faugeras *et al.* [25], and Hartley's method for a rotating camera [40] (see sections 3.1.2 and 3.1.4) find the image (ω) of the absolute conic (Ω_∞). As is shown below, ω is determined by, and determines, the camera calibration in the form of calibration matrix $\mathtt{C}$. So finding ω is equivalent to finding the camera calibration. These methods only use measurements made in the image.

- Hartley's method for an unconstrained camera [38] (see sections 3.1.3) uses a different approach. An iterative search is used to find a set of consistent projection matrices in the form of equation (2.2), which are consistent for the set of matched image points and the corresponding back-projected world points.

Stereo Head When self-calibrating a stereo head, there are other possible approaches in addition to the two mentioned above. From two pairs of views from a stereo head,

two separate projective reconstructions can be computed. The projective transformation relating these is then computed and this can be used for self-calibration. There are two different methods which are described in more detail in section 3.1.5:

- Zisserman *et al.* [9, 107] show that is is possible to obtain constraints on π_∞ and ω from the eigenvalue/vector decomposition of the projective transformation.
- Devernay and Faugeras [20] use a different approach which does not directly utilise $\mathtt{C}$ or ω. They show that the projective transformation can be decomposed into a special form, which allows the projective reconstruction to be upgraded to metric ambiguity.

3.1.1 The Image of Ω_∞ Determines Camera Calibration

In section 2.2.7, it was shown that the absolute conic (Ω_∞) is invariant to rigid motion. However, more interestingly for self-calibration, the image of the absolute conic (ω) is also invariant for rigid motions, and is determined by and determines the camera calibration. Also, it is possible to find ω and thence the camera calibration.

Lemma 3.1 *The image of the absolute conic (ω) is invariant to rigid motions of the camera, determines, and is determined by the internal parameters of the camera.*

Proof: Following Maybank [64]: a point $\mathbf{X}$ on Ω_∞ can be expressed as

$$\mathbf{X} = (\mathbf{Y}, 0)^\top,$$

then from equation (2.2), the image $\mathbf{x}$ of this point is

$$\mathbf{x} = \mathtt{C}\mathtt{R}\mathbf{Y},$$

and it follows

$$\mathbf{Y} = \mathtt{R}^\top \mathtt{C}^{-1}\mathbf{x}.$$

Using the matrix equation for Ω_∞ (see section 2.2.7)

$$\begin{aligned} 0 &= \mathbf{X}^\top \mathtt{Q}_\infty \mathbf{X} \\ &= \mathbf{Y}^\top \mathbf{Y} \\ &= \mathbf{x}^\top \mathtt{C}^{-\top} \mathtt{R}\mathtt{R}^\top \mathtt{C}^{-1} \mathbf{x} \\ &= \mathbf{x}^\top \mathtt{C}^{-\top} \mathtt{C}^{-1} \mathbf{x} \\ &= \mathbf{x}^\top \mathtt{K}^{-1} \mathbf{x}, \end{aligned}$$

gives the definition of a conic in the image plane, which is independent of the rigid displacement ($\mathtt{R}$,$\mathtt{t}$) and only dependent on the camera calibration ($\mathtt{C}$). This conic ($\mathtt{K}^{-1}$) is ω, the image of the absolute conic, and the dual[1] of the conic $\mathtt{K} = \mathtt{C}\mathtt{C}^\top$. Hence, ω determines and is determined by the camera calibration.

□

Once the image of the absolute conic $\mathtt{K}^{-1}$ has been found, it is trivial to determine the camera calibration ($\mathtt{C}$) by Choleski decomposition [86] of $\mathtt{K}$. If there is significant noise on the image, it is possible that $\mathtt{K}$ will not be positive definite, which means that Choleski decomposition will give complex values for the calibration.

3.1.2 Kruppa's Equations

The original method by Faugeras *et al.* [25] involved the computation of the fundamental matrix $\mathtt{F}$, which encodes epipolar geometry between two images [22, 36, 60]. Each fundamental matrix generates two quadratic constraints involving only the five elements of $\mathtt{K}$ (and not the 3D structure or camera motion). From three views a system of polynomial equations is constructed called Kruppa's equations [51]. Originally [25], *homotopy continuation* was used to solve the set of polynomial equations, but the method is computationally

[1] The dual conic is defined as the adjoint of the conic matrix. The adjoint of $\mathtt{A}$ is $\mathtt{A}^+$ which is given by the equation $\mathtt{A}^+\mathtt{A} = \det(\mathtt{A})\mathtt{I}$.

expensive and requires extreme accuracy of computation. Additional views increase the complexity. Since then, Luong [58] has used an iterative search technique to solve the set of polynomial equations, but results were limited by the choice of initial values and the complexity of the equations. More recently Zeller *et al.* [102] solved the equations using energy minimisation. The results obtained are generally better than those achieved by the two previous methods.

Kruppa's equations are based on the relationship between the image of the absolute conic (ω) and the epipolar transformation. If an epipolar line (l) is tangent to ω, then the corresponding epipolar line (l′) is also tangent to ω (see [63] for proof). Kruppa's equations can be derived by several different methods, and the following method follows Vieville and Lingrand [95].

Lemma 3.2 *From a pair of images it is possible to obtain a set of polynomial equations, quadratic in elements of* K*, called Kruppa's equations.*

Proof: It is shown in lemma 3.4 that the infinite homography $\mathtt{H}_\infty$ gives constraints on $\mathtt{K}$ in the form of

$$\mathtt{K} = \mathtt{H}_\infty \mathtt{K} \mathtt{H}_\infty^\top.$$

It has been shown [60] that the relationship between $\mathtt{K}$ and the fundamental matrix $\mathtt{F}$ is

$$\mathtt{F} = [\mathbf{e}]_\times \mathtt{H}_\infty,$$

where $\mathbf{e}$ is the epipole of $\mathtt{F}$. Hence multiplying $\mathtt{K}$ left and right by $[\mathbf{e}]_\times$ gives

$$\begin{aligned} [\mathbf{e}]_\times \mathtt{K} [\mathbf{e}]_\times &= [\mathbf{e}]_\times \mathtt{H}_\infty \mathtt{K} \mathtt{H}_\infty^\top [\mathbf{e}]_\times \\ &= \mathtt{F} \mathtt{K} \mathtt{F}^\top, \end{aligned} \tag{3.1}$$

and the fundamental matrix $\mathtt{F}$ gives constraints on $\mathtt{K}$. However, $\mathtt{F}$ and $\mathtt{K}$ are only defined up to a non-zero scaling and cross multiplying to remove the unknown scale gives quadratic constraints on the elements of $\mathtt{K}$. □

Each pair of views gives two quadratic equations containing the elements of $\mathtt{K}$, and, given three camera displacements (four independent pairs of views), they form an overdetermined set of simultaneous polynomial equations. When there are only two displacements there are only four equations and five unknowns, and another constraint is required to solve for $\mathtt{K}$. Often the assumption that the image axes are perpendicular is used (see section 2.1.2), which, from the definition of $\mathtt{K}$, leads to the additional quadratic constraint of

$$K_{12}K_{33} = K_{23}K_{13}. \tag{3.2}$$

The problem of the calibration being defined up to an one-parameter family, and requiring another constraint, is a recurring theme throughout the area of self-calibration.

3.1.3 Hartley's Unconstrained Motion Method

Hartley [38] uses a set of matched points from images taken with the same camera but from different positions. There is no constraint on the motion allowed between the camera positions. Each set of matched image points has a corresponding point in 3D. The method involves an iterative search to find a set of calibrated camera matrices and 3D world points consistent with the image points.

The method is that a set world points $\mathbf{X}_i$ can be projected onto m images with the projection matrices $\mathtt{P}^j$ ($0 \leq j < m$), which can be expressed in the form of equation (2.2)

$$\mathtt{P}^j = \mathtt{C}[\mathtt{R}^j|\mathtt{t}^j],$$

then in the jth image, $\mathbf{X}_i$ projects to the point $\hat{\mathbf{x}}_i^j$,

$$\hat{\mathbf{x}}_i^j = \mathtt{P}^j\mathbf{X}_i.$$

Using the actual position of the image points $\mathbf{x}_i^j$, solve for $\mathbf{X}_i$, $\mathtt{R}^j$, $\mathbf{t}^j$, $\mathtt{C}$ by an iterative search for the minimum of the function

$$\sum_{i,j} d\left(\hat{\mathbf{x}}_i^j, \mathbf{x}_i^j\right)^2,$$

where $d(\mathbf{a}, \mathbf{b})$ is the Euclidean distance on the image plane between the points defined by the homogeneous vectors $\mathbf{a}$ and $\mathbf{b}$. The initial values are set to those computed for a *quasi-affine* reconstruction achieved using chiral inequalities (see section 2.2.6).

3.1.4 Rotating Camera

Hartley [40] introduced the idea of self-calibration using a rotating camera. When there is no translation of the camera between views, there is an image-to-image projective mapping which can be calculated using point matches. This projective mapping gives linear constraints on $\mathtt{K}$, the dual of ω. Given three or more images, these constraints define $\mathtt{K}$ and hence camera calibration.

The method is: given a set of matched points $\mathbf{x}_i^j$ (same notation as section 3.1.3), compute the 2D projective transformation

$$\mathbf{x}^j = \mathtt{H}^j \mathbf{x}^0.$$

Each projective transformation gives a constraint on $\mathtt{K}$ of the form

$$\mathtt{K}\mathtt{H}^{j-\top} = \mathtt{H}^j \mathtt{K}.$$

Two or more projective transformations give sufficient constraint to solve for $\mathtt{K}$, and hence the calibration $\mathtt{C}$.

3.1.5 Stereo Head

For a stereo head, a projective reconstruction can be computed from each pair of images (see section 2.2.5). The projective transformation between two reconstruction can be computed, and is obviously related to the rigid displacement of the stereo head.

Zisserman *et al.* [107] showed that the projective transformation is conjugate to the rigid (Euclidean) transformation, so that the eigenvalues of the matrices are identical, and

that the eigenvalue/eigenvector decomposition of the projective transformation identifies π_∞, and defines Ω_∞, up to an one-parameter family. Another constraint on the calibration is used to identify Ω_∞ in the one-parameter family.

Devernay and Faugeras [20] also utilise the fact that the projective transformation is conjugate to a rigid displacement. However, rather than use the eigenvalue/eigenvector decomposition to identify π_∞ and Ω_∞, they decompose the projective transformation into rigid displacement and a *reduced projective transformation.* This can be thought of as analogous to the QR decomposition of a matrix. This decomposition of the projective transformation allows metric structure to be recovered. However, one again the metric reconstruction is only defined up to an one-parameter family, but as Ω_∞ is not used in the algorithm, the extra constraints on calibration which are normally used cannot be applied. Rather, they suggest using a second displacement which removes the ambiguity. However, the second displacement will have to have a different rotation axis, and a camera mounted on a vehicle will often have a fixed rotation axis which will not remove the ambiguity.

3.1.6 Affine Structure

Luong and Vieville [61] showed that from two general views of an affine structure, it is possible to calibrate the camera. First the *infinite homography* ($\mathtt{H}_\infty$) is computed from the two projection matrices, and then $\mathtt{H}_\infty$ gives constraints on $\mathtt{K}$. The *infinite homography* maps the image points (corresponding to points on π_∞) from the first to the second image.

Lemma 3.3 *The infinite homography ($\mathtt{H}_\infty$) can be computed from two views of an affine structure, whose projection matrices are $\mathtt{P}_a = [\mathtt{M}|\mathbf{m}]$ and $\mathtt{P}'_a = [\mathtt{M}'|\mathbf{m}']$ respectively, as the matrix $\mathtt{H}_\infty = \mathtt{M}'\mathtt{M}^{-1}$.*

Proof: Points on the plane at infinity ($\mathbf{X}_\infty$) can be represented by $\mathbf{X}_\infty = (X, Y, Z, 0)^\top$, and when projected onto the two image planes by $\mathtt{P}_a$ and $\mathtt{P}'_a$ give the image points $\mathbf{x}$ and $\mathbf{x}'$

$$\begin{aligned} \mathbf{x} &= \mathtt{M}\mathbf{X}_\infty, \\ \mathbf{x}' &= \mathtt{M}'\mathbf{X}_\infty. \end{aligned}$$

The corresponding image points are related by a projective mapping

$$\mathbf{x}' = \mathtt{M}'\mathtt{M}^{-1}\mathbf{x} = \mathtt{H}_\infty\mathbf{x}, \tag{3.3}$$

where $\mathtt{H}_\infty$ is the *infinite homography*.

□

Lemma 3.4 *The infinite homography (H_∞) between two views gives constraints on the dual of the image of the absolute conic (ω), in the form of $\mathsf{K}' = \mathsf{H}_\infty\mathsf{K}\mathsf{H}_\infty^\top$, where $\mathsf{K} = \mathsf{C}\mathsf{C}^\top$ ($\mathsf{K}' = \mathsf{H}_\infty\mathsf{K}\mathsf{H}_\infty^\top$) is the dual of ω in the first (second) image.*

Proof: The projection matrices can be expressed in the form of equation (2.2)

$$\begin{aligned} \mathtt{P} &= \mathtt{C}\left[\mathtt{R}|\mathbf{t}\right], \\ \mathtt{P}' &= \mathtt{C}'\left[\mathtt{R}'|\mathbf{t}'\right]. \end{aligned} \tag{3.4}$$

and comparing to the definition of $\mathtt{P}_a$ and $\mathtt{P}'_a$, and equation (3.3) gives

$$\mathtt{H}_\infty = \mathtt{C}'\mathtt{R}'(\mathtt{C}\mathtt{R})^{-1} = \mathtt{C}'\mathtt{R}'\mathtt{R}^{-1}\mathtt{C}^{-1}, \tag{3.5}$$

where $\mathtt{R}'\mathtt{R}^{-1}$ is the rotation of the camera between the views.

Substituting equation (3.5) into $\mathtt{H}_\infty\mathtt{K}\mathtt{H}_\infty^\top$ gives,

$$\begin{aligned} \mathtt{H}_\infty\mathtt{K}\mathtt{H}_\infty^\top &= (\mathtt{C}'\mathtt{R}'\mathtt{R}^{-1}\mathtt{C}^{-1})(\mathtt{C}\mathtt{C}^\top)(\mathtt{C}'\mathtt{R}'\mathtt{R}^{-1}\mathtt{C}^{-1})^\top \\ &= \mathtt{C}'\mathtt{R}'\mathtt{R}^{-1}\mathtt{C}^{-1}\mathtt{C}\mathtt{C}^\top\mathtt{C}^{-\top}\mathtt{R}^{-\top}\mathtt{R}'^\top\mathtt{C}'^\top \\ &= \mathtt{C}'\mathtt{C}'^\top \\ &= \mathtt{K}'. \quad \square \end{aligned} \tag{3.6}$$

Equation (3.6) is the transformation of a conic under the linear transformation $\mathtt{H}_\infty$. The significance of equation (3.6) is that it provides a linear method for obtaining $\mathtt{K}$. If the camera internal parameters are fixed between views (so $\mathtt{K} = \mathtt{K}'$)

$$\mathtt{K} = \mathtt{H}_\infty \mathtt{K} \mathtt{H}_\infty^\top. \tag{3.7}$$

However, if again only one pair of images is used, the constraints given by equation (3.7) only define $\mathtt{K}$ up to an one-parameter family, and another constraint is required to find it. If another view is used, and the rotation axes are different, then the two[2] constraint equations (3.7) fully define $\mathtt{K}$. More details can be found in chapter 4.

3.1.7 Varying Focal Length

Pollefeys *et al.* have shown that even when the focal length changes it is still possible perform self-calibration. Several different algorithms have been suggested, including self-calibration of a stereo head [75], and self-calibration from a monocular image sequence [76, 77].

The method uses an adaptation of the self-calibration from affine structure (see section 3.1.6) which can deal with a varying focal length. However, the adaptation requires that the position of the principal point is known. Hence, the calibration is found sequentially, with the principal point found first by zooming with a stationary camera, and then the varying focal length is found by zooming with a rotating camera. These deliberate motions can be achieved easily by taking an image sequence with a video camera.

[2] With three views there are three $\mathtt{H}_\infty$'s but only two out of the three are independent, as $\mathtt{H}_\infty$ between views 1 and 3 is the concatenation of the $\mathtt{H}_\infty$'s between views 1 and 2, and 2 and 3. Hence, only two of the three possible constraint equations (3.7) are independent.

3.1.8 Deliberate Motion

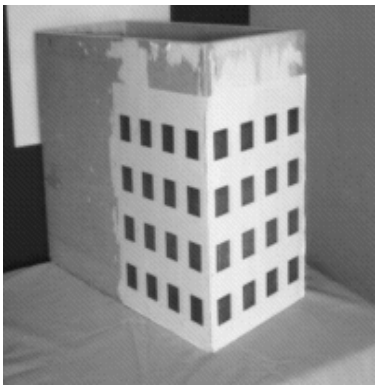
An alternative method, which has been used for active vision, is calibrating a camera using deliberate motions [66]. Here the stereo head [74] is made to perform a certain rotation about one of the camera axes, and by tracking features or using optical flow it is possible to calibrate the camera. The method is different from other self-calibrating algorithms as it requires control of the orientation of the camera, and the rotation between views. It is also different from the deliberate motions described in section 3.1.7, as the rotations have to be pure rotation around one of the camera axes, which generally can only be achieved using mechanical control of the camera.

A more general method is given by Horaud *et al.* [45], where the camera is mounted on a robot arm, and the arm executes general but known rigid displacements. First the scene is recovered up to a projective ambiguity, and then it is shown how the camera calibration can be recovered using two or more displacements. It is shown that to recover the camera calibration using this method, that the displacements must include at least two distinct axes of rotation, and at least one motion where the translation is not perpendicular to the axis of rotation.

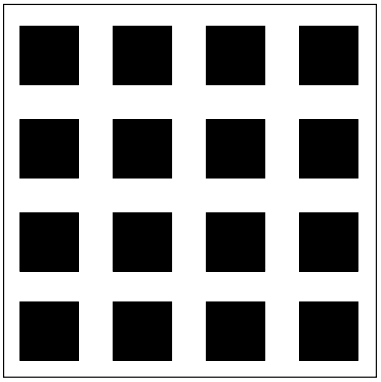
3.2 Traditional Calibration

Chapter 2 introduced the idea of reconstructing a scene using two or more images, and that for a metric reconstruction the camera calibration needs to be known. Traditionally camera calibration has been performed off-line by taking one or more images of a calibration object whose 3D structure is known. Then the position of the object in the image is found, allowing accurate estimation of the world-to-image projection. Finally, the camera calibration can then be found by decomposing this projection.

The calibration objects are designed so that the 3D position of points on the object are



a



b

Figure 3.1: *Calibration Objects: (a) a typical 3D calibration object containing two orthogonal Tsai grids; (b) the planar calibration grid (Tsai grid).*

known *a priori* and the image of these same points can be accurately found. The accuracy of the calibration is improved by reducing the possible error in measured position of the points in the world and in the image.

The points used are the corners created by texture on a planar surface. Figure 3.1 shows a typical calibration grid (Tsai grid) made up of dark squares on a white background, with the corners of the dark squares being used for calibration. Canny edge detection [15] is applied to the image, and orthogonal regression used to fit lines to the eight horizontal and vertical lines on each plane. The intersection of these lines generate 64 vertices (i.e., the corners of the dark squares) which are used as the points to compute the world-to-image projection.

Calibration using a single image of an object requires that the object is 3D rather than planar (see figure 3.1). However, Tsai [94] uses a planar calibration object (see figure 3.1b), and by taking two or more images with a known camera displacement the camera can be calibrated.

3.2.1 Theory of Traditional Camera Calibration

The method used here to calibrate the camera off-line is split into two stages [23]:

1. The projection matrix (P) from equation (2.1) is estimated using the positions of the corresponding 3D world points and 2D image points.

2. The matrix P is decomposed into the form of equation (2.2), which gives the internal parameters (C) and external parameters (R,t).

Estimating the Projection Matrix

The projection matrix P is estimated using a set of n world points $\mathbf{X}_i$ and the corresponding image points $\mathbf{x}_i$, by minimising the *image error*. The *image error* is the distance between the actual image point and the projection of the world point onto the image plane using P.

The projection of the world point onto the image is obtained using equation (2.1), but this only gives the homogeneous position of the image point (i.e., the ray through the camera centre on which the point lies). The position where this ray intersects the image plane has to be found so that the image error can be calculated. The intersection of the ray with the image plane is found by eliminating the unknown scale factor, so that a point on the image plane $(x, y)^\top$ has the homogeneous vector $(x, y, 1)^\top$.

Three equations can be obtained from equation (2.1), but eliminating the unknown scale factor gives two equations for the 12 unknowns in P

$$\begin{aligned} x &= \frac{\mathtt{P}_{11}X + \mathtt{P}_{12}Y + \mathtt{P}_{13}Z + \mathtt{P}_{14}}{\mathtt{P}_{31}X + \mathtt{P}_{32}Y + \mathtt{P}_{33}Z + \mathtt{P}_{34}}, \\ y &= \frac{\mathtt{P}_{21}X + \mathtt{P}_{22}Y + \mathtt{P}_{23}Z + \mathtt{P}_{24}}{\mathtt{P}_{31}X + \mathtt{P}_{32}Y + \mathtt{P}_{33}Z + \mathtt{P}_{34}}. \end{aligned} \tag{3.8}$$

The error to minimise is the geometric distance between the actual image points and the

projected image points from equations (3.8), giving the total image plane error

$$E_g = \frac{1}{n}\sum_{i=1}^{n} \left(\left(x_i - \frac{\mathtt{P}_{11}X_i + \mathtt{P}_{12}Y_i + \mathtt{P}_{13}Z_i + \mathtt{P}_{14}}{\mathtt{P}_{31}X_i + \mathtt{P}_{32}Y_i + \mathtt{P}_{33}Z_i + \mathtt{P}_{34}} \right)^2 + \left(y_i - \frac{\mathtt{P}_{21}X_i + \mathtt{P}_{22}Y_i + \mathtt{P}_{23}Z_i + \mathtt{P}_{24}}{\mathtt{P}_{31}X_i + \mathtt{P}_{32}Y_i + \mathtt{P}_{33}Z_i + \mathtt{P}_{34}} \right)^2 \right). \quad (3.9)$$

The error function (3.9) is non-linear and can be minimised by numerical methods. But before any iterative minimisation can be run, an initial estimate of the solution for $\mathtt{P}$ has to be found. Equations (3.8) can be rearranged, so that instead of minimising the geometric distance (E_g), an algebraic distance is minimised, giving a total algebraic error

$$E_a = \frac{1}{n}\sum_{i=1}^{n} \Big((x_i(\mathtt{P}_{31}X + \mathtt{P}_{32}Y + \mathtt{P}_{33}Z + \mathtt{P}_{34}) - (\mathtt{P}_{11}X + \mathtt{P}_{12}Y + \mathtt{P}_{13}Z + \mathtt{P}_{14}))^2 + (y_i(\mathtt{P}_{31}X + \mathtt{P}_{32}Y + \mathtt{P}_{33}Z + \mathtt{P}_{34}) - (\mathtt{P}_{21}X + \mathtt{P}_{22}Y + \mathtt{P}_{23}Z + \mathtt{P}_{24}))^2 \Big). \quad (3.10)$$

The error function (3.10) has the advantage that it is linear in the unknown elements of $\mathtt{P}$, so that a closed form solution can be found.

Initial Estimate of P

The error function (3.10) can be rearranged into the form

$$\min_{\mathbf{p}} \| \mathtt{Z}\mathbf{p} \|_2, \quad (3.11)$$

subject to $\| \mathbf{p} \|_2 = 1$, where $\mathbf{p}$ is a column vector containing the elements of $\mathtt{P}$ and

$$\mathtt{Z} = \begin{bmatrix} \mathbf{X}_1^\top & \mathbf{0}^\top & -x_1\mathbf{X}_1^\top \\ \mathbf{0}^\top & \mathbf{X}_1^\top & -y_1\mathbf{X}_1^\top \\ \mathbf{X}_2^\top & \mathbf{0}^\top & -x_2\mathbf{X}_2^\top \\ & \vdots & \\ \mathbf{0}^\top & \mathbf{X}_n^\top & -y_n\mathbf{X}_n^\top \end{bmatrix}.$$

The solution to equation (3.11) is the null eigenvector of $\mathtt{Z}$, and this can be found from Singular Value Decomposition (SVD) [78] of $\mathtt{Z}$. The trivial solution $\mathbf{p} = \mathbf{0}$ is avoided because $\mathbf{p}$ is an eigenvector which has unit norm.

Iterative Minimisation

Once the initial estimate of $\mathtt{P}$ has been found, an iterative method can be used to refine the solution. The method used here is Levenberg-Marquardt minimisation [78], and the error function (3.9) is minimised.

When using Levenberg-Marquardt minimisation, the scale factor needs to be taken into account to produce a minimum parameterisation. Either one of the elements of $\mathtt{P}$ can be fixed, or $\mathtt{P}$ can be scaled between the iterations [38] so $\| \mathtt{P} \| = 1$. Here, one element of $\mathtt{P}$ is fixed ($\mathtt{P}_{34} = 1$), and the other 11 parameters are allowed to change. This could produce inaccuracies if, in the solution, $\mathtt{P}_{34}$ is *much* larger or smaller than the other elements in $\mathtt{P}$, but this has not been noticed in practice. If this does then it is very simple to scale another parameter to 1.

Decomposing the Projection Matrix

Once the solution for $\mathtt{P}$ has been found, it has to be decomposed into the form of equation (2.2). The 3×3 submatrix of $\mathtt{P}$ can be expressed as

$$\begin{bmatrix} \mathtt{P}_{11} & \mathtt{P}_{12} & \mathtt{P}_{13} \\ \mathtt{P}_{21} & \mathtt{P}_{22} & \mathtt{P}_{23} \\ \mathtt{P}_{31} & \mathtt{P}_{32} & \mathtt{P}_{33} \end{bmatrix} = \mathtt{CR},$$

where $\mathtt{C}$ is upper triangular and $\mathtt{R}$ is orthonormal. This allows QR decomposition [31] to be used to find $\mathtt{C}$ and $\mathtt{R}$. Faugeras and others [23, 29] decomposed $\mathtt{P}$ using an equivalent series of equations containing all the parameters, but the implementation of QR decomposition provides a more systematic approach.

- Accurately find the image positions of the corners of the Tsai grid by intersecting lines already fitted to the grid by orthogonal regression.
- Estimate the projection matrix using equation (3.11) for the initial estimate, and then iterative non-linear minimisation of the error function (3.9).
- Decompose the projection matrix using QR decomposition to give the camera calibration

Algorithm 3.1: *Off-line camera calibration.*

3.2.2 Results for Traditional Camera Calibration

This section analyses the stability and accuracy of camera calibration when using a calibration object. A set of 24 images with the calibration grid in various positions relative to the camera is used (see figure 3.3). The calibration is computed for each image using algorithm 3.1. The accuracy and stability of the calibration can be shown in two ways:

- The accuracy of `P` can be examined by measuring the difference between the actual corners in the image and the projection of the corresponding 3D points using `P` (see equation (3.9)).
- The stability of the calibration parameters can be examined by comparing the values calculated when the relative position between the grid and camera is varied (generally the grid is centred in the image and is viewed as large as possible).

The calibration object used is shown in figure 3.1a. It consists of two orthogonal Tsai grids. The points used are the corners of the 16 squares on each grid. The 3D points have been accurately measured, and the positions of the corners in the image can be found to sub-pixel accuracies using the intersections of the lines on the grid[3].

[3]The software to automatically find the corners by straight line intersection was written by Paul Beardsley.

Errors in the Image Plane

Table 3.1 shows the average error in the image plane for the projected points. The average is calculated, for the 128 points on the calibrated grid, over the 24 images used. Results are given for two estimates of `P`.

1. The initial (linear) solution for `P` from equation (3.11).

2. The final (non-linear) solution for `P` after Levenberg-Marquardt minimisation.

The initial estimate has a wide variation of errors, but the final estimate typically has errors < 0.2 pixels.

Variation of the Internal Parameters

The internal parameters are calculated for the set of 24 images. The average value and standard deviation for the internal parameters are given in table 3.2.

The results show that α_u, α_v, and ζ are stable, varying by less than 0.5%. In contrast the principal point (u_0,v_0) varies over a 40 $\times$ 40 pixel region in a 512 $\times$ 512 pixel image. This variation commonly occurs in calibration [94] but the magnitude is exaggerated here by having the image of the grid in different parts of the image.

Figure 3.2 shows the calculated position of the principal points when the image of the grid is in the four corners of the image, as well as centred in the image (see figure 3.3). The position of the principal point is definitely related to the relative positioning of the grid and camera, and is probably related to the small amount of radial distortion in the camera.

	Error (pixels)		
Estimate	average	max.	min.
Linear	0.625	2.075	0.256
Non-Linear	0.158	0.283	0.107

Table 3.1: *The reprojection error for the image point, for 128 points over 24 images, with linear and non-linear estimates of* P.

Summary

This chapter has reviewed the different methods of camera calibration. Section 3.2 explained the theory and gave results for a traditional method for the off-line calibration of a camera.

Section 3.1 introduced the idea of self-calibration, where the camera calibration can be computed using only information contained in the images themselves, and explained the actual knowledge required for self-calibration. It reviewed the many methods that have been suggested for self-calibration, and derived many of the basic results. A problem for many of the methods is the algebraic and numerical complexity of self-calibration, and that the methods are slow and require extreme accuracy of computation. Also, increasing the number of views used greatly increases the complexity of the computation. This is not advantageous as increasing the number of images used in the computation should improve the accuracy obtainable.

In the next chapter, two ideas from chapters 2 and 3 are combined in a novel self-calibration method, which by stratifying the problem, it reduces the numerical complexity. The method also allows the extension to the simultaneous use of an arbitrary number of images.

	α_u (pixels)	α_v (pixels)	ζ (-)	u_0 (pixels)	v_0 (pixels)	k (pixels)
Average	648.7	972.7	1.4993	251.8	256.3	2.0
Std. Dev.	2.8	4.7	0.0019	20.1	20.5	0.5
Ratio	0.4%	0.5%	0.1%	8.0%	8.0%	—
Max.	657.8	984.7	1.50383	294.0	306.1	3.8
Min.	643.5	965.1	1.49678	214.9	211.6	1.1

Table 3.2: *The distribution of the internal parameters calculated from 24 images of the calibration grid. The camera focal length is approximately 8.5mm. Ratio is the ratio of the standard deviation to the average value.*

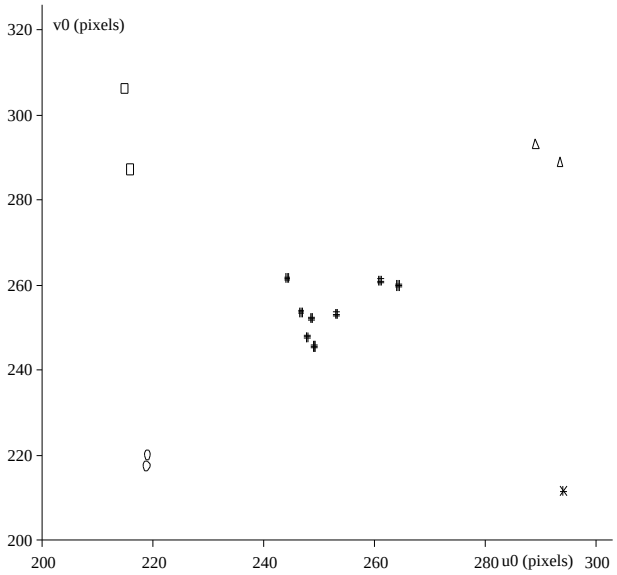


Figure 3.2: *The variation of the principal point with position of the grid in the image (+ centre, * top right, ∘ top left, □ bottom left, and △ bottom right).*

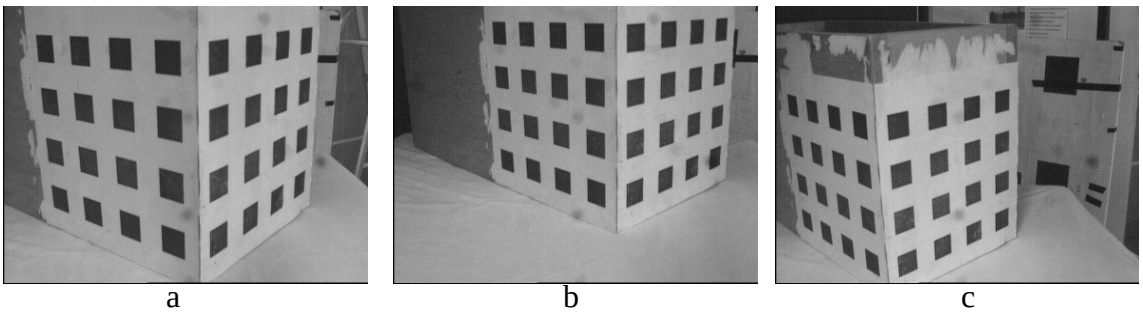


Figure 3.3: *The placement of the calibration grid in different position in the image. (a) the normal centred position, (b) and (c) the typical off-centre placements.*

Chapter 4

Self-Calibration via Affine Structure

Overview

The previous chapter reviewed the different methods that have been suggested for self-calibration. A common problem encountered was the algebraic and numerical complexity of the algorithms. In this chapter, a novel method for self-calibration is presented [4], which stratifies the problem and thereby reduces the numerical complexity. The method extends and combines two ideas (by different authors) from chapters 2 and 3.

First, affine structure is recovered using a translating camera, extending the work by Moons et al. [71]. Then self-calibration can be performed using the affine structure work of Luong and Vieville [61]. Unlike other methods, the algorithm does not require a large non-linear minimisation, and can be applied to an arbitrary number of images without increasing the complexity.

The extension of recovering affine structure from a translating camera is explained in section 4.1.1. It is also shown in section 4.1.2 that affine structure can be recovered using four views from a camera moving with general motion. The theory of self-calibration from affine structure is described in detail in section 4.1.3. An extensive assessment of the accuracy of the algorithm is performed using real images in section 4.2.1. Finally, two applications are described and results given in section 4.2.2.

4.1 Theory of Self-Calibration via Affine Structure

This section introduces a new method for self-calibration [4] which uses a stratified approach to the problem. The method combines and extends the work of Moons *et al.* [71] for computing affine structure with a translating camera (see section 2.2.6), with the work of Luong and Vieville [61] for self-calibration using affine structure (see section 3.1.6). Full implementation details are given as well as the algorithm summaries 4.1 and 4.2.

4.1.1 Affine Structure from a Translating Camera

When there is only translation of the cameras between two views of the scene, Moons *et al.* [71] showed that affine structure can be recovered. The original method used a minimum number of points to form an affine coordinate frame. Here the method is extended to used all the points, and involves estimating the projection matrices, and then back-projecting the rays to intersect in 3-space. With a translating camera the projection matrices have a special form which gives an affine reconstruction.

Lemma 4.1 *When a camera is undergoing pure translation, a special form can be used for the projection matrices of the projective camera, which allows the structure to be recovered up to an affine ambiguity. The projection matrices for two views are* $\mathtt{P}_a = [\mathtt{I}|\mathbf{0}]$ *and* $\mathtt{P}'_a = [\mathtt{I}|\mathbf{e}]$*, where* $\mathbf{e}$ *is the epipole.*

Proof: Without loss of generality [23, 38], the projection matrix for the first view of the affine structure can be set to

$$\mathtt{P}_a = [\mathtt{I}|\mathbf{0}] \,.$$

The pure translation between the first and second views allows a special form to be chosen for the second projection matrix ($\mathtt{P}'_a$). When the cameras are calibrated, the projection

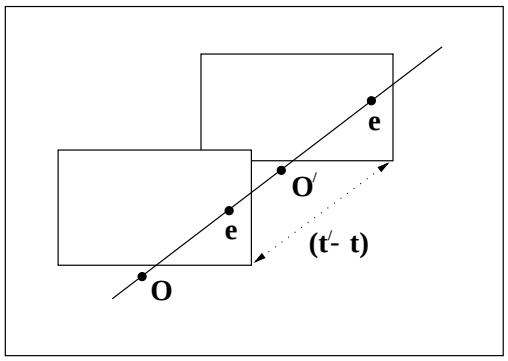


Figure 4.1: *The two translated image planes with optical centres* **O** *and* **O**′*, the same epipole* **e***, and related by the translation (*$\mathbf{t}' - \mathbf{t}$*).*

matrices ($\mathtt{P}_m, \mathtt{P}'_m$) can be expressed in the form of equation (2.2)

$$\begin{aligned} \mathtt{P}_m &= \mathtt{C}[\mathtt{R}|\mathbf{t}], \\ \mathtt{P}'_m &= \mathtt{C}[\mathtt{R}'|\mathbf{t}'], \end{aligned}$$

but for pure translation, $\mathtt{R} = \mathtt{R}'$. The affine and metric structure are related by an affine transformation $\mathtt{H}$ (see equation (2.13)) such that

$$\mathbf{X}_m = \mathtt{H}\mathbf{X}_a,$$

and both project to the same image points,

$$\mathbf{x} = \mathtt{P}_a\mathbf{X}_a = \mathtt{P}_m\mathbf{X}_m = \mathtt{P}_m(\mathtt{H}\mathbf{X}_a).$$

Hence,

$$\mathtt{H} = \begin{bmatrix} (\mathtt{CR})^{-1} & -\mathtt{R}^{-1}\mathbf{t}' \\ \mathbf{0}^\top & 1 \end{bmatrix},$$

and from $\mathtt{P}'_a = \mathtt{P}'_m\mathtt{H}$

$$\mathtt{P}'_a = [\mathtt{I}|\mathtt{C}(\mathbf{t}' - \mathbf{t})]. \tag{4.1}$$

The translation $(\mathbf{t}' - \mathbf{t})$ and calibration $\mathtt{C}$ are unknown, but $\mathtt{C}(\mathbf{t}' - \mathbf{t})$ is the epipole $(\mathbf{e})$ of the images. The epipole is the intersection of the lines joining the camera centres with the

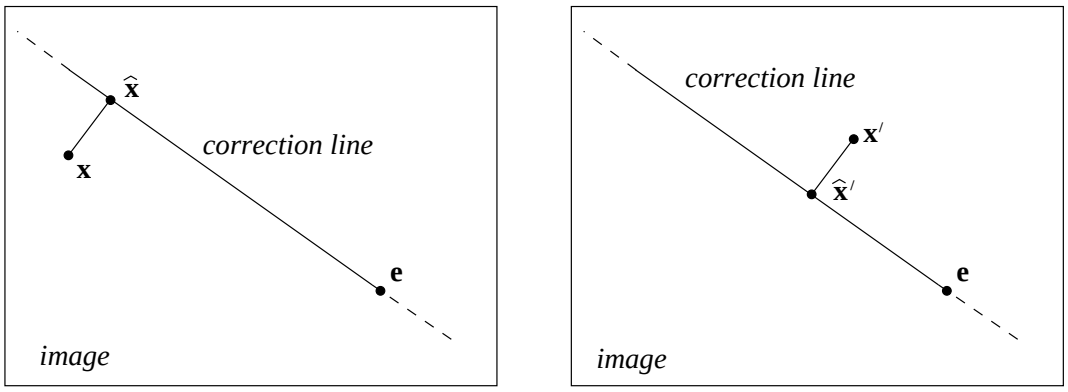


Figure 4.2: *The* image plane correction *used to find the 3D point by back-projection. (a) and (b) are the two translated views, and have the same epipole (***e***) and correction line.*

image planes (see figure 4.1), and can be expressed as the projection onto the image plane, using the calibration matrix $\mathtt{C}$, of the direction of translation $(\mathtt{t}' - \mathtt{t})$. So equation (4.1) can now be written as

$$\mathtt{P}'_a = [\mathtt{I}|\mathbf{e}]. \tag{4.2}$$

The 3-space structure can now be recovered by back-projecting the image points using the projection matrices $\mathtt{P}_a$ and $\mathtt{P}'_a$. The projective transformation $\mathtt{H}$ from the metric structure $\mathbf{X}_m$ to the actual recovered structure $\mathbf{X}_a$ is in the same form as equation (2.13), and so the recovered structure is defined up to an affine ambiguity.

□

Finding the Epipole The epipole is found using the fundamental matrix and solving the equation $\mathtt{F}\mathbf{e} = \mathbf{0}$. For a translating camera with fixed internal parameters, the fundamental matrix is skew symmetric [23] with three homogeneous parameters, which improves the numerical calculation of $\mathtt{F}$. More details can be found in appendix A.

- The fundamental matrix `F` is computed using the point matches between the images. `F` is skew symmetric with 2 degrees of freedom.
- The epipole is found, and is used to form the projection matrix (equation (4.2)).
- Corrected image points are found by projecting the points onto a *correction line*. The *correction line*, radiating from the epipole, is positioned to minimises the perpendicular distance from the matched points to the line.
- The corrected image points are back-projected to intersect in 3-space, to give the points for the affine reconstruction.

Algorithm 4.1: *Recovering affine structure using a translating camera.*

Back-Projection from the Image Plane

The different methods for back-projecting image points were reviewed in section 2.2.2, but as the structure is only recovered up to an affine ambiguity, the problem of intersecting skewed rays still remains. Hartley and Sturm's method [43] can be used, and with the degenerate motion and fundamental matrix, the polynomial equation reduces from degree six to a quadratic with two solutions.

The following constraint gives the same result as Hartley and Sturm's method, but has just one solution rather than the two, and was derived independently [4]. The method *corrects* the corresponding points in the images so that the back-projected rays do intersect. With the epipoles in the same position on both images, the images can be superimposed, and the *image error* for both images can be minimised simultaneously. A *correction line* is defined as the line passing through the epipole that minimises the perpendicular distance from the line to the points $\mathbf{x}$ and $\mathbf{x}'$ (see figure 4.2). The points are then projected onto this line generating corrected points $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$. The corrected back-projected rays intersect, giving a point in 3-space.

4.1.2 Affine Structure from Four Views

This section shows that with four views it is possible to recover affine structure when the camera is rotating as well as translating. The method is not used for the work presented here, but is given to show a more general approach to obtaining affine structure.

