

# High accuracy metrology using low-resolution cameras

D.Phil Thesis

Robotics Research Group  
Department of Engineering Science  
University of Oxford



David Claus  
New College

March 8, 2007

## High accuracy metrology using low-resolution cameras

### Abstract

Camera based localization can provide extremely accurate 3D pose information, even from consumer grade video lenses. Advances in lens distortion correction, pose computation and feature detection would permit low cost cameras to be used in many applications that currently require more expensive equipment. I show how: (1) careful modelling and (2) careful fitting of these models to data; provides increased camera accuracy from the same camera equipment with little or no additional computational overhead.

The primary contribution towards camera modelling is a lens distortion model based on rational functions that can represent standard, fisheye and catadioptric lens systems. Three separate calibration methods are demonstrated, making this a useful technique that can be implemented in a wide range of applications. Evaluation of calibration precision indicates that the proposed model accurately represents real-world lens distortion and provides lower errors than other models in common use. Although sensitivity to image noise can be a problem with such flexible models, several techniques are presented here that yield robust calibration in the midst of image uncertainty. I demonstrate multiple view camera auto-calibration on fisheye lens sequences using point correspondences alone, without first requiring the removal of lens distortion.

Fitting of the camera model is improved by including a non-linear optimization to tune the model parameters against a known error measure. Careful optimizer construction is shown to avoid local minima, converge in realtime and achieve very high levels of precision. Image feature detection error is transmitted through the entire calibration process, so a robust exemplar based learning scheme is proposed to accurately detect known fiducial markers. This efficient classification approach handles the challenges of changing scene conditions (lighting variation, motion blur, clutter) without the large increase in false detections that plague other detection algorithms.

## Acknowledgements

I would like to thank Andrew Fitzgibbon for his patient instruction, guidance, and assistance throughout the course of my research. I could not have asked for a better supervisor.

Thanks to Jamie Paterson, Nicholas Apostoloff, Aeron Buchanan and the others in the Visual Geometry Group for all your assistance and input.

My research was made possible through the funding of the Rhodes Trust; I would like to thank the Warden and all of the staff at Rhodes House for making my stay in Oxford that much more enjoyable.

Finally, I would like to thank my wife, Krista, for all her support and encouragement during both the research and the writing phases of this project. I did finish it in the new year; it is just a different year.

*For Apsley*



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Why Localize? . . . . .	2
1.2	Outline . . . . .	3
1.3	Notational Conventions . . . . .	4
1.4	Authorship . . . . .	4
<b>2</b>	<b>Definitions</b>	<b>5</b>
2.1	Imaging Model . . . . .	5
2.1.1	Pinhole projection . . . . .	6
2.1.2	Aberrations . . . . .	8
2.1.3	Non-central cameras . . . . .	11
2.1.4	Cubic Polynomial Camera . . . . .	14
2.2	Camera Calibration . . . . .	15
2.2.1	Internal Calibration . . . . .	16
2.2.2	External Calibration . . . . .	20
2.3	Nonlinear Optimization . . . . .	21
2.3.1	Levenberg-Marquardt . . . . .	22
2.3.2	Bundle Adjustment . . . . .	25
2.4	Miscellaneous . . . . .	27
2.4.1	Sampson Distance . . . . .	27
2.4.2	Camera ringing . . . . .	28
2.4.3	Direct Linear Transform . . . . .	29
2.4.4	Centre Shift Under Projection . . . . .	30
2.4.5	Pattern Classification . . . . .	31
2.4.6	Receiver Operator Characteristic Curves . . . . .	35
2.5	Summary . . . . .	36
<b>3</b>	<b>Lens Distortion: Modelling</b>	<b>37</b>
3.1	Distortion Models . . . . .	37
3.1.1	Forward and Reverse Camera Models . . . . .	39
3.2	Rational Function Model . . . . .	40
3.2.1	General Mathematical Framework . . . . .	40
3.2.2	Physical Interpretation of $\mathbf{A}$ . . . . .	42
3.2.3	Back-projection and Projection . . . . .	43
3.2.4	Canonicalization of $\mathbf{A}$ . . . . .	44
3.2.5	Parametrizations for Specific Lenses . . . . .	45
3.3	Two-view geometry . . . . .	48
3.3.1	Epipolar curves . . . . .	49
3.4	Summary . . . . .	49

<b>4</b>	<b>Lens Distortion: Calibration</b>	<b>51</b>
4.1	Linear Calibration from an Arbitrary Planar Grid . . . . .	51
4.2	Calibration by Plumb-line Constraints . . . . .	54
4.2.1	Linear Factorization Method . . . . .	55
4.2.2	Optimization Method . . . . .	61
4.2.3	Condensed vs. Full Parametrization . . . . .	63
4.2.4	Plumbline data from multiple views . . . . .	65
4.3	Multiview Calibration from Epipolar Constraints . . . . .	66
4.3.1	Linear method for $\mathbf{G}$ . . . . .	66
4.3.2	Rank 2 nonlinear optimization method for $\mathbf{G}$ . . . . .	68
4.3.3	Recovery of $\mathbf{A}$ from $\mathbf{G}$ . . . . .	69
4.3.4	Parameterizing $\mathbf{G}$ using the reduced RF model . . . . .	70
4.4	Summary . . . . .	72
<b>5</b>	<b>Lens Distortion: Evaluation</b>	<b>73</b>
5.1	Approximation of existing distortion models . . . . .	73
5.2	Planar Grid Calibration . . . . .	74
5.3	Plumbline Calibration . . . . .	80
5.3.1	Reduced RF consistency . . . . .	82
5.3.2	RF vs. the MATLAB Calibration Toolbox . . . . .	83
5.4	Multiview results . . . . .	89
5.4.1	Noise sensitivity of computing $\mathbf{G}$ . . . . .	90
5.4.2	Results on image sequences . . . . .	97
5.5	Three-Dimensional Reconstruction . . . . .	101
5.6	Summary . . . . .	102
<b>6</b>	<b>Camera Localization</b>	<b>104</b>
6.1	World Points and Corresponding Image Locations . . . . .	105
6.2	Initialization . . . . .	106
6.2.1	Camera Intrinsics . . . . .	107
6.2.2	Extrinsics from coplanar data . . . . .	107
6.2.3	Extrinsics from general point data . . . . .	108
6.3	Optimization . . . . .	109
6.3.1	Analytic Derivatives for Reprojection Error . . . . .	110
6.3.2	Timing . . . . .	110
6.3.3	Zoom lenses . . . . .	111
6.4	Surveying Fiducial Positions Optically . . . . .	112
6.4.1	Structure from Motion . . . . .	113
6.4.2	Surveying procedure . . . . .	114
6.5	Evaluation . . . . .	115
6.5.1	Coordinate Measurement System Ground Truth . . . . .	115
6.5.2	ARToolkit comparison . . . . .	130
6.6	Summary . . . . .	134
<b>7</b>	<b>Fiducial Detection</b>	<b>136</b>
7.1	Strategy . . . . .	139
7.1.1	Target Design . . . . .	141
7.1.2	Training Data . . . . .	141
7.1.3	Cascading Classifier . . . . .	142
7.1.4	Cascade Stage One: Ideal Bayes . . . . .	143
7.1.5	Cascade Stage Two: Nearest Neighbour . . . . .	145
7.2	Implementation . . . . .	147
7.2.1	Training Data Filtering . . . . .	147
7.2.2	Target verification . . . . .	149

7.3	Evaluation . . . . .	152
7.3.1	Engineered Detector . . . . .	152
7.3.2	Adaptive Thresholding . . . . .	152
7.3.3	Results . . . . .	153
7.4	Locating a Circle's Projected Centre . . . . .	156
7.4.1	Homography from target coordinates . . . . .	157
7.4.2	Homography Calculation from Four Circles . . . . .	159
7.5	Target Identification . . . . .	160
7.6	Fiducial Localization Trials . . . . .	161
7.7	Summary . . . . .	163
<b>8</b>	<b>Photometric Stereo Application</b>	<b>166</b>
8.1	Motivation . . . . .	167
8.2	Overview . . . . .	168
8.2.1	Lens Distortion Correction . . . . .	170
8.2.2	Light position . . . . .	172
8.2.3	Cone Detection . . . . .	174
8.2.4	Camera pose . . . . .	175
8.2.5	Light Attenuation . . . . .	177
8.2.6	Light Fall-off Correction . . . . .	181
8.2.7	Surface Integration . . . . .	186
8.2.8	Parallax Correction . . . . .	188
8.3	Results . . . . .	191
8.4	Conclusions . . . . .	193
<b>9</b>	<b>Conclusion</b>	<b>194</b>
9.1	Modelling lens distortion . . . . .	194
9.2	Camera calibration and localization . . . . .	196
9.3	Reliable fiducial detection . . . . .	196
9.4	Application areas . . . . .	197
9.5	Summary . . . . .	197
9.6	Further extensions . . . . .	197
	<b>Appendices</b>	<b>199</b>
<b>A</b>	<b>Pinhole Camera Calibration Methods</b>	<b>199</b>
A.1	Extrinsics from Coplanar Data . . . . .	199
A.2	Extrinsics from General Point Data . . . . .	200
<b>B</b>	<b>Rotations</b>	<b>202</b>
B.1	Quaternions . . . . .	202
<b>C</b>	<b>Computing A from G</b>	<b>204</b>
	<b>Bibliography</b>	<b>212</b>

# Chapter 1

## Introduction

This thesis topic was born out of necessity. The original topic was to be surveying (in a building construction sense) with a video camera: merging structure-from-motion (SfM) and simultaneous localization and mapping (SLAM) to accurately measure three dimensional positions over an area the size of a football pitch. It seemed a reasonable starting point to pick up the nearest camera and go outside to record some video of buildings — “Whatever you do, avoid shooting your own video footage” and “Try to get some professional video clips to analyze” were two pieces of advice that came too late in my introduction to computer vision.

Three things became apparent as I processed that early video: 1) detecting and locating features in video recorded under widely varying scene conditions is harder than it looks; 2) precise camera calibration is essential, and often not fully taken advantage of; and finally 3) lens distortion must be carefully compensated for when using low-cost consumer grade lenses. But I’m getting ahead of myself; let us first take a step back and discuss measurements in general.

Measurements are important, particularly in engineering. If an engineer is to apply scientific theory to the solution of a problem he or she will often need to measure some physical quantity to provide an input to theoretical equations or methods. Many times it is also essential that these measurements be taken without disturbing the process under observation.

A camera is a measurement device; it records a projection of both the wavelength and intensity of light. Specifically, it measures the light falling upon its sensor from directions in space as dictated by the lens system. Because light can travel through a vacuum it is a truly non-contact measurement instrument. In practical terms this permits object properties to be recorded at a distance, and the physics of lens construction provides the relationship between image locations and directions in space. Thus the camera is a remote measurement device. This angle measurement function is exploited

in both SfM and vision-based SLAM. However, angle measurement depends on camera calibration: the properties of the lens must be known in order to relate image positions to directions in space. Furthermore, surveying requires the camera location be known so that all the measurements can be tied in (or referenced) to some physical location or object.

## 1.1 Why Localize?

What are the potential uses for a precise camera localization algorithm? First, let us define what we mean by camera localization: camera localization is the process of computing the position and orientation of a camera from the images it recorded.

The most obvious application is as a position sensor. A camera that can report where it is can be attached to robots, aircraft, or even held by the user. Such a camera then provides a visual record of its surroundings and a position measurement for each frame. This has uses in robot navigation, but can also be employed for object tagging inventory systems, mapping and even simply keeping track of where one's holiday photos were taken.

A second application for camera localization is in cinema special effects. A computer rendered object that is to be added into footage of a real scene must be registered relative to the background. If the camera does not move this is a job for artists; the objects are painted in front of a static background. A virtual object inserted into a sequence from a moving camera, however, will appear to float through the scene unless it is aligned correctly in each frame. Re-drawing the scene for each frame is too much for artists, and even slight alignment errors are noticeable. This type of virtual reality requires that the camera location be known so that the virtual object can be rendered from the right viewpoint and then placed at the correct position in the image. In this case the camera could be tracked using other types of sensors (inertial, ultrasonic, etc.) but it is simpler to use the camera alone and the resulting image/virtual object registration is much tighter. A third application area is surveying. If the camera location is known then the angle formed by any two points in an image can be determined.