Section 2.2.5 showed that to transform from a projective to an affine reconstruction requires that the plane at infinity π_∞ is identified. Equation (2.12) showed that the metric and projective reconstruction, $\mathbf{X}_m$ and $\mathbf{X}_p$ respectively, are related by a transformation $\mathtt{H}$ such that

$$\mathbf{X}_m = \mathtt{H}\mathbf{X}_p = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\top & 1 \end{bmatrix} \mathbf{X}_p,$$

where $(\mathbf{v}^\top, 1)^\top$ is the position of π_∞ in the projective reconstruction. The projection matrices for two views of the metric structure can be set to[1]

$$\begin{aligned} \mathtt{P}_m &= \mathtt{C}\,[\mathtt{I}|\mathbf{0}]\,, \\ \mathtt{P}'_m &= \mathtt{C}\mathtt{R}\,[\mathtt{I}|-\mathbf{t}]\,, \end{aligned}$$

while the corresponding projection matrices for the projective structure are

$$\begin{aligned} \mathtt{P}_p &= [\mathtt{I}|\mathbf{0}]\,, \\ \mathtt{P}'_p &= [\mathtt{M}|\mathbf{m}]\,. \end{aligned}$$

The projective and metric structures project to the same image points, such that for the first view

$$\mathbf{x} = \mathtt{P}_p\mathbf{X}_p = \mathtt{P}_m\mathbf{X}_m = \mathtt{P}_m\mathtt{H}\mathbf{X}_p,$$

and hence

$$\mathtt{P}_p = \mathtt{P}_m\mathtt{H} = \begin{bmatrix} \mathtt{A} & \mathbf{t} \\ \mathbf{v}^\top & 1 \end{bmatrix} = \lambda\mathtt{C}[\mathtt{A}|\mathbf{t}],$$

[1] The second projection matrix for the metric structure has a slightly different form from that in equation (2.2), but the only difference is that the translation is measured in a different coordinate frame.

where λ is the unknown scaling in the projection matrices, and this gives

$$\begin{aligned} \mathtt{A} &= \frac{1}{\lambda}\mathtt{C}^{-1}, \\ \mathbf{t} &= \mathbf{0}. \end{aligned}$$

The projection into the second view is

$$\mathbf{x}' = \mathtt{P}'_p\mathbf{X}_p = \mathtt{P}'_m\mathbf{X}_m = \mathtt{P}'_m\mathtt{H}\mathbf{X}_p,$$

which following the same analysis gives

$$\begin{aligned} \mathtt{P}'_p &= \lambda'\mathtt{CR}\left[\mathtt{C}^{-1}/\lambda - \mathbf{t}\mathbf{v}^\top | - \mathbf{t}\right], \\ \mathbf{t} &= -\frac{1}{\lambda'}(\mathtt{CR})^{-1}\mathbf{m}, \end{aligned}$$

and substituting for $\mathbf{t}$ gives

$$\mathtt{M} - \mathbf{m}\mathbf{v}^\top = \frac{\lambda'}{\lambda}\mathtt{CRC}^{-1}. \tag{4.3}$$

The matrix $\mathtt{CRC}^{-1}$ is a conjugate rotation matrix which has the same eigenvalues as $\mathtt{R}$, which are 1 and $e^{\pm i\theta}$. Hence, the matrix $\mathtt{M} - \mathbf{m}\mathbf{v}^\top$ has the eigenvalues (λ'/λ) and $(\lambda'/\lambda)e^{\pm i\theta}$. The sum ($f_1$), sum of pairs ($f_2$), and product ($f_3$) of the eigenvalues of $\mathtt{M} - \mathbf{m}\mathbf{v}^\top$ can be expressed as

$$f_1 = -(\lambda'/\lambda)\left(1 + e^{+i\theta} + e^{-i\theta}\right) = -(\lambda'/\lambda)(1 + 2\cos\theta), \tag{4.4}$$

$$f_2 = (\lambda'/\lambda)^2\left(e^{+i\theta} + e^{-i\theta} + e^{+i\theta}e^{-i\theta}\right) = (\lambda'/\lambda)^2(1 + 2\cos\theta), \tag{4.5}$$

$$f_3 = -(\lambda'/\lambda)^3\left(e^{+i\theta}e^{-i\theta}\right) = -(\lambda'/\lambda)^3. \tag{4.6}$$

The sum, sum of pairs, and product of the eigenvalues of $\mathtt{M} - \mathbf{m}\mathbf{v}^\top$ can also be expressed as linear functions of $\mathbf{v}$

$$\begin{aligned} f_1 &= \mathrm{tr}(\mathtt{M}) - \mathbf{m}^\top\mathbf{v} \\ f_2 &= 1/2\left(\mathrm{tr}(\mathtt{M})^2 - \mathrm{tr}(\mathtt{M}^2)\right) + (\mathtt{M}\mathbf{m} - \mathrm{tr}(\mathtt{M})\mathbf{m})^\top\mathbf{v} \\ f_3 &= \det(\mathtt{M}) - \det(\mathtt{M})(\mathtt{M}^{-1}\mathbf{m})^\top\mathbf{v} \end{aligned}$$

Eliminating λ, λ' and θ from equations (4.4), (4.5) and (4.6) gives a quartic equation in $\mathbf{v}$

$$f_3 f_1^3 = f_2^3. \tag{4.7}$$

Each pair of images will give a constraint in the form of equation (4.7), and four images will give three independent constraints for the three elements of $\mathbf{v}$. Once $\mathbf{v}$ has be found, affine structure can be recovered using equation (2.14).

Independently, Pollefeys *et al.* [76] derived the same result and termed it the *modulus constraint.* They solved the three simultaneous quartic equations using non-linear minimisation.

4.1.3 Self-Calibration from Affine Structure

Luong and Vieville [61] showed that a camera can be calibrated using two or more rotated views of an affine structure (reviewed in section 3.1.6). From each view a projection matrix can be computed using the methods described in section 3.2. Each pair of views can then be used to compute an *infinite homography* using equation (3.3),

$$\mathtt{H}_\infty = \mathtt{M}'\mathtt{M}^{-1},$$

and each infinite homography gives the constraints on $\mathtt{K}$ in form of equation (3.7)

$$\mathtt{K} = \mathtt{H}_\infty \mathtt{K} \mathtt{H}_\infty^\top.$$

The method requires that the internal parameters remain fixed, and if there is no rotation, equation (3.7) reduces to $\mathtt{K} = \mathtt{K}$, and there is no constraint on $\mathtt{K}$.

Solving the Constraint Equations

The constraint equation (3.7) can be rearranged into the form $\mathtt{Z}_i\mathbf{k} = \mathbf{0}$, where $\mathbf{k}$ is the six distinct elements of the symmetric matrix $\mathtt{K}$ written as a vector

$$\mathbf{k} = (K_{11}, K_{12}, K_{13}, K_{22}, K_{23}, K_{33})^\top,$$

- Compute the affine projection matrices by any appropriate method.
- Compute the *infinite homographies* ($\mathtt{H}_\infty$) between the different views using equation (3.3).
- Solve the constraint equation (3.7) for $\mathtt{K}$ using equation (4.8) and possibly equation (4.9) if $\mathtt{Z}$ is Rank deficient.
- Use Choleski decomposition to find $\mathtt{C}$ from $\mathtt{K}$.
- Use equation (2.16) to upgrade from affine to metric structure.

Algorithm 4.2: *Self-calibration and metric structure recovery using affine structure.*

and $\mathtt{Z}_i$ is a 6×6 matrix and a function of the infinite homography. A different $\mathtt{Z}_i$ can be obtained from each pair of views, and these can be compounded into a $6n \times 6$ matrix $\mathtt{Z}$ which gives the equation for n pairs of views as

$$\mathtt{Z}\mathbf{k} = \mathbf{0}. \tag{4.8}$$

The solution to this equation is null space of $\mathtt{Z}$, which for real data can be found using Singular Value Decomposition (SVD) to solve $\min_{\mathbf{k}} \| \mathtt{Z}\mathbf{k} \|_2$ subject to $\| \mathbf{k} \|_2 = 1$.

However, when using a single pair of views or when the rotation between all views is about the same axis, then $\mathtt{Z}$ is Rank 4 [61] and there is a one-parameter family of solutions to equation (4.8). This is the same as the situation described in section 3.1.2, and the constraint of no image skew can be used to identify the correct solution. Now the range of solutions to equation (4.8) is spanned by the two eigenvectors $\mathbf{l}$ and $\mathbf{m}$ associated with the two smallest eigenvalues (found using SVD) to give $\mathbf{k} = \mathbf{l} + \lambda\mathbf{m}$, where λ is the free parameter. The value of λ corresponding to the correct solution can be found by rearranging equation (3.2) to give the quadratic equation

$$\lambda^2(m_2m_6 - m_3m_5) + \lambda(l_2m_6 + l_6m_2 - l_3m_5 - l_5m_3) + (l_2l_6 - l_3l_5) = 0, \tag{4.9}$$

which has two possible solutions, giving two possible solutions for $\mathtt{K}$. Often, one of the solutions for $\mathtt{K}$ is not positive definite and so Choleski decomposition will not give real values for the internal parameters, hence can be eliminated. Once $\mathtt{C}$ has been found, the metric structure can be recovered using equation (2.16).

Degeneracies

If camera rotation is about one of the camera axes, then some of the internal parameters are unconstrained by equation (3.7), and cannot be determined. From equation (3.5), when $\mathtt{C} = \mathtt{C}'$, and $\mathtt{R}$ is the rotation between the camera positions, then

$$\mathtt{H}_\infty = \mathtt{C}\mathtt{R}\mathtt{C}^{-1}. \tag{4.10}$$

The degeneracies can be analysed by examining the infinite homography ($\mathtt{H}_\infty$) and expanding equation (4.10). When the image skew is assumed to be zero ($k = 0$), a rotation of θ about the camera X axis gives

$$\mathtt{H}_\infty(\mathtt{R}(\theta)) = \begin{bmatrix} 1 & u_0 \sin\theta/\alpha_v & u_0(\cos\theta - 1) - u_0 v_0 \sin\theta/\alpha_v \\ 0 & \cos\theta + v_0 \sin\theta/\alpha_v & v_0(\cos\theta - 1) - v_0^2 \sin\theta/\alpha_v - \alpha_v \sin\theta \\ 0 & \sin\theta/\alpha_v & \cos\theta - v_0 \sin\theta/\alpha_v \end{bmatrix}.$$

There are no α_u terms in $\mathtt{H}_\infty$ and consequently no possible constraint on α_u in equation (3.7) (the coefficient of α_u is zero). Consequently, α_u is unconstrained. Similarly, with a rotation about the Y axis, α_v is unconstrained. For a rotation of ϕ about the Z axis (the optical axis) with $k = 0$

$$\mathtt{H}_\infty(\mathtt{R}(\phi)) = \begin{bmatrix} \cos\phi & -(1/\zeta)\sin\phi & -(1/\zeta)v_0 \sin\phi + u_0(1 - \cos\phi) \\ \zeta \sin\phi & \cos\phi & -\zeta u_0 \sin\phi + v_0(1 - \cos\phi) \\ 0 & 0 & 1 \end{bmatrix},$$

in which case α_u and α_v only appear in $\mathtt{H}_\infty$ as the aspect ratio (ζ), so their independent values cannot be determined.

Near-degenerate situations can occur if the magnitude of rotation about one axis is small. In this case the associated internal parameter is poorly constrained.

4.2 Results for Self-Calibration via Affine Structure

In this section, the method of self-calibration described in section 4.1 is implemented, and results obtained using real images. Assessment of the method is made for different number of images and size of rotation between images, and two possible applications are described with results.

4.2.1 Assessment of Method

The object used for the assessment is the calibration object in figure 3.1a. The known 3D structure of the object is **not** used to calibrate (unlike section 3.2) but is used to assess the accuracy of the calibration and structure reconstruction. The accuracy of the calibration is assessed in two ways:

- The internal parameters determined using self-calibration can be compared to the veridical values. The veridical values are obtained by using the calibration object for conventional calibration. The calibration parameters are obtained from each image using the method given in section 3.2, and the veridical values are set to the average over all the images.

- Comparing the recovered 3D metric structure with that of the known calibration object. The comparison is done with several metric invariants, measured on both the actual object and the reconstruction.

In the following, a number of different types of sequences are compared, which differ in the number of views and the rotation between views. In all cases, affine structure is recovered from the first two views of the image sequence, which are related by a translation, using algorithm 4.1. The image points used are the corners of the squares on the calibration grid which are found by line intersection (see section 3.2).

3 Images There is a pure translation between the first two images, and a translation and rotation for the third. The translation between the second and third images is used to fixate on the object which allows the longer sequences used below. $\mathtt{H}_\infty$ is determined between views two and three (it is identical to that obtained from views one and three). As described in section 4.1.3, the additional constraint that $k = 0$ is required in this case of a single rotated view. Two sets of results are given. For the first (5°), the rotation magnitude is 5° about both the camera X and Y axes. For the second (10°), at least one of the rotations has magnitude larger than 5°. The rotation is limited to 15° so that both sides of the calibration grid are still in view in the image. The larger rotations give more stable values, as discussed in section 4.1.3.

4 Images A second rotated view is added to the sequence, and the two $\mathtt{H}_\infty$ matrices, found from the mapping from the second to the third, and second to the fourth images, are used simultaneously to find $\mathtt{K}$. This second $\mathtt{H}_\infty$ generally changes the rank of $\mathtt{Z}$ (see equation (4.8)) from four to five, so the constraint $k = 0$ is not required, and the calculated value for k given. Note, $\mathtt{H}_\infty$ between the third and fourth images adds no additional information. These images are selected from the (5°) and (10°) sets above, such that rotations about corresponding camera axes (e.g., the X axis) in the third and fourth views differ in magnitude.

6 Images Four rotated views are used to give a 24×6 matrix $\mathtt{Z}$ of rank five, with the $\mathtt{H}_\infty$ mapping from the second image to each of the four rotated images. These give the most stable results, with internal parameters approaching the veridical values.

Degenerate A three-image sequence is used, but the rotation is here limited to either the camera X or Y axis. As shown in section 4.1.3, this results in α_u or α_v being unconstrained.

Method (no. of seqs.)	α_u (pixels)	α_v (pixels)	ζ	u_0 (pixels)	v_0 (pixels)	k (pixels)
Known 3D Grid	646.0(3.5)	968.7(4.7)	1.4996(0.0016)	246.5(9.2)	244.3(11.4)	1.5(0.4)
3 Images-5°(9)	625.2(41.1)	954.5(35.5)	1.53(0.11)	242.8(20.1)	225.6(29.4)	*0*
3 Images-10°(6)	684.4(26.7)	1034.0(24.9)	1.513(0.081)	261.0(4.0)	234.2(32.0)	*0*
4 Images (13)	661.0(17.0)	1018.0(22.2)	1.522(0.028)	258.0(7.7)	239.7(11.6)	1.2(36.6)
6 Images (35)	645.6(13.1)	1001.8(16.9)	1.552(0.021)	260.0(4.6)	236.9(6.6)	21.8(14.4)
Degenerate-X (6)	–	1064.0(32.5)	–	245.0(4.2)	208.0(3.0)	*0*
Degenerate-Y (2)	687.2(34.9)	–	–	260.2(27.2)	33.8(13.7)	*0*

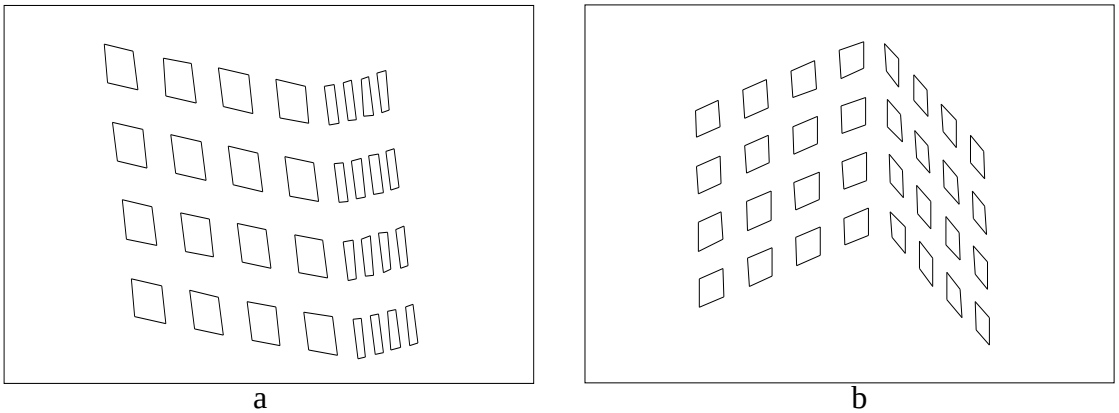
Table 4.1: *Mean (standard deviation) of the internal parameters determined from a varying number of views and rotations. – indicates that no value was obtained, and 0 indicates the skew is set to zero. See text for details.*

Internal Parameters

The results for the internal parameters, calculated from the various image sequences, are given in table 4.1. The stability of the internal parameters increases with the number of views and the size of the rotations. The six image sequence gives the most stable results with internal parameters approaching the veridical values. The parameters α_u, α_v, and ζ have an error of between 2% and 6%. Again (u_0, v_0) varies over a region in the centre of the image, the size of the region varying from 20×30 to 5×5, decreasing with increasing number of views and the magnitude of rotation. In the six-image case, the skew parameter ($k = 21.8$) is significantly larger than the veridical value ($k = 1.5 \pm 0.4$), but this corresponds to only 2° off perpendicular. The results for the degenerate sequences confirm the theory of section 4.1.3 with α_u or α_v being unconstrained.

Metric Reconstruction

Metric structure is recovered from the affine structure and camera calibration using equation (2.16). The accuracy of the reconstruction indicates how errors in camera calibration propagate through to errors in structure. For example, an error in α_u could have a more detrimental effect than one in u_0. The accuracy of the metric reconstruction is assessed



a b

Figure 4.3: *(a) Affine reconstruction from two perspective images of the calibration object with pure translation (note the in plane skew). (b) Metric structure obtained from three perspective images, after determining internal parameters. The planes are now orthogonal to high accuracy (see table 4.2).*

by comparing similarity invariants (angles, distance ratios) with those calculated from the known structure of the calibration object. Results are given in table 4.2 and figure 4.3, and the same image sets given above are used. Three similarity invariants are measured, the first two measure local structure, and the third global.

1. **Distance error** The distance error is the ratio of the standard deviation to mean of 196 measurement of the distance between adjacent points on the grid.

2. **Pattern error** The pattern error is the standard deviation of angles computed for all corners of the squares on the grid.

3. **Plane angle**. The plane angle is the angle between the two planes of the calibration object, determined by orthogonal regression to points on each grid.

In all cases the recovered metric structure is clearly reasonably accurate. Results improve for larger rotation angles and greater number of images.

Method	Distance Error	Pattern Error	Plane Angle
3 Images-5°	4.5%	2.51°	$91.03°(3.80°)$
3 Images-10°	3.9%	1.66°	$91.52°(4.20°)$
4 Images	3.9%	1.12°	$89.33°(2.21°)$
6 Images	3.7%	1.00°	$90.35°(1.07°)$

Table 4.2: *Similarity invariants measured on recovered metric structure: the actual measurements are explained in the text, but for ideal data the distance and pattern error would be zero, and the plane angle 90°. The standard deviation for the plane angle is given in brackets.*

4.2.2 Applications

Active Vision

For a camera mounted on a robot arm or AGV, it is usually not difficult to perform pure translational motion[2]. Consequently, the method in section 4.1.1 for generating affine structure, is particularly appropriate to active vision tasks. The following results are for an implemented system [8], with automatic corner detection, matching and elimination of outliers. See [91] for details of the outlier rejection method. During the motion the camera is calibrated *on the fly* and metric structure recovered. Note, corners are detected here using a corner detector, thus localisation is less accurate than the line intersection method used for the calibration grid.

Figure 4.4 shows two images from a sequence of 20 taken by a camera mounted on an Adept robot arm where the camera rotates by 20° overall about its X and Y axes. The calculated internal parameters are $(\alpha_u = 673.1, \alpha_v = 1005.4, \zeta = 1.494, (u_0, v_0) = (249.3, 282.9))$, compared with values in table 4.1 where the same camera was used. The recovered structure is shown in figure 4.5, and three views of the reconstruction with texture mapping of the image are shown in figure 4.6. The angles between the roof, front

[2]The accuracy of the translational motion which can be achieved using an AGV or robot arm varies considerably.

and side of the house were calculated, by orthogonal regression to points on each plane, to be $53.9°, 94.7°, 87.5°$ compared to measured values of $51.4°, 90.0°, 90.0°$.

Repeated Structure

Structures that repeat in a single image of a scene are equivalent to multiple views of a single instance of the structure [55]. Thus, for example, a view of two identical objects related by a translation is equivalent to a stereo pair of images of one object, with the cameras related by a pure translation. If in a single image there are three identical objects, of which two are related by a simple translation, then the camera can be calibrated and metric structure recovered. Figure 4.7 shows such an image and figure 4.8 shows the reconstruction. The angles between the planar sides of the reconstruction, found by orthogonal regression to points on each plane, are $84°, 81°, 81°$, compared to the actual value of $90°$.

Summary

This chapter has introduced a new method for self-calibration which reduces the numerical complexity of the problem by using stratification. It involved extending the method of obtaining affine structure from a translating camera, and then combining this result with the theory of self-calibration from affine structure. Results were given using real images, and two different applications were explained and results given.

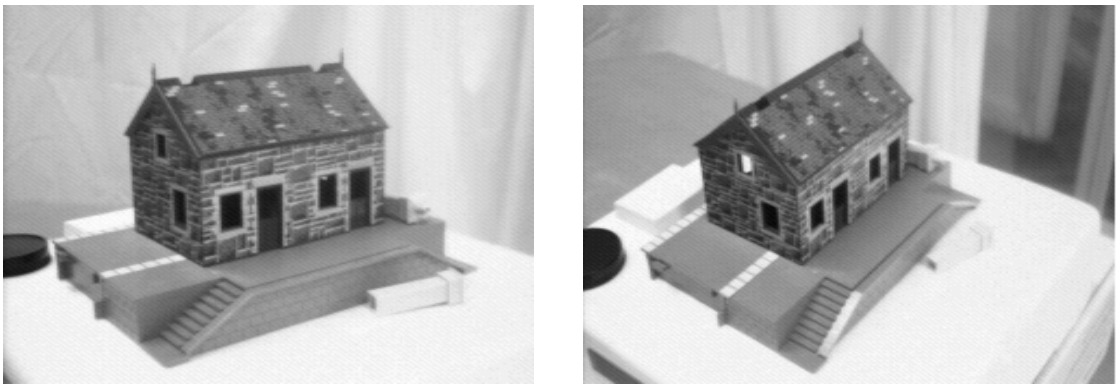


Figure 4.4: *Two images from the sequence of twenty used for self-calibration and metric reconstruction. The images were taken by a camera mounted on an Adept robot arm.*

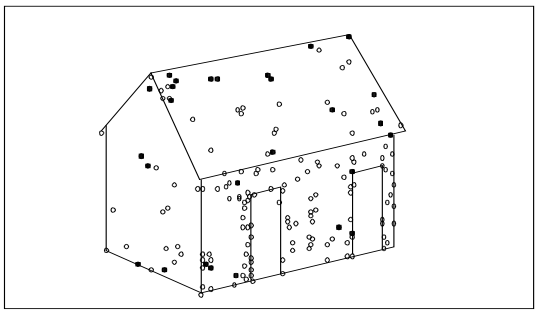


Figure 4.5: *The reconstructed house projected from a different viewpoint, with lines added for clarity. The reconstruction is for 100 points matched between the first two images, of which 30 (bold) were tracked through all 20 images and used to compute calibration.*

Figure 4.6: *Three views of the reconstructed house with the image texture mapped onto the reconstruction.*

Figure 4.7: *A single image containing repeated structure which is used for self-calibration.*

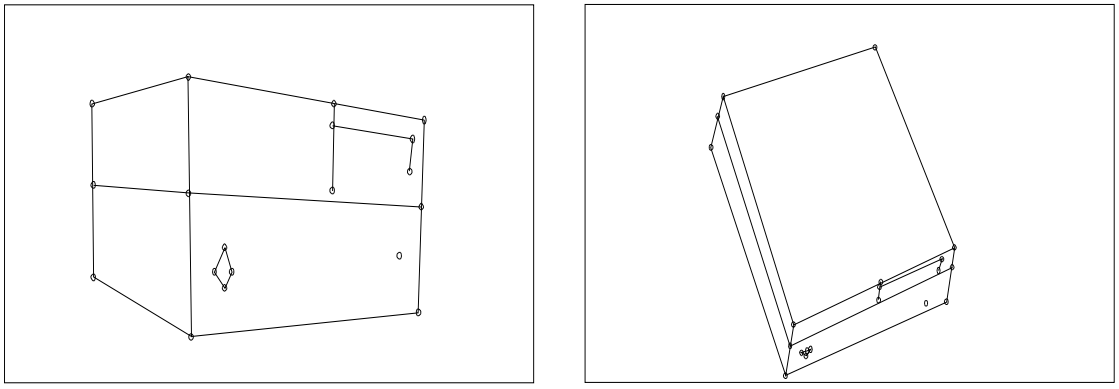


Figure 4.8: *Two views of the recovered metric structure from an image containing repeated structure.*

Chapter 5

Self-Calibration via Fixed Points

Overview

This chapter presents a novel method for self-calibrating a camera when the camera is undergoing planar motion and at least three rotated views are available [5]. The method is based on the idea that there are points and lines in the image and in the scene which remain fixed during planar motion. The positions of these points are determined by the camera calibration, and hence finding their positions enables the camera calibration to be computed.

The idea of fixed entities is introduced in section 5.1.1, and the fixed entities for general and planar motion are described in sections 5.1.2 and 5.1.4. How the fixed points determine camera calibration is discussed in section 5.2, where it is shown how affine and metric structure can be recovered. To recover full metric structure requires that one assumption is made about the camera calibration. If this is not possible, only planar metric structure is achievable. Section 5.2.3 shows that one of the most common assumptions used in self-calibration (zero image skew constraint) can give poor, unstable results, when used with a camera moving under planar motion.

Section 5.4 shows that the position of fixed entities in the image can be found using three images and the trifocal tensor, but that for normal planar motion the solution is a non-trivial algebraic problem. The complexity of the problem can be reduced by stratifying the problem into simpler steps. Section 5.3 shows that the position of some of the fixed

image entities can be found with much less difficulty using the fundamental matrices. These fixed entities can then be used to compute a projective transformation of the images. This transformation reduces the problem of finding the remaining fixed entities, using the trifocal tensor, to solving a cubic in one variable.

The algorithms used are described in detail in section 5.5, then results are given for several real image sequences in section 5.6. The results are shown to be both stable over the image sequences, and accurate when comparing the reconstructions to known ground truths.

5.1 Fixed Points of Planar Motion

5.1.1 Fixed Entities

Fixed entities are geometric objects in space whose position remains invariant during a transformation of the space. Some fixed entities have already been introduced (see section 2.2.4), not least the plane at infinity (π_∞) whose position remains invariant to affine transformations. More important to self-calibration and metric reconstruction is the absolute conic (Ω_∞), which remains invariant during Euclidean transformations (rigid motion).

The other fixed entities which can occur depend on the type of transformation, and the number of transformations considered. Here the fixed entities arising from Euclidean transformation (rigid motion) of 3-space are considered. The fixed entities for a single general Euclidean displacement are explained in section 5.1.2. Then planar motion is defined (section 5.1.3), and the fixed entities for planar motion are described in section 5.1.4.

There are two different types of fixed entities: those which are fixed point-wise, and those which are fixed merely as a set. For example Ω_∞ (π_∞) is a fixed conic (plane), but the points in the conic (plane) are not fixed point-wise (i.e., the conic (plane) is fixed as a set). During a Euclidean (affine) transformation the points in the conic (plane) can be mapped

to other positions in the conic (plane). Point-wise fixed entities are made up of fixed points whose position do not change during the transformation (e.g., a line of fixed points).

The intersection of set-wise fixed entities results in the *creation* of additional fixed entities of a *lower order* (i.e., if two fixed lines (not lines of fixed points) intersect, then the intersection point *is* a fixed point).

5.1.2 Fixed Entities of General Motion

The theory of kinematics state that any general rigid displacement (Euclidean transformation containing both rotation and translation) can be considered as a rotation about a *screw axis* and a translation along the *screw axis* [13].

For a single general displacement there are additional fixed entities in addition to the *normal* fixed entities of π_∞ and Ω_∞. The planes perpendicular to the rotation axis do not change orientation — they are rotated about, and translated along the axis — and so the axis of the pencil of planes is a fixed line on π_∞ (i.e., the line of intersection of the parallel planes remains fixed). The screw axis is a fixed line, as the motion can be considered as a rotation about and translation along it, and so points on the line are fixed set-wise. This results in four extra fixed entities for a single general displacement [107] (see figure 5.1):

1. **A fixed line** ($\mathbf{L}$) on π_∞, the axis of the pencil of planes perpendicular to the rotation (screw) axis.

2. **A fixed line** ($\mathbf{M}$) of 3-space, the screw axis.

3. **A fixed point** ($\mathbf{V}$) on π_∞, the intersection of π_∞ and the screw axis ($\mathbf{M}$).

4. **Two fixed points** ($\mathbf{I}$,$\mathbf{J}$) on Ω_∞, the intersection of the fixed line ($\mathbf{L}$) and Ω_∞, which are known as the circular points [81].

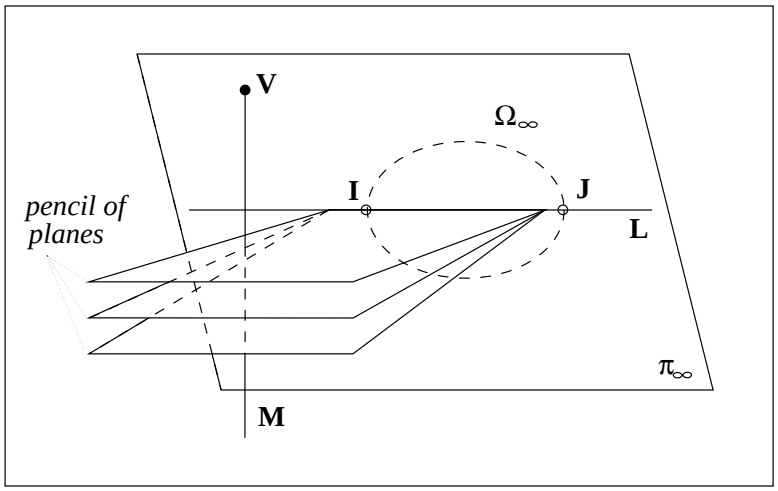


Figure 5.1: *The fixed entities for general motion. Fixed lines:* **M** *the screw axis; and* **L** *the axis of the pencil of planes perpendicular to the screw axes. Fixed points:* **V** *the intersection of* π_∞ *and* **M***; and the circular points* **I** *and* **J***, the intersection of* **L** *and* Ω_∞*. (Note: this is projective representation so that the parallel planes are not drawn as such.)*

However, for two or more general displacements, the screw axes will generally not be parallel (i.e., different rotation axes), and so the four extra fixed entities for a single general displacement will not exist when considering multiple general displacements.

5.1.3 Planar Motion

Planar motion occurs when the rotation axis is fixed throughout the sequence, and the translation is confined to a plane perpendicular to the rotation axis.

This type of motion is typical for camera based on a moving vehicle (e.g., a car or an Autonomous Guided Vehicle). The plane of translation does not have to correspond to any camera axes (unlike the Ground Plane Motion work of Wiles [101]), and hence the camera can point in any direction relative to the translation plane and rotation axis.

These extra constraints on the motion result in extra fixed entities existing for an arbitrary number of displacements of planar motion.

5.1.4 Fixed Entities of Planar Motion

As stated in section 5.1.1, a general displacement can be considered as a rotation about and translational along the *screw axis*. However, with planar motion, the displacement can be considered purely as a rotation about the screw axis, with no translation, as the translation is confined to a plane perpendicular to the rotation (screw) axis.

Single Displacement With planar motion, there is no translation along the screw axis, which changes the fixed entities. The screw axis is now a line of fixed points rather than just a fixed line. So the intersection of the screw axis and π_∞ is no longer uniquely defined as all points on the screw axis are fixed. Also there is a fixed pencil of planes rather than just the fixed line corresponding to the axis of the pencil of planes (as each plane is just rotated about the screw axis and not translated). However, $\mathbf{L}$ is still uniquely defined as π_∞ and the pencil of planes are just fixed point-wise, and hence the intersection line ($\mathbf{L}$) is a fixed line.

Multiple Displacements When considering multiple planar displacements, the screw axes will all be parallel (same rotation axis), but generally they will not be coincident because of the different planar translations. So there is no longer a line of fixed points for the screw axis, but now the intersection of the parallel screw axes will be a fixed point on π_∞. This is the fixed point $\mathbf{V}$ which is now uniquely identifiable (see figure 5.2a).

The pencil of planes remains fixed over multiple displacements, and so the axis of the pencil also remains fixed ($\mathbf{L}$ on π_∞, see figure 5.2a). The fixed line $\mathbf{L}$ still intersects Ω_∞ (see figure 5.2b), and so the circular points ($\mathbf{I}$ and $\mathbf{J}$) also remain fixed over multiple displacements. So for planar motion with multiple displacements, there are 3 fixed points and one fixed line:

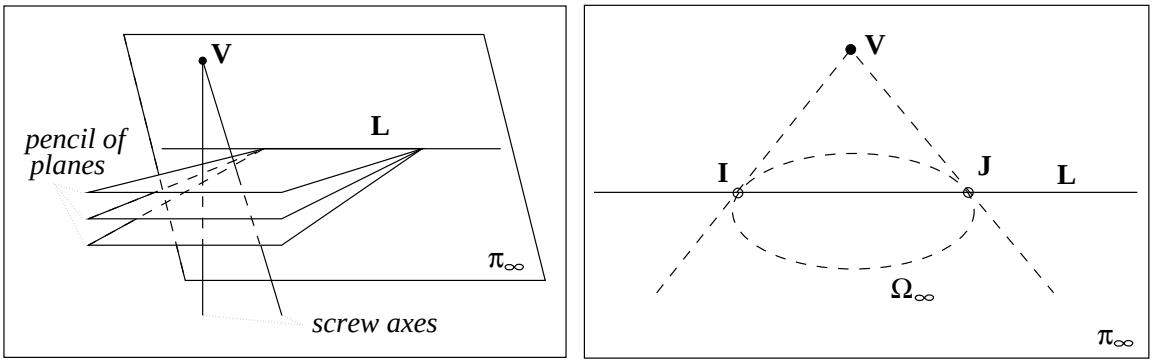


Figure 5.2: *The fixed entities for planar motion. (a) The fixed point* $\mathbf{V}$*, the intersection on* π_∞ *of the screw axes; the fixed line* $\mathbf{L}$ *the axis of the pencil of planes perpendicular to the rotation axis. (b) The circular points* $\mathbf{I}, \mathbf{J}$*, the intersection of* $\mathbf{L}$ *and* Ω_∞ *on* π_∞*. The points* $\mathbf{V}$*,* $\mathbf{I}$*, and* $\mathbf{J}$ *are an orthogonal triad of directions, and* $\mathbf{L}$ *and* $\mathbf{V}$ *are pole and polar with respect to* Ω_∞*.*

1. **The fixed line** $\mathbf{L}$ on π_∞, axis of the pencil of planes perpendicular to the screw axes.

2. **The fixed point** $\mathbf{V}$ on π_∞, the intersection of the screw axes.

3. **The two circular points** $\mathbf{I}, \mathbf{J}$ on π_∞, the intersection of $\mathbf{L}$ and Ω_∞.

The fixed entities $\mathbf{V}$ and $\mathbf{L}$ are pole and polar with respect to Ω_∞ [81]. Pole and polar are a point and line associated with a conic, where the two lines passing through the pole, which are tangential to the conic, are the tangents to the points of intersection of the polar and the conic (see figure 5.2b).

The three fixed points on π_∞ ($\mathbf{V}$, $\mathbf{I}$, and $\mathbf{J}$) are an orthogonal triad of directions, and this becomes important in section 5.2. The orthogonality can be shown by the examining the role of Ω_∞ in determining angles in $\mathcal{R}^3$, such that two lines are perpendicular if their intersections with π_∞ are conjugate[1] [81]. The fixed point $\mathbf{V}$ is the intersection of the screw axis and π_∞, while $\mathbf{I}$ and $\mathbf{J}$ lie in the plane perpendicular to the screw axis. Hence, both $\mathbf{I}$ and $\mathbf{J}$ are conjugate to $\mathbf{V}$, and the line $\mathbf{L} = \mathbf{I} \times \mathbf{J}$ is polar with respect to the pole $\mathbf{V}$.

[1]Two points on π_∞ are conjugate if their relationship to Ω_∞ is such that one point lies on the polar, which is defined by the other point as pole.

5.1.5 Fixed Entities associated with the Camera

The fixed entities associated with a camera consist of the entities in the image that are unchanged under motion of the camera (or equivalently, a motion of space). These include images of the fixed entities in 3-space for the given motion, but will also include other entities that are specific to the camera being considered. Only cameras with fixed internal parameters (calibration) are considered.

In the following, a geometric argument is given for the existence and form of the fixed entities associated with the camera. More details can be found in Maybank [63] and Semple and Kneebone [81] which give algebraic arguments and proofs.

Two Views (Single Displacement)

General Motion Consider a pair of views which are separated by a general Euclidean transformation. Corresponding points (i.e., those with the same image coordinates) can be back-projected to give rays in 3-space, but only some of these pairs of rays will intersect. The locus of all such intersections is known as the *horopter* [63]. In fact, using a theorem from [81] (chapter 12, theorem 14), it follows directly that the horopter is a twisted cubic[2]. By definition this must intersect the optical centres ($\mathbf{O}$ and $\mathbf{O}'$) and the three fixed points ($\mathbf{V}$, $\mathbf{I}$, and $\mathbf{J}$).

The horopter is imaged as a conic in the two images (since a twisted cubic is imaged as a conic if the projection centre lies on the curve). Note, the horopter contains the fixed points of 3-space under the Euclidean motion (such as $\mathbf{V}$), but other points on the horopter are not fixed points (such as $\mathbf{O}$).

[2] A twisted cubic is an algebraic space curve of the third order, which meets a general plane in three points [81]. Algebraically, let $\mathbf{t} = (1, t, t^2, t^3)^\mathsf{T}$, then a twisted cubic is a curve c in $\mathcal{P}^3$ with a parameterisation $\mathbf{t} \rightarrow \mathsf{A}\mathbf{t}$ where A is a 4×4 matrix.

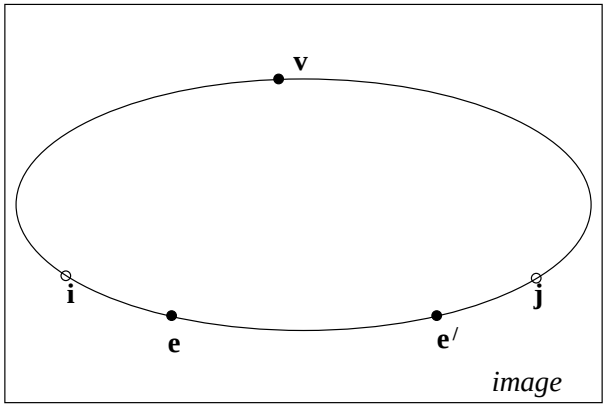


Figure 5.3: *The fixed entities of a pair of images for a camera undergoing general motion. The conic has the equation* $\mathbf{x}^\top \mathsf{F}\mathbf{x} = 0$*, where* F *is the fundamental matrix. It is the image of the horopter curve in 3-space.* $\mathbf{v}$ *is the apex,* $\mathbf{e}$ *and* $\mathbf{e}'$ *are the two epipoles, and* $\mathbf{i}, \mathbf{j}$ *are the images of the circular points.*

With fixed internal parameters, the points lying on the horopter project to the same image coordinates (x). These points also satisfy the epipolar constraint (see equation (A.1)) which gives

$$\mathrm{x}^\top \mathrm{F} \mathrm{x} = 0. \tag{5.1}$$

where F is the fundamental matrix between the two views. The fundamental matrix can be split into two parts, the symmetric part ($\mathrm{F}_s = (\mathrm{F} + \mathrm{F}^\top)/2$), and the asymmetric part ($\mathrm{F}_a = (\mathrm{F} - \mathrm{F}^\top)/2$), such that

$$\mathrm{F} = \mathrm{F}_s + \mathrm{F}_a.$$

The asymmetric part has no affect on equation (5.1) as

$$\mathrm{x}^\top \mathrm{F}_a \mathrm{x} = 0 \qquad \forall \mathrm{x},$$

hence equation (5.1) is equivalent to

$$\mathrm{x}^\top \mathrm{F}_s \mathrm{x} = 0. \tag{5.2}$$

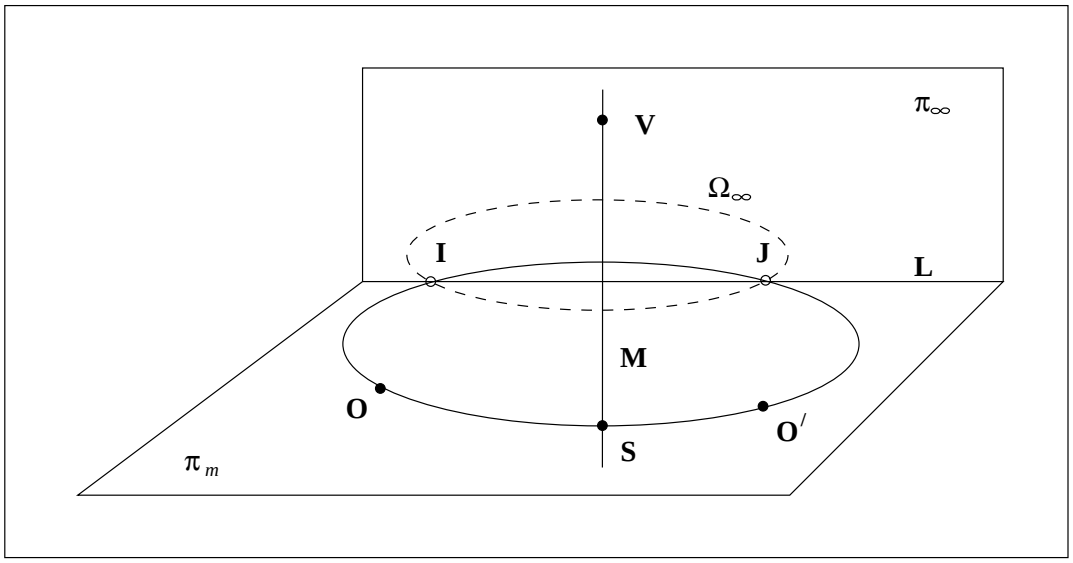


Figure 5.4: *Horopter curve for planar motion. The curve consists of the screw axis (***M***), which is a line of fixed points under the motion, and a conic in the plane of the motion (*π_m*). The conic contains the circular points (***I** *and* **J***), and the optical centres of the camera (***O** *and* **O**$'$*).*

Equation (5.2) defines a conic (the same in both images) which is the image of the horopter, and it contains the image of the five distinguished points on the horopter: the epipoles $\mathbf{e}$, $\mathbf{e}'$ (the images of the optical centres); the points $\mathbf{i}, \mathbf{j}$, the images of the circular points $\mathbf{I}, \mathbf{J}$; and, the apex $\mathbf{v}$ which is the image of $\mathbf{V}$. These points are shown in figure 5.3. Although the imaged circular points and apex lie on the conic, there is, as yet, no algorithm for recovering these points given equation (5.2).

Planar motion The screw axis is a fixed line and must be part of the horopter. Hence, for planar motion the horopter is degenerate and consist of an intersecting line and conic [63]. The conic lies in π_m(the plane of the motion) and contains the optical centres, the circular points, and the intersection of the screw axis with the plane of motion. These five points define the conic. Note that under planar motion the line part of the horopter is a line of fixed points. The conic, on the other hand is not fixed by the motion at all, either point-wise or

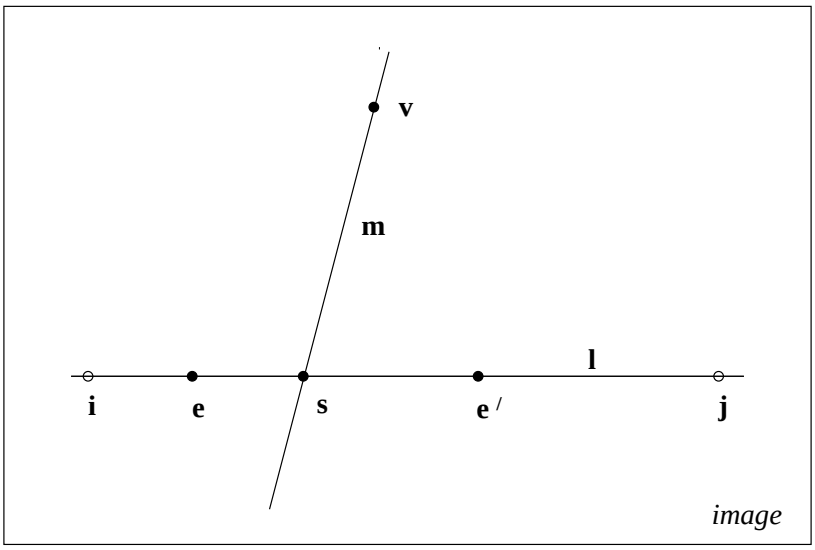


Figure 5.5: *The imaged horopter curve for planar motion. The imaged screw axis (***m***) is a line of fixed points in the image under the motion. The conic is imaged to a fixed line (***l***) under the motion.*

set-wise. Only its image in the two cameras is fixed. On the conic, only the circular points, and the intersection with the screw axis are fixed points of 3-space — the other points move under the planar motion (see figure 5.4).

The imaged horopter is still given by the conic equation (5.2). However, now the horopter is degenerate, consisting of a line and a conic, so the imaged horopter is also degenerate and consists of two distinct lines. The optical centres lie in the same plane as the conic part of the horopter (actually the optical centres are part of the conic), and so the image of the horopter conic is just a line. So the imaged horopter consists of the two distinct lines (see figure 5.5):

1. The image of the screw axis (m). This is a line of fixed points in the images (since it is the image of a line of fixed points in 3-space), and it contains $\mathbf{v}$.