This list of potential applications for camera localization techniques is quite long; here we will close with one final example. Where the relative positions of several cameras must be determined it can be faster and more accurate to localize each of the cameras in a common reference frame than it would be to physically measure their positions. This sort of self-calibration could be used to set up surveillance systems or rigs for studio filming. This is the basis for the single moving camera photometric

stereo system described in Chapter 8.

## 1.2 Outline

So if camera localization is such a useful technology, how does one go about it? Or, since the theory is well known and not too difficult, perhaps the more pertinent question is “What are the tricks to include and the pitfalls to avoid so that my localization algorithm is a success?” That is what I describe in this thesis. The background chapter deals with some of the physics of image formation and provides technical details on several of the concepts employed later on. After that is established we shall look at a method for correcting lens distortion. The rational function model can render the images from a wide variety of lenses as pinhole images. This greatly simplifies the mathematics of localization, and even permits Structure from Motion autocalibration from distorted image correspondences (please refer to Chapter 3 for details on what that means). A chapter is then devoted to each of distortion model calibration and evaluation: a model is only useful if a lens can be calibrated, and this calibration must also be compared with existing models. The proposed model is shown to be both simple to calibrate and extremely accurate for modelling a wide range of lens distortion. The theory is presented first, and then followed in a later chapter by the description of experimental results. This structure was adopted to help the reader; hopefully it is easy to follow.

With the distortion out of the way Chapter 6 is able to focus on localization itself. This is best handled with a nonlinear optimizer, and I present methods for assembling fast and precise fitting algorithms. These were tested both in a registration framework for placing virtual objects in cinema productions and against absolute position values obtained by physically measuring the camera path. Central to localization is the ability to reliably and repeatedly detect the same features in an image. Chapter 7 presents a method for detecting known markers (fiducials) in video. It is an exemplar-based classifier that locates fiducials which are similar in appearance to those it was shown beforehand. This allows us to deal with detection challenges such as motion blur, variable illumination and oblique viewpoints.

To bring it all together we will examine a system that relies upon accurate camera localization. The photometric stereo application of Chapter 8 uses a single camera as input. Although the camera is free to move, the application requires images that were all recorded from a common viewpoint. For this we use distortion correction and localization from fiducial markers to determine the camera position of each image. The images are re-rendered and parallax corrected to yield the required fronto-parallel

views. Not only does this system produce highly detailed photometric data, the moving camera permits depth recovery so that accurate 3D surface models can be produced. Thus a simple camera and flash rig can be used as a 3D scanner provided the camera localization is accurate.

### 1.3 Notational Conventions

Throughout this thesis a number of notational conventions will generally apply. Matrices are denoted by uppercase Roman letters in typewriter font (*e.g.*  $\mathbf{H}$ ). All vectors are column vectors. Points in Euclidean space  $\mathbb{E}^3$  are denoted by bold uppercase Roman letters (*e.g.*  $\mathbf{X} = [x \ y \ z]^\top$ ) while points within an image are denoted by bold lowercase Roman letters (*e.g.*  $\mathbf{x} = [u \ v]^\top$ ). Both types of points will often be expressed in homogeneous coordinates:

$$\begin{aligned} \mathbf{X} &= [kx \ ky \ kz \ k]^\top & \mathbf{x} &= [ku \ kv \ k]^\top \\ &= [x \ y \ z \ 1]^\top & &= [u \ v \ 1]^\top \end{aligned}$$

where  $k$  is an arbitrary scale factor and equality is up to scale. Such homogeneous points define a projective space; setting the third coordinate to zero produces points at infinity. Let us also define the perspective projection function  $\pi(x, y, z) = [x/z \ y/z]^\top$  so that  $\mathbf{X} = \pi(\hat{\mathbf{X}})$  and  $\mathbf{x} = \pi(\hat{\mathbf{x}})$ . Where several vectors are collected into a set it will be denoted by uppercase scripted letters (*e.g.*  $\mathcal{E} = \{\mathbf{e}_1 \dots \mathbf{e}_m\}$ ).

### 1.4 Authorship

This thesis has been solely authored by me, and describes the research I carried out under the supervision of Andrew Fitzgibbon. Several portions of the work have been previously published in conference papers. The fiducial detection methods of Chapter 4 were presented in (Claus and Fitzgibbon 2004, Claus 2004). Camera localization and photogrammetry for virtual reality environment initialization was demonstrated in (Claus and Fitzgibbon 2005c). The rational function lens distortion model appeared in (Claus and Fitzgibbon 2005b), while the reduced parametrization and plumbline methods were covered in (Claus and Fitzgibbon 2005a). The photometric stereo application (Chapter 8) was joint work done with Jamie Paterson (Paterson et al. 2005). I handled the image preparation (distortion correction, camera localization and registration) and surface rendering while he covered the photometric stereo aspects of the project (Paterson 2005). This division is reflected in our respective theses; please refer to (Paterson 2005) for a detailed description of the photometric portions of that work.

## Chapter 2

# Definitions

*We will occasionally use this arrow notation unless there is danger of no confusion. – Ronald Graham, “Rudiments of Ramsey Theory”*

This chapter sets out the framework in which we shall discuss camera calibration and pose estimation. A brief review of the standard pinhole projection model and non-linear distortion relates the physics of image formation to the practicalities of modern cameras. Internal and external camera calibration are then explained. We close the chapter with a discussion of several important tools in estimation and camera modeling including nonlinear optimization.

### 2.1 Imaging Model

This section examines the physics of the imaging device and the mathematics used to represent the image formation process. The conceptual problem, illustrated in Figure 2.1, is to record some representation of a scene as a two dimensional image. There are many sensor systems capable of performing this task, but we will concentrate on cameras operating in the visible light spectrum.

The task of taking photons and converting to an electrical signal is performed by a Charge Coupled Device (CCD) or, increasingly in consumer cameras, a Complementary Metal-Oxide Semiconductor (CMOS) sensor. These are generally arranged in a two

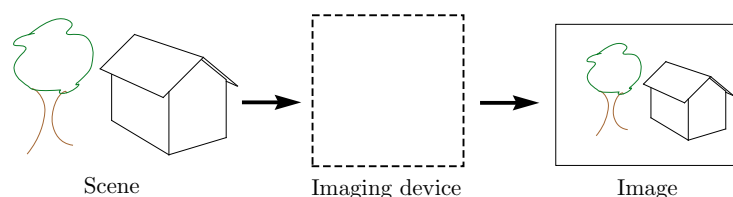


Figure 2.1: An imaging device measures photons emanating from a scene and produces a two dimensional image.



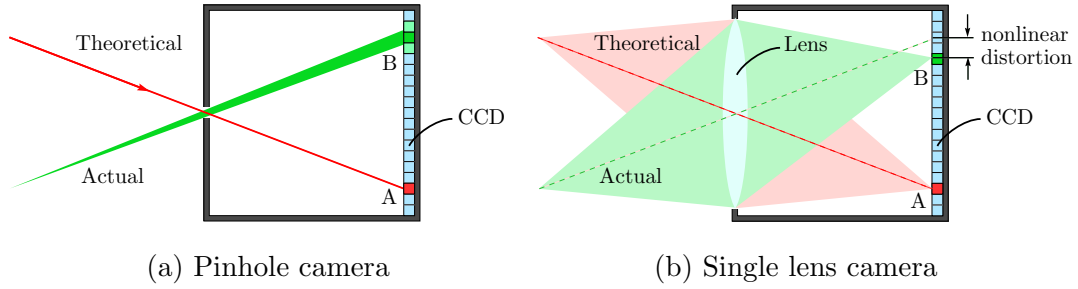


Figure 2.2: Pinhole camera model (a) A pinhole camera records only the light which passes through a tiny hole. Although theoretically a ray, the actual light enters as a cone which causes blur. (b) A lens camera still follows the pinhole model, but focuses light from a larger diameter onto the image plane. Careful lens design aims to minimize aberrations (§2.1.2) such as nonlinear distortion.

dimensional array on a silicon chip that replaces the film in an analog film camera. This dense grid of sensors (pixels) integrates incident light over a period of time (exposure length) and convert the result into a number (the intensity value). A single pixel integrates the light from all directions; it is necessary to impose geometric restrictions in order to produce a structured, coherent image. The simplest way to do this is with a pinhole camera.

### 2.1.1 Pinhole projection

A pinhole camera (Figure 2.2) produces an image from the light which passes through a single point, the pinhole. A theoretical pinhole has zero area so the light falling on each pixel is restricted to that travelling along a specific ray in space (pixel A in Figure 2.2). An actual pinhole must have a finite diameter in order to let light through, and thus a cone of light falls on each pixel rather than a ray. The sharpness of the image is therefore directly related to the size of the pinhole. A smaller hole will allow less light through, and requires a longer exposure time to produce an image. This makes it difficult or impossible to record moving subjects.

A lens can be used to increase both the amount of light falling on the CCD and the sharpness of the image. As shown in Figure 2.2(b), a lens captures light from a larger area than the pinhole, and then focuses it to a point. An ideal lens contains a single point through which all light rays will pass. This *camera centre* is analogous to a pinhole. Using a lens results in a brighter image in a shorter exposure time, but restricts the amount of the scene which can be in focus at any one time. While a pinhole camera reproduces the entire scene at a moderate level of sharpness, a lens camera can only focus objects at a specific distance from the camera. For objects that are effectively

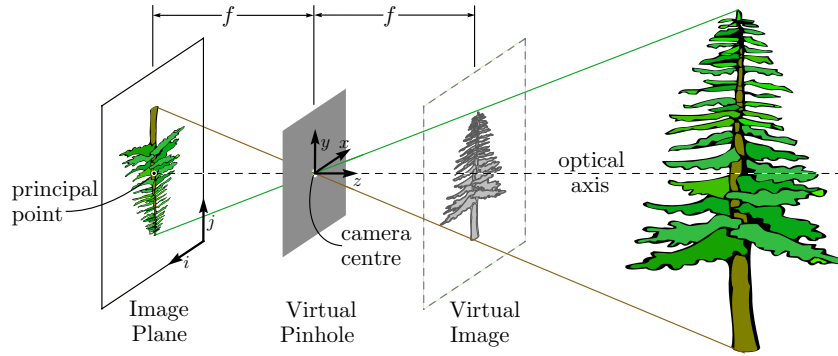


Figure 2.3: Perspective projection. All light rays from the scene pass through a virtual pinhole, the camera centre. The principal point is the intersection of the optical axis and the image plane. The focal length  $f$  is the distance between the camera centre and the image plane; for simplicity we often consider a correctly oriented virtual image located in front of the camera centre.

at infinity (parallel light rays reach the lens) the distance along the optical axis to the focused image plane is the *focal length*,  $f$ , as shown in Figure 2.3. Varying the distance between the lens and the image plane changes the distance to the world plane that will be in focus. The optical axis passes through the camera centre, and is perpendicular to the lens surfaces and the image plane. The point at which the optical axis intersects the image plane is termed the *principal point* and denoted by  $(u_0, v_0)$ . Two additional parameters are required to describe digital image formation. The image is recorded by a CCD array consisting of many rectangular detectors arranged in a dense grid. The *aspect ratio*  $a$  relates the horizontal and vertical axes scales, while the *skew*  $s$  measures the angle of the coordinate axes (actually, the difference between this angle and a right angle). For most modern lenses the skew is negligible, but the aspect ratio can rarely be ignored because CCDs are manufactured to produce images of different aspect ratios. For example, the European PAL video standard uses frame dimensions of  $720 \times 576$  while the normal display format is 4:3. The pixel aspect ratio is set to 1.067 so that the images are displayed without any scaling of the axes.

We have now introduced all of the elements needed to define a mathematical model for image formation. Despite its simplicity, the pinhole perspective model provides an accurate approximation to many real camera systems. It relates 3D points in camera coordinates  $\mathbf{X} = (x, y, z)$  in  $\mathbb{R}^3$  to image pixel coordinates  $\mathbf{x} = (i, j)$  in  $\mathbb{R}^2$  according to

$$i = f \frac{x}{z} + s \frac{y}{z} + u_0 \quad (2.1)$$

$$j = \alpha f \frac{y}{z} + v_0. \quad (2.2)$$

This can be expressed more concisely in matrix notation as  $\mathbf{x} = \pi(\mathbf{K}\mathbf{X})$  where  $\mathbf{K}$  encompasses the parameters of the pinhole projection model:

$$\mathbf{K} = \begin{bmatrix} f & s & u_0 \\ 0 & af & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad \left\{ \begin{array}{l} f \text{ is focal length;} \\ (u_0, v_0) \text{ is principal point;} \\ a \text{ is aspect ratio; } s \text{ is skew.} \end{array} \right. \quad (2.3)$$

The pinhole projection (or central projection) model provides a linear relationship between image and world points, and forms an underlying assumption in much computer vision work. While it is a reasonable first approximation, a number of factors cause deviation from this simplified theory. The next section examines a number of these effects.

### 2.1.2 Aberrations

Real lens systems do not follow the pinhole projection model exactly, but suffer from a number of aberrations. The task of the lens designer is to minimize these aberrations, however a knowledge of their attributes is useful for image understanding. Their effects are observed in high precision computer vision applications, and some compensation methods are proposed later in this thesis.

The pinhole projection model presented in the previous section is based on the approximation that the angle  $\alpha$  between a light ray and the optical axis is small and therefore  $\sin \alpha \approx \alpha$ . This is referred to as paraxial or first-order optics. Without going into exhaustive detail (the reader is referred to (Hecht 1998) for a more thorough treatment), Snell's law describes surface refraction, upon which lenses depend, as a sine relationship between the incident and refracted ray directions. The paraxial refraction equation describes thin lenses using the small angle assumption. Additional accuracy can be obtained by taking the first two terms of the Taylor expansion  $\sin \alpha \approx \alpha - \frac{\alpha^3}{3!}$ . The differences between the resulting third-order model and the (ideal) first-order model are referred to as the Seidel aberrations, after Philipp Ludwig von Seidel (1821-1896).

An *aberration* is any departure from the desired behavior. In lenses, chromatic aberrations vary with wavelength, while the five principal (or Seidel) aberrations influence all wavelengths equally. It is these latter effects which are of primary interest in visual geometry, although the former will crop up during the course of this thesis as well. We will now examine each of the principal aberrations in turn, subsequent chapters will concentrate on the distortion aspect. A lens designer compensates for these effects using a combination of stops, multiple lens elements, and aspherical lens surfaces.

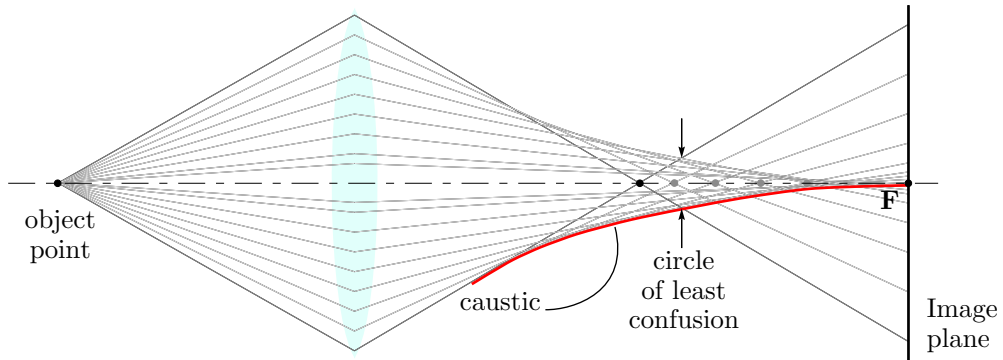


Figure 2.4: Spherical aberration causes the light from a point on the optical axis to be focused to a line along the axis.

**Spherical aberration** Spherical aberration is a dependence of focusing point on the radius at which a ray strikes the lens. It affects only light rays emanating from object points *along the optical axis*, and it is unique in that it can alter the centre of an image. The result is a blurring of the image. As illustrated in Figure 2.4, the envelope of all rays originating along the optical axis forms a caustic. The *circle of least confusion* lies in the plane where the blur caused by spherical aberration will be minimal. An aperture can be used to restrict light passing through the perimeter of the lens, and thus reduce the effects of spherical aberration.

**Coma** Coma prevents off-axis objects from focusing at a single point due to unequal magnification of light rays striking the outer portions of the lens (Figure 2.5). This produces a characteristic comet-shaped blur of off-axis light, often observed when photographing sunlight at an oblique angle. The effect is similar to spherical aberration except that it operates on off-axis points and the defocus is not restricted to the focal axis.

**Astigmatism** In a stigmatic image, each object point is focused at a single position somewhere in the image volume. This is not the case if spherical aberration or coma are present, as then a point is “focused” over some image region. A lens that suffers from astigmatism focuses small object details differently depending on their orientation. The observed effects are radial and tangential (Slama 1980). As shown in Figure 2.6, radial lines in the image are blurred by the tangential focus while the sagittal focus affects the sharpness of tangential lines. A lens has surfaces for which all objects will be in either sagittal or tangential focus. These surfaces need not be the same, nor planar, and will generally not coincide with the image plane. This results in complex patterns

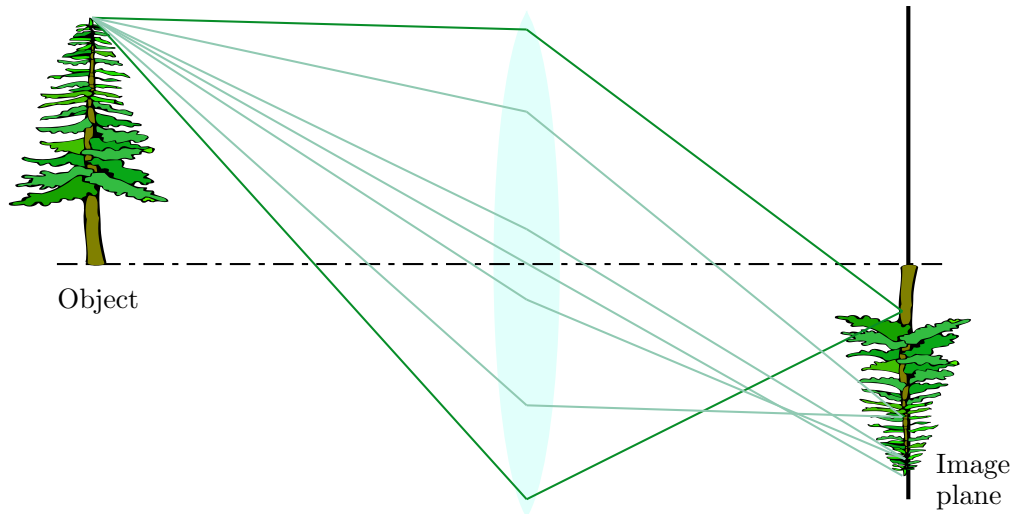


Figure 2.5: Coma prevents object points that are not on the optical axis from focusing at a single point. Although the light rays which pass through the central portion of the lens are focused on the correct location in the image plane, rays from the lens perimeter are incorrectly magnified.

of lost focus.

**Field curvature** Field (or Petzval) curvature images planar objects onto a non-planar focal surface (Figure 2.7). It is a relationship between focal length and the distance an object point is from the optical axis. This is closely related to astigmatism since both produce non-planar surfaces which are in focus.

**Distortion** Distortion denotes any deviation from geometric similarity between an object and its image. For a rotationally symmetric lens this takes the form of a scaling in the distance from the optical axis, or radial distortion (Figure 2.7). Although this radial

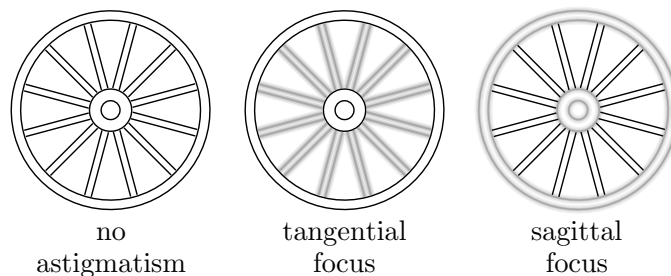


Figure 2.6: Astigmatism affects the sharpness of an image, but depends on the local orientation of the feature. Tangential focus has blurred the radial spokes, while sagittal focus blurs the hubs and rim. This separation between the two planes of focus is an oversimplification; in practice the two overlap and vary spatially.

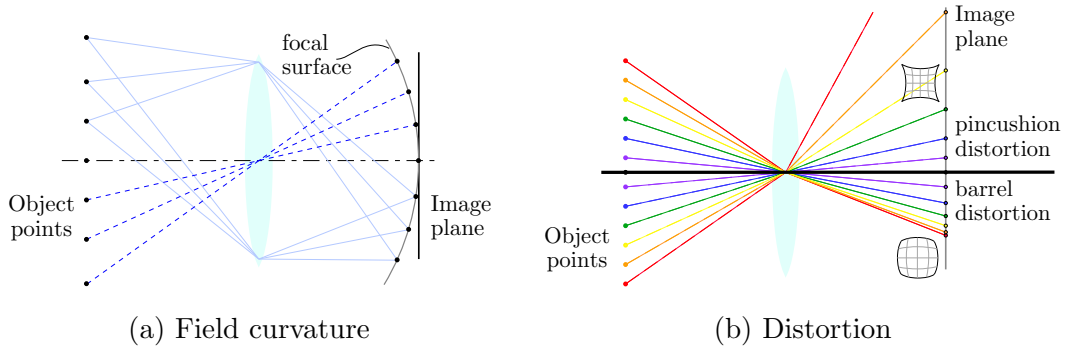


Figure 2.7: (a) Field curvature focuses planar points onto a curved focal surface. (b) Distortion results from spatial variation of the magnification power of a lens. Most distortion is radial, and a given lens will exhibit either pincushion or barrel distortion.

component is often the dominant distortion, any spatial variation in the magnifying power of a lens will induce some form of distortion. Compensation for this type of aberration can be performed in image space, and is examined in Chapter 3.

All of the Seidel aberrations describe a deviation from the pinhole camera assumption, and thus give rise to non-central cameras. The next section describes computer vision models for such cameras.

### 2.1.3 Non-central cameras

The cameras discussed so far have all been central; that is, all light rays pass through a single point (the *camera centre* or pinhole). This is a reasonable assumption for most modern optics. However, there is a large class of optical instruments which are clearly non-central. These include fish-eye lenses, catadioptric cameras, and many flatbed scanners.

Removing the constraint that all light rays must pass through a single point allows us to model cameras that are non-central by construction (such as fish-eyes and some catadioptric systems) as well as sets of images from multiple perspective viewpoints. In the latter case, pixels from several pinhole images are treated as coming from a single non-central camera. This construct has been successfully applied to the problems of mosaicing (Pajdla 2002) and image based rendering (Rademacher and Bishop 1998).