2. The horizon (l), which is a fixed line (not a line of fixed points). This is the intersection of π_m (containing the horopter conic) and the image planes. The epipoles and imaged

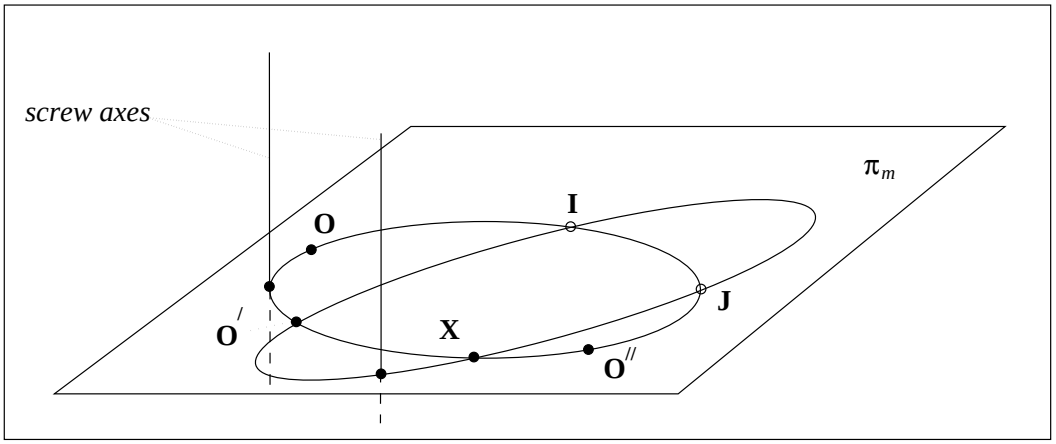


Figure 5.6: *Horopter curves for planar motion over three views. Only the horopter curves between views 1 and 2, and 2 and 3 are shown (i.e., not that between views 1 and 3). The screw axes are not coincident in general, but are parallel, and intersect on π_∞. The conics intersect in four points. Two of these points are the circular points, the other two are the optical centre in common (here $\mathbf{O}'$) and a point* **X**.

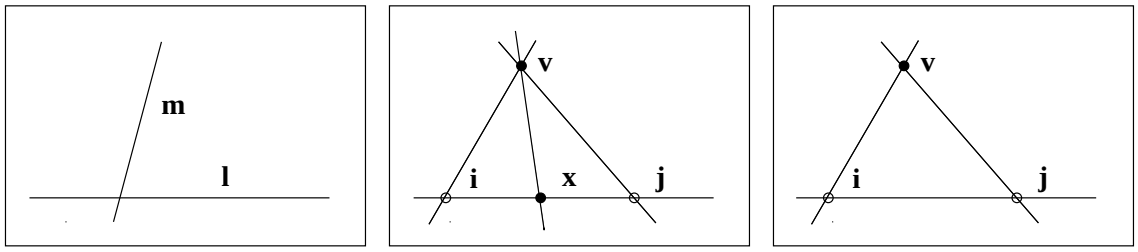
circular points lie on this line.

The intersection of these two lines ($\mathbf{s}$) is the image of the intersection of the screw axis and the plane π_m, and this is the kernel of $\mathtt{F}_s$ (i.e., $\mathtt{F}_s\mathbf{s} = 0$). Equation (5.2) now defines a degenerate conic of two distinct lines, hence $\mathbf{F}_s$ is rank 2 [63, 81].

To summarise, from equation (5.2) with two views under planar motion, the image of the fixed points ($\mathbf{V}$, $\mathbf{I}$, and $\mathbf{J}$) are restricted to two lines. In order to determine these points an additional view is required.

Three Views (Two Displacements)

Planar Motion For three views under planar motion, the horopters are shown in figure 5.6. The second movement generates another conic in the plane π_m, as well as a new screw axis. The two screw axes intersect at $\mathbf{V}$, which is a fixed point of 3-space under both motions. Consider the two conics in the plane; the first conic (views 1 and 2) passes through the circular points ($\mathbf{I}$ and $\mathbf{J}$), and the first and second optical centres ($\mathbf{O}$ and $\mathbf{O}'$). The second



a b c

Figure 5.7: *The fixed image entities for: (a) Two views — two fixed lines; (b) Three views — four fixed points and four fixed lines; (c) Multiple views — three fixed points and three fixed lines*

one (views 2 and 3) passes through $\mathbf{I}$, $\mathbf{J}$, and $\mathbf{O}'$ and $\mathbf{O}''$, the second and third optical centres. This means that the two conics intersect in $\mathbf{I}$, $\mathbf{J}$ and $\mathbf{O}'$. Clearly there is a further intersection point, hereafter called $\mathbf{X}$. Since $\mathbf{X}$ lies on the horopter conic, it appears at the same point in images 1 and 2. Since it lies on the second conic, it appears at the same point in images 2 and 3. In short, it appears at the same point in all three images. This means also that the conic for the views 1 and 3 must also pass through $\mathbf{X}$. It is therefore a point, other than the two circular points, that must be a fixed point in the image over three views. However, it does not correspond to a fixed point of 3-space under the motion.

Multiple Views

When more than three views are considered, the point $\mathbf{X}$ is no longer a fixed point. The position of $\mathbf{X}$ is peculiar to the three views chosen for the triplet, and differs depending on the three views. The horopter conics only intersect at two points, the circular points $\mathbf{I}$ and $\mathbf{J}$, and together with $\mathbf{V}$ are the three fixed points for planar motion.

Summary

For a camera undergoing planar motion, the type and number of fixed entities in the image depend on the number of views being considered. The difference as summarised below and

illustrated in figure 5.7:

- **Two Views** Two fixed lines ($\mathbf{l}$ the horizon, and $\mathbf{m}$ the image of the screw axis). These lines contain the three fixed points, but these cannot be determined from only two views.
- **Three Views** Four fixed points ($\mathbf{v}$ the apex, $\mathbf{i}$ and $\mathbf{j}$, the image of the circular points, and $\mathbf{x}$). Four fixed lines, the four lines joining the fixed points (this becomes important in section 5.4).
- **Multiple Views** Three fixed points ($\mathbf{v}, \mathbf{i}$ and $\mathbf{j}$), and three fixed lines joining the points.

5.2 Calibration from Fixed Points and Structure Recovery

The previous section introduced the idea of fixed entities, and showed that for planar motion there are three fixed points[3] (and the corresponding intersecting fixed lines). These points are fixed both in the image and in 3-space. If they can be identified, and back-projected into the structure, then the ambiguity in the structure can be upgraded from projective to affine or metric.

5.2.1 Affine Structure

To reduce structure ambiguity from projective to affine, π_∞ needs to be identified (see section 2.2.6). The three fixed points ($\mathbf{V}$, $\mathbf{I}$, and $\mathbf{J}$) all lie on π_∞, and are not collinear, and hence uniquely identify π_∞. When the corresponding image points ($\mathbf{v}$, $\mathbf{i}$, and $\mathbf{j}$) are identified, they can be back-projected to identify π_∞, and upgrade the structure to affine ambiguity using equation (2.14).

[3] The fourth fixed points $\mathbf{x}$ provides no useful information at present, and is only fixed for an image triplet, not planar motion in general.

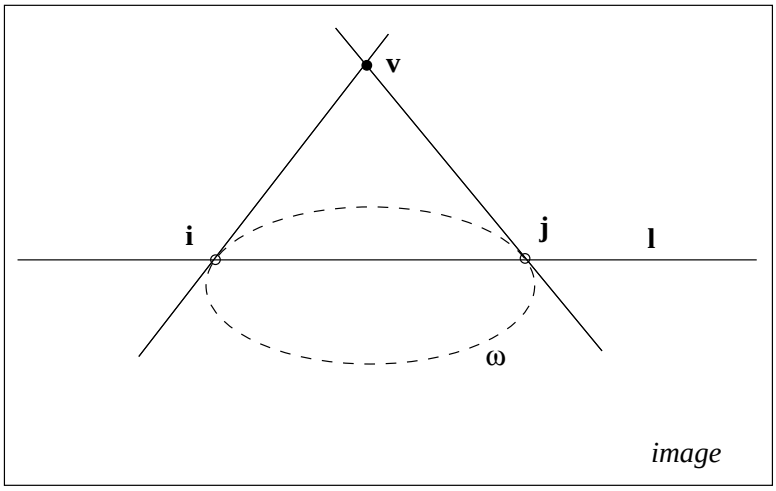


Figure 5.8: *The fixed image points (***v***,* **i***, and* **j***) and corresponding fixed lines. The fixed line* **l** *and point* **v** *are pole and polar with respect to* ω *(image of* Ω_∞*), and hence give some constraints on* ω.

5.2.2 Metric Structure

To reduce the structure ambiguity to metric, Ω_∞ needs to be identified. Section 3.1 introduced the idea that it is sometimes easier to find ω, the image of Ω_∞, rather than Ω_∞ itself. Section 5.1.4 stated that the fixed line $\mathbf{L}$ and fixed point $\mathbf{V}$ are pole and polar with respect to Ω_∞ [81]. Comparing the figures 5.2 and 5.7c, the same relationship holds between the fixed image line l, the fixed image point v, and ω (see figure 5.8). Hence, the positions of l and v give some constraints on ω. The four constraints are that ω has to pass through the circular points i and j, and at these points, the conic must be tangent to the lines from v, $\mathrm{v} \times \mathrm{i}$ and $\mathrm{v} \times \mathrm{j}$ respectively (as shown in figure 5.8).

Planar Metric

A conic has 5 degrees of freedom, so, using the four constraints given above, defines a pencil of conics (one-parameter family) which contains ω. This is illustrated in figure 5.9. Without any other information this allows the structure to be upgraded to planar metric.

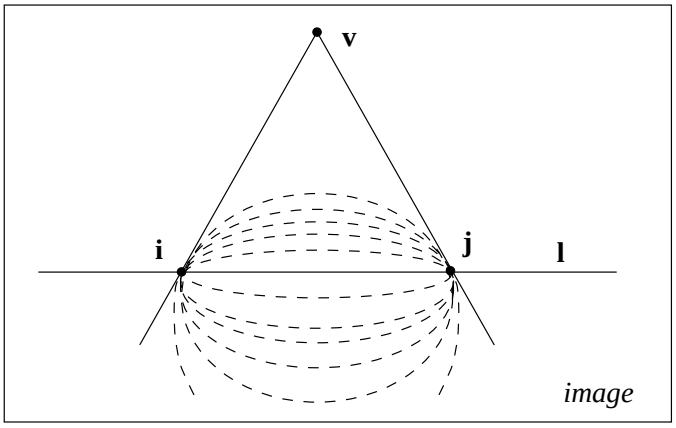


Figure 5.9: *The pencil of conics defined by the fixed image points, which contains* ω.

Planar metric structure is when the planes perpendicular to the rotation axis are defined up to metric ambiguity (i.e., angles in these planes are correct), while there is an unknown scaling in the direction of the rotation axis (i.e., an affine skew). This is the ambiguity introduced by the pencil of conics.

To obtain planar metric structure, a transformation `H` is applied to the affine structure to give the planar metric structure. The transformation `H` is computed using the following constraints:

- Keep π_∞ fixed at $(0,0,0,1)^\top$.
- Keep the fixed point $\mathbf{V}$ fixed at $(0,1,0,0)^\top$.
- Map the circular points $\mathbf{I}$ and $\mathbf{J}$ to $(\pm i,0,1,0)^\top$.

Full Metric

However, it is common to assume that some information is known about the camera calibration. This can be that the image axes are perpendicular [61], that the aspect ratio is known, that the principal point is known (or assumed to be the centre of the image), or a

combination of all three [44]. Each of these gives the extra constraint required to identify ω in the pencil of conics.

First, the pencil of conics obtained from the fixed point constraints, is described algebraically, and then it is explained how the extra constraints are used to identify ω.

Pencil of Conics Semple and Kneebone [81] give the equation for the pencil of conics, defined by two points/tangents, as

$$\mathtt{S} = \mathtt{S}' + \lambda \mathtt{l}\mathtt{l}^\top, \tag{5.3}$$

where $\mathtt{S}'$ is any conic satisfying the constraints, $\mathtt{l}$ is the line connecting the two points on the conic (here $\mathtt{l} = \mathtt{i} \times \mathtt{j}$), and λ is the free parameter of the pencil. The conic $\mathtt{l}\mathtt{l}^\top$ is the degenerate conic of two repeated lines through $\mathtt{i}$ and $\mathtt{j}$. For $\mathtt{S}'$, the degenerate conic formed by the 2 separate lines, $\mathtt{l}_i = \mathtt{v} \times \mathtt{i}$ and $\mathtt{l}_j = \mathtt{v} \times \mathtt{j}$, is used, and this is given by

$$\mathtt{S}' = \mathtt{l}_i \mathtt{l}_j^\top + \mathtt{l}_j \mathtt{l}_i^\top. \tag{5.4}$$

Constraints on Calibration The image of the absolute conic is given by the dual of $\mathtt{K}$ ($\mathtt{K} = \mathtt{C}\mathtt{C}^\top$). Using the definition of $\mathtt{C}$ given in equation (2.4) gives

$$\mathtt{K} = \begin{bmatrix} \alpha_u^2 + u_0^2 + k^2 & u_0 v_0 + k\alpha_v & u_0 \\ u_0 v_0 + k\alpha_v & \alpha_v^2 + v_0^2 & v_0 \\ u_0 & v_0 & 1 \end{bmatrix}, \tag{5.5}$$

$$\mathtt{K}^{-1} = \begin{bmatrix} \frac{1}{{\alpha_u}^2} & -\frac{k}{{\alpha_u}^2 \alpha_v} & \frac{-({\alpha_v}^2 u_0) + \alpha_v k v_0}{{\alpha_u}^2 {\alpha_v}^2} \\ -\frac{k}{{\alpha_u}^2 \alpha_v} & \frac{{\alpha_u}^2 + k^2}{{\alpha_u}^2 {\alpha_v}^2} & \frac{\alpha_v k u_0 - {\alpha_u}^2 v_0 - k^2 v_0}{{\alpha_u}^2 {\alpha_v}^2} \\ \frac{-({\alpha_v}^2 u_0) + \alpha_v k v_0}{{\alpha_u}^2 {\alpha_v}^2} & \frac{\alpha_v k u_0 - {\alpha_u}^2 v_0 - k^2 v_0}{{\alpha_u}^2 {\alpha_v}^2} & \frac{{\alpha_u}^2 {\alpha_v}^2 + {\alpha_v}^2 {u_0}^2 - 2\alpha_v k u_0 v_0 + {\alpha_u}^2 {v_0}^2 + k^2 {v_0}^2}{{\alpha_u}^2 {\alpha_v}^2} \end{bmatrix}. \tag{5.6}$$

$\mathtt{K}^{-1}$ is in the pencil of conics defined by $\mathtt{S}$, and each of the different constraints on the camera calibration gives a solution for λ, hence $\mathtt{K}^{-1}$ (ω).

- **Image skew** The most common assumption about camera calibration is that there is no image skew, so that $k = 0$. Substituting this into equation (5.6) gives

$$\mathtt{K}^{-1} = \begin{bmatrix} \frac{1}{{\alpha_u}^2} & 0 & -\frac{u_0}{{\alpha_u}^2} \\ 0 & \frac{1}{{\alpha_v}^2} & -\frac{v_0}{{\alpha_v}^2} \\ -\frac{u_0}{{\alpha_u}^2} & -\frac{v_0}{{\alpha_v}^2} & \frac{{\alpha_u}^2{\alpha_v}^2+{\alpha_v}^2{u_0}^2+{\alpha_u}^2{v_0}^2}{{\alpha_u}^2{\alpha_v}^2} \end{bmatrix}.$$

  It follows that the constraint of no image skew gives $S_{12} = 0$, which in turn gives a solution for λ,

$$\lambda = -S'_{12}/(l_1 l_2). \tag{5.7}$$

  Previous methods [61] have applied the constraint of no image skew to the dual of ω (i.e., $\mathtt{K}$), which from equation (5.5) gives the quadratic constraint of $K_{12}K_{33} = K_{13}K_{23}$, with two possible solutions. Here the constraint is applied directly to ω giving the linear constraint of $K_{12}^{-1} = 0$, which has only one solution. These are related, as can be seen by examining the cofactors of $\mathtt{K}^{-1}$ which for element K_{12}^{-1} is $K_{12}K_{33} - K_{13}K_{23}$. Their relationship is examined further in section 5.2.3.

However, as will also be explained in section 5.2.3, in some circumstances common to planar motion the constraint of no image skew is unstable and can give poor results. Rather than assume that the image skew is zero, the assumption is that the image skew is small ($k \ll \alpha_u$), then other constraints can be utilised which give better results for planar motion.

- **Aspect Ratio** If the aspect ratio ($\zeta = \alpha_v/\alpha_u$) is known, then from equation (5.6)

$$S_{11} = \zeta^2 S_{22},$$

  which gives

$$\lambda = -(\mathtt{S}'_{11} - \zeta^2 \mathtt{S}'_{22})/(l_1^2 - \zeta^2 l_2^2). \tag{5.8}$$

- **Principal Point** If the principal point (u_0, v_0) is known (or assumed to be in the centre of the image) then

$$1 - u_0 S_{13} - v_0 S_{23} = S_{33},$$

which gives

$$\lambda = (1 - u_0 S'_{13} - v_0 S'_{23} - S'_{33})/(l_3(u_0 l_1 + v_0 l_2 + l_3)). \tag{5.9}$$

Whichever constraint is used, an estimate is found for K which can be decomposed into C via Choleski decomposition. Metric structure can then be recovered from affine using equation (2.16).

5.2.3 The Image Skew Constraint

The assumption that there is no image skew is used often in self-calibration, but the actual affect of the constraint on ω and its dual has not been examined in detail. Examining equation (5.6) shows that forcing $k = 0$ constrains ω to be of the form

$$\mathtt{K}^{-1} = \begin{bmatrix} \times & 0 & \times \\ 0 & \times & \times \\ \times & \times & \times \end{bmatrix}, \tag{5.10}$$

where $\times$ signifies a possible non-zero entry. A conic of this form has its axes aligned with the coordinate frame[4] which here are the complex image axes. Using this constraint to identify ω in a pencil of conics gives a single solution (see equation (5.7)). Alternatively, the constraint can be applied to the dual of ω which gives the quadratic equation

$$K_{12}K_{33} = K_{13}K_{23}$$

which has two solutions. Expanding this equation with the internal parameters gives

$$u_0 v_0 + k\alpha_v = u_0 v_0$$

which shows that the two solutions, if distinct, correspond to $k = 0$ or $\alpha_v = 0$. The second solution gives a singular matrix for K, in which case the conic ω has a degenerate form.

[4]Consider a conic consisting of real points and having the form of equation (5.10), then its major and minor axes are aligned with the real coordinate axes.

Careful examination of equation (5.6) and setting $\alpha_v = 0$ gives the following degenerate form for ω

$$\mathtt{K}^{-1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix}. \tag{5.11}$$

Applying the constraint to ω rather than its dual avoids the second incorrect solution.

The Unstable Image Skew Constraint

When the horizon line is nearly horizontal, the zero image skew constraint can give unstable results. This situation is common for planar motion, because often the orientation of the camera with respect to the plane of motion is just elevation with no roll, which results in a horizontal horizon line.

The pencil of conics defined by the three fixed points is shown in figure 5.9, but the orientation of the pencil depends on the position of the fixed points. Figure 5.10 shows two possible pencils of conics, defined by the fixed points, which are at different orientations to the image axes. Figure 5.10a shows the situation with an almost horizontal horizon line, while figure 5.10b the horizon line is inclined substantially. A difference between the two pencils is the range of conics therein, which are nearly aligned with the image axes. The pencil with the almost horizontal horizon line has the much larger range of conics. In the limit, with the horizon line horizontal, all the conics in the pencil will be aligned with the image axes and the constraint will not give a solution. When the horizon line is nearly horizontal, a small change in the image skew will results in significant differences in the conic chosen.

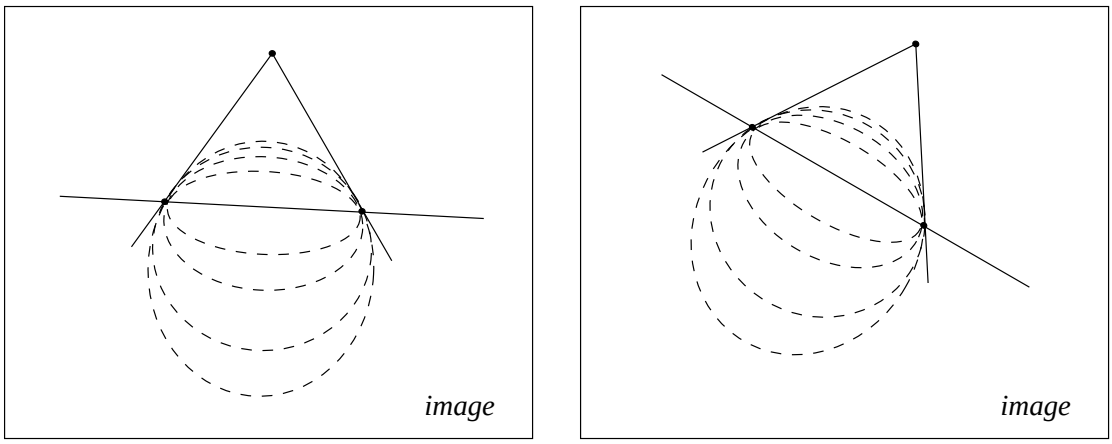


Figure 5.10: *The pencil of conics defined by two sets of fixed points. (a) with a nearly horizontal horizon line, and (b) with the horizon line at an angle to the image.*

The circular points are complex conjugates and can be represented as

$$\begin{aligned} \mathbf{i} &= (a+bi, c+di, 1)^\top, \\ \mathbf{j} &= (a-bi, c-di, 1)^\top, \end{aligned}$$

which gives $\mathbf{l}$ and $\mathbf{l}\mathbf{l}^\top$ (see equation (5.3)) as

$$\mathbf{l} = (2di, -2bi, 2(bc-ad)i)^\top,$$

$$\mathbf{l}\mathbf{l}^\top = \begin{bmatrix} 4d^2 & 4bd & 4d(bc-ad) \\ 4bd & 4b^2 & 4b(bc-ad) \\ 4d(bc-ad) & 4b(bc-ad) & 4(bc-ad)^2 \end{bmatrix}.$$

The matrix $\mathbf{l}\mathbf{l}^\top$ defines the variation in the pencil of conics, and the image skew assumption is concerned with the element $[\mathbf{l}\mathbf{l}^\top]_{12}$ and its relationship to the other elements.

It is shown in section 5.4 that the positions of the circular points can be expressed in terms of the horizon line ($\mathbf{l}$), the vanishing point ($\mathbf{v}$), and a complex number (z), such that

$$\begin{aligned} \mathbf{i} &= (-(l_1 l_3) + l_2(l_2 v_1 - l_1 v_2) + l_2 z, -(l_2 l_3) - l_1(l_2 v_1 - l_1 v_2) - l_1 z, l_1^2 + l_2^2)^\top, \\ \mathbf{j} &= (-(l_1 l_3) + l_2(l_2 v_1 - l_1 v_2) + l_2 \bar{z}, -(l_2 l_3) - l_1(l_2 v_1 - l_1 v_2) - l_1 \bar{z}, l_1^2 + l_2^2)^\top. \end{aligned}$$

However, only the imaginary parts are significant for $\mathtt{l}\mathtt{l}^\top$, and these are

$$\begin{aligned} b &= l_2 \Im[z], \\ d &= l_1 \Im[z]. \end{aligned}$$

For a horizontal horizon line, $l_2 \gg l_1$, which gives a difference in order of magnitude between the elements of $\mathtt{l}\mathtt{l}^\top$,

$$[\mathtt{l}\mathtt{l}^\top]_{11} \ll [\mathtt{l}\mathtt{l}^\top]_{12} \ll [\mathtt{l}\mathtt{l}^\top]_{22}.$$

As $[\mathtt{l}\mathtt{l}^\top]_{12}$ is used for the image skew constraint, any small change in $[\mathtt{l}\mathtt{l}^\top]_{12}$ (i.e., the image skew is small but not zero) will give a significant change in $[\mathtt{l}\mathtt{l}^\top]_{22}$ but a negligible change in $[\mathtt{l}\mathtt{l}^\top]_{11}$. Following this through shows that the estimate for α_v is highly sensitive to the image skew when the horizon line is nearly horizontal.

All of the above analysis has ignored the effect of $\mathtt{S}'$, the degenerate conic of two distinct lines, on the pencil and the image skew constraint. However, it can be shown that when the horizon line is nearly horizontal the ratio of the elements of $\mathtt{S}'$ and $[\mathtt{l}\mathtt{l}^\top]$ further exacerbates the instability, such that

$$\begin{aligned} \frac{S'_{11}}{[\mathtt{l}\mathtt{l}^\top]_{11}} &\gg 1, \\ \frac{S'_{12}}{[\mathtt{l}\mathtt{l}^\top]_{12}} &\approx 1, \\ \frac{S'_{22}}{[\mathtt{l}\mathtt{l}^\top]_{22}} &\ll 1. \end{aligned}$$

The result of these ratios is that changing λ in equation (5.3) affects the values of $\mathtt{S}_{12}$ and $\mathtt{S}_{22}$, but has negligible effect on $\mathtt{S}_{11}$.

The normal assumption about the image skew is that it is small enough to be approximated by zero, but as shown here a small difference in the image skew can give significant differences in the calibration obtained. Other methods of self-calibration avoid this problem

either by having the rotation between views around different axes (which removes the need for a constraint altogether), or just choosing the rotation axis so that the pencil of conics is not aligned with the image axes. However, for planar motion the horizontal horizon line is a common scenario and so the zero image skew constraint has to be used with care.

5.3 Finding the Horizon Line and Vanishing Point

For reasons that will be explained in section 5.4, the fixed image points for planar motion are not found simultaneously, rather sequentially, with the apex $\mathbf{v}$ and fixed line $\mathbf{l}$ being found first.

As stated in section 5.1.5, for planar motion the symmetric part of the fundamental matrix $\mathtt{F}_s$ is a degenerate conic of 2 lines, which correspond to the fixed horizon line ($\mathbf{l}$) and the image of the screw axis ($\mathbf{m}$).

Decomposing F_s The matrix $\mathtt{F}_s$ is a degenerate conic of two non-coincident lines. If the two lines are $\mathbf{f}_1$ and $\mathbf{f}_2$, then $\mathtt{F}_s$ can be expressed in the form [81, 95]

$$\mathtt{F}_s = \mathbf{f}_1\mathbf{f}_2^\top + \mathbf{f}_2\mathbf{f}_1^\top. \tag{5.12}$$

$\mathtt{F}_s$ is also Rank 2, and so can be expressed in terms of eigenvalues (d_i) and eigenvectors ($\mathbf{v}_i$),

$$\mathtt{F}_s = \mathtt{V}\mathtt{D}\mathtt{V}^\top = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 \end{bmatrix} \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^\top \\ \mathbf{v}_2^\top \\ \mathbf{v}_3^\top \end{bmatrix}. \tag{5.13}$$

Comparing equations (5.12) and (5.13) gives an expression for the lines $\mathbf{f}_1$ and $\mathbf{f}_2$. The eigenvalues are such that $d_1 d_2 < 0$, so that there are two alternative expressions.

If $d_1 > 0$,

$$\begin{aligned} \mathbf{f}_1 &= \sqrt{d_1}\mathbf{v}_1 + \sqrt{-d_2}\mathbf{v}_2, \\ \mathbf{f}_2 &= \sqrt{d_1}\mathbf{v}_1 - \sqrt{-d_2}\mathbf{v}_2, \end{aligned} \tag{5.14}$$

or if $d_1 < 0$,

$$\begin{aligned} \mathbf{f}_1 &= \sqrt{d_2}\mathbf{v}_2 + \sqrt{-d_1}\mathbf{v}_1, \\ \mathbf{f}_2 &= \sqrt{d_2}\mathbf{v}_2 - \sqrt{-d_1}\mathbf{v}_1. \end{aligned} \tag{5.15}$$

Which lines correspond to $\mathbf{l}$ and $\mathbf{m}$ is determined by the fact that the epipoles lie on $\mathbf{l}$ (see section 5.1.5). When using a non-linear minimisation to compute $\mathtt{F}$ the parameterisation for planar motion given in equation (A.5) uses $\mathbf{f}_1$ and $\mathbf{f}_2$ as some of the parameters, which avoids the need to decompose $\mathtt{F}_s$.

Finding the Fixed Line l

As is shown above, the fixed line $\mathbf{l}$ can be estimated from the symmetric part of the fundamental matrix $\mathtt{F}_s$. However, for real images with noise, the estimate for $\mathbf{l}$ from different $\mathtt{F}_s$'s will not necessarily be the same. Some method is required to find a best estimate from several different estimates.

Finding the Fixed Point v

Each pair of images (via $\mathtt{F}_s$) gives a line in the image which is an image of the associated screw axis. All these lines intersect at the point $\mathbf{v}$, hence it is termed the apex or vanishing point (see section 5.1.4). For real images they do not intersect, and a least-squares method is used (see section 5.5) to give a best estimate. If the camera moves with *circular motion*, so that it fixates on a point in the scene, the screw axes will be the same line and the images of the axis will be coincident and the intersection point will not be defined.

5.4 Finding the Circular Points

The circular points are found using the trifocal tensor, which allows points which are fixed over three images to be found. Generally, the method is algebraically complex and involves solving three simultaneous fourth-order polynomials of three variables. However, by stratifying the problem and using the fundamental matrices to find the position of one of the fixed points and the fixed line containing the three remaining fixed points, the algebra can be reduced to solving a single cubic in one variable.

5.4.1 The Trifocal Tensor

The trifocal tensor describes the fundamental geometric relationship between three images, and can be calculated without knowledge of the camera calibration or relative positions. It is the culmination of developments by a number of authors [26, 41, 83, 85, 92, 93, 96, 99, 106]. The tensor maps both points and lines, and the points/lines are mapped from two images to the corresponding point/line in the third. This is different from the fundamental matrix, which describes the epipolar geometry between two views, and maps points in one image to the corresponding epipolar lines in the second.

The trifocal tensor is a Rank 3 tensor containing 27 elements and is given by

$$T_i^{jk} \qquad i, j, k = 1, \ldots, 3. \tag{5.16}$$

The tensor has 18 degrees of freedom [92, 106], and the 27 homogeneous parameters satisfy 8 algebraic constraints. This is analogous to the fundamental matrix having 9 homogeneous parameters satisfying one algebraic constraint. The underlying geometry and the mathematics of the transfer of points and lines is described by Zisserman [106], and involves both transfer and incidence relationships. The points/lines are transferred from the first image to the third (second) image using a homography, and the homography is

computed using the trifocal tensor and a line in the second (third) image. When transferring points, the line in the second (third) image has to pass through the corresponding point. The tensor also encodes the incidence relationship of points and/or lines over the three images.

However, the work here uses the trilinearity equations [41, 83] where the corresponding points in three views are related by

$$x''_l = x'_i x_k T_k^{jl} - x'_j x_k T_k^{il}, \tag{5.17}$$

and corresponding lines are related by

$$l_i = l'_j l''_k T_i^{jk}, \tag{5.18}$$

where $\mathbf{x}$, $\mathbf{x}'$, and $\mathbf{x}''$ (l, l', and l'') are the points (lines) in the first, second, and third views respectively.

The tensor is also related to the three camera matrices $\mathtt{P}$, $\mathtt{P}'$, and $\mathtt{P}''$ which map the point $\mathbf{X}$ in 3-space to $\mathbf{x}$, $\mathbf{x}'$, and $\mathbf{x}''$ respectively. $\mathtt{P}$ is set to the canonical form $[\mathtt{I}|\mathbf{0}]$, and $\mathbf{p}'_i$ and $\mathbf{p}''_i$ are the columns of $\mathtt{P}'$ and $\mathtt{P}''$, such that

$$T_i = \mathbf{p}''_i \mathbf{p}'^{\top}_4 - \mathbf{p}''_4 \mathbf{p}'^{\top}_i \tag{5.19}$$

The tensor is useful for finding fixed entities because equations (5.17) and (5.18) can be used to find the positions for fixed image points and lines, such that substituting $\mathbf{x} = \mathbf{x}' = \mathbf{x}''$ into equation (5.17) gives

$$x_l = x_i x_k T_k^{jl} - x_j x_k T_k^{il}. \tag{5.20}$$

Substituting $\mathrm{l} = \mathrm{l}' = \mathrm{l}''$ into equation (5.18) gives

$$l_i = l_j l_k T_i^{jk}. \tag{5.21}$$

5.4.2 Fixed Points and Lines from the Trifocal Tensor

Section 5.1.5 showed that for planar motion there are four fixed points for any triplet of images. It also explained that the lines joining these points are also fixed over the three views. Section 5.2 showed that knowing the position of the fixed points allows affine or metric structure to be recovered.

However, finding the positions of the fixed lines is equivalent to finding the position of the fixed points. So rather than use equation (5.20) to find the position of the fixed points, it is easier to proceed with equation (5.21) to find the position of the fixed lines. Equation (5.21) is only equivalent up to a scale factor, so by taking the cross product of each side and equating to a zero vector, results in three simultaneous homogeneous equations for the components of $\mathbf{l}$.

Normally, solving simultaneous polynomial equations is a difficult numerically problem. However, it is shown below that for planar motion it is possible to reduce the solution to a single cubic equation in one variable. This is achieved by a plane projective transformation of the images.

Special Planar Motion

Initially, the case of a calibrated camera is considered, where the rotation is about the camera Y (vertical) axis, and the translation is confined to the XZ (horizontal) plane. The camera calibration can be set to the identity ($\mathtt{C} = \mathtt{I}$), and it can be assumed that the camera is pointing horizontally. From equation (2.2), the three camera matrices are

$$\begin{aligned} \mathtt{P}_{spm} &= [\mathtt{I} \mid \mathbf{0}], \\ \mathtt{P}'_{spm} &= [\mathtt{R}' \mid \mathbf{t}'], \\ \mathtt{P}''_{spm} &= [\mathtt{R}'' \mid \mathbf{t}''], \end{aligned}$$

where $\mathtt{R}'$ and $\mathtt{R}''$ are rotations about the Y axis, and $\mathtt{t}'$ and $\mathtt{t}''$ are translations in the XZ plane. It follows that both $\mathtt{P}'_{spm}$ and $\mathtt{P}''_{spm}$ are of the form

$$\begin{bmatrix} \times & 0 & \times & \times \\ 0 & \times & 0 & 0 \\ \times & 0 & \times & \times \end{bmatrix}, \tag{5.22}$$

where 0 represents a zero entry and $\times$ represents a non-zero entry. This type of motion is called *special planar motion*.

Now the format of the trifocal tensor is computed from equation (5.19). Using the format of the projection matrices in equation (5.22), it is easy to show that

$$\begin{aligned} T_i^{\cdot\cdot} &= \begin{bmatrix} \times & 0 & \times \\ 0 & 0 & 0 \\ \times & 0 & \times \end{bmatrix} \quad \text{for } i = 1, 3 \\ T_2^{\cdot\cdot} &= \begin{bmatrix} 0 & \times & 0 \\ \times & 0 & \times \\ 0 & \times & 0 \end{bmatrix}, \end{aligned} \tag{5.23}$$

so that the reduced trifocal tensor has only 12 non-zero elements.

Using this reduced form of the tensor, equation (5.21) can be written in the form

$$\begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = \begin{pmatrix} a_1 l_1^2 + b_1 l_1 l_3 + c_1 l_3^2 \\ d_2 l_1 l_2 + e_2 l_2 l_3 \\ a_3 l_1^2 + b_3 l_1 l_3 + c_3 l_3^2 \end{pmatrix} \tag{5.24}$$

which has eight parameters $(a_1, \ldots, c_3)$, a linear combination of the non-zeros elements of the reduced tensor,

$$\begin{aligned} a_1 &= T_1^{11}, \\ b_1 &= T_1^{13} + T_1^{31}, \\ c_1 &= T_1^{33}, \\ d_2 &= T_2^{12} + T_2^{21}, \\ e_2 &= T_2^{23} + T_2^{32}, \\ a_3 &= T_3^{11}, \end{aligned}$$

$$\begin{aligned} b_3 &= T_3^{13} + T_3^{31}, \\ c_3 &= T_3^{33}. \end{aligned}$$

The fixed lines can then be found by solving this set of equations.

One of the fixed points in the image is $\mathbf{v}$, the image of the intersection of the rotation axis and π_∞. Here the rotation axis is $(0,1,0,1)^\top$, so $\mathbf{V} = (0,1,0,0)^\top$, and the image point is $\mathbf{v} = (0,1,0)^\top$. Also, the horizon line (see figure 5.7b) is the image of the intersection of the plane of motion and π_∞. Here the plane of motion is the XZ plane, and so the fixed line on π_∞ is $(0,1,0,0)^\top$, and image of this (the horizon line) is $(0,1,0)^\top$. So only the position of the three fixed lines through $\mathbf{v}$ (see figure 5.7b) are unknown. All lines passing through $\mathbf{v}$ have the form $(l_1, 0, l_3)$, and so it can be assumed that $l_2 = 0$, and equation (5.24) reduces to the form

$$\begin{pmatrix} l_1 \\ l_3 \end{pmatrix} = \begin{pmatrix} a_1 l_1^2 + b_1 l_1 l_3 + c_1 l_3^2 \\ a_3 l_1^2 + b_3 l_1 l_3 + c_3 l_3^2 \end{pmatrix}. \tag{5.25}$$

Cross-multiplying to remove the scale factor, reduces it to a single equation

$$l_3(a_1 l_1^2 + b_1 l_1 l_3 + c_1 l_3^2) = l_1(a_3 l_1^2 + b_3 l_1 l_3 + c_3 l_3^2). \tag{5.26}$$

This is a homogeneous cubic, and may be solved easily for the ratio $l_1 : l_3$.

The solutions to this cubic are the three lines passing through the apex $\mathbf{v}$, which intersect the horizon line at 3 distinct points. These three points are the images of the two circular points ($\mathbf{i}$ and $\mathbf{j}$), and the third fixed point ($\mathbf{x}$). In the particular calibrated case considered here, the circular points have coordinates $(\pm i, 0, 1)^\top$, and hence the lines in question joining the circular points to the apex are $(1, 0, \pm i)$. This gives two complex solutions to the equation (5.26). The image of the third fixed point is a real point corresponding to the third solution of (5.26).

General Planar Motion

Now, *general planar motion* is considered, where the calibration and the orientation are not known (i.e., it cannot be assumed that the camera is pointing horizontally). All that can be assumed is that the rotation is about the Y axis in the world coordinate frame, the translation is confined to the XZ plane, and that the first camera is centred at the origin of the world coordinate frame. The projection matrices are now of the form

$$\begin{aligned} \mathtt{P}_{gpm} &= \mathtt{H}[\mathtt{I} \mid \mathbf{0}], \\ \mathtt{P}'_{gpm} &= \mathtt{H}[\mathtt{R}' \mid \mathbf{t}'], \\ \mathtt{P}''_{gpm} &= \mathtt{H}[\mathtt{R}'' \mid \mathbf{t}''], \end{aligned} \tag{5.27}$$

where $\mathtt{H}$ contains the unknown (constant) camera calibration, and orientation with respect to the world coordinate frame.

When comparing the *special planar motion* and the *general planar motion*, it can be shown that the fixed lines under each type if motion are in one-to-one correspondence. Consider a line in space that maps to the same line $\mathbf{l}$ in all three images under the action of the camera projections matrices $\mathtt{P}_{gpm}$, $\mathtt{P}'_{gpm}$ and $\mathtt{P}''_{gpm}$ given in equations (5.28). The transformation $\mathtt{H}$ represents the same projective transformation for each of the three images. If a further projective transformation, represented by $\mathtt{H}^{-1}$, is applied to each of the images, the net effect is that of projecting via camera transformations

$$\begin{aligned} \mathtt{P} &= \mathtt{H}^{-1}\mathtt{P}_{gpm} = [\mathtt{I} \mid \mathbf{0}], \\ \mathtt{P}' &= \mathtt{H}^{-1}\mathtt{P}'_{gpm} = [\mathtt{R}' \mid \mathbf{t}'], \\ \mathtt{P}'' &= \mathtt{H}^{-1}\mathtt{P}''_{gpm} = [\mathtt{R}'' \mid \mathbf{t}']. \end{aligned}$$

However, the projective transformation $\mathtt{H}^{-1}$ maps the line $\mathbf{l}$ to $\mathtt{H}^{\top}\mathbf{l}$. This shows that $\mathbf{l}$ is a fixed line for three views under a general planar motion, if and only if $\mathbf{l} = \mathtt{H}^{\top}\mathbf{l}$ is a fixed

line for three views under special planar motion. So the fixed points/lines under general planar motion are in one-to-one correspondence to those under special planar motion.

Unfortunately, the projective transformation is not known *a priori*, and the trifocal tensor does not have the same form as in equation (5.23). It is still possible to solve for the fixed lines using the same method as used for the special planar motion, but rather than achieving the vector equation (5.24), the following is obtained

$$\begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = \begin{pmatrix} h^{(2)}(l_1, l_2, l_3) \\ h'^{(2)}(l_1, l_2, l_3) \\ h''^{(2)}(l_1, l_2, l_3) \end{pmatrix}, \tag{5.28}$$

where $h^{(2)}, h'^{(2)}, h''^{(2)}$ are quadratic polynomials. It also cannot be assumed that $l_2 = 0$, so setting the cross product to zero leads to three simultaneous polynomial equations

$$\begin{aligned} l_2 h''^{(2)}(l_1, l_2, l_3) - l_3 h'^{(2)}(l_1, l_2, l_3) &= 0, \\ l_3 h^{(2)}(l_1, l_2, l_3) - l_1 h''^{(2)}(l_1, l_2, l_3) &= 0, \\ l_1 h'^{(2)}(l_1, l_2, l_3) - l_2 h^{(2)}(l_1, l_2, l_3) &= 0. \end{aligned} \tag{5.29}$$

In principle, these equations can be solved to give the positions of the four fixed lines. However, the three equations in (5.30) are not linearly independent, and there are just two linearly independent cubic equations. So given a trifocal tensor computed from real images with noise, different solutions will be obtained depending on the two cubics equation chosen. Also, the number of solutions will increase, with a worst case scenario of 27 possible solutions. It will be very difficult to choose the four correct solutions, especially if there are several solutions close to the horizon line.

However, as there is a one-to-one correspondence between the fixed lines of general planar motion and those of special planar motion, it is possible to reduce the algebraic complexity of the solution for general planar motion. This is explained in the next section.

Normalised Planar Motion

In special planar motion, the horizon line ($\mathbf{l}$) and vanishing point ($\mathbf{v}$) are at $(0,1,0)^\top$, while for general planar motion their position can be found using the fundamental matrices (see section 5.3). The *canonical transformation* is the image-to-image projectivity which maps the horizon line and vanishing point to their *canonical* positions at $(0,1,0)^\top$. The concatenation of an image-to-image projectivity onto the projection matrices is effectively a new camera projection for the three views and camera motion. This scenario is called the *normalised planar motion*, where the horizon line and vanishing point are in their canonical position.

Canonical Transformation The canonical transformation is required to map both the line $\mathbf{l}$ to $(0,1,0)^\top$, and the point $\mathbf{v}$ to $(0,1,0)^\top$. A possible transformation, close to a Euclidean transformation (see appendix B.3) is

$$\mathtt{H}_c = \begin{bmatrix} l_2 & -l_1 & l_1 v_2 - l_2 v_1 \\ l_1 & l_2 & l_3 \\ \frac{-l_1}{\mathbf{l}.\mathbf{v}} & \frac{-l_2}{\mathbf{l}.\mathbf{v}} & \frac{-l_3}{\mathbf{l}.\mathbf{v}} + 1 \end{bmatrix}. \tag{5.30}$$

The new projection matrices, for the normalised planar motion, still differ from those for the special planar motion by a further plane projectivity, but the projectivity has a special simple form. The vanishing point and horizon line have the coordinates $(0,1,0)^\top$ for both types of planar motion. This means that the plane projective transformation ($\mathtt{H}$) must map the point $(0,1,0)^\top$ to $(0,1,0)^\top$, and the line $(0,1,0)^\top$ to $(0,1,0)^\top$, which gives the equations

$$\begin{aligned} (0,1,0)^\top &= \mathtt{H}(0,1,0)^\top, \\ (0,1,0)^\top &= \mathtt{H}^\top(0,1,0)^\top, \end{aligned}$$

which results in $\mathtt{H}$ having the form

$$\mathtt{H} = \begin{bmatrix} \times & 0 & \times \\ 0 & \times & 0 \\ \times & 0 & \times \end{bmatrix}. \tag{5.31}$$

For special planar motion, the trifocal tensor has the reduced form given in equation (5.23). The special planar motion and normalised planar motion are related by a plane projective transformation of form given in equation (5.31). The affect of a projective transformation of the image on the trifocal tensor is described in appendix B.2, and given in equation (B.3). Using the reduced trifocal tensor, and applying the projective transformation of the form of equation (5.31), shows that the trifocal tensor has the same reduced form for the normalised planar motion. This result allows the method of finding the circular points, via the reduced tensor and a single cubic equation, to be used for the normalised planar motion.

5.5 Algorithms for Self-Calibration from Fixed Points

In this section, the implemented algorithms used for self-calibration from fixed points and planar motion are explained. Following the general theme, the algorithms have been stratified such that: first, the fixed image points/lines are found (see algorithm 5.1); then, affine structure is recovered; finally, either planar metric or full metric structure is obtained (see algorithm 5.2).

5.5.1 Algorithm for Finding the Fixed Points

Section 5.1.5 showed that for an image triplet from a camera undergoing planar motion, there are four fixed points and four fixed lines intersecting the fixed points. Section 5.4.2 introduced a two-stage method for finding the positions of these fixed points. First, the fundamental matrices are used to find a fixed line and fixed point. Then the trifocal tensor

is used to find the positions of the three remaining fixed points, with the original line and point being used to reduce the numerical complexity of this stage.

The Fundamental Matrices The corresponding point matches are found automatically, with outlier detection, by the methods of Beardsley *et al.* [8]. The fundamental matrix is then computed for all image pairs, using the matched image points, and details are given in appendix A.3. The planar motion parameterisation [95] given in appendix A.2 is used when computing the fundamental matrices, and this ensures that $\mathtt{F}_s$ is Rank 2.

The symmetric part of the fundamental matrix is a degenerate conic of two lines, corresponding to the horizon line and the image of the screw axis. Using two or more matrices, the vanishing point, as well as the horizon line, can be found. However, when using real images, there will not be an exact solution for the horizon line or the vanishing point.

The Horizon Line At least two methods exist for finding the position of the horizon line from several estimates (one estimate from each image pair):

- **Fitting to Epipoles** All the epipoles are constrained to lie on the horizon line (see section 5.1.5), and so the fixed line can be estimated as the best fit line, using orthogonal regression, through all the epipoles. However there are several scenarios where this could give erroneous results:
  - if the general parameterisation is used for $\mathtt{F}$, then (due to image noise) $\mathtt{F}_s$ may not be Rank 2, and the epipoles will not lie on $\mathtt{l}$. However, using the planar motion parameterisation for the fundamental matrix removes this problem and it is no longer an issue.

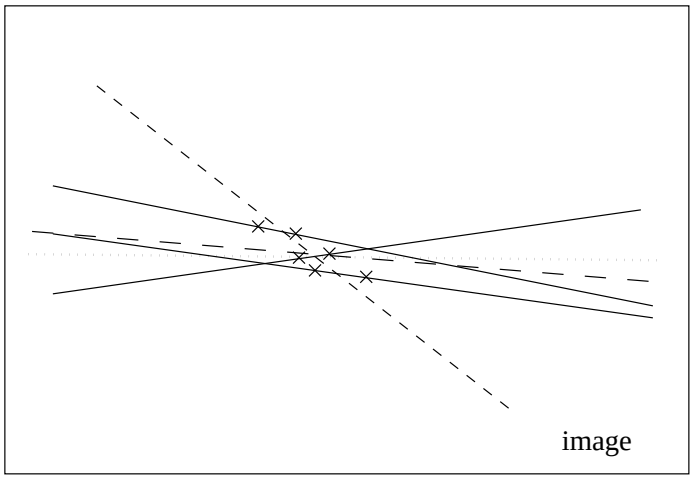


Figure 5.11: *The incorrect fitting of the horizon line. The actual horizon line is shown dotted. The epipoles ($\times$) and image lines (solid) from each fundamental matrix are also shown. The estimate of the horizon line obtained by fitting a line to the epipoles is shown short-dashed, and is obviously incorrect, while the estimate obtained by taking the mean position of the individual estimates is shown long-dashed and is more accurate.*

  - if only a few image pairs are used and the rotation between views is small then the line through the epipoles could give an erroneous results (see figure 5.11).

- **Mean Position** Rather than use the epipoles, it is simpler to take the mean position of the lines which are the different estimates for the horizon lines from each `F`. Care needs to be taken as outlying results can produce significant changes in the mean position. Section 6.2.1 discusses the weighting of each estimate with its covariance, which reduces the effect of outlying estimates.

Two typical sets of estimates (from real data) for the horizon line are shown in figure 5.12, and figure 5.12b shows a set of estimates where fitting to the epipoles gives an erroneous result compared to taking the mean position.

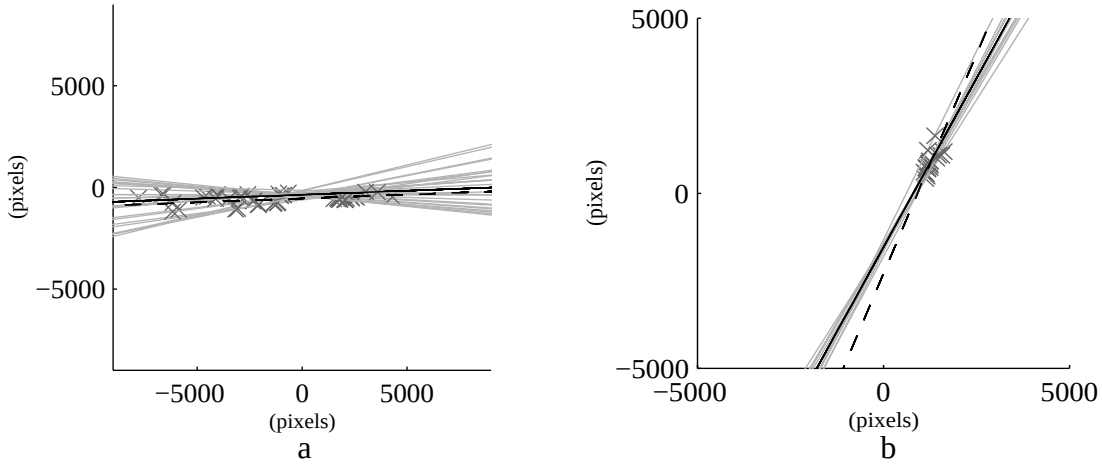


Figure 5.12: *Finding the position of the horizon line with different methods. The solid black line is the mean position of the different estimates shown in grey, while the dashed black line is fitted to the epipoles (×). For some images (a), both methods produce similar results, but the fitting to epipoles can produce an erroneous value (b), depending on the position of the epipoles.*

The Vanishing Point Each pair of images (via $\mathtt{F}_s$) gives a line in the image, which is the image of the associated screw axis, and these lines should intersect at the point $\mathbf{v}$ (see section 5.1.4). With real images, they will not intersect, and the vanishing point is estimated as the point which minimises the perpendicular distance from each line to the point. The details are given in appendix C.2.

A Batch Method Rather than compute each fundamental matrix and then use the image lines to compute the position of the vanishing point and horizon line, it is possible to use a batch parameterisation for several fundamental matrices where the horizon line and vanishing point are used directly as parameters. As a result, the positions of these fixed entities are obtained when computing the fundamental matrices which avoids having to find a best estimate. More details on the batch parameterisation can be found in section 6.2.2. The remainder of this chapter uses the method described above to compute the position of the vanishing point and horizon line.

- Compute $\mathtt{F}$ for all image pairs.
- Decompose each $\mathtt{F}_s$ to give two image lines.
- Compute the position of the horizon line ($\mathbf{l}$) and vanishing point ($\mathbf{v}$) using the set of image lines.
- Form the canonical transformation ($\mathtt{H}_c$), and apply to all images.
- Compute the reduced trifocal tensor for all image triplets.
- For each tensor, solve the cubic equation (5.32), and the complex solutions correspond to the image of the circular points.
- Use the inverse canonical transformation, to give the position of circular points ($\mathbf{i}$ and $\mathbf{j}$) in the real image.

Algorithm 5.1: *Finding the fixed image points of planar motion.*

The Canonical Transformation The canonical transformation ($\mathtt{H}_c$) is computed using the position of the horizon line and vanishing point, using equation (5.30). Then each image is transformed with this projective transformation, so that the image sequence corresponds to normalised planar motion.

The Reduced Trifocal Tensor With normalised planar motion, the trifocal tensor has only 12 non-zero elements. This reduced tensor is computed using point matches which have been tracked through the image triplet. Details are given in appendix B.1.

The Fixed Points A single cubic equation for the positions of fixed lines passing through $\mathbf{v}$ is formed using equation (5.26)

$$T_1^{33}z^3 + (T_1^{13} + T_1^{31} - T_3^{33})z^2 + (T_1^{11} - T_3^{13} - T_3^{31})z - T_3^{11} = 0. \qquad (5.32)$$

A closed form solution exists [1]. There are three solutions (one real and two imaginary). The real solution corresponds to the fourth fixed point $\mathbf{x}$, peculiar to the image triplet (see section 5.1.5), and is discarded. The complex solutions are the fixed lines $(1, 0, z)^\top$ and $(1, 0, \bar{z})^\top$ which intersect the horizon line $(0, 1, 0)^\top$ at the image of the circular points $(-z, 0, 1)^\top$ and $(-\bar{z}, 0, 1)^\top$. The inverse canonical transformation is applied to the image of the circular points, to give the position of $\mathbf{i}$ and $\mathbf{j}$ in the real image.

$$\begin{aligned} \mathbf{i} &= \mathtt{H}_c^{-1}(-z, 0, 1)^\top, \\ \mathbf{j} &= \mathtt{H}_c^{-1}(-\bar{z}, 0, 1)^\top. \end{aligned} \tag{5.33}$$

5.5.2 Algorithm for Recovering Affine and Metric Structure

As mentioned previously, the three fixed image points ($\mathbf{v}$, $\mathbf{i}$, and $\mathbf{j}$) are images of three non-collinear points on the plane at infinity, and if these points are back-projected into a projective reconstruction, they uniquely identify π_∞. This allows affine structure to be recovered. The three points also give 4 constraints on the position of the image of the absolute conic, so given one further constraint on camera calibration, the conic ω can be identified, and thence metric structure recovered.

The projective structure is computed by back-projecting the matched image points from each image of the triplet. This requires computing the camera projection matrices. It is essential that these three matrices are consistent (i.e., that rays back-projected from the fixed points are not skew). This is especially important here because the points being back-projecting are complex, and have no metric for measuring projection errors for complex points in the image plane.

Consistent camera projection matrices can be obtained directly by decomposing the trifocal tensor. Hartley [41] showed that the three projection matrices can be computed from the trifocal tensor, but that this is a difficult problem prone to instability. However,

- Compute three consistent camera projection matrices from the reduced trifocal tensor and the inverse canonical transformation.
- Back-project the image points to recover a projective reconstruction.
- Back-project the fixed points, and identify π_∞ in the projective structure.
- Upgrade to affine structure using equation (2.14) and the position of π_∞.
- Use the fixed points to compute the pencil of conics containing ω.
- Use another assumption to identify ω, and obtain the camera calibration.
- Upgrade to metric structure using equation (2.16).

Algorithm 5.2: *Affine and metric reconstruction using the fixed points.*

for normalised planar motion, the tensor has a reduced form with 12 parameters, and it also known that the projection matrices have a reduced form with only 12 degrees of freedom in total (see section 5.4.2). An algorithm for computing consistent projection matrices for normalised planar motion is given in appendix B.4.

The image points can now be back-projected to recover a projective structure. The position of the fixed points are also back-projected using the same projection matrices, and the plane at infinity is computed as the plane passing through these fixed points. Conventional affine structure can be recovered using equation (2.14). Using the method in section 5.2, and one other constraint on camera calibration, the image of the absolute conic can be identified and hence the calibration $\mathtt{C}$ can be found. Metric structure can be recovered from the affine by using equation (2.16).

If no assumption can be made about the camera calibration, then the structure can only be recovered up to planar metric. The planar metric structure can be recovered by two methods. The aspect ratio can be set to an arbitrary value, and the method proceeds as

if recovering full metric structure from the affine reconstruction. Alternatively, the affine structure can be transformed so that the 3-space position of the circular points **I** and **J** map to $(1, 0 \pm i, 0)^\top$, the vanishing point **V** is fixed at $(0, 1, 0, 0)^\top$, and π_∞ is fixed at $(0, 0, 0, 1)^\top$.

5.6 Results for Self-Calibration via Fixed Points

Several real images sequences have been used to demonstrate the effectiveness of self-calibration and affine/metric reconstruction using fixed points. In some sequences, known measurements allow the accuracy of the reconstruction to be measured. The sequences are:

- **Sequence I** The camera is mounted on the Adept robot arm, and there are eight images in the sequence (see figure 5.13). The plane of motion is parallel to the desk, and the axis of rotation is *vertical* relative to the desk. The camera is elevated at approximately 25° to the camera Y axis with no yaw or roll. 149 points are automatically matched and tracked through the eight images. The Tsai grids in the scene are used to provide veridical values for the structure, which allow the accuracy of the reconstruction to be measured (they are **not** used to calibrate). The approximate calibration is ($\alpha_u = 1000, \zeta = 1.02, (u_0, v_0) = (400, 300)$).

- **Sequences II** Three sequences are taken with a camera mounted on an AGV, and only the elevation of the camera is changed between sequences. The approximate calibration is ($\alpha_u = 345, \zeta = 1.1, (u_0, v_0) = (126, 126)$). The different sequences are:

  - **Sequence IIa** elevation angle of 10°.
  - **Sequence IIb** elevation angle of 20°.
  - **Sequence IIc** elevation angle of 30°.

Points are tracked over at least four images (a *quartet*), and the fundamental matrices and trifocal tensors are computed for the possible combinations in each image quartet. The number of points matched for the quartets varies between 24 and 80 points. The low number of matched points occur because for some quartets there is significant rotation between views and the camera is not fixated on any particular object.

5.6.1 Self-Calibration

The results for each sequence are split into several parts and given in the figures and tables on pages 107–117.

Horizon Line and Vanishing Point

The fundamental matrix is computed for all possible image pairs, and then decomposed to give the two image lines and epipoles. For each sequence these are displayed on a figure which contains each estimate for the horizon line (*dotted*), the image of each screw axis (*dashed*), and the epipoles ($\times$). The figure also shows the estimated position of the vanishing point ($\circ$), and the estimate of the horizon line (*solid*) computed using the mean position of all estimates.

The results show that the estimate for the horizon line and image of the screw axes show some variation over the image sequence, but that an accurate estimate of the horizon line and the vanishing point can be made given enough image pairs. Sequence II (see figure 5.24) show that the vanishing point is better constrained with the increasing elevation angle, but that the horizon line is not.

Circular Points

The trifocal tensor is computed for all possible image triplets and each tensor gives an estimate for the positions of the circular points. The distribution of the circular points for each sequence is shown in the appropriate table. The results show that the circular points computed from different triplets are stable, and that with sequence II this stability increases with elevation angle.

Calibration

The camera calibration is computed using each estimate of the circular points with the single estimate for the position of vanishing point. Results are given for two assumptions about the camera calibration, that either the image skew is zero or the aspect ratio is known. The distribution of the computed calibration is shown in the appropriate table.

The results show that computed calibration is stable over the image sequence and similar to the approximate values. For each sequence, the results obtained with the aspect ratio assumption are more stable (and accurate) than those obtained with the zero image skew constraint. This observation agrees with the theory in section 5.2.3.

The results for sequence II show that the position of v_0 changes with the elevation angle, and moves from the nominally correct position of 110 pixels to 24 pixels. The cause of this drifting has not been identified, but as the next section shows, the accuracy of the metric reconstruction is not affected. This change in position could be due to the normal variation of principal point observed in other calibration methods.

5.6.2 Structure Recovery

For sequence I, the structure in the scene is known, and so the accuracy of the affine and metric structure can be measured by comparing affine and metric invariants to the veridical

values.

The affine invariant used is the ratio of parallel lines, and ten random measurements are made using the points on the calibration grid and compared to the veridical value of 1.0. The ten measurements are made on the reconstruction using each triplet, and the distribution of the results are shown in table 5.4.

The metric invariants are the angles between the planes of the calibration grid (see figure 5.15). Planes are fitted, using orthogonal regression, to 23 and 18 points on the left and right faces, and 15 points on the top row. The angle between the planes is then compared to the veridical value of 90°. The distribution of the measured angles is shown in table 5.4.

The values computed for the affine and metric invariants show that the reconstruction is stable and accurate. Figures 5.16 and 5.17 show one of the affine and metric reconstructions, with the image texture mapped onto the reconstruction to aid visualisation. Figure 5.16 contains affine skew in the reconstruction which has been removed in figure 5.17.

For sequence II, fewer invariants can be measured in the reconstructions, but figure 5.25 shows the lines joining three points whose positions in the world and in the reconstruction are known. This allows the Euclidean invariants of an angle and the ratio of non-parallel lines to be measured in the reconstruction. Using the reconstruction from the first quartet (four images) from sequences IIb and IIc, the angle is measured as 89° (IIb) and 84° (IIc) compared to the veridical value of 90°, and the ratio of the lines is measured as 0.61 (IIb) and 0.59 (IIc) compared to the approximate veridical value of 0.65.

Summary

This chapter has presented a novel method for self-calibrating a camera moving under planar motion. The method is based on the idea that there are points and lines in the image, with corresponding points and lines in the world, that remain fixed during planar motion, and whose position is closely related to the camera calibration.

The theory of how these fixed entities arise and how their position determines the camera calibration was explained in detail. The algorithm to find the position of these fixed entities using three images was given, and it utilised the idea of stratification. First a fixed point and line are found using the fundamental matrices. Then the positions of these are used to reduce the algebraic complexity of finding the remaining fixed points using the trifocal tensor. Extensive results were given for real image sequences and those results were shown to be both stable and accurate.

The zero image skew constraint, a very common assumption used for calibration, was shown to be unstable under certain circumstances, which are common for a camera moving under planar motion.

The next chapter tries to estimate the covariance of the calibration parameters computed using the algorithm presented in this chapter.

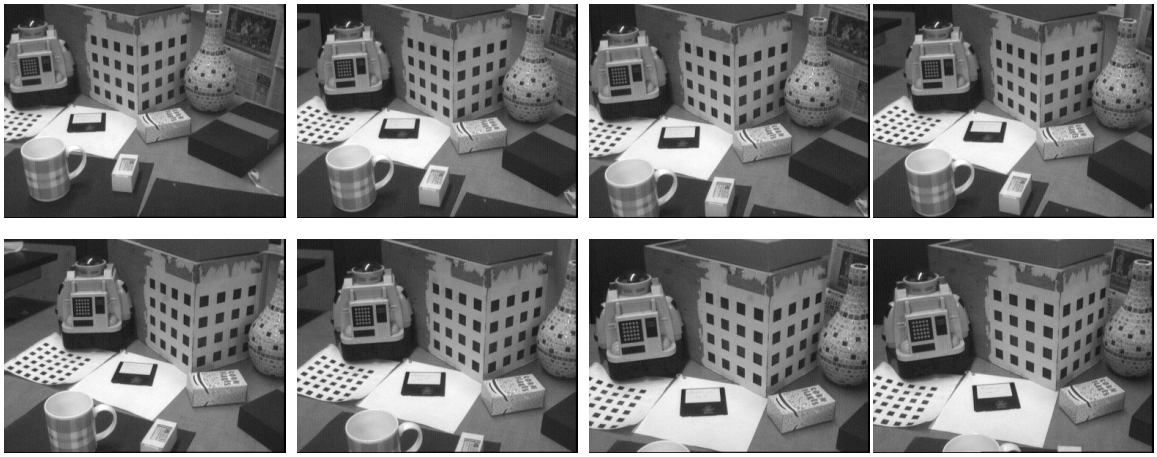


Figure 5.13: *The eight images of sequence I.*

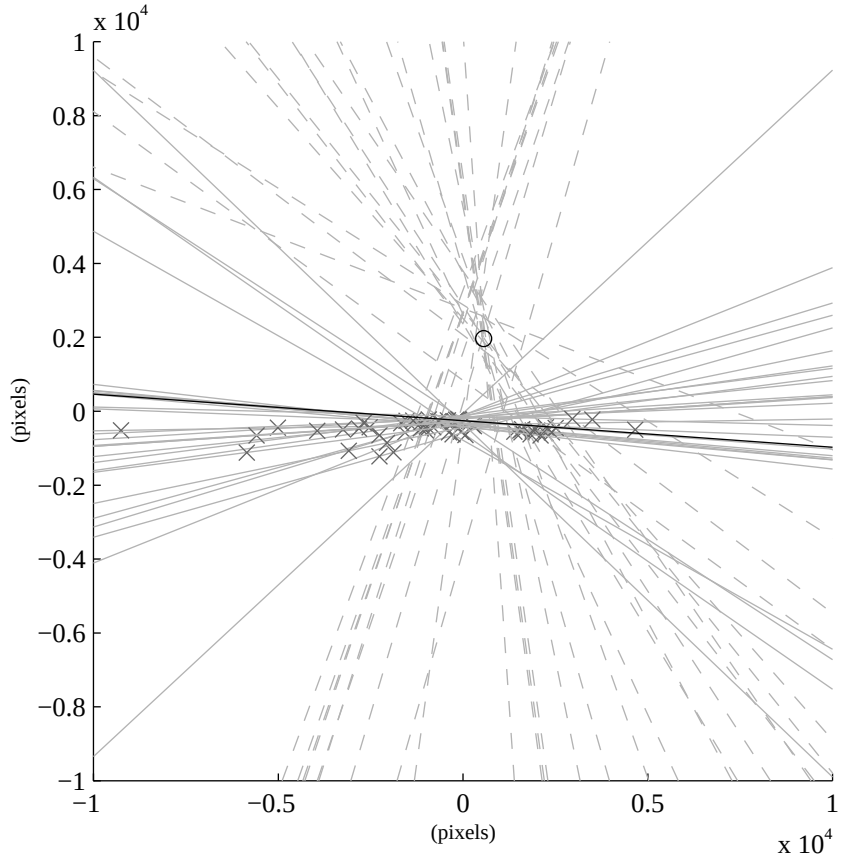


Figure 5.14: *Estimating the horizon line and vanishing point for sequence I. The epipoles* $(\times)$*, the horizon lines (grey solid), and the image of the screw axes (grey dashed) are shown for all image pairs. The vanishing point* $(\circ)$ *is at (558,1967), and the horizon line (black solid) at* $(-2.8e^{-4}, 3.9e^{-3}, 1)$ *which passes through (0,-255).*

	Circular point	
	x	y
Mean	$315 \pm 1223i$	$-278 \mp 88i$
Median	$321 \pm 1225i$	$-278 \mp 88i$
Std. Dev.	$38 \pm 38i$	$3 \mp 3i$

Table 5.1: *The statistics for the position of the circular point in sequence I estimated using all possible trifocal tensors and the values for* **v** *and* **l** *from figure 5.14.*

	α_u	α_v	ζ	u_0	v_0	k
Mean	1020	1041	1.021	391	422	41
Median	1021	1044	1.021	396	429	38
Std. Dev.	18	18	0.001	23	46	18

Table 5.2: *The statistics for the calibration in sequence I estimated using all possible circular points from table 5.1 and the known aspect ratio of 1.02.*

	α_u	α_v	ζ	u_0	v_0	k
Mean	1097	900	0.822	363	173	0
Median	1095	908	0.825	369	179	0
Std. Dev.	39	44	0.061	37	54	0

Table 5.3: *The statistics for the calibration in sequence I estimated using all possible circular points from table 5.1 and the zero image skew constraint.*

Figure 5.15: *The orthogonal triad of angles on the calibration object (highlighted) which are measured in the metric reconstruction from sequence I, and compared to the veridical values of* 90°. *The angles are measured between the planes of calibration grid, and the planes are fitted by orthogonal regression to 23 and 18 points on the left and right grids, and 15 points on the top row.*

	Affine	Metric Invariants		
	Invariant	Angle 1	Angle 2	Angle 3
Mean	1.017	84.4	87.6	89.9
Median	1.012	84.0	88.4	91.0
Std. Dev.	0.061	3.0	5.2	5.8

Table 5.4: *The statisticss for the invariants measurements made in the affine and metric reconstructions for sequence I. The affine invariant is the ratio of parallel lines measured on the calibration grid, with 10 measurements made in each reconstruction, and with a veridical value of 1.0. The metric invariants are the three orthogonal angles shown in figure 5.15 with a veridical value of* 90°.

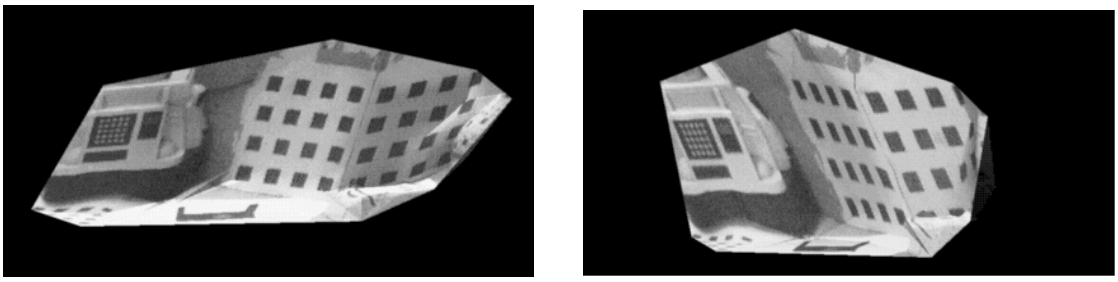


Figure 5.16: *Two views of the affine reconstruction from sequence I, with the image texture mapped onto the reconstruction. Note, the affine skew in the grid and between the grid and the desk.*

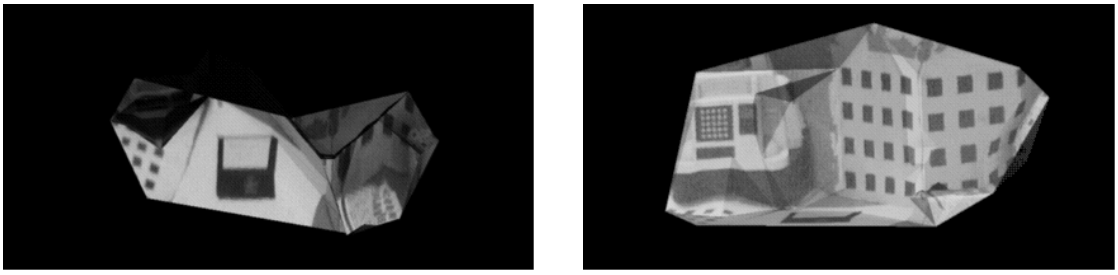


Figure 5.17: *Two views of the metric reconstruction from sequence I, with the image texture mapped onto the reconstruction. Note, all the affine skew has been removed with the grid being orthogonal to the desk.*

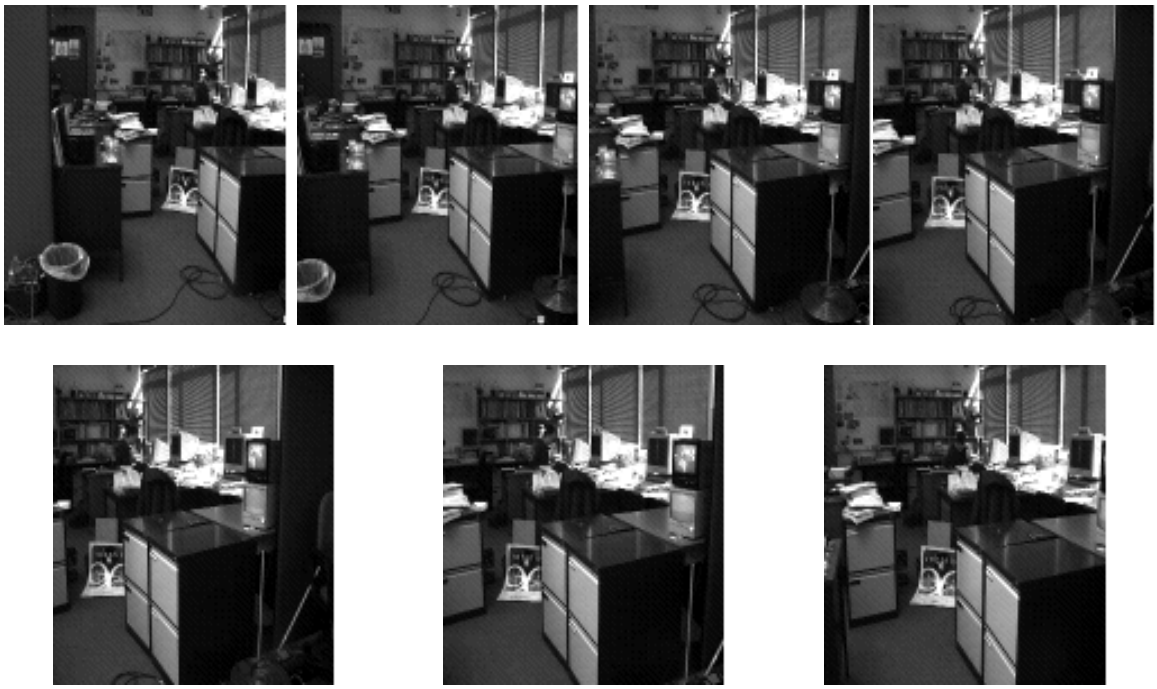


Figure 5.18: *The seven images of sequence IIa.*

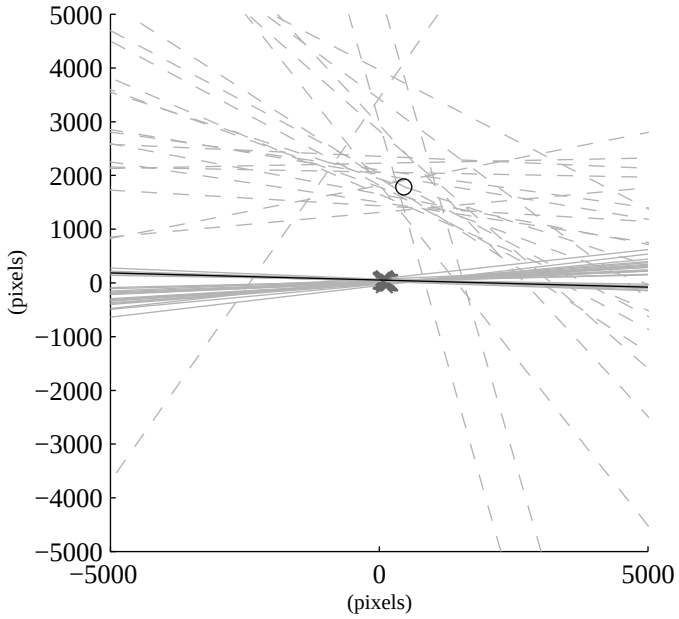


Figure 5.19: *Estimating the horizon line and vanishing point for sequence IIa. The epipoles* $(\times)$*, the horizon lines (grey solid), and the image of the screw axes (grey dashed) are shown for all image pairs. The vanishing point* $(\circ)$ *is at (455,1787), and the horizon line (black solid) at* $(-4.9e^{-4}, -1.9e^{-2}, 1)$ *which passes through (0,54).*

	Circular point	
	x	y
Mean	$150 \pm 294i$	$50 \mp 8i$
Median	$128 \pm 289i$	$50 \mp 8i$
Std. Dev.	$57 \pm 19i$	$2 \pm 0i$

Table 5.5: *The statistics for the position of the circular point in sequence IIa estimated using all possible trifocal tensors.*

	α_u	α_v	ζ	u_0	v_0	k
Mean	290	324	1.117	161	113	50
Median	285	319	1.119	140	111	54
Std. Dev.	18	20	0.007	55	9	12

Table 5.6: *The statistics for the calibration in sequence IIa estimated using all possible circular points, assuming the aspect ratio is 1.1.*

	α_u	α_v	ζ	u_0	v_0	k
Mean	294	116	0.393	152	58	0
Median	289	111	0.374	129	57	0
Std. Dev.	19	15	0.043	57	1	0

Table 5.7: *The statistics for the calibration in sequence IIa estimated using all possible circular points, assuming the image skew is zero.*

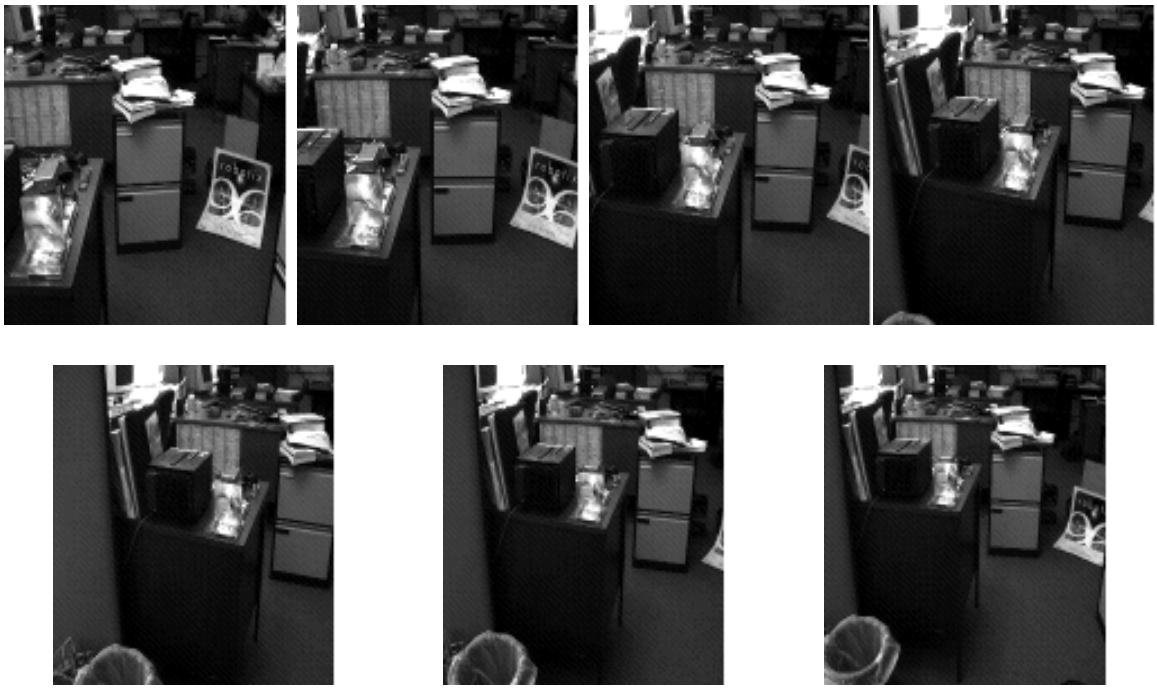


Figure 5.20: *The seven images of sequence IIb.*

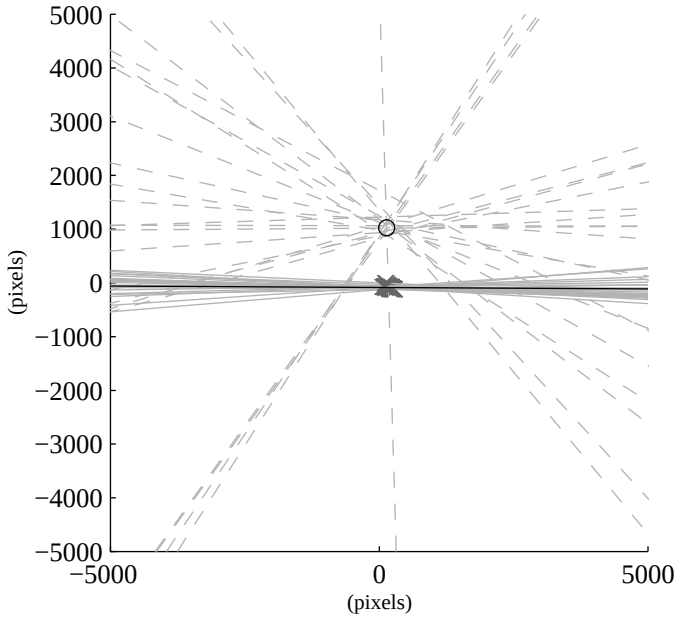


Figure 5.21: *Estimating the horizon line and vanishing point for sequence IIb. The epipoles* $(\times)$*, the horizon lines (grey solid), and the image of the screw axes (grey dashed) are shown for all image pairs. The vanishing point* $(\circ)$ *is at (139,1023), and the horizon line(black solid) at* $(6.7e^{-5}, 1.5e^{-1.2}, 1)$ *which passes through (0,-84).*

	Circular point	
	x	y
Mean	$104 \pm 362i$	$-86 \mp 2i$
Median	$125 \pm 344i$	$-86 \mp 2i$
Std. Dev.	$89 \pm 103i$	$1 \pm 1i$

Table 5.8: *The statistics for the position of the circular point in sequence IIb estimated using all possible trifocal tensors.*

	α_u	α_v	ζ	u_0	v_0	k
Mean	330	363	1.100	123	50	5
Median	324	356	1.100	127	44	3
Std. Dev.	20	22	0.001	14	19	6

Table 5.9: *The statistics for the calibration in sequence IIb estimated using all possible circular points, assuming the aspect ratio is 1.1.*

	α_u	α_v	ζ	u_0	v_0	k
Mean	338	253	0.801	121	2	0
Median	337	225	0.672	126	-38	0
Std. Dev.	39	119	0.531	16	117	0

Table 5.10: *The statistics for the calibration in sequence IIb estimated using all possible circular points, assuming the image skew is zero.*

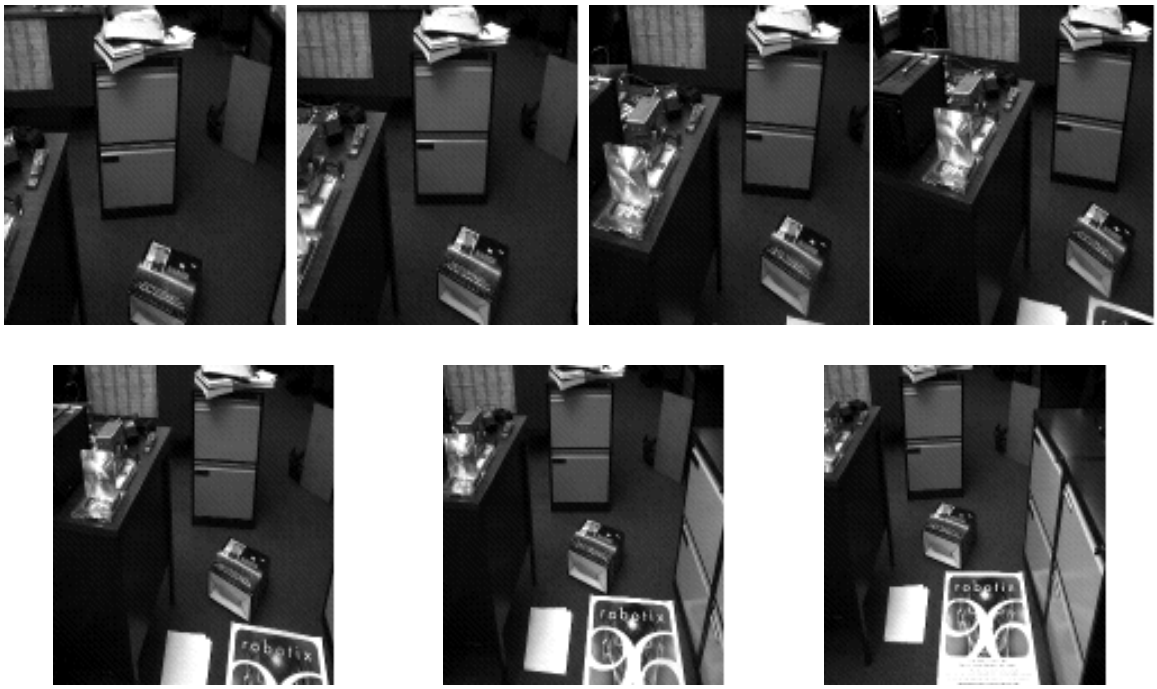


Figure 5.22: *The seven images of sequence IIc.*

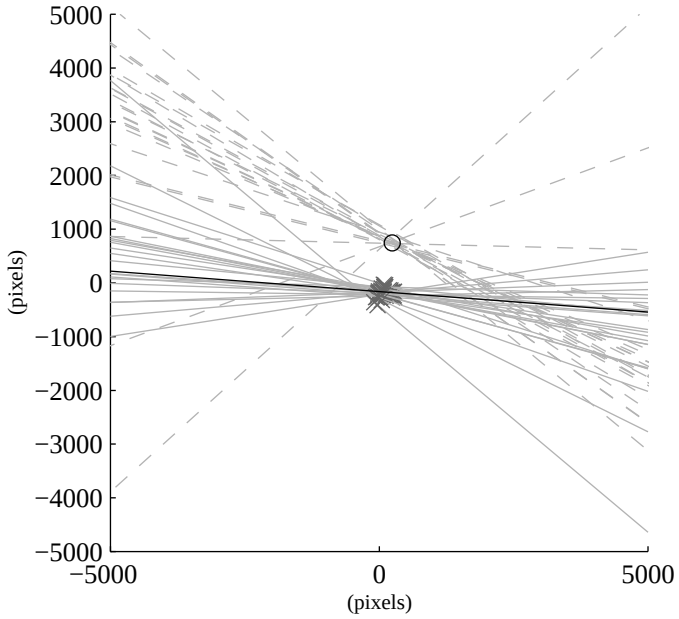


Figure 5.23: *Estimating the horizon line and vanishing point for sequence IIc. The epipoles* $(\times)$*, the horizon lines (grey solid), and the image of the screw axes (grey dashed) are shown for all image pairs. The vanishing point* $(\circ)$ *is at (241,743), and the horizon line (black solid) at* $(4.7e^{-4}, 6.2e^{-3}, 1)$ *which passes through (0,-160).*

	Circular point	
	x	y
Mean	$110 \pm 377i$	$-169 \mp 29i$
Median	$110 \pm 372i$	$-169 \mp 28i$
Std. Dev.	$17 \pm 20i$	$1 \pm 2i$

Table 5.11: *The statistics for the position of the circular point in sequence IIc estimated using all possible trifocal tensors.*

	α_u	α_v	ζ	u_0	v_0	k
Mean	338	373	1.105	138	24	30
Median	335	370	1.105	137	20	31
Std. Dev.	12	13	0.002	15	19	7

Table 5.12: *The statistics for the calibration in sequence IIc estimated using all possible circular points, assuming the aspect ratio is 1.1.*

	α_u	α_v	ζ	u_0	v_0	k
Mean	361	265	0.733	122	-83	0
Median	359	259	0.726	121	-90	0
Std. Dev.	14	29	0.060	18	25	0

Table 5.13: *The statistics for the calibration in sequence IIc estimated using all possible circular points, assuming there is no image skew.*

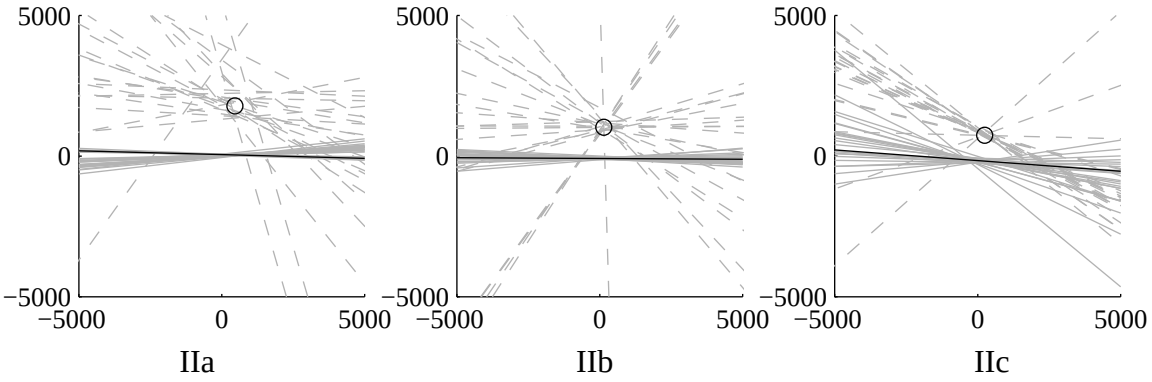


Figure 5.24: *The three results for sequences II juxtaposed, showing that the vanishing point is better constrained with increasing elevation angle, but that the horizon line is not. The full results for each sequence are shown in figures 5.19, 5.21, and 5.23 which also explain the notation.*

Figure 5.25: *The Euclidean invariants measured in the reconstruction from sequences IIb and IIc. The right angle is measured as* 89° *(IIb) and* 84° *(IIc) respectively. The ratio of the non-parallel lengths is measured as 0.61 (IIb) and 0.59 (IIc) compared to the approximate veridical value of 0.65.*

Chapter 6

Fixed Points and Image Sequences

Overview

The previous chapter introduced the idea of self-calibration using fixed points for cameras undergoing planar motion, and results using real images were shown to be stable and accurate. However, computer vision not only requires a numerical solution to a problem but also a measure of confidence in the solution. This chapter examines how a measure of confidence can be computed for self-calibration using fixed points. This is very closely linked to the idea of how to combine the results for an image sequence rather than a single image triplet. Is it better to treat the sequence as a series of image triplets, or use the whole sequence simultaneously as a batch which might give better results but at greater computational cost?