A typical pinhole camera has too narrow a field of view for many applications. In an effort to increase the field of view of a lens yet still have a planar focal volume (to match the planar CCD) the central point constraint is often relaxed or discarded from the design criteria altogether. Such lenses induce parallax effects in the outer regions of an image, even under pure rotation. A linear pushbroom camera consists of a 1D sensor array that records an image as the camera is moved, often through rotation

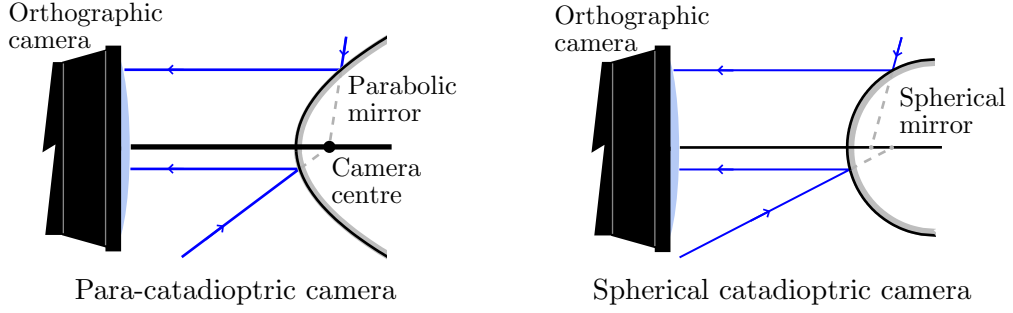


Figure 2.8: *Left* A para-catadioptric camera is constructed from a parabolic mirror and an orthographic lens. This produces a central camera; all the world rays project onto a single point, even though that point is neither inside the camera nor does the light actually pass through it. *Right* A catadioptric camera with a spherical mirror is non-central because the light rays do not intersect at a single point.

(orbiting satellite based photogrammetry) or translation (flatbed document scanners). Each line of pixels in the 2D image is recorded from a different position, which ensures a non-central camera. Fish-eye lenses compose another common example of cameras that are non-central by construction. The light rays that pass through the perimeter of the lens are furthest from the virtual camera centre (the point that is closest to all light rays passing through the lens). This divergence causes motion parallax even in images taken by rotating the camera about its virtual centre. Even so the circle of least confusion is typically small enough that the central assumption provides a reasonable approximation.

Catadioptric cameras enable the field of view to be expanded even beyond what is attainable with a fish-eye lens. A catadioptric system consists of a camera (generally pinhole but orthographic cameras are often used) viewing light reflected from a mirror surface. This mirror is typically a surface of revolution; the use of a parabolic mirror (para-catadioptric system) produces a central camera, provided the lens is orthographic and the axes are collinear. A hyperbolic mirror and a pinhole camera also constitute a central system. The following paragraph outlines the unifying theory for central camera projections, as described by Geyer and Daniilidis (2000).

**Spherical-perspective projection** Central panoramic cameras can be represented as a projection onto a sphere combined with a subsequent perspective projection onto a plane (Geyer and Daniilidis 2000), as illustrated in Figure 2.9. The homogeneous world point  $\mathbf{X} = [x \ y \ z \ w]^\top$  is projected (via the origin) onto the sphere as  $\mathbf{q} = [x \ y \ z]^\top / r$  and  $\mathbf{q}' = -[x \ y \ z]^\top / r$ , where  $r = \sqrt{x^2 + y^2 + z^2}$ . The image point  $\mathbf{x} = [i \ j \ -a]^\top \simeq [i \ j]^\top$

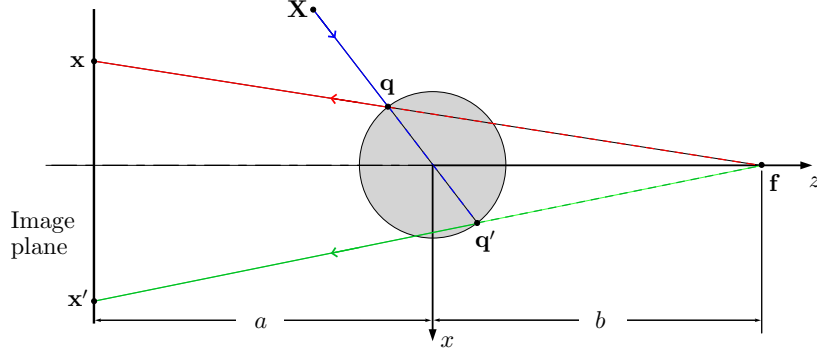


Figure 2.9: Projection onto a sphere followed by perspective projection onto the image plane. Catadioptric and radially distorted pinhole image formation can be modelled as this type of double projection. Note that points  $\mathbf{q}$  and  $\mathbf{q}'$  are projected onto the image, they are not points of reflection; the sphere is not a mirror.

is given by

$$i = \pm x \left( \frac{b+a}{br \mp z} \right) \quad (2.4)$$

$$j = \pm y \left( \frac{b+a}{br \mp z} \right). \quad (2.5)$$

Perspective projection is obtained by setting  $a = 1, b = 0$  so that  $\mathbf{x}$  becomes  $(-x/z - y/z + 1)$ . When  $a = 0, b = 1$  this double projection yields a special case of stereographic projection.

Consider the parabolic catadioptric camera with orthographic projection illustrated in Figure 2.8. If the paraboloid mirror is defined as  $z = f - \frac{1}{4f}(x^2 + y^2)$  then the projection from world coordinates to the image plane is

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} c_x + \frac{2fx}{-z + \sqrt{x^2 + y^2 + z^2}} \\ c_y + \frac{2fy}{-z + \sqrt{x^2 + y^2 + z^2}} \\ 1 \end{pmatrix} \quad (2.6)$$

As shown in Figure 2.8, a spherical mirror produces a non-central catadioptric camera.

The methods for scene reconstruction using para-catadioptric cameras were laid out by (Peleg et al. 2001, Shum and He 1999). This was extended to establish the epipolar constraint for general non-central cameras (Pajdla 2001). Sturm (2002) presented models for back-projection of rays into affine, perspective and para-catadioptric images. These models permit the mixing of catadioptric and perspective cameras by defining the fundamental matrices between such image types. The paper also extends



calibration transfer and plane-homography self-calibration to these types of views. The work was later extended to incorporate structure from motion with more general camera types (including non-central), both for triangulation and analytical bundle adjustment (Ramalingam and Sturm 2004).

This was taken a step further by Grossberg and Nayar (2001) to work with general cameras. This treats a camera as a black box which maps light rays onto a 2D image, but there need not be any sort of ordering or correlation between adjacent pixels. Each pixel samples from an arbitrary ray in 3D space. Although this approach disregards the structure inherent in physical optical devices, it facilitates the analysis of unique collections of cameras and has stimulated much recent work in the field of computer vision (Nistér et al. 2005, Ramalingam et al. 2005, Stewénus et al. 2005). Pless (2003) defined the generalized epipolar constraint, and Sturm (2005) put forward multiview geometry for non-central cameras using the general model.

### 2.1.4 Cubic Polynomial Camera

Hartley and Saxena (1997) proposed a cubic polynomial camera to model a wide variety of imaging sensors. The transformation from world points to image coordinates is given by

$$\mathbf{x} = \pi(\mathbf{C}\chi_3(\mathbf{X})) \quad (2.7)$$

where

$$\chi_3(x, y, z) = [x^3 \ x^2y \ x^2z \ xy^2 \ xyz \ xz^2 \ y^3 \ xy^2 \ y^2z \ z^3 \ xz^2 \ yz^2 \ x^2 \ xy \ xz \ y^2 \ yz \ z^2 \ x \ y \ z \ 1]$$

and  $\mathbf{C}$  is a  $22 \times 3$  matrix of coefficients. Fitting the coefficients of so many parameters directly from observed data was found to be an extremely noise-sensitive operation.

The generality and modelling power of the polynomial camera has led to its adoption as a standard means for providing the properties of the imaging device used to record a set of images (Tao and Hu 2001). It has become something of a *lingua franca* in the satellite imaging industry, facilitating the use of and translation between various imaging models. It allows a vendor to supply the parameters required for stereo reconstruction without having to disclose the methods and models used for their particular camera calibration. Generic photogrammetry and analysis software can then load the rational polynomial parameters for post-processing. Because the specific calibration procedure used to generate the polynomial coefficients is tailored to the camera (and thus has a lower overall parameter count) the noise and over-fitting problems associated with the polynomial camera model applied to real data are avoided.

A camera model based on rational functions will be presented in Chapter 3. We show that it provides a very general camera model while accurately fitting the physics of the camera/lens system with a minimal number of parameters.

## 2.2 Camera Calibration

Camera calibration involves determining the mapping between world rays and image pixels for a specific camera. The techniques used for measuring the camera parameters depend upon which underlying model will be used. It is possible to calibrate a camera by taking physical measurements of its components on an optical bench, with the aid of a collimator and goniometer. However, if a lens has already been assembled and mounted in a camera this process is physically so awkward that is not a practical camera calibration method (Brown 1971). Thus, I shall focus on camera calibration by optical methods. Typically an image or sequence of images are taken of a scene or object that possesses some particular geometry. That scene property is then exploited to compute the camera model coefficients. The scene properties used for calibration vary greatly depending on the chosen model and calibration technique. Some procedures require that the camera be held in a specific position relative to some precisely machined object (Foxlin and Naimark 2003), some require straight lines (Devernay and Faugeras 2001) or circles (Kim et al. 2005), while others only ask for multiple views of a static scene recorded by the same camera. This last type establishes point correspondences between the images in order to use the underlying 3D positions as invariants for calibration. This is commonly referred to as camera auto-calibration because no specialist knowledge or object is required, only generic footage from the camera.

Camera calibration is often divided into two phases: internal calibration to determine the parameters of the camera itself, and external calibration to establish its position in some world coordinate frame. Compensating for nonlinear lens distortion is one aspect of internal calibration. Many cameras can be modeled as pinhole projection systems once the distortion has been removed from an image. The transformation between world points and image points is then a projective homography. Given a set of world/image point correspondences it is a straightforward task to compute this homography  $P$ ; it is often desirable to separate into internal and external camera parameters:

$$P = K [R \mid \mathbf{t}]. \quad (2.8)$$

Here  $K$  is the matrix of camera internal parameters that describes the pinhole model and the rotation  $R$  and translation  $\mathbf{t}$  specify the external parameters or camera orientation

in space.

### 2.2.1 Internal Calibration

Internal calibration measures the parameters that pertain to the camera itself; they are independent of the camera's orientation in space. Prior to discussing the history of pinhole calibration with lens distortion, we shall turn our attention to general camera calibration, the most recent statement of the problem (Grossberg and Nayar 2001).

**General Calibration** The general camera model states that a camera acquires images consisting of pixels; each pixel captures light that travels along a ray in 3D. In this way we can relate world rays to camera pixels without any parametric model. These rays need not pass through the photosensitive element associated with the pixel; alternatively they can be constrained to pass through a virtual element that is at some arbitrary location in space. Generally it is safe to assume that the photosensitive elements in a conventional camera are adjacent to one another and can be integer indexed, but they may be on a 3D curved surface or scattered through space.

An early formulation of a general camera was provided by Grossberg and Nayar (2001), who also described a patterned light method for calibration. One of the chief challenges encountered when working with the general camera model comes in representing the ray directions for all of the pixels in an image. Grossberg and Nayar handled this by associating each pixel with a position and tangential direction on a caustic surface.

Another means of recording the direction for each pixel is to calibrate a subset of the pixel directions and interpolate between these for the remaining pixels. This is the approach taken in the efficient and straightforward method for calibration model introduced by Sturm and Ramalingam (2004). This closed form algorithm provides calibration of unknown camera positions from three or more images recording some known structure. We will now describe this calibration technique in a little more detail in order to better understand the advantages and disadvantages of the general camera model.

The first step is to record three images of the same object in different positions. Assign the coordinate frame attached to the object in the first image to be the common frame in which the camera rays will be determined. The unknown motions between this frame and the remaining two are given by  $\mathbf{R}'$ ,  $\mathbf{R}''$  and  $\mathbf{t}'$ ,  $\mathbf{t}''$ . Throughout this section we will denote the second view with a single prime and the third view with a double prime.