The chapter organisation follows the structure of the self-calibration algorithm. First, the different methods for computing the confidence in the vanishing point and horizon line are discussed. A novel batch parameterisation for the fundamental matrices of multiple images is given, and it is shown that this batch parameterisation gives a more accurate estimate for the covariance of the vanishing point and horizon line.

Then the next step of the algorithm, using the trifocal tensor to find the circular points, is analysed. It is shown that the noise on the image points in the normalised planar motion frame is neither isotropic, homogeneous, nor Gaussian, which greatly increases the problem

of computing the covariance of the circular points. Various techniques are explained and are shown to be unsatisfactory. Nevertheless, the final steps of the algorithm, computing the camera calibration using the fixed points, is analysed and a method suggested for computing the covariance of the internal parameters.

For all parts of the chapter, synthetic data with added noise is used with a Monte Carlo simulation to show that the covariances (the measure of confidence) computed for the different parameters are correct. Section 6.1 describes the synthetic sequence used, and the different measurements used to assess the accuracy of the covariance matrices. Results are also given for the real image sequences used in chapter 5.

6.1 Error Analysis

The objective of this chapter is to derive an analytical solution for the covariance of the camera calibration computed using the fixed points of planar motion. To show that the suggested algorithms are correct, Monte Carlo simulations [78] are used throughout the chapter to allow the comparison of the actual distribution of solutions and the derived analytical solutions.

All the Monte Carlo simulations are based on the synthetic sequence SI, and for each run Gaussian image noise is added to the correct image points. The number of runs (samples) and the amount of noise added is given with each set of results. The noise is only added to the image points and the uncertainty is then transformed through each step of the algorithm.

- **Sequence SI** Fifty points are arranged as a double Tsai grid (see figure 3.1a) with the size of the grid 1 unit squared, and viewed by an ideal pinhole camera at a distance of 4 units. The camera calibration is ($\alpha_u = 900, \zeta = 1.0, (u_0, v_0) = (400, 300)$). A single image triplet, consisting of three distinct images of the structure, is taken with the camera rotating 10° around a fixed axis between views and the translation confined

to the plane perpendicular to the rotation axis. To avoid degenerate situations the orientation of the camera is such that the rotation axis is not close to one of the camera axes and the horizon line is not horizontal.

The results are assessed with two different measures:

1. The covariance of the distribution of results obtained with the Monte Carlo simulation (*statistical covariance*) is compared to the predicted distribution (*analytical covariance*). For each run of the simulation a covariance is computed using the equations derived below, and the *analytical covariance* shown in the results is the mean of the computed covariance from all the runs.

2. Using the χ^2 distribution and the hyper-ellipsoid of uncertainty [17] (see below), the distribution of results can be combined with either the statistical or analytical covariances to show that the distribution of the results is Gaussian. This is done by comparing the number of samples lying inside the hyper-ellipsoid of uncertainty to the total number of samples. Results are given for different levels of uncertainty, which should give the ratio of samples inside to the total number of samples as 0.6, 0.8, 0.9, or 0.95.

The Hyper-Ellipsoid of Uncertainty

The hyper-ellipsoid of uncertainty can be used to compare a covariance matrix to the actual distribution of parameters it is attempting to model, and tests that the distribution is Gaussian. If a population $\mathbf{y}$ has a distribution with the covariance $\Lambda_{\mathbf{y}}$, then a hyper-ellipsoid can be defined using the mean and covariance of $\mathbf{y}$

$$(\mathbf{y} - \mathrm{E}[\mathbf{y}])^{\top} \Lambda_{\mathbf{y}}^{-1} (\mathbf{y} - \mathrm{E}[\mathbf{y}]) = k^2. \tag{6.1}$$

	$P_{\chi^2}(k,r)$			
r	0.6	0.8	0.9	0.95
2	1.833	3.219	4.605	5.991
6	6.211	8.558	10.64	12.59

Table 6.1: *The percentage points of the χ^2-distribution. The value of k^2 required to give the probability $P_{\chi^2}(k,r)$ (ratio) for the different values of r, taken from [53].*

The size of the hyper-ellipsoid is dependent on the value of k. The probability that a sample from the population $\mathbf{y}$ lies inside this hyper-ellipsoid is given by the χ^2-distribution [53], $P_{\chi^2}(k,r)$, where r is the rank of $\Lambda_{\mathbf{y}}$. For more details on the derivation see [17, 104].

The hyper-ellipsoid can now be used to assess how accurately the predicted covariance matrix describes the actual distribution. Using the results of the Monte Carlo simulations, the ratio of the number of samples lying inside a hyper-ellipsoid to the total number of samples can be calculated. This calculated ratio is then compared to the theoretical ratio given by $P_{\chi^2}(k,r)$. The calculation is repeated for several hyper-ellipsoid with different values of k, hence different ratios, and this gives a qualitative measure of how well the covariance matrix models the distribution. Results are given for the four ratios of 0.6, 0.8, 0.9, and 0.95, and table 6.1 gives the corresponding values of k for different r. For two or three-dimensional variables, the hyper-ellipsoid can be used to give a graphical representation of the covariance as either a conic or as an ellipsoid.

6.2 Horizon Line and Vanishing Point

The algorithm used to compute the position of the horizon line and vanishing point is described in detail in section 5.3, but can be summarised as: compute the fundamental matrix between image pairs, then decompose the symmetric part of the matrix to give two image lines; finally use the image lines from two or more image pairs to give the positions

of the horizon line and vanishing point.

First, the image pairs are treated separately so that the covariance of each fundamental matrix and the corresponding image lines can be computed. Then the covariance of the image lines can be taken into account when computing the position and covariance of the horizon line and vanishing point.

An alternative *batch* approach is explained where the fundamental matrices for several images are computed simultaneously, so that the position and covariance of the horizon line and vanishing points can be computed directly. The *batch* approach can deal with an an arbitrary number of images but the results given here use only the three images of the image triplet in sequence SI.

6.2.1 Image Pairs

The fundamental matrix is computed using the planar motion parameterisation of Vieville and Lingrand [95]. The advantage of this, apart from being the minimum parameterisation, is that the image lines are used as part of the parameterisation. Each planar fundamental matrix is parameterised with six elements

$$\mathtt{F}_i \leftrightarrow \mathbf{f}_i = \mathbf{f}(\theta_i, \mathbf{l}_i, \mathbf{m}_i, \rho_i), \tag{6.2}$$

which are explained in appendix A.2, but $\mathbf{l}_i$ and $\mathbf{m}_i$ correspond to the image lines used to compute the horizon line and vanishing point.

The fundamental matrix is computed using the non-linear minimisation described in appendix A.3, and using the results of lemma C.1 (covariance of an implicit function) the covariance of the matrix can be computed. Hence, the covariance of the image lines is also computed as they are part of the parameterisation of the fundamental matrix. It follows that

for the i^{th} image pair the following parameters and covariances can be computed

$$\begin{aligned} \{\mathbf{l}_i, \mathbf{m}_i\} &\subset \mathbf{f}_i, \\ \{\Lambda_{\mathbf{l}_i}, \Lambda_{\mathbf{m}_i}\} &\subset \Lambda_{\mathbf{f}_i}, \end{aligned} \tag{6.3}$$

where $\mathbf{f}_i$ is the six parameter representation of $\mathtt{F}_i$ (see equation (6.2)), and $\Lambda_{\mathbf{x}}$ is the covariance of the parameter $\mathbf{x}$.

The vanishing point is found as the point which minimises the squared distance to the lines $\mathbf{m}_i$ (the images of the screw axes), and the covariance of each line $\Lambda_{\mathbf{m}_i}$ is taken into account. Both the position and covariance of the vanishing point are computed as $\mathbf{v}$ and $\Lambda_{\mathbf{v}}$. More details are given in appendix C.2 and [16].

The horizon line $\mathbf{l}$ is computed as the weighted least-squares average [6] of the set of estimates $\mathbf{l}_i$, such that

$$\mathbf{l} = \left[\sum_i \Lambda_{\mathbf{l}_i}^{-1}\right]^{-1} \left(\sum_i \Lambda_{\mathbf{l}_i}^{-1}\mathbf{l}_i\right), \tag{6.4}$$

$$\Lambda_{\mathbf{l}} = \left[\sum_i \Lambda_{\mathbf{l}_i}^{-1}\right]^{-1}. \tag{6.5}$$

Results

Csurka *et al.* [17] gave extensive results to show that the estimation of the covariance of the fundamental matrix, using the covariance of an implicit function minimisation, is correct, and so that work will not be repeated here. However, the planar parameterisation of the fundamental matrix was not used and so the distribution of the image lines is analysed and shown to be correctly estimated by the analytical solution. Then the estimation of the vanishing point and horizon line is discussed.

Image Lines The three fundamental matrices for sequence SI are computed for 1000 runs where the image points have either 0.1 or 0.3 pixels of noise added, and 500 runs

with 1.0 pixel of noise. The position and covariance of the image lines are a subset of the parameters and covariance of the fundamental matrices. The distribution of the lines is shown in figures 6.1 and 6.2, and tables 6.2, 6.3, and 6.4 give a numerical comparison of the statistical (actual) covariance of the image lines with the analytical (predicted) covariance. The covariances are listed, and using the χ^2 distribution gives a numerical comparison of the different covariances. All results show that the covariance of the lines predicted using analytical methods give very similar values to the true covariance.

Vanishing Point For each run of the Monte Carlo simulation, the three images of the screw axes ($\mathbf{m}_{1,2,3}$) are intersected to give the Monte Carlo distribution of the vanishing point. The covariance of the vanishing point is also computed for each run, hence giving the analytical covariance. Figures 6.3, 6.4, and 6.5 show the results for the three sequences described above.

The three lines being intersected are nearly parallel, so as expected the distribution of the intersection points is mainly along the mean direction of the lines. This is shown by the major axis of the 3σ ellipse for the *statistical covariance* being aligned with the intersecting lines (see figures 6.3, 6.4, and 6.5). However, although the 3σ ellipse for the *analytical covariance* is also aligned with the intersecting lines, it is significantly overestimated in that direction. The numerical results are given in tables 6.5, 6.6, and 6.7 and show that the analytical covariances are significantly different from the true statistical covariances.

Tables 6.2, 6.3, and 6.4 show that the covariance of one of the image lines ($\mathbf{m}_2$) is significantly smaller than the others. As a result of the weighting of the equations used to find the vanishing point, this line dominates the position and covariance of the vanishing point. However, this is not the whole story, as will be shown in section 6.2.2. A large part of the overestimation of the covariance is due to the fact that the correlation between

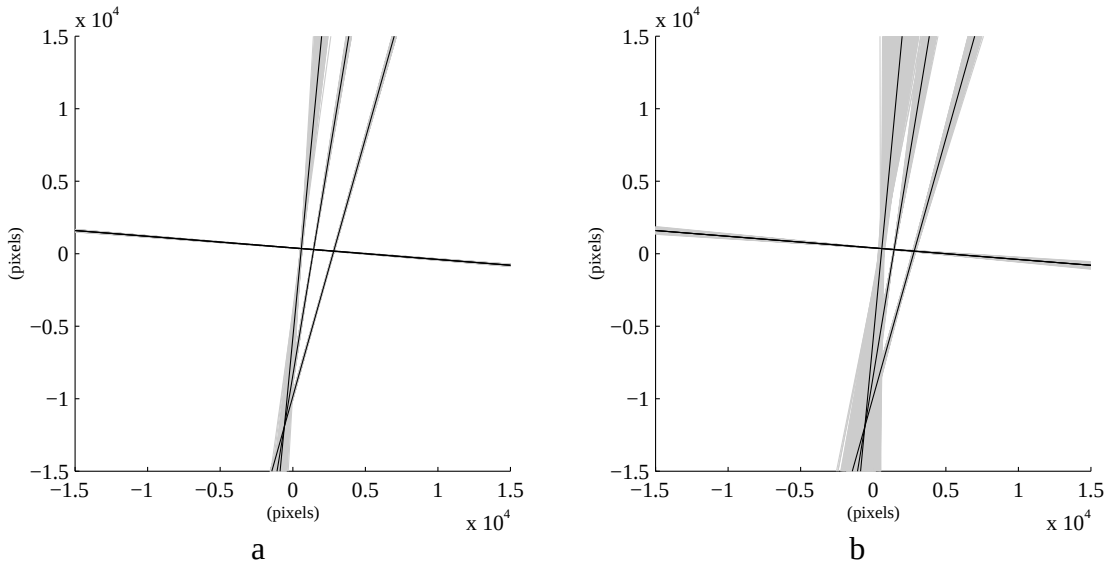


Figure 6.1: *The distribution of the image lines computed from 1000 runs of sequence SI with (a) 0.1 and (b) 0.3 pixels of Gaussian image noise added. The three fundamental matrices from the image triplet are decomposed to give the four lines shown: the three images of the screw axes* $\mathbf{m}_i$ *(one from each fundamental matrix), and the horizon line* $\mathbf{l}_i$ *(repeated in each fundamental matrix). The true values are shown in black while the results from each run of the Monte-Carlo simulation are shown in grey.*

the three fundamental matrices is not utilised. There is correlation between the matrices because they are calculated using the same image points and should have some identical parameters. In section 6.2.2 this correlation is taken into account and the overestimation of the covariance along the direction of the intersecting lines is removed.

Horizon Line The horizon line is computed for each run using equations (6.4) and (6.5) and the three estimates of the line from the three fundamental matrices. The results are given in tables 6.5, 6.6 and 6.7 for the three sequences, and show that the analytical results are similar to the true distributions.

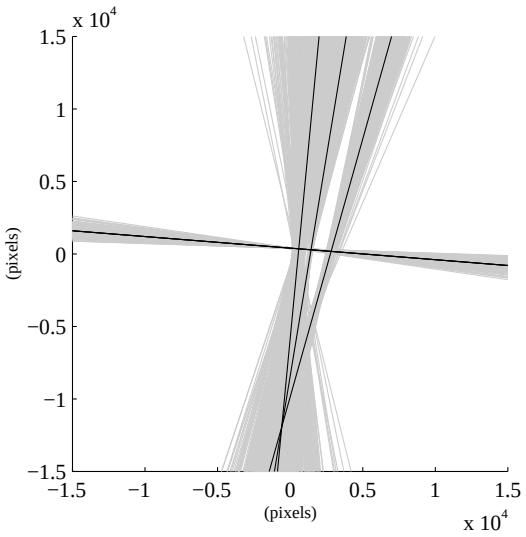


Figure 6.2: *The distribution of the image lines computed from 500 runs of sequence SI with 1.0 pixel of Gaussian image noise added. See figure 6.1 for explanation.*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
$\mathbf{m}_1$	Statistical	4.753e-09	-6.551e-10	4.358e-10	0.621	0.815	0.899	0.941
	Analytical	4.913e-09	-6.788e-10	4.506e-10	0.631	0.824	0.906	0.944
$\mathbf{m}_2$	Statistical	2.240e-12	2.890e-13	7.008e-13	0.621	0.809	0.901	0.950
	Analytical	2.593e-12	3.098e-13	7.107e-13	0.635	0.834	0.916	0.955
$\mathbf{m}_3$	Statistical	3.526e-12	6.936e-13	7.868e-12	0.615	0.799	0.909	0.948
	Analytical	3.875e-12	4.873e-13	7.736e-12	0.619	0.815	0.917	0.955
$\mathbf{l}_1$	Statistical	1.972e-13	2.213e-12	4.829e-11	0.591	0.788	0.891	0.955
	Analytical	2.095e-13	2.433e-12	5.245e-11	0.606	0.802	0.901	0.960
$\mathbf{l}_2$	Statistical	1.157e-12	-3.889e-12	1.352e-11	0.605	0.791	0.905	0.957
	Analytical	1.136e-12	-3.773e-12	1.300e-11	0.607	0.801	0.914	0.960
$\mathbf{l}_3$	Statistical	1.785e-11	-1.297e-11	9.789e-12	0.580	0.795	0.905	0.951
	Analytical	1.867e-11	-1.354e-11	1.021e-11	0.601	0.814	0.917	0.967

Table 6.2: *Comparing the statistical and analytical distributions of the lines shown in figure 6.1a (0.1 pixels of noise). The line is scaled so that* $\mathbf{l} = (x, y, 1)^\top$ *and the covariance has the three elements* σ_x^2, σ_{xy}, *and* σ_y^2. *The statistical covariance gives the actual distribution of the 1000 lines, while the analytical covariance is the mean of the 1000 computed line covariances. Also shown are the ratio of number samples lying inside the hyper-ellipsoid of uncertainty (see section 6.1) to the total number of samples with four values given which should be 0.6, 0.8, 0.9, and 0.95.*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
m_1	Statistical	5.023e-08	-6.692e-09	4.342e-09	0.632	0.832	0.903	0.945
	Analytical	5.361e-08	-7.353e-09	4.524e-09	0.653	0.837	0.912	0.950
m_2	Statistical	2.643e-11	3.831e-12	6.265e-12	0.583	0.803	0.909	0.952
	Analytical	1.958e-11	2.320e-12	6.183e-12	0.532	0.755	0.870	0.930
m_3	Statistical	3.148e-11	2.294e-12	7.380e-11	0.596	0.804	0.906	0.942
	Analytical	3.484e-11	4.362e-12	6.965e-11	0.599	0.807	0.908	0.943
l_1	Statistical	1.788e-12	2.084e-11	4.590e-10	0.600	0.822	0.910	0.951
	Analytical	1.927e-12	2.229e-11	4.796e-10	0.622	0.840	0.922	0.955
l_2	Statistical	1.035e-11	-3.478e-11	1.209e-10	0.601	0.782	0.903	0.958
	Analytical	9.284e-12	-3.076e-11	1.061e-10	0.583	0.770	0.892	0.949
l_3	Statistical	1.592e-10	-1.164e-10	8.859e-11	0.594	0.811	0.907	0.948
	Analytical	1.678e-10	-1.217e-10	9.174e-11	0.607	0.817	0.912	0.953

Table 6.3: *Comparing the statistical and analytical distributions of the lines shown in figure 6.1b (0.3 pixels of noise). See table 6.2 for explanation of terms.*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
m_1	Statistical	1.143e-06	-1.282e-07	6.814e-08	0.796	0.886	0.912	0.921
	Analytical	5.015e-06	-6.119e-07	1.812e-07	0.906	0.952	0.967	0.980
m_2	Statistical	2.741e-10	3.281e-11	7.129e-11	0.605	0.818	0.895	0.939
	Analytical	1.568e-10	1.658e-11	6.500e-11	0.502	0.713	0.829	0.886
m_3	Statistical	3.960e-10	4.688e-11	6.659e-10	0.605	0.816	0.895	0.936
	Analytical	3.753e-10	4.946e-11	7.640e-10	0.618	0.825	0.906	0.947
l_1	Statistical	2.405e-11	2.763e-10	5.656e-09	0.607	0.789	0.899	0.954
	Analytical	2.161e-11	2.427e-10	5.212e-09	0.586	0.776	0.882	0.939
l_2	Statistical	1.037e-10	-3.543e-10	1.258e-09	0.601	0.805	0.904	0.939
	Analytical	8.803e-11	-2.895e-10	9.983e-10	0.594	0.770	0.882	0.921
l_3	Statistical	1.886e-09	-1.378e-09	1.048e-09	0.612	0.800	0.904	0.956
	Analytical	1.832e-09	-1.334e-09	1.010e-09	0.603	0.794	0.899	0.950

Table 6.4: *Comparing the statistical and analytical distributions of the lines shown in figure 6.2 (1.0 pixel of noise). See table 6.2 for explanation of terms.*

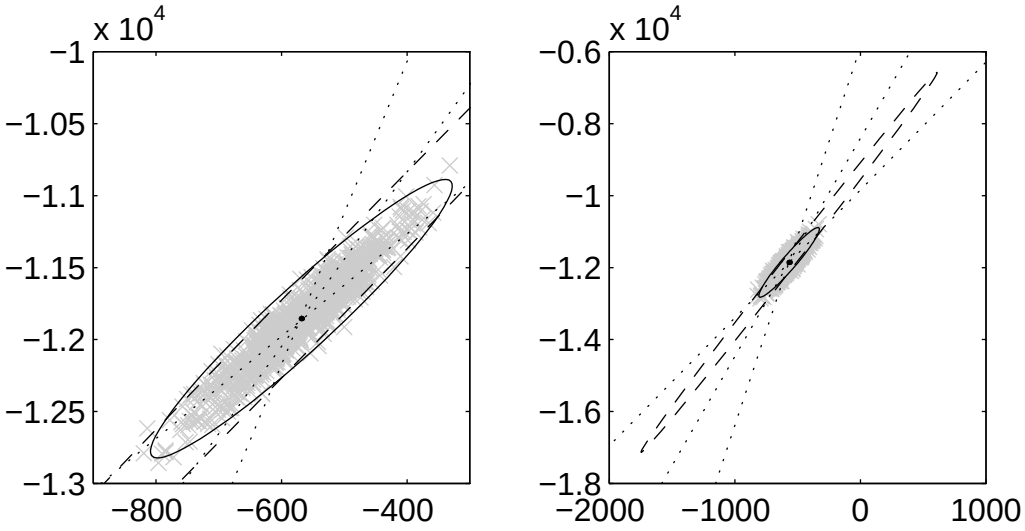


Figure 6.3: *The distribution of the vanishing points (×) computed from the lines shown in figure 6.1a (0.1 pixels of noise). The same results are shown at two different scales. The ellipses show the 3σ confidence limits for the statistical (solid) and analytical (dashed) covariances. The true position of the three intersecting lines are shown (dotted) and the true position of the intersection point is at (-572,-11876).*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
v	Statistical	6.427e+03	2.465e+04	1.040e+05	0.612	0.810	0.891	0.948
	Analytical	1.546e+05	6.909e+05	3.096e+06	0.667	0.724	0.763	0.793
l	Statistical	7.370e-14	-1.317e-13	4.911e-13	0.591	0.799	0.902	0.960
	Analytical	4.677e-14	-8.364e-14	3.572e-13	0.716	0.837	0.899	0.934

Table 6.5: *The statistics for the distribution of the vanishing points shown in figure 6.3 and the corresponding horizon line. The vanishing point and horizon line are scaled so that* $\mathbf{v} = (x, y, 1)^\top$ *and* $\mathbf{l} = (x, y, 1)^\top$ *to give the three covariance values of* σ_x^2, σ_{xy} *and* σ_y^2. *The remaining terms are explained in table 6.2.*

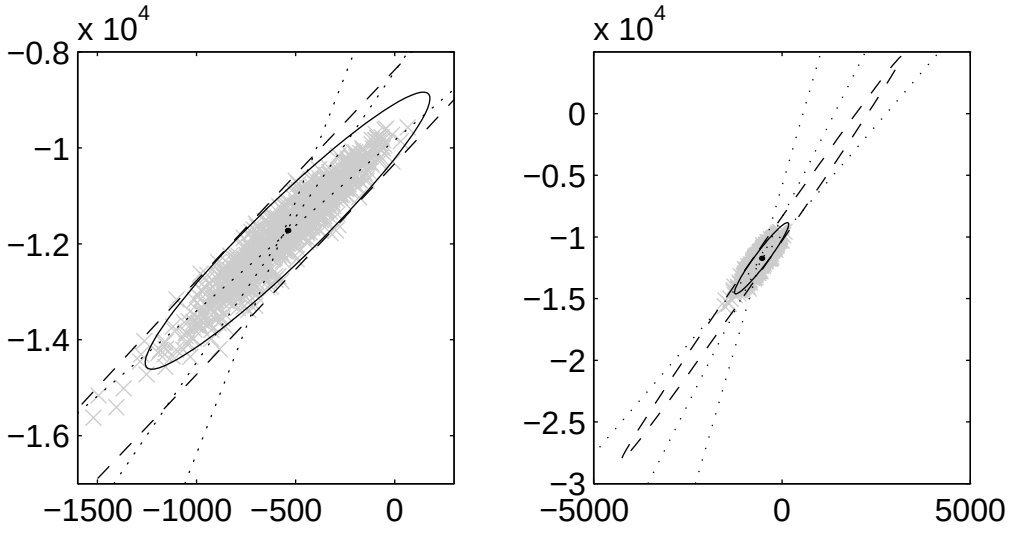


Figure 6.4: *The distribution of the vanishing points (×) computed from the lines shown in figure 6.1b (0.3 pixels of noise). The same results are shown at two different scales. The ellipses show the 3σ confidence limits for the statistical (solid) and analytical (dashed) covariances. The true position of the three intersecting lines are shown (dotted) and the true position of the intersection point is at (-572,-11876).*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
v	Statistical	5.742e+04	2.200e+05	9.265e+05	0.623	0.811	0.904	0.947
	Analytical	1.542e+06	6.776e+06	2.987e+07	0.592	0.599	0.607	0.618
l	Statistical	6.518e-13	-1.245e-12	4.822e-12	0.608	0.805	0.899	0.954
	Analytical	4.258e-13	-7.575e-13	3.212e-12	0.702	0.832	0.900	0.943

Table 6.6: *The statistics for the distribution of the vanishing points shown in figure 6.4 and the terms are explained in table 6.5.*

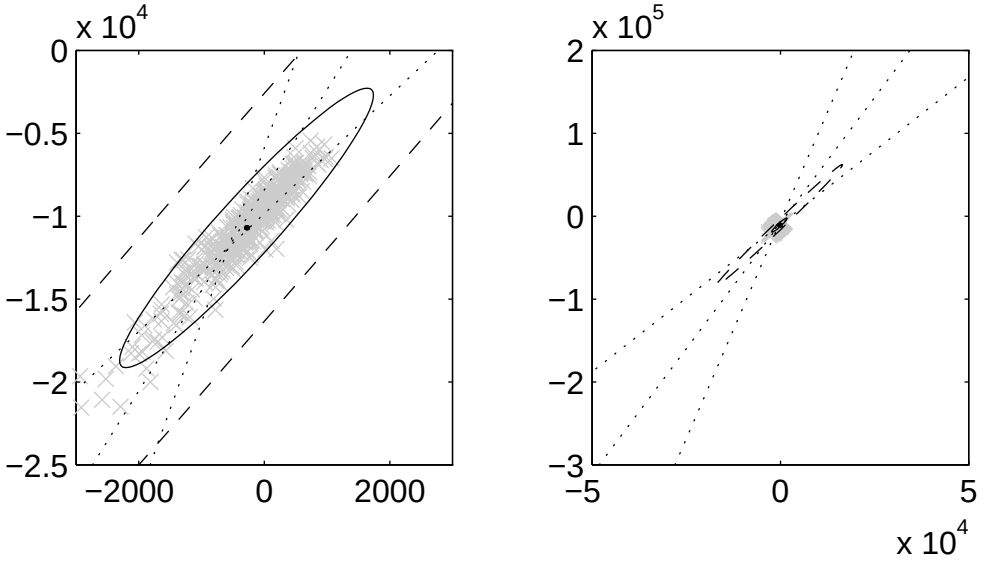


Figure 6.5: *The distribution of the vanishing points (×) computed from the lines shown in figure 6.2 (1.0 pixel of noise). The same results are shown at two different scales. The ellipses show the 3σ confidence limits for the statistical (solid) and analytical (dashed) covariances. The true position of the three intersecting lines are shown (dotted) and the true position of the intersection point is at (-572,-11876).*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
v	Statistical	4.546e+05	1.799e+06	7.897e+06	0.689	0.838	0.899	0.934
	Analytical	3.129e+07	1.359e+08	5.957e+08	0.496	0.504	0.504	0.513
l	Statistical	7.476e-12	-1.440e-11	5.419e-11	0.586	0.796	0.906	0.952
	Analytical	4.656e-12	-8.238e-12	3.496e-11	0.697	0.796	0.886	0.936

Table 6.7: *The statistics for the distribution of the vanishing points shown in figure 6.5 and the terms are explained in table 6.5.*

6.2.2 Batch Parameterisation

Rather than treat each fundamental matrix separately, the fact that parts of the different matrices are correlated can be utilised. The horizon lines $\mathbf{l}_i$ should be the same for every matrix, while the other image lines $\mathbf{m}_i$ should all intersect at the same point $\mathbf{v}$. This gives a *batch parameterisation* for the fundamental matrices for a set of images. All the image lines $\mathbf{l}_i$ are represented by the single image line $\mathbf{l}$, while the image lines $\mathbf{m}_i$ are represented by passing through the image point $\mathbf{v}$ with orientation ϕ_i. The other parameters of $\mathbf{f}_i$ remain unchanged.

For n images with p distinct fundamental matrices[1] the batch parameterisation is

$$\{\mathtt{F}_i\} \leftrightarrow \mathbf{f}_b(\mathbf{v}, \mathbf{l}, \theta_1, \phi_1, \rho_1, \ldots, \theta_p, \phi_p, \rho_p), \tag{6.6}$$

where the i^{th} matrix is a function of $(\mathbf{v}, \mathbf{l}, \theta_i, \phi_i, \rho_i)$, and this changes the number of parameters from $(6n)$ to $(4 + 3p)$.

Using the non-linear methods described in appendix A.3 and results of lemma C.1, and extending the epipolar error of equation (A.2) to p image pairs,

$$E_{ep} = \sum_{i=1}^{p} \sum_{j} \left(d(\hat{\mathbf{x}}'_{ij}, \mathtt{F}_i \hat{\mathbf{x}}_{ij})^2 + d(\hat{\mathbf{x}}_{ij}, \mathtt{F}_i^\top \hat{\mathbf{x}}'_{ij})^2 \right) \tag{6.7}$$

then the fundamental matrices and associated covariances can be computed simultaneously for n images. Now the position and covariance of $\mathbf{v}$ and $\mathbf{l}$ are available directly as

$$\begin{aligned} \{\mathbf{v}, \mathbf{l}\} &\subset \mathbf{f}_b, \\ \{\Lambda_{\mathbf{v}}, \Lambda_{\mathbf{l}}\} &\subset \Lambda_{\mathbf{f}_b}. \end{aligned} \tag{6.8}$$

Results

Three Monte Carlo simulations were completed with 500 runs with image noise of 0.1, 0.3, or 1.0 pixels added. The results are shown graphically in figures 6.6, 6.7, and 6.8,

[1] For n images there are $p = \mathcal{C}_2^n = (n^2 - n)/2$ distinct fundamental matrices.

and numerically in tables 6.8, 6.9, and 6.10. The results for the vanishing points are much closer to the true results than those obtained using just image pairs, with only a slight overestimation of the covariance along the dominant image line. The analytical covariance for the horizon line are slightly underestimated for both sets of results, but the actual covariance of the horizon line is very small compared to the true value of the line. Compare the line $\mathbf{l} = (\text{-2.0e-04}, \text{-2.5e-03}, 1.0)^\top$ with the statistical covariance, from table 6.9, $(\sigma^2_{l_1}, \sigma^2_{l_2}) = (\text{4.5e-13}, \text{3.4e-12})$).

6.2.3 Comparison of Images Pairs and Batch Methods

To compare the results of the image pair and batch algorithms, the same data is used to compute the distribution of solutions obtained from each method. The results are summarised in tables 6.11, 6.12, and 6.13 for three sequences with 500 runs and either 0.1, 0.3, or 1.0 pixels of added noise. These results show that more accurate results are obtained using the batch method but with disadvantage of increased computational expense.

6.2.4 Results for Real Image Sequences

Some results using sequence I from section 5.6 are shown in tables 6.14 and 6.17. These results show the computed vanishing point and horizon line with associated covariances for both the image pairs and batch algorithm. The batch algorithm is used with both image triplets (three images) and image quartets (four images). The actual values for the vanishing point are similar but the covariances are more realistic for the batch algorithm, and also are smaller for the quartet results which is as expected. The results for the horizon line are very similar for both algorithms.

More results using sequences IIb and IIc are given in tables 6.15–6.19. The results are given for triplets and quartets from the first four images of each sequence.

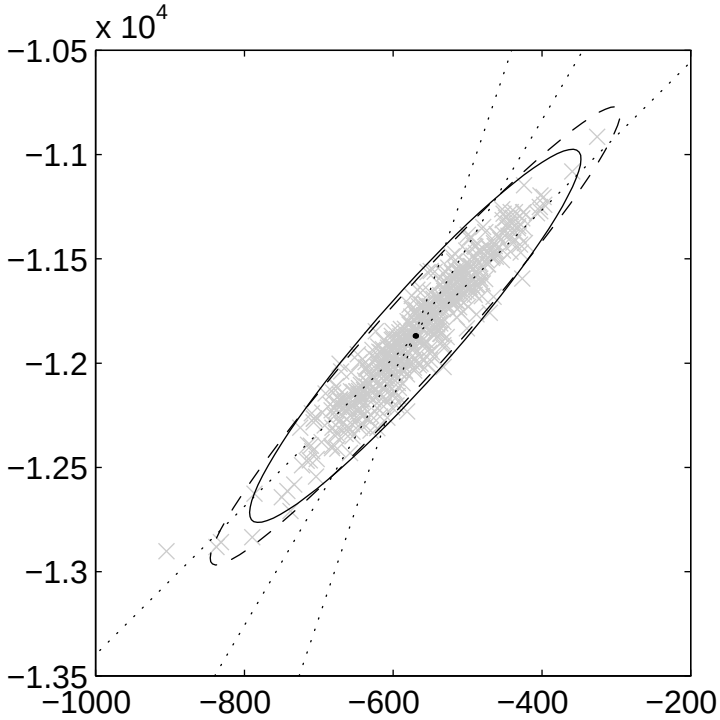


Figure 6.6: *The distribution of the vanishing points (×) computed using 500 runs of the batch algorithm with 0.1 pixels of added Gaussian noise. The ellipses show the 3σ confidence limits for the statistical (solid) and analytical (dashed) covariances. The true position of the three intersecting lines are shown (dotted).*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
v	Statistical	5.513e+03	2.099e+04	8.895e+04	0.627	0.794	0.898	0.956
	Analytical	8.445e+03	3.268e+04	1.341e+05	0.665	0.830	0.900	0.954
l	Statistical	5.868e-14	-8.737e-14	3.739e-13	0.609	0.796	0.896	0.940
	Analytical	3.686e-14	-5.923e-14	2.815e-13	0.493	0.661	0.790	0.866

Table 6.8: *The statistics of the distribution of vanishing points shown in figure 6.6 and the corresponding horizon lines (0.1 pixels of noise and the batch parameterisation). See table 6.5 for details of terms used.*

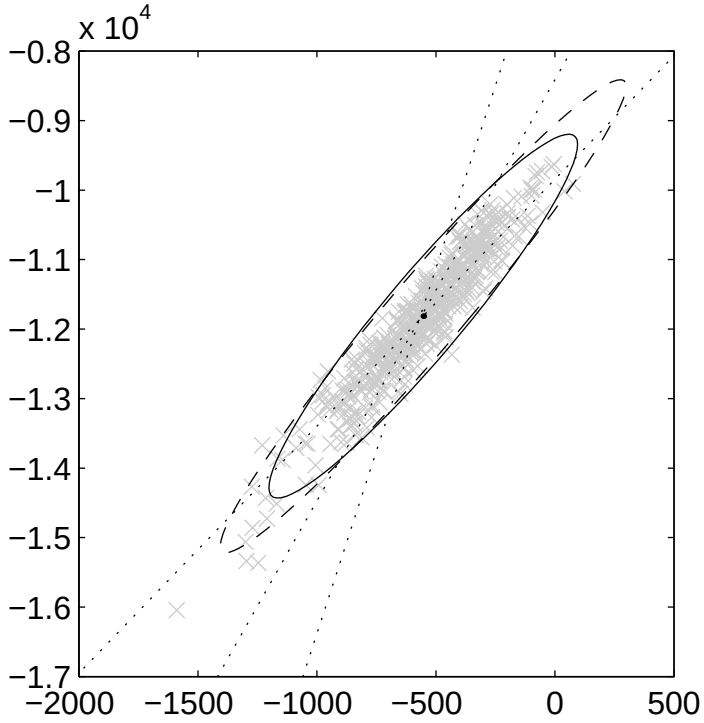


Figure 6.7: *The distribution of the vanishing points (×) computed using 500 runs of the batch algorithm with 0.3 pixels of added Gaussian noise. The ellipses show the 3σ confidence limits for the statistical (solid) and analytical (dashed) covariances. The true position of the three intersecting lines are shown (dotted).*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
v	Statistical	4.670e+04	1.778e+05	7.614e+05	0.626	0.804	0.900	0.947
	Analytical	8.074e+04	3.132e+05	1.287e+06	0.677	0.832	0.904	0.953
l	Statistical	4.498e-13	-6.867e-13	3.362e-12	0.581	0.787	0.902	0.953
	Analytical	3.437e-13	-5.573e-13	2.611e-12	0.502	0.681	0.819	0.894

Table 6.9: *The statistics of the distribution of vanishing points shown in figure 6.7 and the corresponding horizon lines (0.3 pixels of noise and the batch parameterisation). See table 6.5 for details of terms used.*

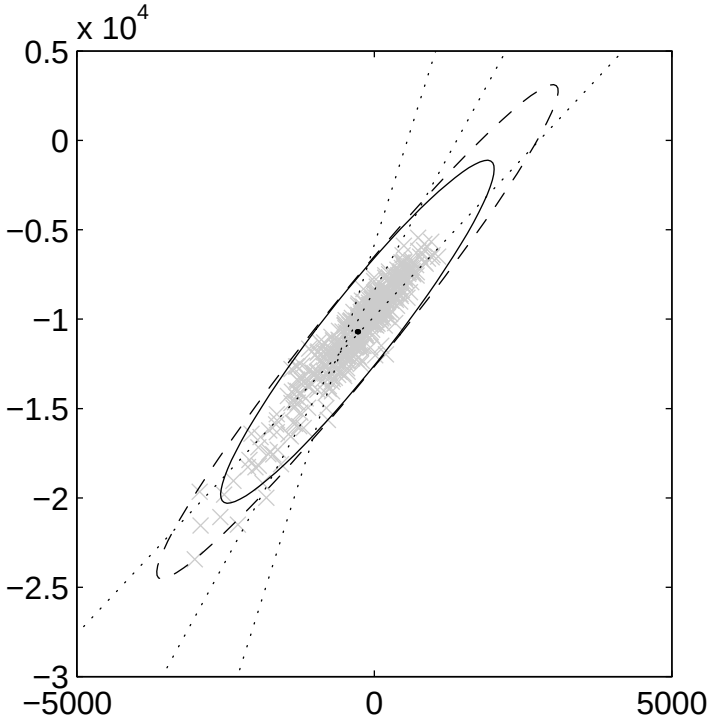


Figure 6.8: *The distribution of the vanishing points (×) computed using 500 runs of the batch algorithm with 1.0 pixel of added Gaussian noise. The ellipses show the 3σ confidence limits for the statistical (solid) and analytical (dashed) covariances. The true position of the three intersecting lines are shown (dotted).*

		Covariance			χ^2 Test			
		σ_x^2	σ_{xy}	σ_y^2	0.60	0.80	0.90	0.95
v	Statistical	5.869e+05	2.324e+06	1.023e+07	0.737	0.853	0.915	0.954
	Analytical	1.264e+06	5.050e+06	2.124e+07	0.788	0.908	0.941	0.972
l	Statistical	5.844e-12	-1.009e-11	4.301e-11	0.600	0.801	0.904	0.952
	Analytical	3.945e-12	-6.315e-12	2.964e-11	0.486	0.676	0.796	0.877

Table 6.10: *The statistics of the distribution of vanishing points shown in figure 6.8 and the corresponding horizon lines (1.0 pixel of noise and the batch parameterisation). See table 6.5 for details of terms used.*

	True	Pairs	Batch
v_x	-5.718e+02	-5.702e+02	-5.741e+02
v_y	-1.188e+04	-1.187e+04	-1.189e+04
$\sigma^2_{v_x}$	–	6.492e+03	5.513e+03
$\sigma^2_{v_y}$	–	1.047e+05	8.895e+04
$\sigma_{v_x v_y}$	–	2.482e+04	2.099e+04
l_x	-2.003e-04	-2.003e-04	-2.003e-04
l_y	-2.510e-03	-2.510e-03	-2.510e-03
$\sigma^2_{l_x}$	–	7.576e-14	5.868e-14
$\sigma^2_{l_y}$	–	4.692e-13	3.739e-13
$\sigma_{l_x l_y}$	–	-1.279e-13	-8.737e-14

Table 6.11: *Comparison of the distribution of the vanishing point and horizon line computed using the image pair algorithm and the batch algorithm, with 500 samples with 0.1 pixels noise added. The results for the batch algorithm have a smaller covariance than the image pair algorithm, with both means having a similar accuracy.*

	True	Pairs	Batch
v_x	-5.718e+02	-5.535e+02	-5.824e+02
v_y	-1.188e+04	-1.181e+04	-1.194e+04
$\sigma^2_{v_x}$	–	5.716e+04	4.670e+04
$\sigma^2_{v_y}$	–	9.378e+05	7.614e+05
$\sigma_{v_x v_y}$	–	2.203e+05	1.778e+05
l_x	-2.003e-04	-2.004e-04	-2.003e-04
l_y	-2.510e-03	-2.510e-03	-2.510e-03
$\sigma^2_{l_x}$	–	5.628e-13	4.498e-13
$\sigma^2_{l_y}$	–	4.461e-12	3.362e-12
$\sigma_{l_x l_y}$	–	-1.046e-12	-6.867e-13

Table 6.12: *Comparison of the distribution of the vanishing point and horizon line computed using the image pair algorithm and the batch algorithm, with 500 samples with 0.3 pixels noise added. The results for the batch algorithm have a smaller covariance and a mean value closer to the true value.*

	True	Pairs	Batch
v_x	-5.718e+02	-2.841e+02	-5.070e+02
v_y	-1.188e+04	-1.070e+04	-1.161e+04
$\sigma^2_{v_x}$	–	4.546e+05	5.869e+05
$\sigma^2_{v_y}$	–	7.897e+06	1.023e+07
$\sigma_{v_x v_y}$	–	1.799e+06	2.324e+06
l_x	-2.003e-04	-2.006e-04	-2.003e-04
l_y	-2.510e-03	-2.510e-03	-2.510e-03
$\sigma^2_{l_x}$	–	7.476e-12	5.844e-12
$\sigma^2_{l_y}$	–	5.419e-11	4.301e-11
$\sigma_{l_x l_y}$	–	-1.440e-11	-1.009e-11

Table 6.13: *Comparison of the distribution of the vanishing point and horizon line computed using the image pair algorithm and the batch algorithm, with 500 samples with 1.0 pixel noise added. The results for the batch algorithm have a smaller covariance and a mean value closer to the true value.*

Images	Algorithm	(v_x, v_y)	$\sigma^2_{v_x}$	$\sigma_{v_x v_y}$	$\sigma^2_{v_y}$
567	Pair	(597,2534)	2.93e+04	-1.91e+05	1.26e+06
	Batch	(665,2293)	1.03e+03	-2.73e+03	4.36e+04
678	Pair	(658,2382)	1.38e+03	5.59e+02	1.11e+05
	Batch	(541,2248)	3.74e+02	-1.10e+02	1.27e+04
5678	Pair	(518,2376)	2.97e+03	-2.45e+04	2.10e+05
	Batch	(474,2526)	3.35e+02	-1.12e+03	1.93e+04

Table 6.14: *The vanishing point computed for various image triplets and image quartets from sequence I using both the image pairs and batch algorithms. Both the vanishing point* $\mathbf{v} = (v_x, v_y)^\top$ *and its covariance are shown in the table.*

Images	Algorithm	(v_x, v_y)	$\sigma^2_{v_x}$	$\sigma_{v_x v_y}$	$\sigma^2_{v_y}$
123	Pair	(421,991)	2.57e+04	-2.13e+04	1.80e+04
	Batch	(430,994)	3.05e+03	-1.16e+03	7.73e+02
234	Pair	(-28,945)	1.05e+04	1.77e+03	1.80e+03
	Batch	(-60,936)	4.63e+03	-1.45e+02	3.00e+02
1234	Pair	(284,981)	2.73e+04	-2.01e+04	1.51e+04
	Batch	(248,1007)	2.15e+03	-1.12e+03	8.61e+02

Table 6.15: *The vanishing point computed for various image triplets and image quartets from sequence IIb using both the image pairs and batch algorithms. Both the vanishing point* $\mathbf{v} = (v_x, v_y)^\top$ *and its covariance are shown in the table.*

Images	Algorithm	(v_x, v_y)	$\sigma^2_{v_x}$	$\sigma_{v_x v_y}$	$\sigma^2_{v_y}$
123	Pair	(270,759)	2.56e+05	-1.33e+05	6.86e+04
	Batch	(232,797)	8.95e+02	-3.56e+02	2.97e+02
234	Pair	(137,731)	1.60e+04	2.46e+03	1.33e+03
	Batch	(146,743)	4.00e+03	7.35e+02	2.36e+02
1234	Pair	(301,721)	1.12e+05	-4.27e+04	1.63e+04
	Batch	(252,748)	8.00e+02	-3.25e+02	2.08e+02

Table 6.16: *The vanishing point computed for various image triplets and image quartets from sequence IIc using both the image pairs and batch algorithms. Both the vanishing point* $\mathbf{v} = (v_x, v_y)^\top$ *and its covariance are shown in the table.*

Images	Algorithm	(l_x, l_y)	$\sigma^2_{l_x}$	$\sigma_{l_x l_y}$	$\sigma^2_{l_y}$
567	Pair	(-4.21e-04,2.60e-03)	2.32e-10	3.50e-10	3.06e-09
	Batch	(-3.90e-04,2.68e-03)	1.15e-09	-1.58e-09	2.69e-09
678	Pair	(-5.02e-04,2.74e-03)	5.22e-10	4.54e-10	7.11e-09
	Batch	(-3.66e-04,3.31e-03)	8.22e-10	-9.74e-10	3.66e-09
5678	Pair	(-3.85e-04,3.03e-03)	1.71e-10	2.46e-10	1.62e-09
	Batch	(-3.46e-04,3.18e-03)	6.76e-10	-8.72e-10	2.20e-09

Table 6.17: *The horizon line computed for various image triplets and image quartets from sequence I using both the image pairs and batch algorithms. The horizon line is scaled such that* $\mathbf{l} = (l_x, l_y, 1)^\top$ *and both the line and its covariance are shown in the table.*

Images	Algorithm	(l_x, l_y)	$\sigma^2_{l_x}$	$\sigma_{l_x l_y}$	$\sigma^2_{l_y}$
123	Pair	(3.48e-05,1.27e-02)	6.03e-09	1.09e-08	3.07e-07
	Batch	(3.80e-05,1.29e-02)	4.17e-08	6.73e-08	1.16e-07
234	Pair	(3.48e-05,1.27e-02)	6.03e-09	1.09e-08	3.07e-07
	Batch	(3.80e-05,1.29e-02)	4.17e-08	6.73e-08	1.16e-07
1234	Pair	(3.21e-04,1.12e-02)	6.10e-09	1.77e-08	3.27e-07
	Batch	(3.25e-04,1.14e-02)	2.35e-08	3.91e-08	6.83e-08

Table 6.18: *The horizon line computed for various image triplets and image quartets from sequence IIb using both the image pairs and batch algorithms. The horizon line is scaled such that* $\mathbf{l} = (l_x, l_y, 1)^\top$ *and both the line and its covariance are shown in the table.*

Images	Algorithm	(l_x, l_y)	$\sigma^2_{l_x}$	$\sigma_{l_x l_y}$	$\sigma^2_{l_y}$
123	Pair	(2.00e-04,5.81e-03)	1.50e-08	1.44e-08	8.22e-08
	Batch	(2.25e-04,5.82e-03)	3.90e-08	2.64e-08	2.63e-08
234	Pair	(2.00e-04,5.81e-03)	1.50e-08	1.44e-08	8.22e-08
	Batch	(2.25e-04,5.82e-03)	3.90e-08	2.64e-08	2.63e-08
1234	Pair	(1.16e-04,4.47e-03)	1.21e-08	2.44e-08	1.29e-07
	Batch	(1.12e-04,4.58e-03)	1.84e-08	7.23e-09	9.46e-09

Table 6.19: *The horizon line computed for various image triplets and image quartets from sequence IIc using both the image pairs and batch algorithms. The horizon line is scaled such that* $\mathbf{l} = (l_x, l_y, 1)^\top$ *and both the line and its covariance are shown in the table.*

6.3 Trifocal Tensor

In chapter 5 the trifocal tensor was computed by the algorithm given in section 5.5, where the images are transformed to the normalised planar motion frame and the reduced trifocal tensor computed using the method given in appendix B.1. This is a linear solution which uses SVD to find the eigenvector corresponding to the minimum eigenvalue. The covariance of an eigenvector can be computed [82, 91, 98], assuming the image noise is homogeneous, isotropic, and Gaussian. However, section 6.3.1 shows that these assumptions do not hold for points in the normalised planar motion frame, and therefore the covariance of the solution cannot be computed using this method. The full tensor of 27 elements and its covariance could be computed using this method in the real frame, but the tensor is severely over-parameterised in this form, and Clarke [16] claims that the results obtained by this method are not accurate. Instead, non-linear methods are described in section 6.3.2, but current limitations are also described and these limitations preclude the use of a non-linear method.