The light ray from a single pixel will intersect the object in a different position for each view; these 3D positions will be denoted as  $\mathbf{Q}$ ,  $\mathbf{Q}'$  and  $\mathbf{Q}''$  respectively. Expressed in the common reference frame these positions are

$$\begin{aligned} & \mathbf{Q} \\ & [\mathbf{R}' | \mathbf{t}'] \mathbf{Q}' \\ & [\mathbf{R}'' | \mathbf{t}''] \mathbf{Q}'' . \end{aligned} \tag{2.9}$$

In the common frame, each of these points must lie along the camera ray for that pixel; the points must be collinear. This constraint means that the following matrix must have rank less than 3 (it is comprised of three collinear points):

$$\begin{bmatrix} Q_1 & [\mathbf{R}'_1 t_1] \mathbf{Q}' & [\mathbf{R}''_1 t_1] \mathbf{Q}'' \\ Q_2 & [\mathbf{R}'_2 t_2] \mathbf{Q}' & [\mathbf{R}''_2 t_2] \mathbf{Q}'' \\ Q_3 & [\mathbf{R}'_3 t_3] \mathbf{Q}' & [\mathbf{R}''_3 t_3] \mathbf{Q}'' \\ Q_4 & Q'_4 & Q''_4 \end{bmatrix} \tag{2.10}$$

where  $\mathbf{R}_i$  denotes the  $i^{th}$  row of a rotation matrix. Each  $3 \times 3$  submatrix must have zero determinant; these submatrices give rise to trifocal tensors whose coefficients depend upon the motion parameters. These can be estimated by solving linear equations; the camera parameters can then be extracted using the methods described in Sturm and Ramalingam (2003, 2004). This formulation does not require a central camera; simplified algorithms are given in the above references for central cameras. It is also possible to use planar calibration targets for both central and non-central cameras.

One aspect of the calibration that hasn't been discussed yet is the task of determining the locations  $\mathbf{Q}'$  and  $\mathbf{Q}''$  on the calibration object. Let us assume that a planar target printed with a rectilinear pattern of dots is to be used. The camera rays used for calibration will be those which pass through the image of each dot in the first (reference or common frame) view.

The point where the camera ray intersects the object is then one of the defined control points (a printed dot), and the physical coordinates of  $\mathbf{Q}$  are known. The corresponding image location  $\mathbf{q}$  then determines which camera ray we are solving for. This position doesn't correspond to a specific pixel; it is the center of the imaged dot interpolated to sub-pixel accuracy. For the subsequent views we need to determine the point on the calibration plane that corresponds to the image location  $\mathbf{q}$ . Because the camera and calibration target undergo arbitrary motion between views this location will not generally fall on one of the printed dots. An interpolation scheme is required to find the positions  $\mathbf{Q}'$  and  $\mathbf{Q}''$ . Sturm and Ramalingam use a homography to warp the four closest dots into the fronto-parallel plane and measure the location in that plane. This

assumes a locally linear (or pinhole) image and is not strictly accurate, particularly for images with large amounts of nonlinear distortion. However for moderate levels of distortion or a dense grid of dots this assumption yields reasonable results without introducing excessive computational overhead.

Because the correspondence matching for  $\mathbf{Q}'$  and  $\mathbf{Q}''$  depends upon the imaged grid of dots, this method is only able to calibrate the portion of the image that lies within the convex hull of the dot pattern in *all* images. For fisheye lenses and other cameras with large fields of view this can restrict the method to a small portion of the image. Bundle adjustment methods that allow the use of many input images and calibration that covers the entire image region have also been developed.

This method for general camera calibration yields precise values for the camera extrinsics: the rotation and translation between views. For central cameras it provides an accurate means to recover the position of the camera centre relative to a known object. The process of intersecting many 3D rays provides a solid constraint on the position of each view and permits an informed definition of the camera centre where a camera is only approximately central. Error measures such as the centre of least confusion or the mean shortest distance to each ray can be attached to proposed camera centres and used to evaluate their merits.

Many applications that depend upon camera calibration also require the camera's intrinsic parameters. These are not easily recovered from such a general camera model, but that isn't its purpose. The model is useful for nonparametric distortion correction. By intersecting the camera rays with a plane and then using the local homography method to warp the image it is possible to rectify images without any knowledge of the form of the distortion. The downsides to this approach are that it is piecewise linear (which may introduce "creases" into the image) and any error in detecting the centre of one of the original calibration dots translates into a local warping error in the final image.

In summary, the general camera model is a useful construct for dealing with multiple camera systems and for accurately determining a camera centre. As a nonparametric technique it is able to handle a wide variety of lens types without the need for special case code. One disadvantage to this type of calibration is that it requires at least four parameters (to specify a line in  $\mathbb{R}^3$ ) for each pixel. Such a verbose parametrization is unwieldy and not necessary for many cameras and applications. It is often desirable to reduce the number of coefficients by using a more specific camera model, such as that used by the pinhole calibration methods described in the next section.

**Pinhole Calibration** For the ubiquitous pinhole camera model these parameters are: focal length  $f$ , aspect ratio  $a$ , skew  $s$ , and principal point  $(u_0, v_0)$ . The pinhole projection model is linear as formulated. However most lenses in common usage are not linear due to the presence of lens distortion. Compensation for lens distortion is therefore also part of internal camera calibration using the pinhole model. Often this is treated as a separate task where distortion correction is performed on all input images prior to computing the classical pinhole parameters. This assumes that the pinhole parameters are independent of the radial distortion. Weng et al. (1992) has observed that these two aspects of a pinhole camera may not be sufficiently de-coupled for this assumption to hold; there is a dependency between focal length and radial distortion. Most auto-calibration methods assume that radial distortion is negligible, either due to the lens used or because it has already been compensated for. Those that do account for it tend to include a single radial distortion component.

Much of the theory for camera calibration within computer vision was initially developed within the discipline of photogrammetry. Brown published a series of papers that dealt with distortion of a magnitude and type that is relevant to the type of camera often used in computer vision. Brown (1966) described the decentering distortion caused by mis-aligned optical elements, and provided analytical means for calibrating such distortion. Most camera calibration methods rely upon images of a pattern of dots or corners; points whose 3D coordinates are known and that can easily be detected in an image (Tsai 1987, Weng et al. 1992). Brown (1971) introduced the concept of plumbline calibration. This method relies upon images of straight lines to compute the parameters of the distortion model.

In 1987 Roger Tsai published a pivotal paper on camera calibration for computer vision. He set forth the criteria for judging a camera calibration technique:

1. Autonomous - no user initialization required
2. Accurate - precise modelling of the imaging process
3. Efficient - able to be implemented efficiently
4. Versatile - able to handle a wide range of cameras, accuracy requirements and applications
5. Common equipment - should be able to accommodate consumer grade optics and camera technology

The key difference was that here was a straightforward method for using a readily available, yet clearly non-metric camera for accurate measurement. To achieve the first four aims while using off-the-shelf camera equipment it was necessary to carefully treat each step of the calibration process. Tsai's work served to define those steps for much of the computer vision community. He drew a clear distinction between

intrinsic and extrinsic camera parameters and stressed the need for lens distortion correction. Only by ignoring lens distortion is it possible to have a linear calibration algorithm. In his two stage procedure the rotation and  $t_x$ ,  $t_y$  for the camera are computed first, and the second stage computes the focal length, distortion parameters and  $t_z$ . The radial alignment constraint upon which this calibration technique is based relies on the assumption of only radial distortion. The method considers only the first two polynomial terms of radial distortion; it states that tangential distortion is insignificant and higher order radial terms merely cause numerical instability. While this may have been the case for the lenses being calibrated at the time, it will be shown in this work that modelling distortion beyond the first two radial terms is stable and increases the accuracy significantly. Nevertheless, Tsai's calibration technique based on the radial alignment constraint encompassed lens distortion and the two stage process was a reasonably straightforward means towards accurate calibration of the cameras in common usage.

Tsai (1987) also noted the difficulty of obtaining high accuracy ground truth for camera calibration. The three methods put forward in that work are 1) triangulate 3D positions from images recorded with the calibrated cameras and compare with physically measured ground truth, 2) use the calibrated camera parameters to project world points into the image and measure the error relative to the actual imaged points, and 3) make relative measurements of lengths on 3D objects within an image.

A different two stage calibration procedure was suggested by Weng et al. (1992). He computes all of the perspective camera parameters in an initial linear stage that does not account for lens distortion. This provides an estimate that can be used to initialize the nonlinear optimization in the second stage. This optimization differs from Tsai's second stage in that it updates *all* of the camera parameters, including those computed in the first stage. The interaction between the perspective and distortion parameters is minimized by updating each independently, with the other fixed. Tsai's distortion model is also expanded to include decentering and thin prism effects.

### 2.2.2 External Calibration

External calibration computes the camera position and orientation relative to some world coordinate system. This is often termed *camera localization*, a topic which is covered in detail in Chapter 6. Generally external calibration begins with known camera intrinsic parameters, but some of the calibration methods described in the previous section compute both sets of variables simultaneously. This is the approach taken by classical photogrammetry: all of the camera parameters are fed into a nonlinear

optimization that tunes the entire model in one step (Slama 1980, Triggs et al. 2000).

To compute the camera extrinsics for a pinhole camera with known intrinsic parameters it is necessary to separate the projection matrix  $P$  into  $P = K[R|t]$ . One simple method for performing this decomposition is described in §6.2.2. The accuracy of this decomposition depends upon the precision of the computed  $P$  matrix; Harker and O’Leary (2005) present an improved computation method ideally suited to this application. Another technique is the POSIT algorithm put forward by Dementhon and Davis (1995). Their method uses several iterative steps to update an initial affine camera model to full projective perspective.

All of the calibration methods described above minimize some error to obtain an optimal set of parameters. Often this minimization is concealed within a singular value decomposition (as with the DLT) and the error is the norm of a convenient design matrix. In the following section we examine non-linear optimization methods where this minimization is performed explicitly on a carefully selected error measure.

## 2.3 Nonlinear Optimization

Nonlinear optimization computes the solution (that is in some sense optimal) to a problem by varying the input parameters until the error reaches some preset threshold. The problem must be expressed in terms of some parameters with a function that computes an error measure from those parameters. Given these components and a reasonable initial estimate of the parameters (where reasonable depends on the shape of the functional manifold you have selected to convert the parameters into errors), an optimization routine is called to vary the parameters until the desired tolerances are reached. The way in which the parameters are varied from one iteration to the next is what differentiates the many optimization strategies. The next section describes the optimizer used in the context of this thesis, but first we shall discuss two common criticisms of iterative methods.

**Complaints** Two complaints often directed towards iterative methods in general, and non-linear optimizations in particular, are: 1) they are inexact, and 2) they take too long. The former statement is misleading and the latter is often not the case, particularly for real world problems.<sup>1</sup>

Optimization is often regarded as an inexact hack, to be attempted only when direct linear methods of solution cannot be tracked. The implication is that the resulting so-

---

<sup>1</sup>These ideas were introduced to me by Prof. L. N. Trefethen and have been published in (Trefethen 1992)



lution is an approximation, and not as accurate as one obtained by direct computation. In reality, every numerical computation on a computer is inexact; the best that can be achieved are errors on the order of machine precision. Iterative methods can converge to the level of machine precision, and therefore are able to match the level of accuracy achieved by any other means. In some cases they can return such an answer in less time than a direct method may require (Trefethen and Bau 1997). A direct method returns no answer until all  $O(m^3)$  steps have been completed, while an optimization method finishes as soon as the requested error tolerance is reached. For well-posed problems, or if functional derivatives are available, or efficient methods for evaluating the function are known, then this process might be very fast indeed. Section 6.3 demonstrates the efficient use of numerical optimization to rapidly obtain highly accurate camera positions.

### 2.3.1 Levenberg-Marquardt

Unless otherwise noted, all optimizations within this thesis were performed using the Levenberg-Marquardt method. MATLAB's `lsqnonlin` function was used for all except the realtime camera pose code; this used the implementation of LMDIF (Moré et al. 1980) in VXL (The TargetJr Consortium 2000).

The Levenberg-Marquardt (Levenberg 1944, Marquardt 1963)(*abbr.* LM) algorithm finds a minimum of  $F(x)$ , a sum of squares

$$F(x) = \frac{1}{2} \sum_{i=1}^m [f_i(x)]^2, \quad (2.11)$$

in a manner that is regularized and fast to converge, even for over-parametrized problems. The functions  $f_i(x)$  are typically nonlinear, though the steps taken for each iteration are linear.

From an initial starting point  $P_k$ , the algorithm progresses by selecting a search direction  $\mathbf{p}_k$ , advancing some distance  $\lambda_k$  in that direction, and re-evaluating the sum of squares at the new point  $F(P_{k+1})$ . These steps comprise one iteration; the search direction and distance are determined by the algorithm's update strategy. The LM update strategy is a hybrid of the Gauss-Newton iteration and gradient descent methods. Prior to discussing any of these methods we must set forth some notation and define the terms.

Given a (vector) set of measurements  $\mathbf{X}$  and a function  $F(\mathbf{P})$  that transforms the parameter vector  $\mathbf{P}$  according to

$$\mathbf{X} = F(\mathbf{P}), \quad (2.12)$$

we seek the set of parameters  $\hat{\mathbf{P}}$  in  $\mathbf{X} = F(\hat{\mathbf{P}}) - \epsilon$  so that the error  $\|\epsilon\|$  is minimized. Let us define

$$\begin{aligned} G(\mathbf{P}) &= \frac{1}{2} \|F(\mathbf{P}) - \mathbf{X}\|^2 \\ &= \frac{1}{2} \|\epsilon(\mathbf{P})\|^2 \\ &= \epsilon(\mathbf{P})^\top \epsilon(\mathbf{P}) / 2 \end{aligned} \quad (2.13)$$

so that our goal in this least squares minimization is to find the minimum of  $G(\mathbf{P})$ . The Taylor series expansion of (2.13) about some initial point  $\mathbf{P}_0$  is

$$G(\mathbf{P}_0 + \mathbf{D}) = G(\mathbf{P}_0) + G'(\mathbf{P}_0)\mathbf{D} + \frac{1}{2}\mathbf{D}^\top G''(\mathbf{P}_0)\mathbf{D} + \dots \quad (2.14)$$

where the prime denotes differentiation and the vector  $\mathbf{D}$  is some small change in all elements of the parameter vector. To minimize (2.14) we differentiate with respect to  $\mathbf{D}$  and set the result equal to zero:

$$\begin{aligned} G'(\mathbf{P}_0) + G''(\mathbf{P}_0)\mathbf{D} &= 0 \\ \implies G''(\mathbf{P}_0)\mathbf{D} &= -G'(\mathbf{P}_0). \end{aligned} \quad (2.15)$$

By differentiating (2.13) we see that the gradient vector  $G'(\mathbf{P}) = \epsilon'(\mathbf{P})^\top \epsilon(\mathbf{P})$ . The *Jacobian* matrix is defined as

$$\mathbf{J} = \frac{\partial F}{\partial \mathbf{P}}, \quad (2.16)$$

which can also be expressed as  $\mathbf{J} = F'(\mathbf{P}) = \epsilon'(\mathbf{P})$ . This last result allows the gradient to be given as

$$G'(\mathbf{P}) = \mathbf{J}^\top \epsilon(\mathbf{P}). \quad (2.17)$$

The second derivative matrix  $\mathbf{H}(\mathbf{P}) = G''(\mathbf{P})$  is the *Hessian* of  $G$ , with entries defined as

$$H_{ij} = \frac{\partial^2 G}{\partial p_i \partial p_j} \quad (2.18)$$

where  $p_i$  and  $p_j$  are the  $i^{th}$  and  $j^{th}$  parameters in  $\mathbf{P}$ . The Hessian and Jacobian allow us to express (2.15) as

$$\mathbf{J}^\top \epsilon(\mathbf{P}_0) = -\mathbf{H}(\mathbf{P}_0)\mathbf{D}. \quad (2.19)$$

The Jacobian and the error term are readily evaluated; the optimization algorithms vary in how they approximate the Hessian. In Newton iteration a quadratic cost function near the minimum is assumed, and the Hessian is approximated as  $\mathbf{H} = \epsilon'(\mathbf{P})^\top \epsilon'(\mathbf{P}) + \epsilon''(\mathbf{P})^\top \epsilon(\mathbf{P})$ . The Gauss-Newton method drops the quadratic term to assume a linear cost function:  $\mathbf{H} = \epsilon'(\mathbf{P})^\top \epsilon'(\mathbf{P}) = \mathbf{J}^\top \mathbf{J}$ . This change allows the approximation to often be computed much more efficiently, and converges well if the

initial estimate is close to the minimum so that the linearity assumption is reasonable. The gradient descent algorithm takes an entirely different approach and sets  $\mathbf{H} = \lambda \mathbf{I}$  where  $\mathbf{I}$  is the identity matrix and  $\lambda$  is some scalar. This sets the search direction to be the steepest descent; convergence is often slow and displays a characteristic zig-zag pattern.

As was mentioned earlier, the LM method is a hybrid of Gauss-Newton and gradient descent. The linear system to be solved (2.15) for LM is

$$(\mathbf{J}^\top \mathbf{J} - \lambda \mathbf{I}) \mathbf{D} = -\mathbf{J}^\top \epsilon. \quad (2.20)$$

The value of  $\lambda$  allows the algorithm to switch seamlessly between Gauss-Newton and gradient descent updates at each iteration. For each update, if the computed  $\mathbf{D}$  yields a reduced error then  $\lambda$  is reduced and execution proceeds to the next iteration. If, however, the solution obtained for  $\mathbf{D}$  does not yield a reduced error then  $\lambda$  is increased and  $\mathbf{D}$  re-solved until there is a reduced error. This corresponds to an iteration where a gradient descent update is preferred.

The performance of LM can be tweaked by adjusting the parameters that control  $\lambda$ : the step size increment/decrement as well as its initial value. As with any optimization strategy, the termination criteria can also be varied to suit the problem. For large problems in particular it is important that parameters be carefully grouped together to enhance the sparsity inherent in the derivative matrices. These matrices can be computed through numerical differentiation (finite difference derivatives) but the speed of the optimization can often be dramatically improved by supplying error functions complete with the analytic Jacobian. The key to success, however, is how closely the two primary assumptions are met: 1) there exists a well-defined minimum in the cost function space, and 2) the initial parameter estimate  $\mathbf{P}_0$  is reasonably close to that minimum.

### Important Practical Considerations

Numerical optimization bears some resemblance to a black art; at first it appears that one can assemble a few ingredients (write an error function), say a few magic words (call the optimization routine) and *voila!* the answer will appear by magic. In practice there are all sorts of ways in which this process can go very much awry. This section sets out a few points that were useful in keeping the minimizations used in this thesis under control.

Scale your parameter vector so that all entries stay within the range  $[-1 \dots 1]$ . Try to formulate the problem so that the distribution of entries in  $\epsilon$  is centered about zero.

While setting up an optimization algorithm it is often helpful to set the convergence parameters very loose (tolerance function of  $1 \times 10^{-3}$ ), and limit the number of iterations (typically to 200). These values are then tightened once the optimization is functioning correctly. Generally, I would set the tolerance function to  $1 \times 10^{-6}$ , maximum number of function evaluations to  $1 \times 10^6$  and total number of iterations to  $1 \times 10^8$ . The iteration counts should be set large enough that the algorithm converges to the tolerance function long before it executes too many iterations.

### 2.3.2 Bundle Adjustment

Within the field of photogrammetry, *bundle adjustment* refers to the process of updating an initial estimate of the camera parameters and world positions to more accurately match the observed image feature locations. It is an iterative method that fits a nonlinear model to measured point correspondences. Here it will be described to provide a tangible example of nonlinear optimization, and in particular the Levenberg-Marquardt algorithm.

This thesis doesn't directly pertain to bundle adjustment, but the technique is used to examine calibration results and demonstrate 3D reconstructions. The topic is briefly described here as a primer for readers who may be unfamiliar with this aspect of photogrammetry (further details are available in (Triggs et al. 2000), for example). A commercially available structure-from-motion software package, *boujou* (2d3 Ltd. 2003), was used to perform any bundle adjustment required for this thesis. It provided a high quality, versatile solver without the need to implement one from scratch.

Bundle adjustment is really the third and final stage in assembling a 3D reconstruction. The first is to establish image point correspondences between multiple views of a scene. These matches are then used to generate a preliminary reconstruction consisting of 1) a camera matrix  $P_i$  at each frame, and 2) a set of world point locations  $\mathbf{X}$ . Pinhole camera parameters are implicit in  $P_i$ , but additional distortion parameters are often also included in the representation. These values form the initial parameter vector estimate which will be refined in the bundle adjustment.

The components of the reconstruction are shown in Figure 2.10. Picture all the rays joining world points to their corresponding image locations (the blue lines in Figure 2.10) as a *bundle* of sticks or dowels. Bundle adjustment is a nonlinear optimization that *adjusts* all of these rays to obtain the optimal reconstruction. This is similar to twisting a handful of sticks back and forth until they all align in a tight bundle. The solution is optimal in that it minimizes the error based on the chosen image point inaccuracy model.

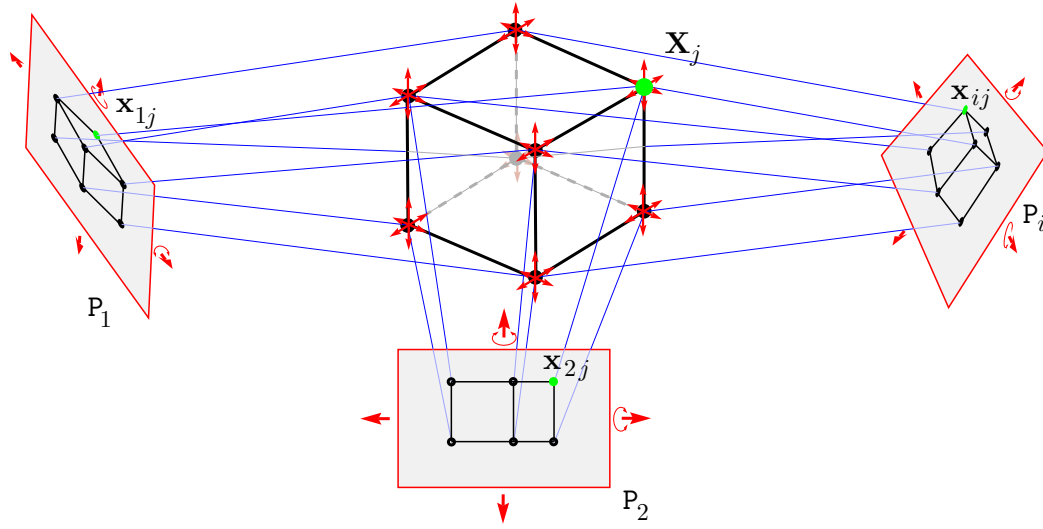


Figure 2.10: Bundle adjustment is the simultaneous optimization of world points and camera parameters over a multiple view sequence. The red arrows indicate the degrees of freedom, both for the world points  $\mathbf{X}_j$  and the camera matrices  $\mathbf{P}_i$ . Note that the drawing does not show all the degrees of freedom for the cameras.

Not all of the real world points will be visible in every view from the image sequence; some points will be obscured part of the time, and sometimes it will not be possible to establish a correspondence. These non-matched points lead to missing data in the optimization framework, and serve to enhance the inherent sparsity. Modern solvers take advantage of this sparsity so that a typical bundle adjustment problem can be solved surprisingly fast.

The choice of cost function affects the accuracy of the final result, the speed of convergence, and also the overall stability of the optimization. Mismatched point correspondences produce erroneous and misleading data (outliers) which the bundle adjustment software must be able to handle. For this reason robust cost functions are often preferred over the default quadratic error measure.