6.3.1 The Canonical Frame is not Homogeneous

The canonical transformation (equation (5.30)) is used to transform the image points from the real frame to the normalised planar motion frame, and is a linear transformation of the homogeneous coordinates. However, the image noise is associated with the non-homogeneous coordinates, and when the canonical transformation is expressed in terms of non-homogeneous coordinates it becomes a non-linear function.

Appendix C discusses the propagation of uncertainty using the first order approximation to a non-linear function, and this can be applied to the noise on the image points. However, as will be shown below, the noise on the image points is neither isotropic nor homogeneous in the normalised planar motion frame. The following derives the covariance of a point in

the normalised planar motion frame using both a first and second order approximations.

The transformation of the image points from the real frame ($\mathbf{x}$) to the normalised planar motion frame ($\hat{\mathbf{x}}$) is expressed in homogeneous coordinates as

$$\hat{\mathbf{x}} = \mathtt{H}_c\mathbf{x}$$

where $\mathtt{H}_c$ is a function of the vanishing point $\mathbf{v} = (u, v, 1)^\top$ and the horizon line $\mathrm{l} = (l_x, l_y, l_z)^\top$. However, the line l is just used to compute a rigid transformation of the image (see appendix B.3) and this has no affect on the assumptions made about the image noise, namely it is homogeneous, isotropic and Gaussian, and so the effect of l does not need to be taken into account in the following analysis. Similarly, the u component of $\mathbf{v}$ is used to translate the image horizontally so that $u \rightarrow 0$, and hence also has no affect on the noise assumptions. So the transformation of non-homogeneous points can be expressed as

$$(\hat{x}, \hat{y})^\top = \mathbf{f}(\mathbf{a}), \tag{6.9}$$

where $\mathbf{a} = (x, y, v)^\top$. The values (x, y, v) are measured in the transformed image plane where $\mathrm{l} \rightarrow (0, 1, 0)^\top$ and $u \rightarrow 0$, but to avoid the proliferation terms the real image notation is used. So using only the projective part of the canonical transformation (part of $\mathtt{H}_2$ in appendix B.3) which involves v but neither l nor u, then the function $\mathbf{f}(\mathbf{a})$ is

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} x/(1 - y/v) \\ y/(1 - y/v) \end{pmatrix}, \tag{6.10}$$

and the covariance of $\mathbf{a}$ can be assumed to be

$$\Lambda_{\mathbf{a}} = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_v^2 \end{bmatrix}.$$

Using the first order approximation to the Taylor expansion of $\mathbf{f}(\mathbf{a})$ around the mean value $\bar{\mathbf{a}}$ (see equation (C.3)) gives

$$\mathbf{f}(\mathbf{a}) = \mathbf{f}(\bar{\mathbf{a}}) + \frac{\partial \mathbf{f}}{\partial \mathbf{a}}\boldsymbol{\Delta}\mathbf{a} + O(\| \boldsymbol{\Delta}\mathbf{a} \|^2).$$

The mean is transformed correctly as

$$\mathrm{E}[\hat{\mathbf{x}}] = \mathbf{f}(\bar{\mathbf{a}}), \tag{6.11}$$

while the covariance is given by

$$\Lambda_{\hat{\mathbf{x}}} = \frac{\partial \mathbf{f}}{\partial \mathbf{a}} \Lambda_{\mathbf{a}} \frac{\partial \mathbf{f}}{\partial \mathbf{a}}^{\top}. \tag{6.12}$$

The Jacobian $(\partial \mathbf{f}/\partial \mathbf{a})$ is not fixed, but is a function of the position of each point. Expanding equation (6.12) gives the first order approximation for the covariance as

$$\Lambda_{\hat{\mathbf{x}}} = \begin{bmatrix} \sigma_{\hat{x}}^2 & \sigma_{\hat{x}\hat{y}} \\ \sigma_{\hat{x}\hat{y}} & \sigma_{\hat{y}}^2 \end{bmatrix},$$

where

$$\begin{aligned}
\sigma_{\hat{x}}^2 &= \sigma_x^2 \frac{1}{(1-y/v)^2} + \sigma_y^2 \frac{(x/v)^2}{(1-y/v)^4} + \sigma_v^2 \frac{(x/v)^2(y/v)^2}{(1-y/v)^4}, \\
\sigma_{\hat{y}}^2 &= \sigma_y^2 \frac{1}{(1-y/v)^4} + \sigma_v^2 \frac{(y/v)^4}{(1-y/v)^4}, \\
\sigma_{\hat{x}\hat{y}} &= \sigma_y^2 \frac{(x/v)}{(1-y/v)^4} + \sigma_v^2 \frac{(x/v)(y/v)^3}{(1-y/v)^4}.
\end{aligned}$$

Careful examination of the above expressions show that in general the image noise in the normalised planar motion frame is no longer homogeneous, as the covariance of each point depends on the original position of the point (x, y), nor is it isotropic, as $\sigma_x^2 \neq \sigma_y^2$. Only with the gross assumption that the vanishing point is sufficiently distant in the y-direction does the noise revert to being homogeneous and isotropic.

When $v \gg y$ and $v \gg x$ then

$$\begin{aligned}
\sigma_{\hat{x}}^2 &\approx \sigma_x^2, \\
\sigma_{\hat{y}}^2 &\approx \sigma_y^2, \\
\sigma_{\hat{x}\hat{y}} &\approx 0.
\end{aligned}$$

However, extending the analysis to include second order terms of the Taylor expansion shows that the canonical transformation can introduce bias. The second order approximation is

$$\mathbf{f}(\mathbf{a}) = \mathbf{f}(\bar{\mathbf{a}}) + \frac{\partial \mathbf{f}}{\partial \mathbf{a}}\boldsymbol{\Delta}\mathbf{a} + \frac{1}{2!}\frac{\partial^2 \mathbf{f}}{\partial \mathbf{a}^2}\boldsymbol{\Delta}\mathbf{a}^2 + O(\| \boldsymbol{\Delta}\mathbf{a} \|^3). \tag{6.13}$$

The second order term of the Taylor series can be computed on a component-by-component basis as

$$\frac{1}{2!}\frac{\partial^2 \mathbf{f}}{\partial \mathbf{a}^2}\boldsymbol{\Delta}\mathbf{a}^2 = \frac{1}{2!}\left(\sum_{i=1}^{4}\Delta a_i \frac{\partial}{\partial a_i}\right)^2 \mathbf{f}(\mathbf{a})\Bigg|_{\mathbf{a}=\bar{\mathbf{a}}}.$$

The mean of the second order approximation is

$$\begin{aligned}
\mathrm{E}[\hat{x}] &= \frac{x}{(1-y/v)} + \sigma_y^2\frac{x}{(1-y/v)^3 v^2} + \sigma_v^2\frac{xy}{(1-y/v)^3 v^3}, \\
\mathrm{E}[\hat{y}] &= \frac{y}{(1-y/v)} + \sigma_y^2\frac{1}{(1-y/v)^3 v} + \sigma_v^2\frac{y^2}{(1-y/v)^3 v^3},
\end{aligned}$$

where the first term in each expression is the true value while the remaining terms are bias. However, the affect will only become noticeable when $y \approx v$ and σ_v^2 is large. The covariance of the second order approximation is given by

$$\begin{aligned}
\mathrm{E}\left[(\mathbf{f}(\mathbf{a}) - \mathbf{f}(\bar{\mathbf{a}}))^2\right] &= \mathrm{E}\left[\left(\frac{\partial \mathbf{f}}{\partial \mathbf{a}}\boldsymbol{\Delta}\mathbf{a} + \frac{1}{2!}\frac{\partial^2 \mathbf{f}}{\partial \mathbf{a}^2}\boldsymbol{\Delta}\mathbf{a}^2\right)^2\right] \\
&= \mathrm{E}\left[\left(\frac{\partial \mathbf{f}}{\partial \mathbf{a}}\boldsymbol{\Delta}\mathbf{a}\right)^2 + \left(\frac{1}{2!}\frac{\partial^2 \mathbf{f}}{\partial \mathbf{a}^2}\boldsymbol{\Delta}\mathbf{a}^2\right)^2\right],
\end{aligned}$$

which is the expected result for a second order approximation.

6.3.2 Non-linear Solution for the Trifocal Tensor

Any non-linear minimisation has three parts: the cost function to minimise, the parameters to minimises over, and the data used. When trying to derive a non-linear minimisation for the trifocal tensor each of these parts introduces problems and limitations.

The cost function should preferably minimise an error which is measured in the image frame, and using the trifocal tensor this can be associated with the transfer of points and

lines. Also, it is better if the cost function is unbiased and treats the measurement made in each image equally. This is not simple to achieve using the tensor, as the tensor uses one of the three images as the *reference* image, such that points are transferred from image one to three via two or from image one to two via three. This requirement for an unbiased cost function is analogous to the fundamental matrix which is estimated more accurately by minimising the epipolar error in both images rather than just one (see equation (A.2)). Currently an unbiased cost function has not been derived.

The data used in any minimisation is the position of the points matched over the image triplets. However, the positions can be measured in either the real frame or the normalised planar motion frame. The covariance of the solution is also required, but section 6.3.1 showed that the image noise in the normalised planar motion frame is neither homogeneous, isotropic, nor Gaussian, and this greatly increases the problem of computing a covariance when using the normalised planar motion frame. Similarly, the tensor can be parameterised in either frame, and it will be either be over-parameterised or the covariance will be difficult to compute.

The ideal solution to these problems will be to obtain a parameterisation of the tensor which includes the fixed points/lines as part of the parameterisation. This will avoid the use of the normalised planar motion frame and the associated problems. This is analogous to using the fundamental matrices to find the horizon line and vanishing point, where section 6.2 showed that better results are obtained using the batch parameterisation which uses the horizon line and vanishing point as actual parameters.

6.4 Circular Points and Camera Calibration

When the trifocal tensor is computed without explicitly using the circular points as parameters, the positions of the circular points are obtained by solving the cubic equation (see equation (5.32))

$$z^3 + b_2 z^2 + b_1 z + b_0 = 0,$$

where $\mathbf{b} = (b_2, b_1, b_0)^\top = \mathbf{f}(\mathbf{t})$, and $\mathbf{f}(\mathbf{t})$ is a non-linear function of the elements of the reduced trifocal tensor. The covariance of the solution of the cubic equation can be computed using a first order approximation to propagate the covariance of the trifocal tensor. The solution required from the cubic equation is the complex conjugate pair, which introduces the problem of computing the real and imaginary parts separately so that the covariance between them can be correctly computed. The complex conjugate solution of the cubic equation [1] is

$$z = -\frac{b_2}{3} - \frac{(s_1 + s_2)}{2} \pm \frac{i\sqrt{3}(s_1 - s_2)}{2},$$

where

$$\begin{aligned} s_1 &= (r + (q^3 + r^2)^{\frac{1}{2}})^{\frac{1}{3}}, \\ s_2 &= (r - (q^3 + r^2)^{\frac{1}{2}})^{\frac{1}{3}}, \\ r &= (b_1 b_2 - 3b_0)/6 - b_2^3/27, \\ q &= b_1/3 - b_2^2/9, \end{aligned}$$

and $(q^3 + r^2) > 0$ for the complex conjugate solution to exist.

Expressing the complex conjugate as $z = (p \pm qi)$ gives the position of the circular point in the normalised planar motion frame as $(-p \pm qi, 0, 1)^\top$, and using the inverse canonical transformation, the position of the circular points in the real frame are

$$\begin{aligned} \mathbf{i} &= \mathtt{H}_c^{-1}(-p + qi, 0, 1)^\top, \\ \mathbf{j} &= \mathtt{H}_c^{-1}(-p - qi, 0, 1)^\top. \end{aligned}$$

The canonical transformation is function of $\mathbf{v}$ and $\mathbf{l}$, hence the position of the circular points in the real frame can be expressed as

$$\mathbf{i} = \mathbf{f}(\mathbf{c}) = \mathbf{f}(\mathbf{t}^\top, \mathbf{v}^\top, \mathbf{l}^\top),$$

and the first order approximation of the covariance is

$$\Lambda_\mathbf{i} = \frac{d\mathbf{f}}{d\mathbf{c}} \Lambda_\mathbf{c} \frac{d\mathbf{f}}{d\mathbf{c}}^\top.$$

Now the position and covariance of all three fixed points is known. Section 5.2 shows how the image of the absolute conic is computed using the fixed points, and it is relatively simple to obtain a first order approximation so that the covariance of the estimate for ω can be computed. Also, when using the known aspect ratio as the extra constraint, it is possible to incorporate any uncertainty about the aspect ratio.

6.4.1 Results for Camera Calibration

As section 6.3 did not produce an algorithm to give an accurate estimate of the covariance of the trifocal tensor, it is not possible to predict the covariance of the calibration by propagating the image noise through the whole algorithm. However, figures 6.9 and 6.10 shows the distribution of the calibration parameters for 1000 runs of sequence SI with 0.1 and 0.3 pixels of noise. The results show a Gaussian distribution and suggest that it will be possible to obtain an accurate prediction of the covariance using a concatenation of first order approximations. The results are also given in table 6.20, which shows the empirical relationship between the image noise and the variance of the calibration.

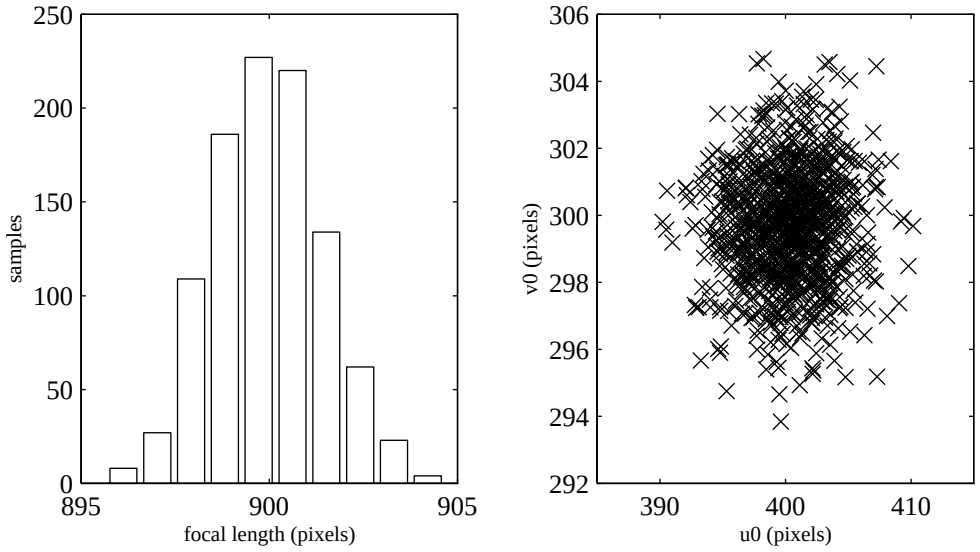


Figure 6.9: *The distribution of the calibration parameters computed using 1000 runs of sequence SI with 0.1 pixels of added noise. The extra constraint of known aspect ratio is used, and (a) shows the distribution of the focal length α_u, while (b) shows the distribution of the principal point (u_0, v_0).*

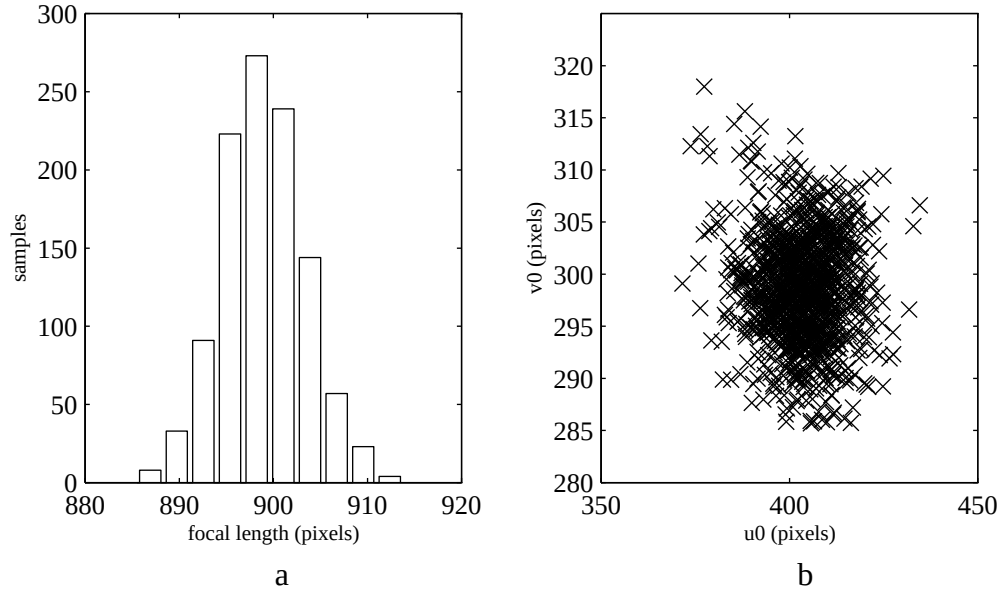


Figure 6.10: *The distribution of the calibration parameters computed using 1000 runs of sequence SI with 0.3 pixels of added noise. The extra constraint of known aspect ratio is used, and (a) shows the distribution of the focal length α_u, while (b) shows the distribution of the principal point (u_0, v_0).*

	Standard Deviation			
Image noise	α_u	u_0	v_0	k
0.1	1.5	3.2	1.7	4.5
0.3	4.4	9.0	5.2	12.5

Table 6.20: *The empirical relationship between the image noise (pixels) and the standard deviation of the distribution of the calibration parameters shown in figures 6.9 and 6.10.*

Summary

This chapter has attempted to estimate the covariance of the calibration parameters obtained using the fixed points of planar motion. It was shown that the horizon line and vanishing point can be obtained using either an image pair algorithm or a novel batch parameterisation, and that the latter gave better results. The noise in the normalised planar motion frame was shown to be anisotropic, non-homogeneous, and not Gaussian, which greatly increases the problem of computing the covariance of the circular points. The problems of computing the covariance of the trifocal tensor when using a non-linear minimisation were discussed.

Finally, the distribution of the calibration parameters obtained using a Monte Carlo simulation with a synthetic data were shown to be Gaussian. This suggests that it will be possible to compute an estimate of the uncertainty of the calibration parameters by propagating the image noise through the whole algorithm.

Chapter 7

Robust Object Tracking

Overview

The previous chapters have presented new algorithms for self-calibration from image sequences, but these chapters did not address the problem of the accurate tracking of objects over long sequences. This chapter increases the robustness and hence accuracy of a rigid object tracker by applying the ideas of stratification and utilising motion constraints. The goal is that in the future the tracking and self-calibration could be combined into a single framework, and this has been partially achieved by using a camera model which incorporates a changing focal length.

Many different approaches have been suggested for tracking moving objects, and these differ in the type of object being tracked. However, most methods are not robust to a number of ambient conditions, such as partial occlusions, which degrade performance. This chapter increases the robustness of the RAPiD tracker [34] to give the Robust RAPiD (RoRAPiD) [3].

The object being tracked is described at a number of levels, and at each level there is a model/hypothesis for which redundant measurements are available; this redundancy allows the hypothesis to be verified or refuted. Also included are several motion models which can improve the performance when the motion is constrained, and several camera models which are used to remove any possible error introduced by using the incorrect camera model. Both the motion models and camera models are constructed so that the

same general approach can be used throughout.

Section 7.2 is a review of RAPiD, then in section 7.3 the extended RoRAPiD is described in detail. Finally in section 7.4 results are given for real image sequences running on standard hardware at frame rate.

7.1 Object Tracking

Object tracking through image sequences is typically achieved by concentrating only on boundaries, such as occluding contours. This reduces the computational demands, and allows frame rate tracking on non-specialised hardware. A number of such methods have been demonstrated: Harris [34] and Lowe [57] track rigid objects with a known 3D model; snakes [11, 48] have been used to track image contours of deformable objects; while deformable templates [54] have been used for objects with known or constrained geometry.

However, the performance of many of the approaches are severely degraded by a number of ambient conditions such as partial occlusions, photometric changes (e.g., shadows), camera modelling errors, and incorrect feature matching. The objective here is to make tracking robust to these conditions by extending the model of the object and utilising known constraints such as restricted motion models and/or different camera models.

7.1.1 Stratifying Object Tracking

All the methods of tracking mentioned above represent the object as a single set of points, lines, or boundaries, which are processed simultaneously. Generally, none of them describe an object as a set of lower order components which still have some *complexity* which can be utilised. This is another use of the idea of stratifying a problem into smaller, more soluble parts. Here the object is described at a number of levels, and at each level there is a model/hypothesis for which redundant measurements are available. These redundant

measurements allow the hypothesis to be verified or refuted, and also allow the identification and removal of outliers. Outliers in this context are grossly incorrect measurements or associations, whose errors have a very large adverse affect on the accuracy of subsequent calculations [91].

The object is described by a set of related **geometric primitives** (hereafter called primitives). Primitives have a known geometry (e.g., lines and conics) which is the *complexity* mentioned above. At a low level the primitives are associated with a set of high contrast edges, and are used to reject outlying edgels measured in the image. At a high level, the primitives are associated with the object pose, and are used to reject outlying model-image associations. The robust methods are applicable to several different methods of tracking, but are demonstrated here by extending the RAPiD tracker of Harris [34].

7.2 RAPiD Tracker

The RAPiD tracker represents a 3D object as a set of control points which lie on high contrast edges. The pose (position and orientation) of the object is estimated and the object outline tracked at field rate (50Hz) on general purpose hardware. The cycle of the RAPiD algorithm is given in the algorithm summary 7.1. The algorithm is split into two parts, the first dealing with making measurements in the image, and the second calculating the new pose of the object.

7.2.1 The RAPiD Approach

Measurements

The control points are projected onto the image using the predicted pose of the object obtained from the Kalman filter [47]. For each control point, a 1D search is then executed to find the strongest image gradient (edgel) in the vicinity of the control point, which

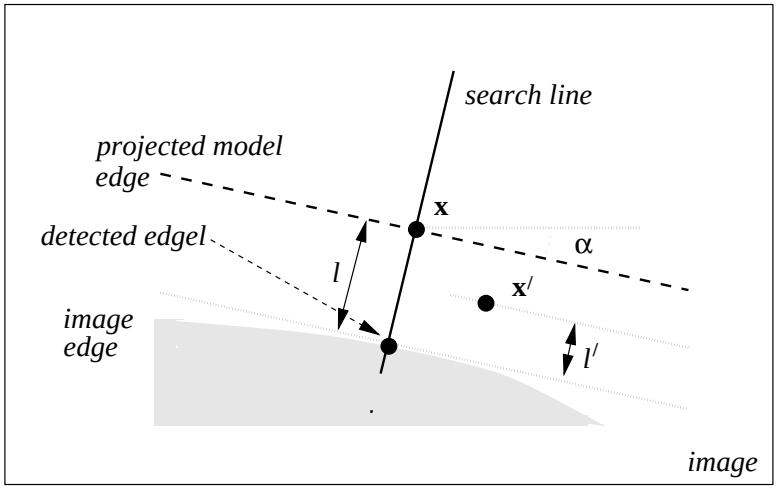


Figure 7.1: *Measurements made by RAPiD at a single control point. The control point (***x***) lies on a projected model edge which is inclined at an angle* α*. A search is carried out perpendicular to the projected edge to find the strongest image edgel, and the distance is given by* l*. When a pose correction is added, the new position of the control point (***x**′*) is now at a distance* l' *from the detected image edgel. See text for more details of the role of* **x**′*,* l'*, and* α*.*

is assumed to be the new position of the edge. The search used is a 1D Canny edge detection [15], and the search direction is perpendicular to the projected model outline in the image[1] (see figures 7.1 and 7.2).

Pose Update

When correcting the pose of the object, it is assumed that the pose differs only by a small translation and a small rotation from the predicted position in 3D. The projection of the object onto the image plane is linearised about the predicted pose. The small pose correction can then be calculated directly using the position of the control points and the distance to the corresponding image edgel.

The position of the control points can be measured in three different coordinate frames:

[1] To avoid interpolating between image pixels, the search direction is either vertical, horizontal, or diagonal, and the measured distance to the edgel is corrected for the actual orientation of the model outline, see [34] for more details.

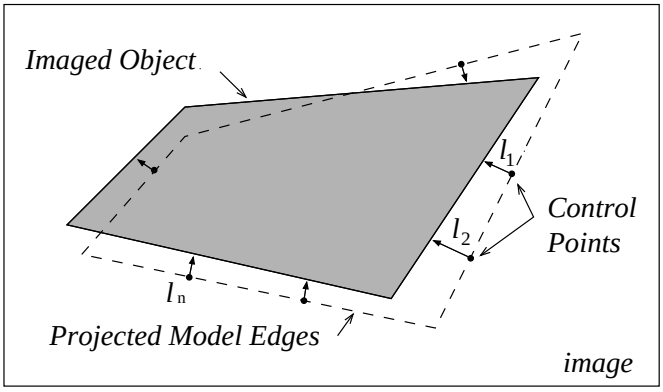


Figure 7.2: *Measurements made by RAPiD, for a set of control points on an object, to correct object pose. Each distance l_i partially constrains the value of the small pose correction* $\mathbf{q}$*, and used together allow* $\mathbf{q}$ *to be computed.*

the model frame ($\mathbf{X}^m = (X^m, Y^m, Z^m)^\top$), the camera-centred frame ($\mathbf{X} = (X, Y, Z)^\top$), and the normalised[2] image frame ($\mathbf{x} = (x, y)^\top$). The pose of the object ($\mathtt{R}$ and $\mathbf{T}$) determines the transformation between the model and camera-centred frames

$$\mathbf{X} = \mathtt{R}\mathbf{X}^m + \mathbf{T}, \tag{7.1}$$

while the transformation between the camera frame and the normalised image frame is a simple projection

$$\mathbf{x} = (x, y)^\top = (X/Z, Y/Z)^\top. \tag{7.2}$$

The actual pose of the object is assumed to differ from the predicted pose by a small rotation about the origin of the model frame ($\mathbf{\Theta} = (\Theta_x, \Theta_y, \Theta_z)^\top$), and a small translation ($\mathbf{\Delta T} = (\Delta T_x, \Delta T_y, \Delta T_z)^\top$). This moves the position (in the camera frame) of the point $\mathbf{X}$ to $\mathbf{X}'$

$$\mathbf{X}' = \mathtt{R}\mathbf{X}^m + \mathbf{T} + \mathbf{\Delta T} + \mathbf{\Theta} \times \mathtt{R}\mathbf{X}^m. \tag{7.3}$$

To simplify the notation, in the following the rotated model coordinates are used ($\hat{\mathbf{X}}^m = \mathtt{R}\mathbf{X}^m = (\hat{X}^m, \hat{Y}^m, \hat{Z}^m)^\top$). Both $\mathbf{X}$ and $\mathbf{X}'$ are projected onto the normalised image plane

[2]The normalised image frame removes the affect of the known camera calibration, so point $\mathbf{x}_r$ measured in the real image is transformed to the point $\mathbf{x}_n = \mathtt{C}^{-1}\mathbf{x}_r$ in the normalised frame.

by equation (7.2) to give the points $\mathbf{x}$ and $\mathbf{x}'$ respectively, and expanding up to first order terms gives

$$x' = x + \frac{\Delta T_x + \Theta_y \hat{Z}^m - \Theta_z \hat{Y}^m - x(\Delta T_z + \Theta_x \hat{Y}^m - \Theta_y \hat{X}^m)}{T_z + \hat{Z}^m}, \tag{7.4}$$

$$y' = y + \frac{\Delta T_y + \Theta_z \hat{X}^m - \Theta_x \hat{Z}^m - y(\Delta T_z + \Theta_x \hat{Y}^m - \Theta_y \hat{X}^m)}{T_z + \hat{Z}^m}. \tag{7.5}$$

Expressing the small changes as a six vector $\mathbf{q} = (\mathbf{\Theta}^\top, \mathbf{\Delta T}^\top)^\top$, equations (7.4) and (7.5) can be given in vector form as

$$\mathbf{x}' = \mathbf{x} + (\mathbf{q}^\top \mathbf{a}, \mathbf{q}^\top \mathbf{b})^\top, \tag{7.6}$$

where

$$\begin{aligned} \mathbf{a} &= (-x\hat{Y}^m, x\hat{X}^m + \hat{Z}^m, -\hat{Y}^m, 1, 0, -x)^\top / (T_z + \hat{Z}^m), \\ \mathbf{b} &= (-y\hat{Y}^m - \hat{Z}^m, y\hat{X}^m, \hat{X}^m, 0, 1, -y)^\top / (T_z + \hat{Z}^m). \end{aligned} \tag{7.7}$$

The perpendicular distance of the new position of the control point to the image edgel (see figure 7.1) is

$$l' = l + \mathbf{q}^\top \mathbf{a} \sin\alpha - \mathbf{q}^\top \mathbf{b} \cos\alpha = l + \mathbf{q}^\top \mathbf{c}, \tag{7.8}$$

where $\mathbf{c} = \mathbf{a}\sin\alpha - \mathbf{b}\cos\alpha$. The sum of the squared perpendicular distance from each control point to their respective image edgel gives E, a measure of the error for a given pose correction,

$$E = \sum_i (l_i + \mathbf{q}^\top \mathbf{c}_i)^2, \tag{7.9}$$

and the pose correction which minimises this error is the solution to

$$\sum_i \mathbf{c}_i \mathbf{c}_i^\top \mathbf{q} = -\sum_i l_i \mathbf{c}_i, \tag{7.10}$$

which can be found using standard linear algebra techniques [86]. The pose correction $\mathbf{q}$ can then be used to update the pose of the object.

- Predict the position of the object.
- For each control point, find the strongest image edgel, perpendicular to the projected model outline.
- Calculate pose correction $\mathbf{q}$ using equations (7.7)–(7.10).
- Update the position of the object using a Kalman filter and $\mathbf{q}$.

Algorithm 7.1: *The cycle of the original RAPiD.*

Tracking

The pose of the object is tracked over time using a Kalman filter. The pose consists of the translation 3-vector $\mathbf{T}$ (see equation (7.1)) and the rotation encoded[3] as a 3-vector $\mathbf{R}$. A constant velocity model is assumed for the object, so that there are 12 state variables (6 for pose and 6 for pose velocity) giving the state vector

$$\mathbf{x} = (\mathbf{T}, \mathbf{R}, \dot{\mathbf{T}}, \dot{\mathbf{R}})^\top.$$

Only the pose of the object is measured at each time step giving the measurement vector

$$\mathbf{z} = (\hat{\mathbf{T}}, \hat{\mathbf{R}})^\top,$$

where $\hat{\mathbf{T}}$ and $\hat{\mathbf{R}}$ are the measured pose. The vectors $\mathbf{x}$ and $\mathbf{z}$ have a covariance of $\mathtt{P}$ and $\mathtt{S}$ respectively.

The Kalman filter operates as follows. Given the pose of the object at time $t-1$ as $\mathbf{x}_{t-1}$, then $\mathbf{x}'_t$ is the estimated pose at time t,

$$\mathbf{x}'_t = \mathtt{F}\mathbf{x}_{t-1}$$

[3]The rotation can be encoded by several different formulations (e.g., Euler angles, quaternions, etc.) but here the angle-axis form is used.

where $\mathtt{F}$ is the state transition model, which for the constant velocity model is

$$\mathtt{F} = \begin{bmatrix} \mathtt{I} & \mathtt{I} \\ \mathtt{0} & \mathtt{I} \end{bmatrix}.$$

The covariance of the state vector increases due to uncertainty of evolution as

$$\mathtt{P}'_t = \mathtt{F}\mathtt{P}_{t-1}\mathtt{F}^\top + \mathtt{Q}_t$$

where $\mathtt{Q}$ is the process noise covariance.

At time t the pose of the object is measured as $\mathbf{z}_t$ and the updated state vector $\mathbf{x}_t$ is computed as

$$\mathbf{x}_t = \mathbf{x}'_t + \mathtt{K}(\mathbf{z}_t - \mathtt{H}\mathbf{x}'_t),$$

where $\mathtt{H} = [\mathtt{I} \mid \mathtt{0}]$ is the observation matrix and $\mathtt{K}$ is the Kalman gain matrix,

$$\mathtt{K} = \mathtt{P}'_t\mathtt{H}^\top(\mathtt{H}\mathtt{P}'_t\mathtt{H}^\top + \mathtt{S}_t)^{-1}.$$

Similarly, the state covariance is updated as

$$\mathtt{P}_t = \mathtt{P}'_t - \mathtt{K}\mathtt{H}\mathtt{P}'_t.$$

More details of the values for the error covariances $\mathtt{Q}$ and $\mathtt{S}$ can be found in [21, 34].

7.2.2 Limitations of RAPiD

RAPiD has a number of limitations:

- At no point are incorrect edgels identified and eliminated. Incorrect edgels arise from shadows, errors in pose causing alignment with erroneous edges, occlusions, or texture on the object itself or in the background.
- The stability of control points (how often/accurately they are found) is not utilised in any calculation.

- The control points are treated individually throughout the algorithm, without taking into account that several control points are often placed on the same edge, and hence there is an association between their positions in the image.
- Only the calibrated perspective camera model is used.
- Only a general motion model is used.

These problems are addressed in the following section.

7.3 RoRAPiD: A Robust Tracker

This section describes a robust object tracker (RoRAPiD) [3], an extension to the RAPiD tracker, which removes the limitations listed above. The cycle of the RoRAPiD is given in the algorithm summary 7.2.

7.3.1 Robust Detection

The object model is considered both at *low* and *high* levels. The low levels contains the primitives, which consists of the high contrast edges with known geometry (e.g., line or conic). At the high level, these primitives are related to give a full description of the object. This gives a model/hypothesis at each level:

Low Level The model is the known geometry of the primitive, while the hypothesis is that the set of detected edgels are constrained to have this geometry.

High Level The model is the relative positions of the primitives in the object and the hypothesis is that the detected primitives are constrained by this relative positioning.

Using redundant measurements at each level allows verification of the models and the elimination of outlying measurements. The robust methods used for each level are described below.

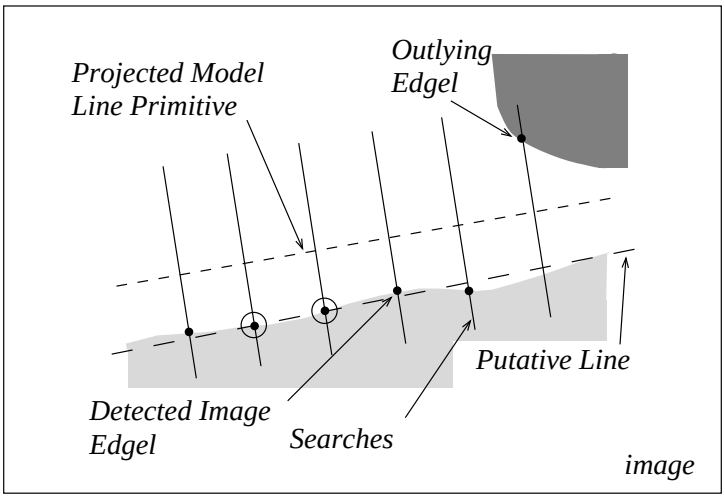


Figure 7.3: *Detection of outlying edgels for a line primitive. The outlying edgel does not support the chosen putative line, which is fitted through the 2 circled edgels.*

Robust Low Level Detection

A mathematical model is known for each type of primitive (line, conic, etc). Sufficient control points are placed on each primitive to ensure redundant measurements (e.g., > 2 control points on a line, > 5 control points on a conic). A RANSAC [27] methodology is used to detect outliers among the edgels detected by each control point. After the elimination of outliers, if the number of remaining edgels falls below a threshold the primitive is not included in the pose update.

RANSAC

The RANSAC algorithm is the opposite to conventional least-squares techniques where as much data as possible is used to obtain a solution. Instead, as small a subset as is feasible is used to estimate the parameters, and the support for this solution measured. For example, in the computation of a line from several edgels (see figure 7.3), putative lines are estimated using random samples of 2 edgels. The distance from the putative line to each edgel is then calculated, and if it is below a threshold that edgel is deemed to support the line. The

process is repeated a set number of times, and the putative line with the most support is the one finally adopted. Outliers are edgels that do not support the putative line chosen. After the elimination of outliers, the position of the line is then re-estimated using a least-squares fit to the remaining (inlying) edgels.

Robust High Level Detection

The pose update is robustly calculated using the remaining primitives, and outlying primitives are detected and eliminated. A typical outlying primitive is shown in figure 7.4. Most of the primitives will have been found correctly, so rather than use a RANSAC methodology, case deletion is used to detect the outlying primitives. Each primitive is deleted in turn, and the pose correction is calculated from the remaining primitives to give a putative corrected pose. A projection error is measured for each putative corrected pose, and if the largest error is significant, the primitive deleted in the calculation of that putative pose is considered outlying and hence eliminated.

The projection error is the image distance between the projected and measured primitive. It is measured only for the primitive deleted in the pose computation (similar to the Cross-Validation method). If a primitive has been incorrectly identified in the image (i.e., an incorrect image-model association) the projection error will be high.

Primitive Stability

The stability of the primitives will vary over time, due to the position of the object, lighting conditions, etc. Consequently, the confidence attached to a particular primitive should also vary. This confidence is computed as a decaying average of the frequency with which the primitive is correctly found. This confidence is then used to weight the effect of the primitive's control points in the subsequent pose correction calculation. The decaying

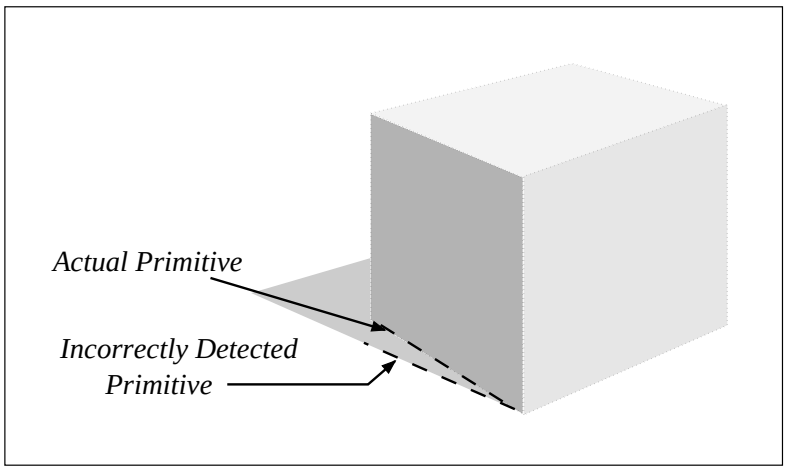


Figure 7.4: *Detection of an incorrect primitive. The lighting condition results in a strong shadow near the predicted position of a line and causes the detection of an incorrect primitive.*

average of the stability is calculated using the binary value $found(t)$, which is true when the control point is found at time t, and used to give the weight $w_i(t)$ for the i^{th} control point at time t as

$$w_i(t) = (found(t) + w_i(t-1))/2.$$

This weight is then used in adapting equations (7.9) and (7.10) to give

$$E = \sum_i w_i (l_i + \mathbf{q}^\top \mathbf{c}_i)^2, \tag{7.11}$$

and

$$\sum_i w_i \mathbf{c}_i \mathbf{c}_i^\top \mathbf{q} = -\sum_i w_i l_i \mathbf{c}_i, \tag{7.12}$$

7.3.2 Camera Models

The original RAPiD used a calibrated perspective camera which is described in section 2.1.2. However, in some circumstances other camera models are more appropriate. The general structure of the tracker is independent of the camera model used, the only differences being the pose parameters and the object-to-image projection. This means that $\mathbf{q}$ contains different

- Predict the position of the Object
- **For** each visible Primitive
  Calculate position of Control Points
  For each Control Point in Primitive
  Find the strongest edgel in image
  Eliminate outlying edgels in Primitive using RANSAC
  If *number of edgels* $>$ *number of edgels threshold*: then
  Primitive is found
- Correct pose using surviving Primitives/Control Points weighted according to stability using equation (7.12).
- **For** each surviving Primitive
  Delete Primitive, recalculate pose correction, and measure projection error
  If largest projection error is significant: then eliminate Primitive
- Update Kalman Filter

Algorithm 7.2 : *The cycle of the Robust Object Tracker RoRAPiD. The values of the thresholds used are given in section 7.4.*

small pose correction parameters, and that **c** is computed using a different expression which is dependent on the object-to-image projection. However, the same equation (7.12) is used to compute the pose correction which minimises equation (7.11).

Weak Perspective Camera Model

The weak perspective camera, described in section 2.1.3, is an useful approximation to the perspective camera [72], which still has 6 external (pose) parameters but the focal length is combined with the average depth of the object to give the scale factor.

From equation (2.6), the projection of a point $(\mathbf{X}^m)$ from the model coordinate frame

to the normalised image plane[4] is given by

$$\mathbf{x} = s \begin{bmatrix} \mathbf{R}_1^\top \\ \mathbf{R}_2^\top \end{bmatrix} \mathbf{X}^m + \mathbf{t} \tag{7.13}$$

where s is the scaling, and the 2-vector $\mathbf{t}$ is the position of the image of the origin of the model coordinate frame. Following the approach of section 7.2, the pose correction can be encoded as

$$\mathbf{q} = (\boldsymbol{\Theta}^\top, \boldsymbol{\Delta}\mathbf{t}^\top, \Delta s)^\top \tag{7.14}$$

where $\boldsymbol{\Theta}$ is the small rotation, $\boldsymbol{\Delta}\mathbf{t}$ is the small change in $\mathbf{t}$, and Δs the small change in the scaling s. Incorporating the small pose change gives the new position $\mathbf{x}'$ as

$$\mathbf{x}' = (s + \Delta s) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} (\hat{\mathbf{X}}^m + \boldsymbol{\Theta} \times \hat{\mathbf{X}}^m) + \mathbf{t} + \boldsymbol{\Delta}\mathbf{t},$$

where points in the model coordinate frame are rotated to correspond to the orientation of camera-centred coordinate frame

$$\hat{\mathbf{X}}^m = \mathtt{R}\mathbf{X}^m.$$

The different world-to-image projection give the following expressions for $\mathbf{a}$ and $\mathbf{b}$ and hence $\mathbf{c}$ (see equations (7.8)–(7.10))

$$\begin{aligned} \mathbf{a} &= (0, s\hat{Z}^m, -s\hat{Y}^m, 1, 0, \hat{X}^m)^\top, \\ \mathbf{b} &= (-s\hat{Z}^m, 0, s\hat{X}^m, 0, 1, \hat{Y}^m)^\top, \\ \mathbf{c} &= \mathbf{a}\sin\alpha - \mathbf{b}\cos\alpha. \end{aligned} \tag{7.15}$$

Affine Camera Model

The affine camera is the uncalibrated version of the weak perspective camera, described in section 2.1.3, where the aspect ratio is unknown and the projection matrices have 8

[4]The use of the normalised image plane for a weak perspective camera removes the aspect ratio (ζ) from equation (2.6).

degrees of freedom. The camera model transforms directly from the model coordinates to the image, and from equation (2.8)

$$\mathbf{x} = \mathtt{M}\mathbf{X}^m + \mathbf{m}. \tag{7.16}$$

The small pose change is now the small change in the elements of $\mathtt{M}$ and $\mathbf{m}$, $\Delta\mathtt{M}$ and $\mathbf{\Delta m}$ respectively, which gives

$$\mathbf{q} = (\Delta M_{11}, \Delta M_{12}, \ldots, \Delta M_{2,3}, \Delta m_1, \Delta m_2)^\top.$$

The new position $\mathbf{x}'$ is now

$$\mathbf{x}' = (\mathtt{M} + \Delta\mathtt{M})\mathbf{X}^m + \mathbf{m} + \mathbf{\Delta m}.$$

This gives a very simple expression for $\mathbf{a}$ and $\mathbf{b}$, and hence $\mathbf{c}$

$$\begin{aligned} \mathbf{a} &= (X^m, Y^m, Z^m, 0, 0, 0, 1, 0)^\top, \\ \mathbf{b} &= (0, 0, 0, X^m, Y^m, Z^m, 0, 1)^\top, \\ \mathbf{c} &= \mathbf{a}\sin\alpha - \mathbf{b}\cos\alpha. \end{aligned} \tag{7.17}$$

When using an affine camera only an affine model of the object is required.

Perspective Camera with Zoom

If camera calibration changes during operation, it is usually confined to a change in the focal length (i.e., zooming in or out of the scene). When using a metric model of an object and the perspective camera[5], it is possible to track changes in the focal length in addition to the changing pose of the object. The projection from camera-centred coordinates onto the image is an adaptation of equations (2.1) and (2.2). The image points are now measured

[5]The weak perspective camera model allows a changing focal length as it is incorporated into the scale factor.

in a *quasi-normalised* frame where the effects of the aspect ratio and the position of the principal point are removed, but not the effect of the focal length. The camera calibration matrix (2.4) can be decomposed into two parts

$$\mathtt{C} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \zeta\alpha_u & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & u_0 \\ 0 & \zeta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_u & 0 & 0 \\ 0 & \alpha_u & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

which, combined with equation (7.2), give the new camera-to-image frame projection for the *quasi-normalised* image frame

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \alpha_u X/Z \\ \alpha_u Y/Z \end{pmatrix}. \tag{7.18}$$

The pose correction $\mathbf{q}$ now contains seven elements, whose seventh element is the small change in the focal length Δ_{α_u}. Following the same steps as before gives the position of the corrected image point as

$$x' = x + \frac{\Delta_x + \Theta_y\hat{Z}^m - \Theta_z\hat{Y}^m - x(\Delta_z + \Theta_x\hat{Y}^m - \Theta_y\hat{X}^m)}{T_z + \hat{Z}^m} + \Delta_{\alpha_u}x, \tag{7.19}$$

$$y' = y + \frac{\Delta_y + \Theta_z\hat{X}^m - \Theta_x\hat{Z}^m - y(\Delta_z + \Theta_x\hat{Y}^m - \Theta_y\hat{X}^m)}{T_z + \hat{Z}^m} + \Delta_{\alpha_u}y. \tag{7.20}$$

Hence, $\mathbf{a}$ and $\mathbf{b}$ are now seven vectors containing

$$\begin{aligned} \mathbf{a} &= (-x\hat{Y}^m, x\hat{X}^m + \hat{Z}^m, -\hat{Y}^m, 1, 0, -x, x(T_z + \hat{Z}^m))^\top/(T_z + \hat{Z}^m), \\ \mathbf{b} &= (-y\hat{Y}^m - \hat{Z}^m, y\hat{X}^m, \hat{X}^m, 0, 1, -y, y(T_z + \hat{Z}^m))^\top/(T_z + \hat{Z}^m), \\ \mathbf{c} &= \mathbf{a}\sin\alpha - \mathbf{b}\cos\alpha. \end{aligned} \tag{7.21}$$

So when tracking a metric model any changes in the focal length will be detected and tracked over time.

However, the perspective-camera-with-zoom model can be unstable. This occurs when the perspective effects are small. Examining equation (7.21), the final two elements of $\mathbf{a}$ and $\mathbf{b}$, corresponding to the small change in depth ΔT_z and the small change in focal length

Δ_{α_u} respectively, have the ratio of $(T_z + \hat{Z}^m)$. In an argument analogous to that used for affine imaging conditions [82], if the depth of the object (the difference in the values of $\hat{Z}^m$) is small compared to the depth of the object T_z the ratio of the final two elements of $\mathbf{c}$ is fixed for all points on the object. Hence, the matrix $\sum c_i c_i^\top$ in equation (7.12) is singular and the values of the small change in depth and small change in focal length are unconstrained, which introduces instability.

7.3.3 Motion Models

The robust tracker and RAPiD assume a general 3D motion model with 6 dof's. However, in many situations a more restricted motion occurs. A number of restricted motion models have been implemented, including (1) pure translation, and (2) planar motion. In all cases, the change in pose has fewer parameters than the 6 used in general 3D motion, which results in improved performance from the Kalman filter.

Pure Translation The algorithm used for the pure translation motion model is a simple adaptation of the general model in which only the small translation is corrected for, and only the translation is tracked. Similarly a pure rotation model can be used but this scenario is less likely to occur when tracking objects — with the exception of security cameras — and has the problem of defining the centre of rotation.

Planar Motion Planar motion occurs when the translation is confined to a plane and the rotation is around a fixed axis which is perpendicular to the plane of translation. Planar motion is a very common when tracking objects for traffic scene analysis, such as vehicles on roads or at junctions [28, 50, 88] or aeroplanes at an airport [87]. However, the planar motion model is not a simple adaptation of the general motion model used above, and the required equations are derived below.

The planar motion model has three degrees of freedom: the angle of rotation about the fixed axis, and the two degrees of planar translation. The transformation between model and camera-centred coordinate frames, equation (7.1), is altered so that the rotation is split into two parts. First, the rotation between the model frame and camera frame $\mathtt{R}_f$ which remains fixed throughout the motion, and then the rotation $\mathtt{R}_a$ around a fixed axis $\mathbf{a} = (a_x, a_y, a_z)^\top$

$$\mathbf{X} = \mathtt{R}_a \mathtt{R}_f \mathbf{X}^m + \mathbf{T}. \tag{7.22}$$

The axis of rotation $\mathbf{a}$ is perpendicular to the plane, but measured in the camera-centred frame, and the single degree of freedom for the rotation is the angle of rotation about this axis. The fixed axis rotation matrix $\mathtt{R}_a$ is parameterised using the angle-axis form with axis $\mathbf{a}$ and angle θ

$$\mathtt{R}_a = \begin{bmatrix} a_x^2(1-\cos\theta)+\cos\theta & a_x a_y(1-\cos\theta)-a_z\sin\theta & a_x a_z(1-\cos\theta)+a_y\sin\theta \\ a_x a_y(1-\cos\theta)+a_z\sin\theta & a_y^2(1-\cos\theta)+\cos\theta & a_y a_z(1-\cos\theta)-a_x\sin\theta \\ a_x a_z(1-\cos\theta)-a_y\sin\theta & a_y a_z(1-\cos\theta)+a_x\sin\theta & a_z^2(1-\cos\theta)+\cos\theta \end{bmatrix},$$

which, when θ is small, simplifies to

$$\mathtt{R}_a \approx \begin{bmatrix} 1 & -a_z\theta & a_y\theta \\ a_z\theta & 1 & -a_x\theta \\ -a_y\theta & a_x\theta & 1 \end{bmatrix}.$$

The translation is confined to a plane and has only two degrees of freedom, and so two perpendicular directions are defined in the plane of translation ($\mathbf{D}_1$ and $\mathbf{D}_2$), measured in the camera frame, and these form the coordinate frame for the small translation.

The small pose variation is now given as a single small angle θ and the small planar translation (α, β), which gives the corrected position, equation (7.3), as

$$\mathbf{X}' = \mathtt{R}_a \mathtt{R}_f \mathbf{X}^m + \mathbf{T} + \alpha\mathbf{D}_1 + \beta\mathbf{D}_2 + \theta\mathbf{a} \times \mathtt{R}_a \mathtt{R}_f \mathbf{X}^m. \tag{7.23}$$

Following the steps of section 7.2.1 gives the corrected image positions, equations (7.4) and (7.5), as

$$x' = x + \frac{\alpha D_{1_x} + \beta D_{2_x} + \theta(a_y\hat{Z}^m - a_z\hat{Y}^m) - x(\alpha D_{1_z} + \beta D_{2_z} + \theta(a_x\hat{Y}^m - a_y\hat{X}^m))}{T_z + \hat{Z}^m},$$

$$y' = y + \frac{\alpha D_{1_y} + \beta D_{2_y} + \theta(a_z \hat{X}^m - a_x \hat{Z}^m) - y(\alpha D_{1_z} + \beta D_{2_z} + \theta(a_x \hat{Y}^m - a_y \hat{X}^m))}{T_z + \hat{Z}^m},$$

where $\hat{\mathbf{X}} = \mathtt{R}_a \mathtt{R}_f \mathbf{X}$. The pose correction vector has the three elements $\mathbf{q} = (\theta, \alpha, \beta)^\top$, and the remaining vectors $\mathbf{a}$ and $\mathbf{b}$ are simple to derive as

$$\begin{aligned}
\mathbf{a} &= \left(a_y \hat{Z}^m - a_z \hat{Y}^m - x(a_x \hat{Y}^m - a_y \hat{X}^m), D_{1_x} - x D_{1_z}, D_{2_x} - x D_{2_z}\right)^\top / (T_z + \hat{Z}^m), \\
\mathbf{b} &= \left(a_z \hat{X}^m - a_x \hat{Z}^m - y(a_x \hat{Y}^m - a_y \hat{X}^m), D_{1_y} - y D_{1_z}, D_{2_y} - y D_{2_z}\right)^\top / (T_z + \hat{Z}^m), \\
\mathbf{c} &= \mathbf{a} \sin\alpha - \mathbf{b} \cos\alpha, \qquad (7.24)
\end{aligned}$$

which gives the new form of the pose correction equation (7.10).

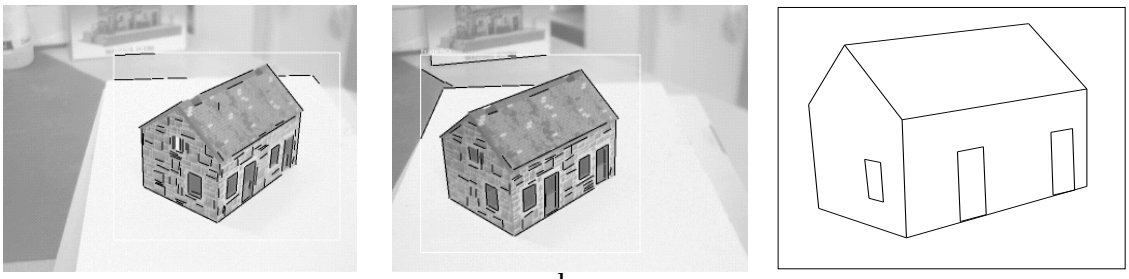
The axis of rotation and plane of translation need to be known *a priori* for the planar motion model. These values can learned using the general motion model to track an object known to be moving under planar motion, then fitting a plane to the path of translation.

7.3.4 Object Models

Model Acquisition

A model is constructed from a set of primitives, which are the high contrast edges associated with surface creases, occluding contours, or surface markings. Currently the primitives have to have a known geometry (e.g., line, conic), to allow the placing of control points along their length, and the detection of outliers. Also defined in the model are the faces (opaque surfaces) which control which primitives are visible in the image, thus allowing hidden primitive removal. The faces are defined as areas enclosed by one or more primitives.

Currently, models are obtained using two views of the object, where the camera is translating, to compute an affine model (see section 4.1.1). Then the known camera calibration is used to upgrade from affine to a metric model using equation (2.16). Figure 7.5 shows two such images, with the lines found in the image highlighted, and the actual wire frame model constructed.



a b c

Figure 7.5: *Obtaining a model of an object: (a),(b) the two images used with the camera translating between views, and (c) the wire frame model. Note the internal (texture) primitives.*

Hidden Primitive Removal

To detect which primitives are hidden (self-occluded) from a particular view, the primitives and faces are ordered according to distance from the camera. Then each primitive is checked to see if any, or all, of it is occluded by a face between itself and the camera. For a calibrated camera this can be done off-line for all possible view directions, and the visibility of each primitive stored in a table.

7.4 Results for RoRAPiD

Implementation In RoRAPiD there are several variables which are chosen heuristically and the values used are listed below. Six (10) control points are placed on each line (conic) primitive with four (7) being the threshold for the elimination of the primitive (the *number of edgel threshold* in algorithm 7.2). The threshold for an edgel to support a putative primitive is 2 pixels. The search area for the edgel at each primitive varies between 8 and 24 pixels, and increases when the edgel is not found and decreases when the edgel found is significantly closer to the projected primitive than the size of search area.

When the number of control points used is less than 50 (i.e., 10 line primitive with 5 control points on each line) RoRAPiD can run at 50Hz on a SUN IPX workstation [3] with

an S2200 image board used purely for the image input/output. However, the results below were obtained with the tracker running at frame rate on either a SUN ULTRA1 workstation or SGI workstation.

Robustness When tracking an object in a simple, uncluttered environment, the original RAPiD generally works as well as our robust tracker. However, as soon as the object becomes partly occluded (see figure 7.6), the original RAPiD fails and starts to track a phantom object. Figure 7.7 shows the variation of pose parameters for both trackers, on a sequence where the object becomes partially occluded. RAPiD fails, while the robust tracker continues to track correctly until the object moves out of view of the camera.

Figure 7.8 shows another sequence when the object is rotating, and as a result the parts of the object which are self-occluded changes over the sequence. The performance of the tracker is not degraded by the hand occluding part of the object.

Camera Models Figure 7.11 shows two images from a sequence where the focal length of the camera is set incorrectly. Using the perspective-camera-with-zoom model, the focal length is corrected during the sequence. Figure 7.12 shows the changing focal length over the sequence from the initial 400 pixels to the correct 540 pixels.

When the focal length is changing and the perspective effects are small, the perspective-camera-with-zoom model can be unstable (see section 7.3.2), and it is better to use the weak perspective camera model where the depth of the object and the focal length are combined into the scale factor. Figure 7.13 shows such a sequence where the focal length is changed from 12mm (1500 pixels) to 50mm (6000 pixels) over the sequence while the object is moving away from the camera. The resulting change in the scale factor is shown in figure 7.14.

The affine camera model is less stable than the other models, because the two extra

degrees of freedom, which remove the rigid motion constraint, mean that quite often the primitives found do not fully constrain the pose correction (i.e., excessive skew is produced). This problem could be alleviated by have more primitives defined on the object, which should result in enough being found to constrain the pose correction.

Motion Models Figure 7.9 shows images from the sequence used to compare the general motion model and translation motion model. The object in the sequence is moving under pure translation and was tracked using both motion models. Figure 7.10 shows the motion parameters being tracked over the sequence, and also the 95% confidence limits for each parameter. The confidence limits are computed using the covariance from the Kalman filter. The restricted motion model tracks the parameters slightly better than the general model.

Summary

This chapter described the extension of a rigid object tracker to make it more robust to ambient conditions such as partial occlusions, photometric changes, etc. The object model is described at two levels, both of which have redundant measurements thus allowing the detection of outliers. Several different camera models were used to reduce the affect of errors introduced by an incorrect camera model, and also extending the use of the tracker to only partially calibrated cameras. Several different motion models were described as better results can be obtained when using the restricted motion models rather than the general full motion model. Results were given for several real images sequences with the tracker running at frame rate on standard hardware.

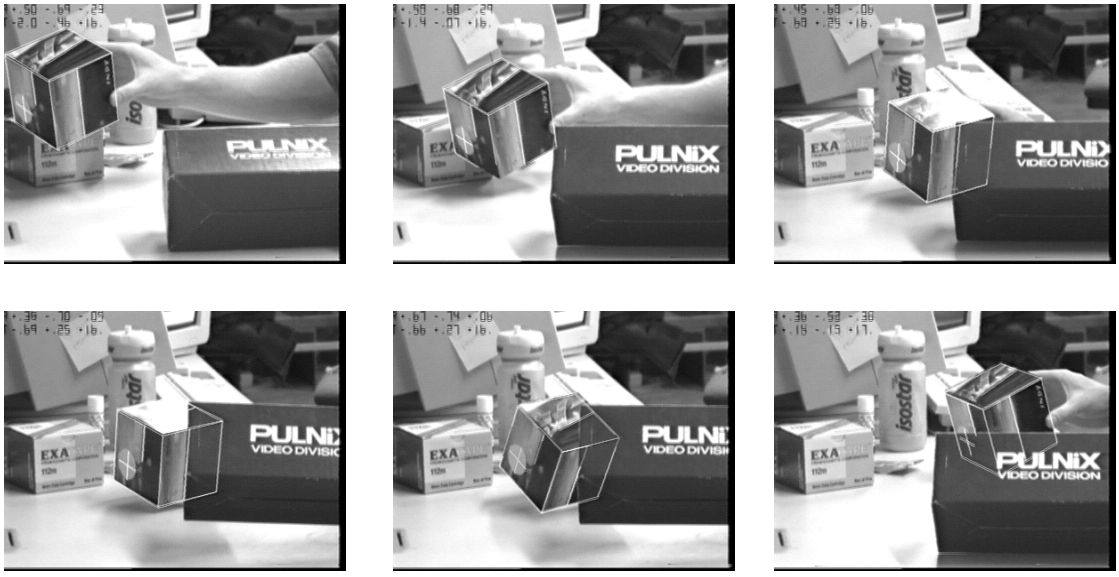


Figure 7.6: *Images from a sequence when the object is partially occluded, but the robust tracker remains locked onto the object. Note, both internal (texture) and external (occluding) boundaries are tracked. The cross marks a tracked ellipse.*

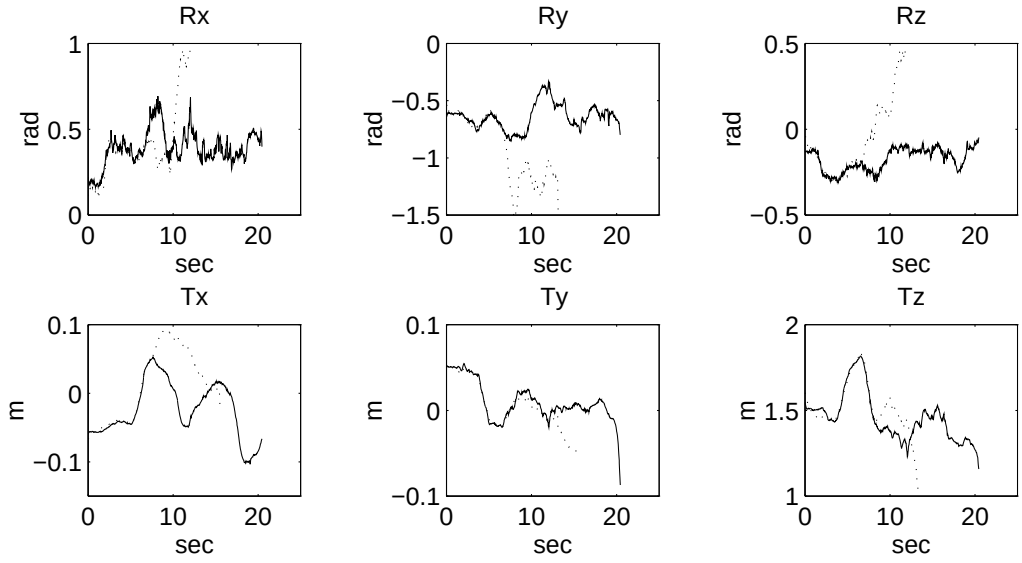


Figure 7.7: *Object Pose Parameters for different trackers using the same image sequence: (1) RoRAPiD (solid), (2) RAPiD (dotted). The pose parameters are 3 for rotation of the object (given in the angle/axis form), and 3 for the translation from the camera to object coordinate frames. The object is partially occluded after 8 sec., causing RAPiD to fail, while RoRAPiD continues to track correctly until, after 20s, the object moves out of view of the camera.*

Figure 7.8: *Images from a sequence when the object is rotating and the parts of the object which are self-occluded changes mid-sequence. Also, the hand occludes part of the object during the sequence but the performance of the tracker is not affected.*

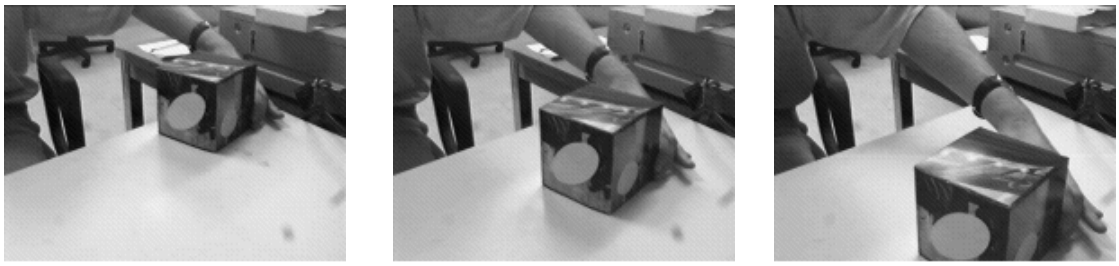


Figure 7.9: *Images from the sequence where the object is only translating, and moving towards the camera.*

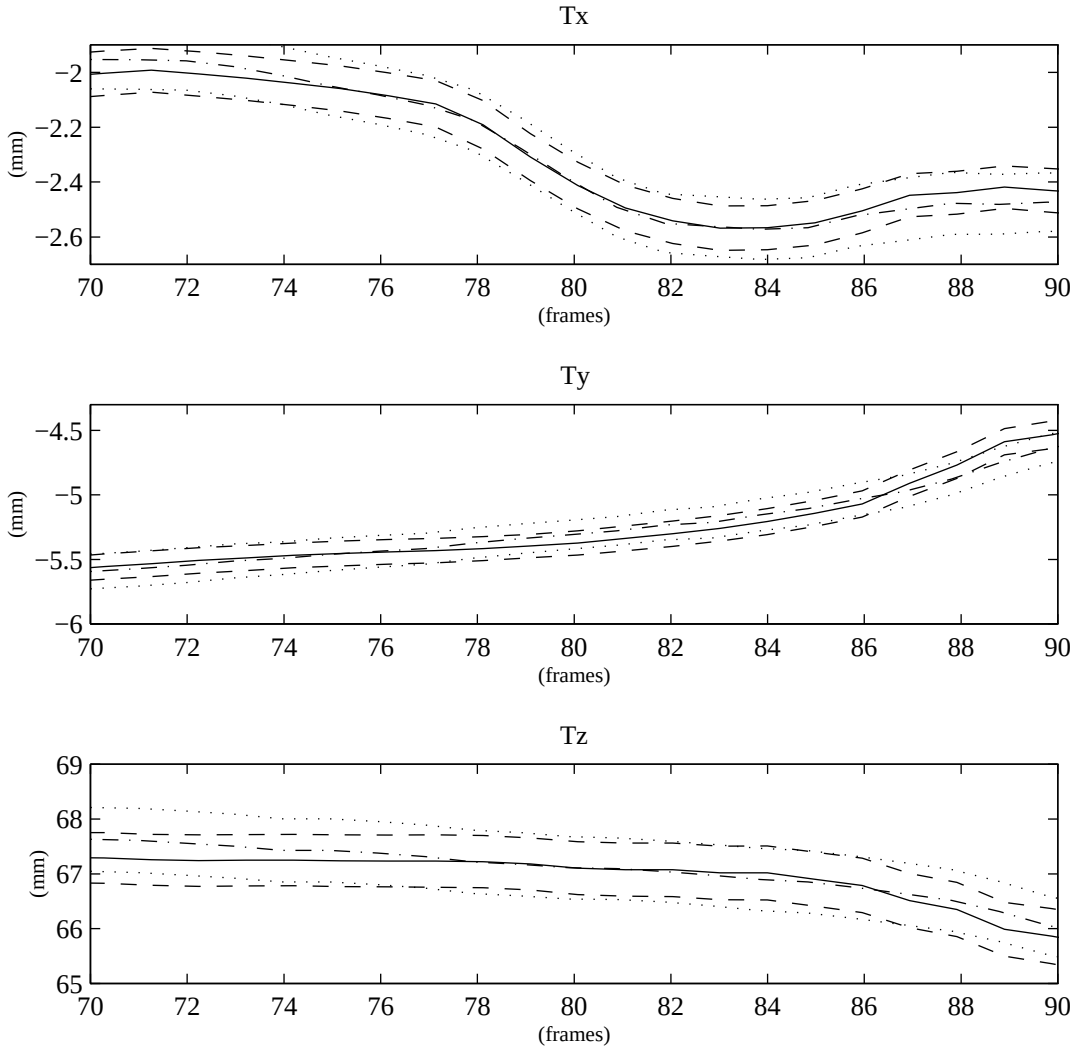


Figure 7.10: *The comparison of the results for the full motion model and the pure translation model when the object being tracked is only translating (see figure 7.9). The three translation parameters are shown for a sequence of 20 frames from the 200 frames the object is tracked over. The different lines are: solid - translational model parameters, dashed - translational model 95% confidence limits, dot/dashed - full model parameters, and dotted - full model 95% confidence limits. The confidence limits for the translational model are smaller than those of the full model.*

a b

Figure 7.11: *Perspective Camera with Zoom Model: The (a) first and (b)* 75^{th} *images from a sequence where the incorrect initial focal length of 400 pixels changes to the correct value of 550 pixels. The pose parameters and focal length are shown in the top left hand corner of the images. Note how in image (b) the projected outline fits the object unlike image (a).*

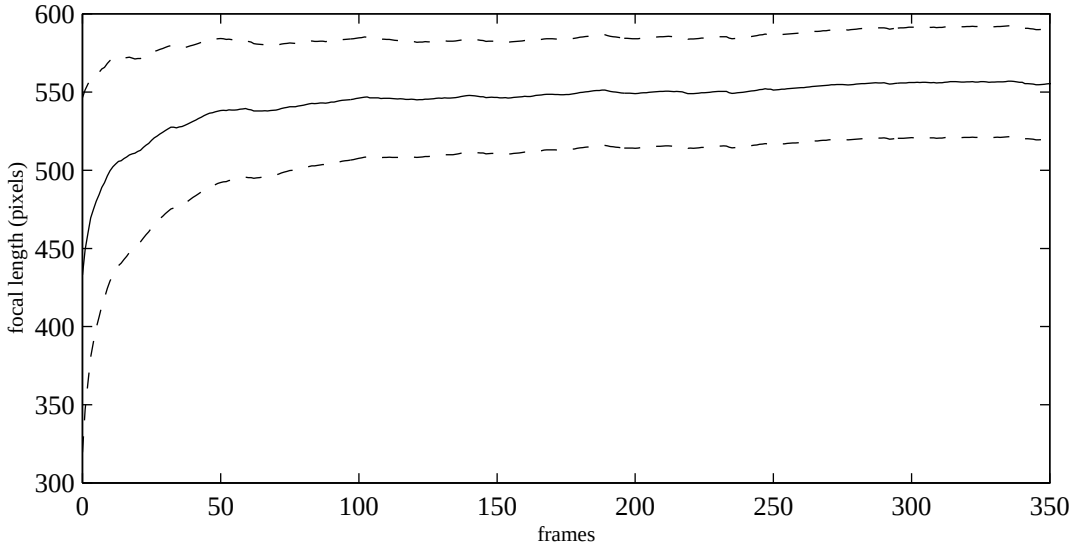


Figure 7.12: *Perspective Camera with Zoom Model: Graph showing changing focal length from the sequence shown in figure 7.11. The focal length (solid) converges from the initial value of 400 pixels to 555 pixels after 350 frames, with the* 1σ *confidence limits shown dashed.*

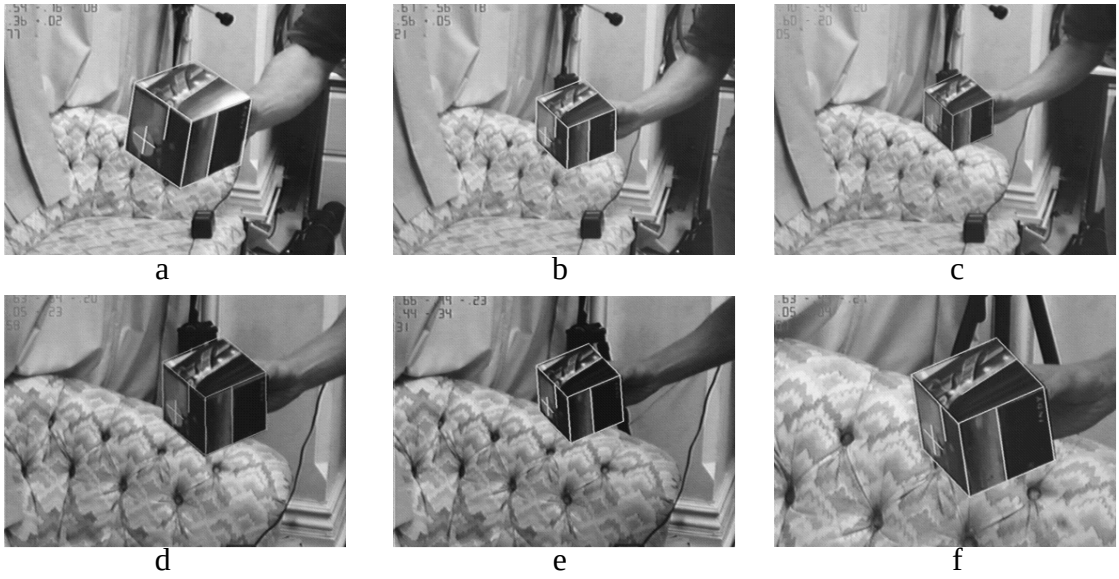


Figure 7.13: *Weak Perspective Camera Model: Several images from a sequence where the object is moving away from the camera and the focal length is increasing. Images (a), (b) and (c) show the object moving away from the camera over frames 1–400 when the focal length is fixed, then the object remains still while the camera zooms in until frame 580 (d), then again the focal length is fixed and the object moves away until frame 700 (e), and finally the camera zooms with the object still moving until frame 1100 (f).*

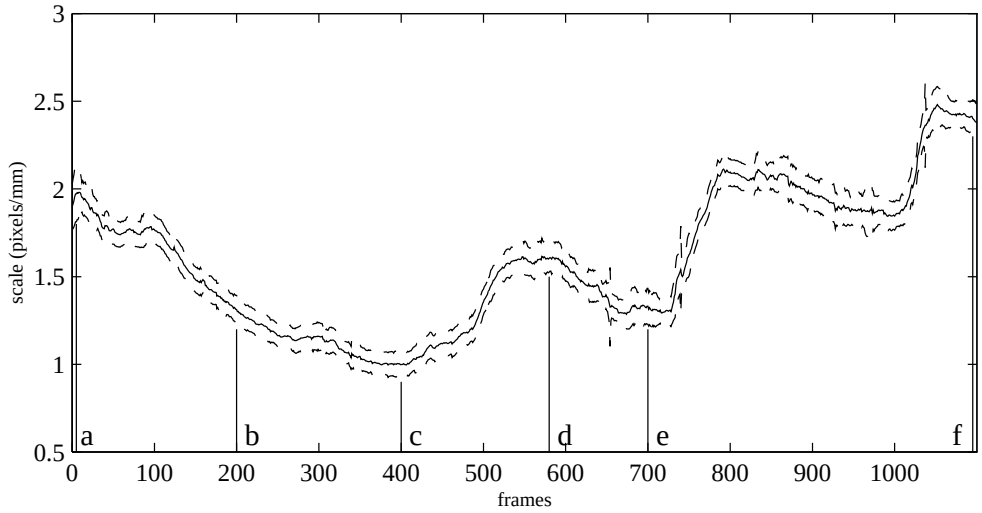


Figure 7.14: *Weak Perspective Camera Model: Graph showing changing scale factor for the weak perspective camera from the sequence shown in figure 7.13. The 1σ confidence limits are shown dashed. The spikes at frames 650, 740, and 1040 are where the lock on the object is temporarily lost giving an obvious increase in the uncertainty. The motion occurring is described in figure 7.13.*

Chapter 8

Conclusions

8.1 Summary

This thesis has presented work in two areas of computer vision: two novel methods for self-calibrating a pinhole camera were described, and a common rigid object tracker was extensively modified to increases its robustness. The general themes applied to these areas included stratification of a problem and utilising known constraints to simplify a problem, with the aim of obtaining a concise algorithm which can be easily extended to multiple images rather than the minimum number required.

The first self-calibration algorithm (chapter 4), utilised the known motion constraint of an initial pure translation to allow the stratification of the self-calibration problem. As a result, the calibration algorithm is linear and can be easily extended to an arbitrary number of images.

The second self-calibration algorithm (chapter 5) demonstrated the geometric importance of fixed entities as tools for camera calibration. The fixed entities of planar motion were described, and it was shown how the position of these determine the camera calibration. Then chapter 6 estimated the covariance of the position of the fixed entities, and showed how this can give a measure of uncertainty for the computed calibration. However, this was only partially successful, and the limitations and problems encountered were described.

The tracker RAPiD was extended to give the robust object tracker RoRAPiD (chapter 7),

which by utilising the complexity inherent in rigid objects and robust detection methods, improved the robustness of the tracker to conditions such as partial occlusion. The constraints of the type of object motion or the different camera models were used to extend the operation of the tracker.

Throughout the thesis, extensive results were given for a number of cameras mounted on various platforms, and these results have been shown to be both accurate and stable.

8.2 Future Work

Although the theory and algorithms presented here have been shown to perform successfully, there are still many areas which require further work. Some of the more general areas of interest are described below.

- Most camera calibration algorithms use static scenes, but these are not the norm for computer vision, where the scene usually contains independently moving objects. So an interesting avenue of work will be to examine if it is possible to calibrate a camera when tracking a moving object. This could involve combining the work presented here on rigid object tracking and self-calibration, especially with the planar motion constraint. This approach has the advantage that if more than one object is being tracked, then each object will give independent constraints on the calibration.

- Camera calibration is generally presented as single independent topic in computer vision, but it should really be seen as a small part of a larger vision task. As a result, can camera calibration be approached as a background process to other vision tasks, so that it can track the camera calibration and detect when the calibration changes?

- Uncertainty analysis is important as it gives a measure of the confidence in the computed parameters, but is not well understood in the area of self-calibration and

scene reconstruction. How does the uncertainty in camera calibration affect the Euclidean reconstruction? How can the uncertainty of the motion constraints be propagated, e.g., what is the effect of a small rotation during an assumed pure translation?

- The work presented here uses a large number of points matched over a few images. How does this compare to using a few points tracked over a large number of images?

Appendix A

The Fundamental Matrix

Overview

A large amount of work has been done on understanding and computing the fundamental matrix. This appendix is only a brief review of the fundamental matrix containing the theory of epipolar geometry which leads to the fundamental matrix, and the different parameterisations for the matrix. A description of the implementation used to compute the matrix is given. More details on the matrix and its computation can be found in [60, 91].

The Fundamental Matrix

The fundamental matrix [22, 36] algebraically encodes the epipolar geometry between two views. It is the extension of the essential matrix [56] to uncalibrated cameras. The epipolar geometry constraint, which defines the fundamental matrix is stated without proof.

Theorem A.1 *Given an uncalibrated camera moving under the rigid displacement defined by* $(\mathsf{R}, \mathbf{t})$ *such that* $\mathbf{t} \neq \mathbf{0}$*, and a set of homogeneous image points* $\{\mathbf{x}_i\}$ *in the first view which are transformed to the images points* $\{\mathbf{x}'_i\}$ *in the second view, then there exists a* 3×3 *matrix* F *which satisfies the epipolar geometry constraint*

$$\mathbf{x}'^{\top}_i \mathsf{F} \mathbf{x}_i = 0 \quad \forall i. \tag{A.1}$$

Proof: See Faugeras [22] or Hartley [36].

□

A.1 Epipolar Geometry

Equation (A.1) is the epipolar constraint for corresponding points in two views. Geometrically, a point in the first image $\mathbf{x}$ is mapped by $\mathtt{F}$ to a line $\mathbf{l}'$ in the second image (the *epipolar line*), where $\mathbf{l}' = \mathtt{F}\mathbf{x}$, and the corresponding point $\mathbf{x}'$ lies on this line giving the constraint

$$\mathbf{x}'^{\top}\mathbf{l}' = \mathbf{x}'^{\top}\mathtt{F}\mathbf{x} = 0.$$

Conversely, the point $\mathbf{x}'$ is mapped to the epipolar line in the first image, $\mathbf{l} = \mathtt{F}^{\top}\mathbf{x}'$, and $\mathbf{x}$ lies on this line to give the simple rearrangement of equation (A.1)

$$\mathbf{x}^{\top}\mathbf{l} = \mathbf{x}^{\top}\mathtt{F}^{\top}\mathbf{x}' = 0.$$

For real images, where the position of the points is perturbed by noise, the epipolar constraint is not always satisfied. So that for the real points $\mathbf{x}$ and $\mathbf{x}'$, corresponding to the exact points $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ respectively, then the points do not necessarily lie on the epipolar line and

$$\mathbf{x}'^{\top}\mathtt{F}\mathbf{x} = \epsilon,$$

where ϵ is the algebraic error of the epipolar constraint. However, rather than use the algebraic error ϵ, a geometric error measured on the image plane is often used. This leads to the definition of epipolar distance (error), which is the perpendicular distance from a point $\mathbf{x}'$ to the corresponding epipolar line $\mathtt{F}\mathbf{x}$, and is given by $d(\mathbf{x}', \mathtt{F}\mathbf{x})$ where $d(\mathbf{a}, \mathbf{l})$ is the perpendicular distance[1] from the point $\mathbf{a}$ to the line $\mathbf{l}$.

Generally, the epipolar distance is computed for both images to avoid any bias in any computation using the epipolar distance, and for a set of points the sum of the squared

[1]The actual function is $d(\mathbf{a}, \mathbf{l}) = (a_1 l_1 + a_2 l_2 + l_3)/(a_3\sqrt{l_1^2 + l_2^2})$.

epipolar distance is given by

$$E_{ep} = \sum_i \left(d(\mathbf{x}'_i, \mathtt{F}\mathbf{x}_i)^2 + d(\mathbf{x}_i, \mathtt{F}^\top \mathbf{x}'_i)^2 \right) . \tag{A.2}$$

A.2 Parameterising the Fundamental Matrix

General Motion Parameterisation

For general motion, $\mathtt{F}$ has seven independent degrees of freedom. The constraint equation (A.1) only requires $\mathtt{F}$ to be defined up to a non-zero scaling, and $\mathtt{F}$ also has to satisfies the additional algebraic constraint of $\det(\mathtt{F}) = 0$. This leaves seven independent degrees of freedom.

Several different parameterisations have been suggested for $\mathtt{F}$ which encode the above constraints [25, 59], but the simplest form is where one row (column) is computed as a linear combination of the other two rows (columns) such as

$$\mathtt{F} = \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ \alpha F_{11} + \beta F_{21} & \alpha F_{12} + \beta F_{22} & \alpha F_{13} + \beta F_{23} \end{bmatrix}, \tag{A.3}$$

and one of the elements (or a row/column/matrix norm) is fixed to be unity to remove the non-zero scaling. Possible errors can be introduced when the fixed element is significantly smaller than the other elements, but this can be detected and a different parameterisation used.

However, for degenerate motions (e.g., pure translation) the fundamental matrix has reduced forms, which are important if the degenerate situations are not to adversely effect numerical computations [91]. The parameterisations for pure translation or planar motion are described below. Several other degenerate forms of $\mathtt{F}$ exist, but are not utilised for the work presented here, and more details can be found in [91, 95].