Careful planning prior to recording the video shoot is important for successful 3D reconstruction. The camera path and the number of easily recognizable features both affect the results. The distance between furthest camera positions (baseline) should be as large as possible; including both rotation and translation aids in avoiding degenerate camera paths. The detected features should neither be coplanar nor grouped in a subset of the image area. The resulting reconstruction should be checked for plausibility: triangulated points to the rear of the camera often indicate an inverted solution, while reprojected points that jump from one location to another between frames may be the

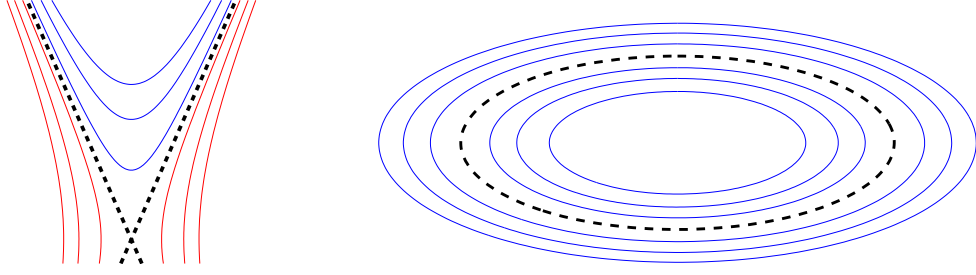


Figure 2.11: Concentric conic sections: hyperbolas and ellipses. These families represent equal error contours under Bookstein’s conic fitting error measure. Points along the “flat” portions of the conics receive undue weighting; this is the basis for the curvature based re-weighting in the Sampson distance.

result of mismatched point correspondences. Analytic evaluation of the reconstruction accuracy requires some knowledge of the true positions of the reconstructed points. Such position data is often not available. An alternative is to measure the reconstruction accuracy based on the precision of the computed camera path. Section 6.5.1 reports a set of test reconstructions where camera path ground truth was obtained independently to provide some measure against which to evaluate the reconstruction results.

## 2.4 Miscellaneous

This section describes several approximations and algorithms useful in distortion correction, camera calibration and scene reconstruction. The background on these techniques is not essential for understanding this thesis, so readers who are pressed for time or are already familiar with these subjects are free to proceed to Chapter 3 without loss of continuity.

### 2.4.1 Sampson Distance

The Sampson distance (Sampson 1982) is a first order approximation to the distance from a point to a conic. Computing the Euclidean distance along the normal to the conic requires solving a quartic equation and is therefore too expensive to be included in the inner loop of an optimization.

Sampson proposed an improvement to Bookstein’s (1979) conic fitting error measure that amounts to a re-weighting by the inverse of the gradient magnitude at each point. Bookstein’s error of fit is

$$Q(i, j) = \theta \chi(i, j) = A_{xx}i^2 + A_{xy}ij + A_{yy}j^2 + A_xi + A_yj + A_0 \quad (2.21)$$

where  $\theta$  are the conic parameters and  $\chi$  is the lifted point vector. Note that the

contours of equal error are conic sections concentric with the fit conic  $Q(i, j) = 0$ . This error measure tends to over-fit in the “flat” portions of conics where the concentric sections are closer to each other (see Figure 2.11).

Sampson suggested approximating the distance from a point to a conic by the perpendicular distance from the concentric conic through the point (shown in Figure 2.12). A linear approximation to this distance is

$$Q(i_{a''}, j_{a''}) \approx Q(i_a, j_a) - d'' |\nabla Q(i_a, j_a)| \quad (2.22)$$

where  $|\nabla Q(i_a, j_a)|$  is the norm of the concentric conic’s gradient at  $\mathbf{a}$ . Since  $\mathbf{a}''$  is on the conic,  $Q(i_{a''}, j_{a''}) = 0$  and

$$d'' = \frac{Q(i_a, j_a)}{|\nabla Q(i_a, j_a)|} \quad (2.23)$$

is the approximate distance we wish to perform the optimization with. Expressed in terms of the conic parameters, our minimization is

$$\sum_{k=1}^n \left[ \frac{\boldsymbol{\theta}^\top \boldsymbol{\chi}(i_k, j_k)}{[(2A_{xx}i_k + A_{xy}j_k + A_x)^2 + (2A_{yy}j_k + A_{xy}i_k + A_y)^2]^{1/2}} \right]^2 \quad (2.24)$$

where the denominator represents the re-weighting by the cross-product matrix.

### 2.4.2 Camera ringing

A digital image of a white/black transition (*e.g.* a checkerboard pattern, or a fiducial from Chapter 7) does not exhibit a perfect step-edge in intensity. Rather, there is an

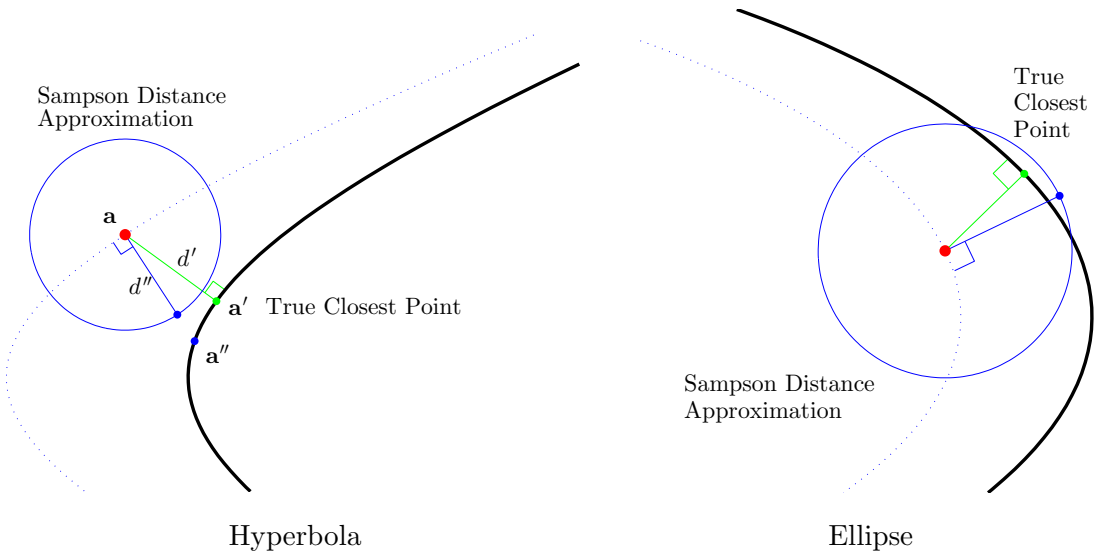


Figure 2.12: Sampson distance is a linear approximation to the distance from a point to a conic.

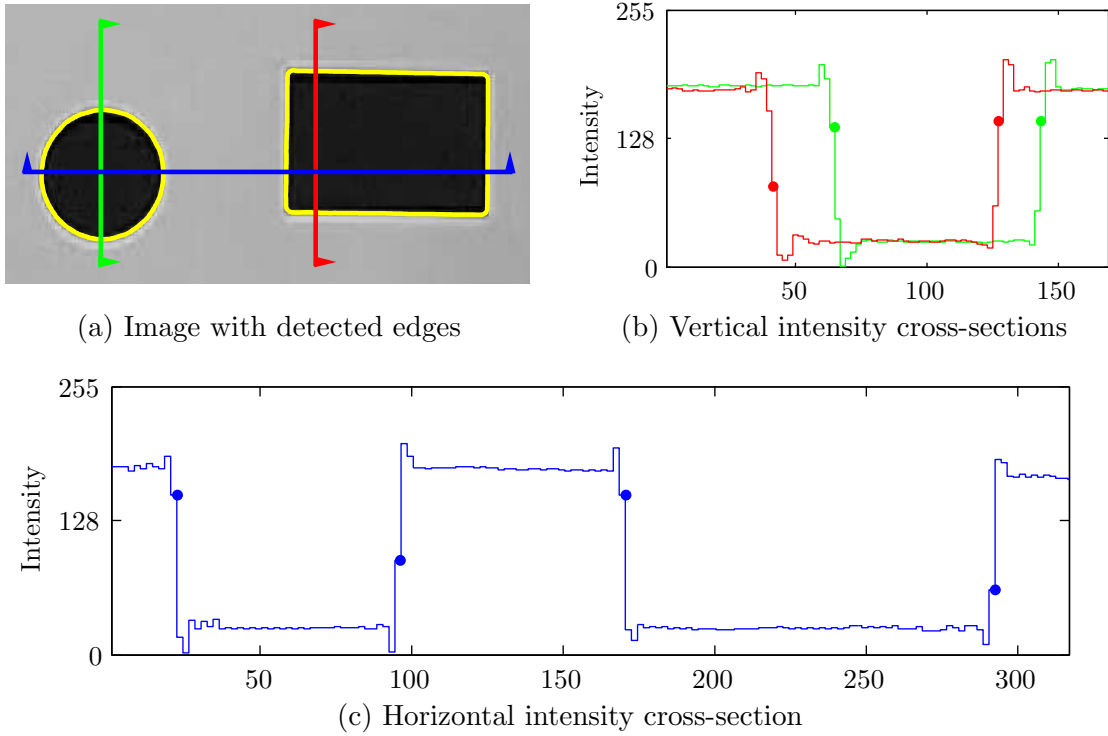


Figure 2.13: Electronic ringing causes spikes in intensity at the step edges in an image; this can bias the location of Canny detected edges. (a) An image of two black and white shapes with detected edges shown in yellow. (b) A vertical cross-section of the image intensity shows the spikes at each step change in intensity. The Canny-detected edges are marked by round dots; the detections are nearly balanced in this direction. (c) The horizontal cross-section also exhibits intensity spikes, but here the detected edges are biased to the left.

intensity spike at one or both intensity changes, as shown in Figure 2.13. This effect is known as “ringing” and results from the camera electronics’ inability to reproduce the step edge. In computer vision, the result can be an inaccuracy in detecting image points corresponding to large changes in greylevel. Subpixel Canny edge detection (Canny 1986) is used in the course of this thesis to locate the perimeter of dots. This detection method can return spurious results at either edge of the spike, and the true edge detections can be biased. Due to the sequence in which pixel intensities are read off a chip, ringing is often symmetric only in one direction. In the example of Figure 2.13 the ringing on horizontal edges is not symmetrical and there is an overall downward bias in the horizontal Canny edges.

### 2.4.3 Direct Linear Transform

The Direct Linear Transform (DLT) (Sutherland 1963) is a method for computing the matrix  $\mathbf{H}$  that relates two corresponding sets of homogeneous points  $\mathbf{x}' = \mathbf{H}\mathbf{x}$ . This



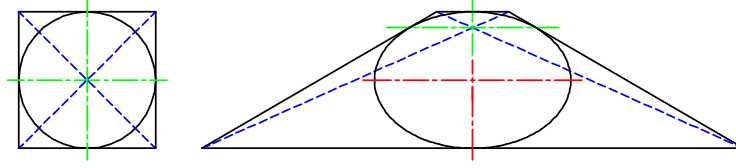


Figure 2.14: Centres of circles and squares under projective transformation. Note that the centre of the square can still be located as the intersection of the diagonals. The circle transforms to an ellipse whose centre of gravity no longer corresponds to the centre of the original circle.

technique has applications in homography estimation, in determining the projection of world points to image points, and in fundamental matrix computation; in §4.3.1 the method is used on Veronese lifted points to compute a  $6 \times 6$  fundamental matrix.

Here we shall briefly outline the DLT method; a more detailed presentation can be found in Hartley and Zisserman (2003). Consider two points  $\mathbf{x}_i = (x_i, y_i, w_i)^\top$  and  $\mathbf{x}'_i = (x'_i, y'_i, w'_i)^\top$ . By assuming that  $w'_i = 1$  and with a little rearranging we see that each point correspondence gives rise to two equations

$$\begin{bmatrix} \mathbf{0}^\top & -\mathbf{x}_i^\top & y'_i \mathbf{x}_i^\top \\ \mathbf{x}_i^\top & \mathbf{0}^\top & x'_i \mathbf{x}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^{1\top} \\ \mathbf{h}^{2\top} \\ \mathbf{h}^{3\top} \end{pmatrix} = \mathbf{0}. \quad (2.25)$$

Equations from each point correspondence are then stacked into a  $2i \times 9$  design matrix  $\mathbf{D}$  so that  $\mathbf{D}\mathbf{h} = \mathbf{0}$ . Four point correspondences yield an exact solution for  $\mathbf{h}$  that can be rearranged into the solution  $\mathbf{H}$ .