Translational Motion Parameterisation

For a pure translating camera, the fundamental matrix $\mathtt{F}$ is skew symmetric with two independent degrees of freedom [63], and has the form

$$\mathtt{F} = \begin{bmatrix} 0 & F_{12} & F_{13} \\ -F_{12} & 0 & F_{23} \\ -F_{13} & -F_{23} & 0 \end{bmatrix}. \tag{A.4}$$

Planar Motion Parameterisation

For a camera undergoing planar motion, Maybank [63] showed that the fundamental matrix satisfies an additional constraint, that $\det(\mathtt{F} + \mathtt{F}^\top) = 0$ (i.e., the symmetric part of $\mathtt{F}$ is also Rank 2), which results in six independent degrees of freedom for $\mathtt{F}$. Vieville and Lingrand [95] introduced a minimal parameterisation for this degenerate matrix of the form

$$\mathtt{F} = \left[\sin(\theta)[\mathbf{f}_0]_\times + (1 - \cos(\theta))[\mathbf{f}_1\mathbf{f}_2^\top + \mathbf{f}_2\mathbf{f}_1^\top]\right], \tag{A.5}$$

where θ is related to the rotation angle between views, and $\mathbf{f}_{0,1,2}$ are unit vectors which satisfy the additional constraint of $\mathbf{f}_0^\top\mathbf{f}_1 = 0$. Section 5.3 shows that this parameterisation is very closely related to the fixed image entities for planar motion. The two unit vectors $\mathbf{f}_1$ and $\mathbf{f}_2$ correspond to the image lines l and m.

The constraint of $\mathbf{f}_0^\top\mathbf{f}_1 = 0$ is explicitly encoded in the parameterisation by expressing $\mathbf{f}_0$ as a function of $\mathbf{f}_1$ and an angle ρ (i.e., $\mathbf{f}_0 = \mathbf{f}(\mathbf{f}_1, \rho)$). The constraint is such, that if the vector $\mathbf{f}_1$ is taken to define a normal to a plane in 3-space, then $\mathbf{f}_0$ is constrained to lie in that plane. A reference direction on the plane is obtained by projecting the X axis onto the plane. Then ρ is the angle measured between the direction of $\mathbf{f}_0$ and the reference direction, hence $\mathbf{f}_0$ is constrained to lie in the plane and the constraint $\mathbf{f}_0^\top\mathbf{f}_1 = 0$ is satisfied.

So the minimum parameterisation used for each planar motion fundamental matrix is

$$\mathtt{F} \leftrightarrow \mathbf{f}(\theta, \mathrm{l}, \mathrm{m}, \rho)$$

where the image lines are encoded using spherical coordinates to give unit vectors.

A.3 Computing the Fundamental Matrix

Several methods have been suggested for computing the fundamental matrix using matched points[2] [8, 19, 36, 60, 91]. All methods have a similar approach: first, an estimate of $\mathtt{F}$ is computed using linear methods which minimises the algebraic error using equation (A.1); then a non-linear method is used to minimise the geometric error, the epipolar distance given in equation (A.2). Where the methods differ, is in how the fundamental matrix is parameterised (see appendix A.2), if and how outlying points (mismatches) are detected, and if the degenerate situations are detected and utilised.

The method used to compute the fundamental matrix for this work is given below, with a linear method used to obtain an initial estimate of the solution, and then a non-linear minimisation which iteratively refines the answer.

Linear Method

The linear method is independent of the final parameterisation of $\mathtt{F}$, and does not enforce the Rank 2 constraint. Equation (A.1) can be rearranged into the form

$$F_{11}xx' + F_{12}yx' + F_{13}x' + F_{21}xy' + F_{22}yy' + F_{23}y' + F_{31}x + F_{32}y + F_{33} = 0,$$

where $\mathbf{x} = (x, y, 1)^\top$ and $\mathbf{x}' = (x', y', 1)^\top$, and each pair of points give one constraint on $\mathtt{F}$. So eight[3] or more pairs of points can be used to solve for $\mathtt{F}$ with the equation

$$\mathtt{Z}\mathbf{f} = \mathbf{0}, \tag{A.6}$$

where $\mathbf{f} = (F_{11}, F_{12}, \ldots, F_{33})^\top$ and each row of $\mathtt{Z}$ is a function of a pair of points

$$\mathbf{z}_i = (x_i x'_i, y_i x'_i, x'_i, x_i y'_i, y_i y'_i, y'_i, x_i, y_i, 1).$$

[2] Matched points are obtained using software developed by Paul Beardsley, and the approach used is described in [8].

[3] $\mathtt{F}$ is only defined up to scale and requires eight constraints.

For real points there is generally not an exact solution to equation (A.6), and so the solution is found using SVD [78] to solve

$$\min_{\|\mathbf{f}\|_2=1} \| \mathtt{Z}\mathbf{f} \|_2 \ . \tag{A.7}$$

The solution to equation (A.7) is the right singular vector associated with the minimum singular value of $\mathtt{Z}$. However, $\mathtt{Z}$ is a $n \times 9$ matrix where n, the number of points, is often $n \gg 9$, and so better numerical results can be found by decomposing the 9×9 matrix $\mathtt{Z}^\top\mathtt{Z}$ to give the eigenvector solution. The extra constraint of $\| \mathbf{f} \|_2 = 1$ is required to avoid the trivial solution of $\mathbf{f} = \mathbf{0}$. This constraint is automatically achieved by using the eigenvector solution which has unit norm by definition. Following [36], numerically better results are obtain by scaling the points to have the mean position at the origin, and a mean distance of $\sqrt{2}$ from the origin.

When the motion is pure translation, the reduced form for the matrix (A.4) can be used. Now there are only three elements to find and equation (A.1) can be rearranged into the form

$$F_{12}(yx' - xy') + F_{13}(x' - x) + F_{23}(y' - y) = 0.$$

Again the each pair of points gives one constraint, and these can be rearranged into a reduced form of equation (A.6)

$$\mathtt{Z}^*\mathbf{f}^* = \mathbf{0}, \tag{A.8}$$

where $\mathbf{f}^* = (F_{12}, F_{13}, F_{23})^\top$ and each row of $\mathtt{Z}^*$ has the form

$$\mathbf{z}_i^* = (y_i x_i' - x_i y_i', x_i' - x_i, y_i' - y_i),$$

and SVD is used to find a solution corresponding to the right singular vector of $\mathtt{Z}^{*\top}\mathtt{Z}^*$ associated with the minimum singular value.

Non-Linear Method

Once an initial solution has been found using linear methods, a non-linear iterative scheme can be used. This has the advantage of minimising the geometric (epipolar) error rather than an algebraic error, and also allows the correct minimum parameterisation, which satisfies the applicable constraints, to be used. A Levenberg-Marquardt [78] minimisation scheme is used to minimise the sum of the squared epipolar distances given in equation (A.2). Any of the possible parameterisations for $\mathtt{F}$ can be used, and the covariance of the final solution can be obtained using the results of lemma C.1.

Appendix B

The Trifocal Tensor

Overview

The trifocal tensor was reviewed in section 5.4.1, and was used to compute the position of the circular points. In this appendix, it is shown how the tensor was computed, and several results associated with the tensor are derived.

B.1 Computing the Trifocal Tensor

Several methods have been suggested for computing the trifocal tensor [41, 52, 92], some of which enforce the minimum parameterisation. However, the method here follows Hartley [41] by computing the 27 parameters without enforcing the additional constraints.

The method used is analogous to the linear methods used to compute the fundamental matrix (see appendix A.3). From each triplet of matched points and equation (5.17), nine constraints on the elements of the tensor are obtained, of which only four are independent. Similarly from a triplet of matched lines and equation (5.18), two independent constraints on the tensor are obtained. So given n_p triplets of matched points and n_l triplets of matched lines which satisfy $4n_p + 2n_l \geq 26$, then the tensor can be computed. The constraints are arranged into the equation

$$\mathtt{Z}\mathbf{t} = \mathbf{0} \tag{B.1}$$

where $\mathbf{t} = (T_1^{11}, T_1^{12}, \ldots, T_3^{33})^\top$ and $\mathtt{Z}$ contains the constraints from the matched points and

lines. The solution for real images is found using SVD to solve

$$\min_{||\mathbf{t}||_2=1} \| \mathtt{Z}\mathbf{t} \|_2 \tag{B.2}$$

with the method described in appendix A.3. Again to improve numerical accuracy the images are transformed to have mean position at the origin and an average distance $\sqrt{2}$ from the origin [36].

Computing the Reduced Trifocal Tensor

The method used to compute the reduced trifocal tensor is the same as that for the full tensor. However, as only 12 elements are being estimated, the calculation should be better conditioned than when estimating the full tensor.

The points and lines are transformed into the normalised planar motion frame, and the corresponding triplets are used with equations (5.17) and (5.18) to give constraints on the 12 non-zero elements of the tensor. Now, only n_p triplets of matched points and n_l triplets of matched lines which satisfy $4n_p + 2n_l \geq 11$ are required to find a solution.

B.2 Trifocal Tensor Transformation

When an image is transformed, either to improve the numerical computation of the tensor (appendix B.1) or to transform to the canonical frame, a different tensor will be computed. Given below is the transformation of the tensor given an image-to-image transformation. The same image transformation has to be applied to all three images.

Lemma B.1 *Given an image point transformation* $\mathtt{H}$*, such that* $\hat{\mathbf{x}} = \mathtt{H}\mathbf{x}$*, which is applied to all three images, then the corresponding trifocal tensor transformation can be expressed as* $\hat{T}_n^{op} = [H^{-1}]_{in} H_{oj} H_{pk} T_i^{jk}$.

Proof: The point and line transformation in tensor notation are $\hat{x}_i = H_{ij}x_j$ and $l_i = H_{ji}\hat{l}_j$ respectively. The trifocal tensor transfers lines between 3 images with equation (5.18)

$$l_i = l'_j l''_k T_i^{jk},$$

with the standard notation for the corresponding lines in three images. If all the images are transformed with the same function H, then the three new lines are $l_i = H_{ni}\hat{l}_n$, $l'_j = H_{oj}\hat{l}'_o$, and $l''_k = H_{pk}\hat{l}''_p$. Substituting these into equation (5.18) gives

$$H_{ni}\hat{l}_n = H_{oj}\hat{l}'_o H_{pk}\hat{l}''_p T_i^{jk},$$

and simple rearrangement of the indices and careful treatment of H_{ni} gives

$$\hat{l}_n = \hat{l}'_o \hat{l}''_p [H^{-1}]_{in} H_{oj} H_{pk} T_i^{jk}.$$

Comparison with $\hat{l}_n = \hat{l}'_o \hat{l}''_p \hat{T}_n^{op}$ shows that

$$\hat{T}_n^{op} = [H^{-1}]_{in} H_{oj} H_{pk} T_i^{jk}. \tag{B.3}$$

□

B.3 Canonical Transformation

The canonical transformation, introduced in section 5.4.2, is an image-to-image projective mapping which is required to map the vanishing point $\mathbf{v}$ to $(0,1,0)^\top$ and the horizon line $\mathbf{l}$ to $(0,1,0)^\top$.

Lemma B.2 *The canonical transformation* H_c *which maps the image point* $\mathbf{v}$ *to* $(0,1,0)^\top$ *and the image line* $\mathbf{l}$ *to* $(0,1,0)^\top$ *is*

$$\mathsf{H}_c = \begin{bmatrix} l_2 & -l_1 & l_1 v_2 - l_2 v_1 \\ l_1 & l_2 & l_3 \\ \frac{-l_1}{\mathbf{l}.\mathbf{v}} & \frac{-l_2}{\mathbf{l}.\mathbf{v}} & \frac{-l_3}{\mathbf{l}.\mathbf{v}} + 1 \end{bmatrix}. \tag{B.4}$$

Proof: First l is mapped to $(0,1,0)^\top$ with the transformation $\mathtt{H}_1$ (i.e., $\mathrm{l} = \mathtt{H}_1^\top(0,1,0)^\top$), which can be the rigid transformation of the plane

$$\mathtt{H}_1 = \begin{bmatrix} l_2 & -l_1 & 0 \\ l_1 & l_2 & l_3 \\ 0 & 0 & 1 \end{bmatrix}.$$

This transforms $\mathbf{v}$ to $\hat{\mathbf{v}}$ where

$$\hat{\mathbf{v}} = \begin{pmatrix} l_2 v_1 - l_1 v_2 \\ l_1 v_1 + l_2 v_2 + l_3 \\ 1 \end{pmatrix}.$$

Then the projective transformation $\mathtt{H}_2$ is used to map $\hat{\mathbf{v}}$ to $(0,1,0)^\top$ whilst keeping the line $(0,1,0)^\top$ fixed, such that

$$\begin{aligned} (0,1,0)^\top &= \mathtt{H}_2\hat{\mathbf{v}}, \\ (0,1,0)^\top &= \mathtt{H}_2^\top(0,1,0)^\top, \end{aligned}$$

which to be non-singular has the form

$$\mathtt{H}_2 = \begin{bmatrix} 1 & 0 & -\hat{v}_1 \\ 0 & 1 & 0 \\ 0 & -1/\hat{v}_2 & 1 \end{bmatrix}.$$

Hence, the canonical transformation $\mathtt{H}_c = \mathtt{H}_2\mathtt{H}_1$ has the form

$$\mathtt{H}_c = \begin{bmatrix} l_2 & -l_1 & l_1 v_2 - l_2 v_1 \\ l_1 & l_2 & l_3 \\ \frac{-l_1}{\mathrm{l}.\mathbf{v}} & \frac{-l_2}{\mathrm{l}.\mathbf{v}} & \frac{-l_3}{\mathrm{l}.\mathbf{v}} + 1 \end{bmatrix}. \tag{B.5}$$

□

The inverse of the canonical transformation is

$$\mathtt{H}_c^{-1} = \begin{bmatrix} l_2 & l_1 - \frac{l_1 l_3}{\mathrm{l}.\mathbf{v}} + \frac{l_2(l_2 v_1 - l_1 v_2)}{\mathrm{l}.\mathbf{v}} & -l_1 l_3 + l_2(l_2 v_1 - l_1 v_2) \\ -l_1 & l_2 - \frac{l_2 l_3}{\mathrm{l}.\mathbf{v}} - \frac{l_1(l_2 v_1 - l_1 v_2)}{\mathrm{l}.\mathbf{v}} & -l_2 l_3 - l_1(l_2 v_1 - l_1 v_2) \\ 0 & \frac{l_2^2 + l_2^2}{\mathrm{l}.\mathbf{v}} & l_2^2 + l_2^2 \end{bmatrix}.$$

B.4 Projection Matrices

Section 5.5.2 uses the reduced trifocal tensor to obtain three consistent projection matrices, which can then be used to back-project the image points to give a projective reconstruction. Hartley [41] first decomposed the trifocal tensor to give consistent projection matrices, and was derived from equation (5.19),

$$T_i^{\cdot\cdot} = \mathbf{p}''_i \mathbf{p}'^{\top}_4 - \mathbf{p}''_4 \mathbf{p}'^{\top}_i,$$

where $\mathtt{P}$, $\mathtt{P}'$, and $\mathtt{P}''$ are the three projection matrices, and $\mathtt{P}$ is set to the canonical form $[\mathtt{I}|\mathbf{0}]$, and $\mathbf{p}'_i$ and $\mathbf{p}''_i$ are the columns of $\mathtt{P}'$, and $\mathtt{P}''$. However, the method is complex and prone to instability. For normalised planar motion, the trifocal tensor ($\tilde{T}_i^{jk}$) has a reduced form with only 12 non-zero parameters (see equation (5.23)), and this allows a simpler method to be used to obtain the projection matrices.

Lemma B.3 *The reduced trifocal tensor $\tilde{T}_i^{jk}$ can be simply decomposed into three consistent projection matrices $\tilde{\mathtt{P}}$, $\tilde{\mathtt{P}}'$, and $\tilde{\mathtt{P}}''$, which can then be transformed using $\mathtt{H}_c$ from the normalised planar motion frame into the real frame using the canonical transformation.*

Proof: Section 5.4.2 showed that the projection matrices for the normalised planar motion frame, $\tilde{\mathtt{P}}$, $\tilde{\mathtt{P}}'$, and $\tilde{\mathtt{P}}''$, have a special form

$$\begin{aligned}
\tilde{\mathtt{P}} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \\
\tilde{\mathtt{P}}' &= \begin{bmatrix} \tilde{P}'_{11} & 0 & \tilde{P}'_{12} & \tilde{P}'_{13} \\ 0 & \tilde{P}'_{22} & 0 & 0 \\ \tilde{P}'_{31} & 0 & \tilde{P}'_{33} & \tilde{P}'_{34} \end{bmatrix}, \\
\tilde{\mathtt{P}}'' &= \begin{bmatrix} \tilde{P}''_{11} & 0 & \tilde{P}''_{12} & \tilde{P}''_{13} \\ 0 & \tilde{P}''_{22} & 0 & 0 \\ \tilde{P}''_{31} & 0 & \tilde{P}''_{33} & \tilde{P}''_{34} \end{bmatrix}.
\end{aligned}$$

Using equation (5.19) in the normalised planar motion frame

$$\tilde{T}_i^{jk} = \tilde{P}'_{j4}\tilde{P}''_{ki} - \tilde{P}'_{ji}\tilde{P}''_{k4}$$

the following equations are obtained

$$\tilde{T}_1^{jk} = \begin{bmatrix} \tilde{P}'_{11}\tilde{P}''_{14} - \tilde{P}'_{14}\tilde{P}''_{11} & 0 & \tilde{P}'_{11}\tilde{P}''_{34} - \tilde{P}'_{14}\tilde{P}''_{31} \\ 0 & 0 & 0 \\ \tilde{P}'_{31}\tilde{P}''_{14} - \tilde{P}'_{31}\tilde{P}''_{34} & 0 & \tilde{P}'_{34}\tilde{P}''_{11} - \tilde{P}'_{34}\tilde{P}''_{31} \end{bmatrix},$$

$$\tilde{T}_2^{jk} = \begin{bmatrix} 0 & -\tilde{P}'_{14}\tilde{P}''_{22} & 0 \\ \tilde{P}'_{22}\tilde{P}''_{14} & 0 & \tilde{P}'_{22}\tilde{P}''_{34} \\ 0 & -\tilde{P}'_{34}\tilde{P}''_{22} & 0 \end{bmatrix},$$

$$\tilde{T}_3^{jk} = \begin{bmatrix} \tilde{P}'_{13}\tilde{P}''_{14} - \tilde{P}'_{14}\tilde{P}''_{13} & 0 & \tilde{P}'_{13}\tilde{P}''_{34} - \tilde{P}'_{14}\tilde{P}''_{33} \\ 0 & 0 & 0 \\ \tilde{P}'_{33}\tilde{P}''_{14} - \tilde{P}'_{33}\tilde{P}''_{34} & 0 & \tilde{P}'_{34}\tilde{P}''_{13} - \tilde{P}'_{34}\tilde{P}''_{33} \end{bmatrix}.$$

Six parameters $\{\tilde{P}'_{22}, \tilde{P}''_{22}, \tilde{P}'_{14}, \tilde{P}''_{14}, \tilde{P}'_{34}, \tilde{P}''_{34}\}$ can be found up to scale from $\tilde{T}_2^{jk}$. Projection matrices are only defined up to scale, which is fixed such that $\|\tilde{P}'_{i4}\| = 1.0$ and $\|\tilde{P}''_{i4}\| = 1.0$.

From $\tilde{T}_1^{jk}$ and $\tilde{T}_3^{jk}$, 8 linear equations in the 8 remaining unknown parameters are obtained. However, there is a two-parameter family of solutions, and any solution from this family will give consistent projection matrices. So $\tilde{P}''_{31}$ and $\tilde{P}''_{33}$ are set to unity, and the remaining 6 parameters can be found. The two-parameter family of solutions exist because of the arbitrary choice for the plane at infinity. Normally, when forming projective projection matrices there is a four-parameter family [10], but the constrained form of the projection matrices reduces this ambiguity to just two-parameters.

Three consistent projection matrices for the real images can be computed by using the inverse canonical transformation $\mathtt{P}_i = \mathtt{H}_c^{-1}\tilde{\mathtt{P}}_i$. This is from the simple observation that the same projective reconstruction projects to the transformed image points.

□

Appendix C

First Order Error Propagation

Overview

This appendix gives a brief review of error propagation for non-linear functions, and shows how using a first order approximation to a non-linear function it is possible to get a first order estimate of the covariance of the output. Also stated, without proof, is a result showing how to estimate the covariance of the solution of a non-linear minimisation. The theory of first order error propagation is applied to the intersection of multiple lines.

C.1 Non-Linear Functions

Chapter 6 examined the method of self-calibration using fixed points, and tried to estimate the covariance of the computed calibration. The initial assumption was that the image points are perturbed by Gaussian noise, and this uncertainty is then tracked through the algorithm using error propagation. However, many of the steps of the algorithm are non-linear and so first order approximations have to be used to propagate the covariance.

Error Propagation Given some data $\mathbf{x}$ with covariance $\Lambda_{\mathbf{x}}$, and a known function $\mathbf{y} = \mathbf{f}(\mathbf{x})$, what is the covariance of the output of the function $\Lambda_{\mathbf{y}}$?

If the function $\mathbf{f}$ is linear and is expressed as a matrix $\mathtt{M}$, then the output and its covariance can be simply stated as

$$\mathbf{y} = \mathtt{M}\mathbf{x}, \tag{C.1}$$

$$\Lambda_{\mathbf{y}} = \mathtt{M}\Lambda_{\mathbf{x}}\mathtt{M}^{\top}. \tag{C.2}$$

Once the function $\mathbf{f}$ becomes non-linear, a first order approximation has to be used. Expanding the function using a Taylor series and truncating to first order terms give

$$\mathbf{f}(\bar{\mathbf{x}} + \boldsymbol{\Delta}\mathbf{x}) = \mathbf{f}(\bar{\mathbf{x}}) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\boldsymbol{\Delta}\mathbf{x} + O(\| \boldsymbol{\Delta}\mathbf{x} \|^2), \tag{C.3}$$

where $\bar{\mathbf{x}}$ is the expected value, and $\boldsymbol{\Delta}\mathbf{x}$ models the covariance such that

$$\begin{aligned} \mathrm{E}[\boldsymbol{\Delta}\mathbf{x}] &= \mathbf{0}, \\ \mathrm{E}\left[(\boldsymbol{\Delta}\mathbf{x} - \mathrm{E}[\boldsymbol{\Delta}\mathbf{x}])^2\right] &= \mathrm{E}\left[\boldsymbol{\Delta}\mathbf{x}\boldsymbol{\Delta}\mathbf{x}^{\top}\right] \\ &= \Lambda_{\mathbf{x}}. \end{aligned}$$

The covariance of the output can now be calculated as

$$\begin{aligned} \Lambda_{\mathbf{y}} &= \mathrm{E}\left[(\mathbf{f}(\bar{\mathbf{x}} + \boldsymbol{\Delta}\mathbf{x}) - \mathbf{f}(\bar{\mathbf{x}}))^2\right] \\ &\approx \mathrm{E}\left[(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\boldsymbol{\Delta}\mathbf{x})^2\right] \\ &= \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Lambda_{\mathbf{x}}\frac{\partial \mathbf{f}}{\partial \mathbf{x}}^{\top}. \end{aligned} \tag{C.4}$$

Covariance of an Implicit Function

Several algorithms in this thesis use an unconstrained minimisation of an implicit cost function when iteratively solving a non-linear minimisation (see section 6.2.2 and appendix A.3). Using the implicit function theorem [62] and the case where the cost function is a sum of squares it is possible to compute a first order approximation to the covariance of the solutions.

Lemma C.1 *Given an implicit function of the form*

$$C(\mathbf{x}_i, \mathbf{z}) = \sum_{i=1}^{n} C_i^2(\mathbf{x}_i, \mathbf{z})$$

where $\mathbf{x}_i$ *are the measurements,* $\mathbf{z}$ *contains the* p *parameters being minimised over, and* $\mathbf{z} = \mathbf{z}_0$ *at the minimum of the function* $C(\mathbf{x}_i, \mathbf{z})$*. Then the covariance of the solution* $\mathbf{z}_0$ *is given by*

$$\Lambda_{\mathbf{z}_0} = \frac{2C_{min}}{n-p}\mathsf{H}^{-\top},$$

where the Hessian H *is approximated by*

$$\mathsf{H} \approx \sum_i \frac{\partial C_i}{\partial \mathbf{z}}^\top \frac{\partial C_i}{\partial \mathbf{z}},$$

and

$$C_{min} = \sum_i C_i^2(\mathbf{x}_i, \mathbf{z}_0).$$

Proof: See Csurka *et al.* [17] (or books on classical analysis [62]).

□

C.2 Covariance of the Intersection of Lines

The following is taken from Clarke [16], and deals with the problem of estimating the position of the vanishing point by intersecting the image lines (see section 5.5). The problem can be stated generally as that given a set of lines lying in a plane, find the point which minimises the sum of the perpendicular distance from that point to the lines. The n lines are given as homogeneous 3 vectors ($\mathrm{l}_i = (l_{i1}, l_{i2}, l_{i3})^\top$) which are scaled such that $l_{i1}^2 + l_{i2}^2 = 1$, and the point is represented by $\mathbf{p} = (p_x, p_y, 1)^\top$. The sum of the distances squared is

$$C = \sum_{i=1}^{n} \left(\mathrm{l}_i^\top \mathbf{p}\right)^2.$$

It can be shown [16] that the solution is

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix} = -\frac{1}{S_{xx}S_{yy} - S_{xy}^2} \begin{bmatrix} S_{yy} & -S_{xy} \\ -S_{xy} & S_{xx} \end{bmatrix} \begin{bmatrix} S_{xz} \\ S_{yz} \end{bmatrix}, \tag{C.5}$$

where $S_{uv} = \sum_{i=1}^{n} l_{iu} l_{iv}$. The covariance of the solution is given by

$$\Lambda_{\mathbf{p}} = \nabla \mathbf{f} \Lambda_{\mathbf{l}} \nabla \mathbf{f}^{\top}, \tag{C.6}$$

where $\Lambda_{\mathbf{l}}$ is a block diagonal matrix with the i^{th} block containing $\Lambda_{\mathbf{l}_i}$, the covariance of the i^{th} line. The Jacobian $\nabla \mathbf{f}$ is estimated using

$$\nabla \mathbf{f} = -\left(\frac{\partial^2 C}{\partial \mathbf{p}^2}\right)^{-1} \frac{\partial^2 C}{\partial \mathbf{p} \partial \mathbf{l}},$$

where

$$\frac{\partial^2 C}{\partial \mathbf{p}^2} = \begin{bmatrix} S_{xx} & S_{xy} \\ S_{xy} & S_{yy} \end{bmatrix},$$

and

$$\frac{\partial^2 C}{\partial \mathbf{p} \partial \mathbf{l}} = \begin{bmatrix} \dots & 2l_{ix}p_x & l_{ix}p_y & l_{ix} & \dots \\ \dots & 2l_{iy}p_x & l_{iy}p_y & l_{iy} & \dots \end{bmatrix}.$$

If the lines have different covariance matrices, then the equations are adjusted such that

$$S_{uv} = \sum_{i=1}^{n} \frac{l_{iu} l_{iv}}{w_i},$$

and

$$\frac{\partial^2 C}{\partial \mathbf{p} \partial \mathbf{l}} = \begin{bmatrix} \dots & \frac{2l_{ix}p_x}{w_i} & \frac{l_{ix}p_y}{w_i} & \frac{l_{ix}}{w_i} & \dots \\ \dots & \frac{2l_{iy}p_x}{w_i} & \frac{l_{iy}p_y}{w_i} & \frac{l_{iy}}{w_i} & \dots \end{bmatrix},$$

where w_i is the trace of the i^{th} line covariance matrix.

Bibliography

[1] Abramowitz, M. and Stegun, I. *Handbook of Mathematical Functions.* Dover, 1972.

[2] Armstrong M. *Euclidean Structure and Camera Calibration from Image Sequences.* First year report, Dept. of Engineering Science, University of Oxford, 1994.

[3] Armstrong, M. and Zisserman A. Robust object tracking. In *Proc. Asian Conference on Computer Vision*, volume I, pages 58–61, 1995.

[4] Armstrong, M., Zisserman A., and Beardsley P. Euclidean reconstruction from uncalibrated images. In *Proc. British Machine Vision Conference*, pages 509–518, 1994.

[5] Armstrong, M., Zisserman, A., and Hartley, R. Self-calibration from image triplets. In *Proc. European Conference on Computer Vision*, LNCS 1064/5, pages 3–16. Springer-Verlag, 1996.

[6] Bar-Shalom, Y. and Fortmann, T.E. *Tracking and Data Association.* Academic Press, 1988.

[7] Beardsley, P. *Correction of radial distortion.* Technical Report OUEL 1896/91, Dept. of Engineering Science, University of Oxford, 1991.

[8] Beardsley, P., Torr, P., and Zisserman, A. 3d model acquisition from extended image sequences. In *Proc. European Conference on Computer Vision*, LNCS 1064/1065, pages 683–695. Springer-Verlag, 1996.

[9] Beardsley, P. and Zisserman, A. Affine calibration of mobile vehicles. In *Europe-China workshop on Geometrical Modelling and Invariants for Computer Vision* [69], pages 214–221.

[10] Beardsley, P., Zisserman, A., and Murray, D. Navigation using affine structure and motion. In *Proc. European Conference on Computer Vision*, LNCS 800/801, pages 85–96. Springer-Verlag, 1994.

[11] Blake, A., Curwen, R., and Zisserman, A. A framework for spatiotemporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11(2):127–145, 1993.

[12] Blake, A. and Yuille, A. *Active Vision*. MIT Press, 1992.

[13] Bottema, O. and Roth, B. *Theoretical Kinematics.* Dover, New York, 1979.

[14] Brady, M. and Wang, H. Vision for a mobile robot. *Phil. Trans. Royal Society of London*, pages 341–350, 1992.

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

[16] Clarke, J.C. *First order error propagation: A primer.* Submitted I.V.C, 1996.

[17] Csurka, G., Zeller, C., Zhang, Z., and Faugeras, O. *Characterizing the Uncertainty of the Fundamental Matrix.* Technical Report 2560, I.N.R.I.A., France, 1995.

[18] Deriche R. Using canny's criteria to derive a recursively implemented optimal edge detector. *International Journal of Computer Vision*, 1(2):167–187, 1987.

[19] Deriche, R., Zhang, Z, Luong, Q.T., and Faugeras, O. Robust recovery of epipolar geometry for an uncalibrated stereo rig. In *Proc. European Conference on Computer Vision*, LNCS 800/801, pages 567–576. Springer-Verlag, 1994.

[20] Devernay, F. and Faugeras, O. From projective to euclidean reconstruction. In *Proc. Computer Vision and Pattern Recognition*, pages 264–269, 1996.

[21] Evans, R.J. Filtering of pose estimates generated by the rapid tracker in applications. In *Proc. British Machine Vision Conference*, pages 79–84, 1990.

[22] Faugeras, O.D. What can be seen in three dimensions with an uncalibrated stereo rig? In *Proc. European Conference on Computer Vision*, LNCS 588, pages 563–578. Springer-Verlag, 1992.

[23] Faugeras, O.D. *Three-dimensional computer vision: a geometric viewpoint.* MIT Press, 1993.

[24] Faugeras, O.D. Stratification of three-dimensional vision: projective, affine, and metric representation. *Journal of Optical Society of America*, A12:465–484, 1995.

[25] Faugeras, O.D., Luong, Q.T., and Maybank, S. Camera self-calibration: theory and experiments. In *Proc. European Conference on Computer Vision*, LNCS 588, pages 321–334. Springer-Verlag, 1992.

[26] Faugeras, O.D. and Mourrain, B. On the geometry and algebra of point and line correspondences between n images. In *Proc. International Conference on Computer Vision*, pages 951–962, 1995.

[27] Fischler, M. A. and Bolles, R. C. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. Assoc. Comp. Mach.*, 24:381–95, 1981.

[28] Frank, T, Haag, M., Kollnig, H., and Hagel, H.H. Tracking of occluded vehicles in traffic scenes. In *Proc. European Conference on Computer Vision*, LNCS 1064/5, pages 485–494. Springer-Verlag, 1996.

[29] Ganapathy, S. Decomposition of transformation matrices for robot vision. In *Proc. International Conference on Robotics and Automation*, pages 130–139, 1984.

[30] Ghosh, S. *Analytical Photogrammetry.* Pergammon Press, 1993.

[31] Golub, G. and Van Loan, C. *Matrix Computations.* John Hopkins University Press, 1983.

[32] Harris, C. *The Droid 3D Vision System.* Technical Report 72/88N488U, Plessey Research, Roke Manor, 1988.

[33] Harris, C. Structure-from-motion under orthographic projection. In *Proc. European Conference on Computer Vision*, LNCS 427, pages 118–123. Springer-Verlag, 1990.

[34] Harris, C. Tracking with rigid models. In *Active Vision* [12], chapter 4, pages 263–284.

[35] Harris C. and Stephens M. A combined corner and edge detector. In *Proc. Alvey Vision Conference*, pages 147–151, 1988.

[36] Hartley, R. Estimation of relative camera positions for uncalibrated cameras. In *Proc. European Conference on Computer Vision*, LNCS 588, pages 579–587. Springer-Verlag, 1992.

[37] Hartley, R. Cheirality invariants. In *Proc. D.A.R.P.A. Image Understanding Workshop*, pages 745–753, 1993.

[38] Hartley, R. Euclidean reconstruction from uncalibrated views. In *Applications of Invariance in Computer Vision* [73].

[39] Hartley, R. Projective reconstruction from line correspondence. In *Proc. Computer Vision and Pattern Recognition*, 1994.

[40] Hartley, R. Self-calibration from multiple views with a rotating camera. In *Proc. European Conference on Computer Vision*, LNCS 800/801. Springer-Verlag, 1994.

[41] Hartley, R. A linear method for reconstruction from lines and points. In *Proc. International Conference on Computer Vision*, pages 882–887, 1995.

[42] Hartley, R., Gupta, R., and Chang, T. Stereo from uncalibrated cameras. In *Proc. Computer Vision and Pattern Recognition*, 1992.

[43] Hartley, R. and Sturm, P. Triangulation. In *Proc. Conference Computer Analysis of Images and Patterns*, Prague, Czech Republic, 1995.

[44] Hippisley-Cox, S.D. and Porril, J. Auto-calibration — kruppa's equations and intrinsic parameters of a camera. In *Proc. British Machine Vision Conference*, pages 771–779, 1994.

[45] Horaud, R., Mohr, R., Dornaika, F, and Boufama, B. The advantage of mounting a camera onto a robot arm. In *Europe-China workshop on Geometrical Modelling and Invariants for Computer Vision* [69], pages 206–213.

[46] Hu, X, and Ahuja, N. Matching point feature with ordered geometric, rigidity and disparity constraints. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 16(10):1041–1048, 1984.

[47] Kalman, R.E. A new approach to linear filtering and prediction problems. *Transactions A.S.M.E., Journal of Basic Engineering*, March 1960.

[48] Kass, M., Witkin, A., and Terzopoulos, D. Snakes: Active contour models. In *Proc. International Conference on Computer Vision*, pages 259–268, 1987.

[49] Koenderink, J. and van Doorn, A. Affine structure from motion. *Journal of Optical Society of America*, 8(2):377–385, 1991.

[50] Koller, D., Daniilidis, K., and Nagel, H.H. Model-based object tracking in monocular sequences of road traffic scenes. *International Journal of Computer Vision*, 10:257–281, 1993.

[51] Kruppa, E. Zur ermittlung eines objektes aus zwei perspektiven mit innerer orientierung. *Sitz.-Ber. Akad. Wiss., Wien, math. naturw. Abt. IIa*, 122:1939–1948, 1913.

[52] Laveau, S. *Geometry of a system of N cameras. Theory, estimation, and applications*. PhD thesis, INRIA, 1996.

[53] Lindley, D.V. and Scott, W.F. *New Cambridge Elementary Statistical Tables*. Cambridge University Press, 1984.

[54] Lipson, P., Yuille, A., Keefe, D.O., Cavanauch, J., Taffe, J., and Rosenthal, D. Deformable templates for feature extraction from medical images. In *Proc. European Conference on Computer Vision*, LNCS 427, pages 413–417. Springer-Verlag, 1990.

[55] Liu, J., Mundy, J., and Walker, E. Recognizing arbitrary objects from multiple projections. In *Proc. Asian Conference on Computer Vision*, pages 422–426, 1993.

[56] Longuet-Higgins, H.C. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[57] Lowe, D. Integrated treatment of matching and measurement errors for robust model-based motion tracking. In *Proc. International Conference on Computer Vision*, pages 436–440, 1990.

[58] Luong, Q.T. *Matrice Fondamentale et Autocalibration en Vision par Ordinateur*. PhD thesis, Université de Paris-Sud, France, 1992.

[59] Luong, Q.T., Deriche, R., Faugeras, O.D., and Papadopoulo, T. *On determining the fundamental matrix: Analysis of different methods and experimental results.* Technical Report 1894, INRIA, Sophia-Antipolis, France, 1993.

[60] Luong, Q.T. and Faugeras, O.D. The fundamental matrix: theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17(1):43–76, 1996.

[61] Luong, Q.T. and Vieville, T. Canonic representations for the geometries of multiple projective views. In *Proc. European Conference on Computer Vision*, LNCS 800/801, pages 589–597. Springer-Verlag, 1994.

[62] Marsden, J.E. *Elementary Classical Analysis.* W.H. Freeman, San Francisco, 1974.

[63] Maybank, S. *Theory of reconstruction from image motion*. Springer-Verlag, Berlin, 1993.

[64] Maybank, S. *Lecture Series: Reconstruction from Image Motion*. Dept. of Engineering Science, University of Oxford, 1994.

[65] Maybank, S. and Faugeras, O.D. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8:123–151, 1992.

[66] P.F. McLauchlan and D.W. Murray. Active camera calibration for a Head-Eye platform using the variable State-Dimension filter. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 18(1):15–22, 1996.

[67] McLauchlan, P. and Murray, D. A unifying framework for structure from motion recovery from image sequences. In *Proc. International Conference on Computer Vision*, pages 314–320, 1995.

[68] Mohr, R., Boufama, B., and Brand, P. Accurate projective reconstruction. In *Applications of Invariance in Computer Vision* [73].

[69] Mohr, R. and Chengke, W. *Europe-China workshop on Geometrical Modelling and Invariants for Computer Vision*. Xidan University Press, Xi'an, China, 1995.

[70] Mohr, R., Veillon,F., and Quan, L. Relative 3d reconstruction using multiple uncalibrated images. In *Proc. Computer Vision and Pattern Recognition*, pages 543–548, 1993.

[71] Moons, T., Van Gool, L., Van Diest, M., and Pauwels, E. Affine reconstruction from perspective image pairs. In *Applications of Invariance in Computer Vision* [73].

[72] Mundy, J. and Zisserman, A. *Geometric Invariance in Computer Vision*. MIT Press, 1992.

[73] Mundy, J., Zisserman, A., and Forsyth, D. *Applications of Invariance in Computer Vision*. LNCS 825. Springer-Verlag, 1994.

[74] Murray, D.W., Du, F., McLauchlan, P.F., Reid, I.D., Sharkey, P.M., and Brady, M. Design of robot heads. In *Active Vision* [12].

[75] Pollefeys, M., van Gool, L., and Moons, T. Euclidean 3d reconstruction from stereo sequences with variable focal length. In *Proc. Asian Conference on Computer Vision*, volume II, pages 6–10, 1995.

[76] Pollefeys, M., van Gool, L., and Oosterlinck, A. The modulus constraint: a new constraint for self-calibration. In *Proc. I.C.P.R.*, 1996.

[77] Pollefeys, M., van Gool, L., and Proesmans, M. Euclidean 3d reconstruction from image sequences with variable focal lengths. In *Proc. European Conference on Computer Vision*, LNCS 1064/1065. Springer-Verlag, 1996.

[78] Press, W., Flannery, B., Teukolsky, S., and Vetterling, W. *Numerical Recipes in C*. Cambridge University Press, 1988.

[79] Quan, L. and Mohr, R. Affine shape representation from motion through reference points. *Journal of Mathematical Imaging and Vision*, 1:145–151, 1992.

[80] I. D. Reid and D. W. Murray. Active tracking of foveated feature clusters using affine structure. *International Journal of Computer Vision*, 18(1):41–60, 1996.

[81] Semple, J. and Kneebone, G. *Algebraic Projective Geometry*. Oxford University Press, 1979.

[82] Shapiro, L. *Affine Analysis of Image Sequences*. PhD thesis, University of Oxford, England, 1993.

[83] Shashua, A. Trilinearity in visual recognition by alignment. In *Proc. European Conference on Computer Vision*, LNCS 800/801, pages 479–484. Springer-Verlag, 1994.

[84] Slama, C. *Manual of Photogrammetry*. American Society of Photogrammetry, Falls Church, VA, USA, 4th edition, 1980.

[85] Spetsakis, M.E. and Aloimonos, J. Structure from motion using line correspondences. *International Journal of Computer Vision*, 4(3):171–183, 1990.

[86] Strang, G. *Linear Algebra and its Applications*. Harcourt Brace Jovanovich, San Diego, 1988.

[87] Sullivan, G.D. Visual interpretation of known objects in constrained scenes. *Phil. Trans. Royal Society of London*, 337:361–370, 1992.

[88] Tan, T.D., Sullivan, G.D., and Baker, K.D. Fast vehicle localisation and recognition without line extraction and matching. In *Proc. British Machine Vision Conference*, pages 85–94, 1994.

[89] Taylor, M. *Visually Guided Grasping*. PhD thesis, University of Oxford, England, 1995.

[90] Tomasi, C. and Kanade, T. Shape and motion from image streams under orthography: a factorisation method. *International Journal of Computer Vision*, 9:137–154, 1992.

[91] Torr, P. *Outlier Detection and Motion Segmentation*. PhD thesis, University of Oxford, England, 1995.

[92] Torr, P. and Zisserman, A. Robust parameterisation and computation of the trifocal tensor. In *Proc. British Machine Vision Conference*, 1996.

[93] Triggs, B. The geometry of projective reconstruction i: Matching constraints and the joint image. In *Proc. International Conference on Computer Vision*, pages 338–343, 1995.

[94] Tsai, R. An efficient and accurate camera calibration technique for 3D machine vision. In *Proc. Computer Vision and Pattern Recognition*, 1986.

[95] Vieville, T. and Lingrand, D. Using singular displacements for uncalibrated monocular vision systems. In *Proc. European Conference on Computer Vision*, LNCS 1064/1065, pages 207–216. Springer-Verlag, 1996.

[96] Vieville, T. and Luong, Q. *Motion of points and lines in the uncalibrated case*. Technical Report 2054, I.N.R.I.A., 1993.

[97] Wang, H. and Brady, M. Corner detection: some new results. In *IEE Colloquium Digest of System Aspects of Machine Perception and Vision*, pages 1.1–1.4, London, 1992.

[98] Weng, J., Huang, T.S., and Ahuja, N. Motion and structure from two perspective views: Algorithms, error analysis, and error estimation. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 11(5):451–476, 1989.

[99] Weng, J., Huang, T.S., and Ahuja, N. Optimal motion and structure estimation. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 15(9):864–884, 1993.

[100] Wiles, C. *Closing the Loop on Multiple Motion*. PhD thesis, University of Oxford, England, 1996.

[101] Wiles, C and Brady, M. Ground plane motion camera models. In *Proc. European Conference on Computer Vision*, LNCS 1064/5. Springer-Verlag, 1996.

[102] Zeller, C. and Faugeras, O.D. *Camera self-calibration from video sequences: the Kruppa equations revisited*. Technical Report 2793, INRIA, 1995.

[103] Zeller, C. and Faugeras, O.D. Projective, affine and metric measurements from video sequences. In *Proc. of the International Symposium on Optical Science, Engineering and Instrumentation*, 1995.

[104] Zhang, Z. and Faugeras, O. *3D Dynamic Scene Analysis.* Springer-Verlag, 1992.

[105] Zisserman, A. *Lecture Series: The Geometry of Multiple Views.* Dept. of Engineering Science, University of Oxford, 1995.

[106] Zisserman, A. *A Users Guide to the Trifocal Tensor*. Dept. of Engineering Science, University of Oxford, 1996.

[107] Zisserman, A., Beardsley P., and Reid, I. Metric calibration of a stereo rig. In *IEEE Workshop on Representation of Visual Scenes, Boston*, 1995.

[108] Zisserman, A., Forsyth, D., Mundy, J., Rothwell, C., Liu, J., and Pillow, N. 3d object recognition using invariance. *Artificial Intelligence*, 78(1-2):239–287, 1995.