In the over-determined case where more than four point correspondences are known, the Singular Value Decomposition (SVD) is generally used. Then the solution  $\mathbf{h}$  is taken to be the unit singular vector corresponding to the smallest singular value. This amounts to finding the matrix  $\hat{\mathbf{D}}$  which is closest to  $\mathbf{D}$  in Frobenius norm, and for which an exact solution does exist. This minimizes  $\|\mathbf{D}\mathbf{h}\|$  subject to the constraint  $\|\mathbf{h}\| = 1$ . Although convenient, this constraint has no geometric significance and the resulting solution can be unstable at even moderate levels of image noise. Careful data normalization can help to stabilize the solution, and should be included in any DLT scheme. Discussions of normalization for the DLT can be found in Hartley (1997), Chojnacki and Brooks (2003) and Hartley and Zisserman (2003); an example of its importance is described in §5.4.

#### 2.4.4 Centre Shift Under Projection

As shown in Figure 2.14, the centres of both circles and squares shift under projective transformation. The original centre of the square (marked in green) can still be located

as the intersection of the diagonals; this requires fitting lines to a large number of edge sample points, determining the corners by intersection, and finally the centre by another intersection. The centre of the ellipse formed by the circle projection can be efficiently and accurately computed as a weighted centroid. However, this can only yield its centre (marked by the red cross), which no longer corresponds to the original centre position (marked by the green cross). Thus it is not possible to recover the centre of a single circle.

### 2.4.5 Pattern Classification

The fiducial detector described in Chapter 7 employs a series of classifiers to label every pixel within an image as either target or non-target. This section provides a review of several non-parametric classification strategies as background. Particular emphasis is given to the nearest neighbour technique, which despite its simplicity is regaining influence as processing speeds increase and memory limitations are relaxed.

Given two classes of points in  $n$ -D space, a “classifier” seeks to assign additional query points to the correct class, often by employing knowledge of the probability density function over each class. For example, let us consider a binary classification problem where each 2-D point (2-D being chosen for ease of visualization only) is in one of two classes: positive or negative. The training data consists of a set of points labeled positive and a set labeled negative; these sets are used to generate the probability density functions for each class. For an observed point  $x$  the classifier produces the positive and negative *a posteriori* probabilities (or *loss functions*)  $g_p(x)$  and  $g_n(x)$ . The class of the query point is then assigned according to:

$$\text{classification}(x) = \begin{cases} +1 & \text{if } \lambda \cdot g_p(x) > g_n(x) \\ -1 & \text{otherwise} \end{cases} \quad (2.26)$$

where  $\lambda$  is the relative cost of a false negative over a false positive. Classifiers include neural networks, support vector machines, and nearest neighbour techniques; they vary in the way that the loss functions are defined.

### Ideal Bayes Classification

With only the *a priori* probability for each of two states  $\omega_1$  and  $\omega_2$ , the Bayes decision surface that results in the lowest error rate is simply:

$$\text{decide } \omega_1 \text{ if } P(\omega_1) > P(\omega_2); \text{ otherwise decide } \omega_2.$$

When a measurement  $x$  is incorporated this surface becomes:

$$\text{decide } \omega_1 \text{ if } P(\omega_1|x) > P(\omega_2|x); \text{ otherwise decide } \omega_2.$$

Through the application of Bayes Rule this can be written in terms of conditional and *a priori* probabilities:

$$\text{decide } \omega_1 \text{ if } p(x|\omega_1)P(\omega_1) > p(x|\omega_2)P(\omega_2); \text{ otherwise decide } \omega_2.$$

For problems where the the state-conditional probability density  $p(x|\omega_j)$  can be computed this provides the classification  $\omega_j$  which will minimize the probability of error.

### Nearest Neighbour

Among the various methods of supervised statistical pattern recognition, the nearest neighbour rule (Cover and Hart 1967) achieves consistently high performance (Ripley 1997). The strategy is very simple: given a training set of examples from each class, a new sample is assigned the class of the nearest training example. In contrast with many other classifiers, this makes no *a priori* assumptions about the distributions from which the training examples are drawn, other than the notion that nearby points will tend to be of the same class.

For a binary classification problem given sets of positive and negative examples  $\{p_i\}$  and  $\{n_j\}$ , subsets of  $\mathbb{R}^d$  where  $d$  is the dimensionality of the input vectors, the NN classifier is formally written as:

$$\text{classification}(x) = -\text{sign}(\min_i \|p_i - x\|^2 - \min_j \|n_j - x\|^2) \quad (2.27)$$

This is extended in the  $k$ -NN classifier, which reduces the effects of noisy training data by taking the  $k$  nearest points and assigning the class of the majority. The choice of  $k$  should be performed through cross-validation, though it is common to select  $k$  small and odd to break ties (typically 1, 3 or 5).

One of the chief drawbacks of the nearest neighbour classifier is that it is slow to execute. Testing an unknown sample requires computing the distance to each point in the training data; as the training set gets large this can be a very time consuming operation. A second disadvantage derives from one of the technique's advantages: that *a priori* knowledge cannot be included where it is available. Both have been the subject of much research, which is summarized in the remaining part of this section.

### Speeding Up Nearest Neighbour

There are many techniques available for improving the performance and speed of a nearest neighbour classification (Wilson and Martinez 2000). One approach is to pre-sort the training sets using  $kd$ -trees (Sproull 1991) or Voronoi cells (Berchtold et al. 1998), however these strategies become less effective as the dimensionality of the data increases.

```

store = {pos(1), neg(1)}
grabbag = {pos(2 : n), neg(2 : m)}

repeat
    num_transfers = 0
    for i = 1 : length(grabbag)
        if classify_NN(grabbag[i], store) ≠ class(grabbag[i]) then
            grabbag = grabbag \ {grabbag[i]}
            store = store ∪ {grabbag[i]}
            num_transfers = num_transfers + 1
    until num_transfers = 0 or is_empty(grabbag)

return store

```

Figure 2.15: Condensed nearest neighbour dataset reduction technique (Hart 1968).

Another solution is to choose a subset of the training data such that classification by the 1-NN rule (using the subset) approximates the Bayes error rate (Ripley 1997). This can result in significant speed improvements as  $k$  can now be limited to 1 and redundant data points have been removed from the training set. These data modification techniques can also improve the classifier performance because points that cause mis-classifications can be removed. Two of the many techniques for obtaining a training subset will be examined here: condensed nearest neighbour and edited nearest neighbour.

The problem that these two techniques solve is to choose  $S \subset T$  such that the performance of a 1-NN search on  $S$  is equal to the performance of a  $k$ -NN search on  $T$ , and the size of  $S$  is minimized.

**Condensed Nearest Neighbour** The condensed nearest neighbour algorithm (Hart 1968) is a simple pruning technique that begins with one example in the subset and recursively adds any examples that the subset misclassifies. The algorithm is given in Figure 2.15 and a binary classification example is shown in Figure 2.16. Drawbacks to this technique include sensitivity to noise and no guarantee of the minimum consistent training set. The minimum consistent training set is the smallest subset of training examples that will correctly classify all of the original training examples (see Figure 2.16). Condensation is an incremental technique: the subset is built up by adding samples one at a time. The first few samples to be added have a larger effect on the final subset as there are only a couple of (possibly misleading) samples present with which to classify them. These effects are minimized if the subset is initialized with a representative positive and negative sample.

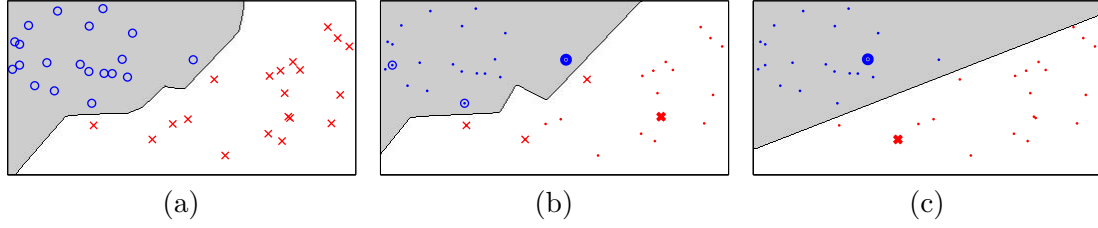


Figure 2.16: Example of condensed nearest neighbour technique for removing redundant training data. (a) The complete two class dataset; the nearest neighbour classifier will label any point within the shaded region as a circle. (b) Condensing drastically reduces the total number of examples. Note that the decision boundary has moved, but the new boundary still correctly classifies all of the original training examples. The two bold points denote the (randomly selected) initialization points. (c) Minimum consistent dataset obtained by manually specifying the initialization points. The boundary defined by these two initial samples (shown in bold) correctly classifies all the original training data (shown as dots for reference).

### Parameterization of Nearest Neighbour

Another enhancement to the nearest neighbour classification involves favouring specific training data points through weighting (Cost and Salzberg 1993). In cases where the cost of a false positive is greater than the cost of a false negative it is desirable to weight all negative training data so that negative classification is favoured. This cost parameter allows a ROC curve (see §2.4.6) to be constructed, which is needed later.

To introduce this weighting, a transition is made from a strict nearest neighbour classifier to a Parzen window density estimator (Duda et al. 2001) with a Gaussian smoothing function. The Parzen window estimate of the density of positive examples  $\{p_i\}$  is:

$$f_p(x) = \frac{1}{n} \sum_i N(p_i, \sigma_p; x) \quad (2.28)$$

where  $n$  is the number of examples and

$$N(\mu, \sigma_p; x) = \frac{1}{\sqrt{2\pi}\sigma_p} e^{-\frac{(x-\mu)^2}{2\sigma_p^2}} \quad (2.29)$$

with an analogous formula for the negative distribution  $f_n(x)$ . Then the Bayes decision boundary occurs at  $f_n(x) = f_p(x)$ , and the classification is:

$$\text{classification}(x) = \begin{cases} +1 & \text{if } f_n(x) < f_p(x) \\ -1 & \text{otherwise} \end{cases} \quad (2.30)$$

If all the  $\sigma_p$  are small, then  $f_p(x) \approx \max_i N(p_i, \sigma_p; x)$  and the classifier output is then determined by the maximum probability at a given test point:

$$\text{classification}(x) = \begin{cases} +1 & \text{if } \max_i N(p_i, \sigma_p; x) < \max_j N(n_j, \sigma_n; x) \\ -1 & \text{otherwise} \end{cases} \quad (2.31)$$

If  $p$  and  $n$  denote the points which maximize the positive and negative probabilities respectively, this amounts to a scaling and offsetting of the negative distances. This can be seen as follows:

$$N(p, \sigma_p; x) < N(n, \sigma_n; x) \quad (2.32)$$

$$\frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{(x-n)^2}{2\sigma_n^2}} > \frac{1}{\sqrt{2\pi}\sigma_p} e^{-\frac{(x-p)^2}{2\sigma_p^2}} \quad (2.33)$$

$$(x-p)^2 > \left(\frac{\sigma_p}{\sigma_n}\right)^2 (x-n)^2 + 2 \cdot \sigma_p^2 \ln \left(\frac{\sigma_n}{\sigma_p}\right) \quad (2.34)$$

If the distance metric for the classifier is then defined to be the square of the Euclidean distance and the weight  $w = (\sigma_p/\sigma_n)^2$  this simplifies to:

$$d_{\text{pos}} > w \cdot d_{\text{neg}} - \sigma_p^2 \ln w \quad (2.35)$$

The standard deviation of each class can then be varied to reflect the relative costs of false positives and false negatives. If  $\sigma_p$  is set to be small then there remains only the single parameter  $w$ .

This weighting  $w$  is the parameter varied to produce an ROC curve. If  $w$  is set to unity then positive and negative training points at equal distance from the sample point are given equal consideration. If the weighting is set to 0.5 then negative training data up to twice as far away as positive data will cause the example to be labeled as negative.

#### 2.4.6 Receiver Operator Characteristic Curves

A receiver operator characteristic (ROC) curve (Courtney and Thacker 2001) displays the response of the classifier as the weighting parameter is varied. This allows the ideal weighting to be determined for each specific set of costs. The axes on a ROC curve are conventionally *sensitivity* on the vertical and *1-specificity* on the horizontal where:

$$\text{sensitivity} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \quad (2.36)$$

$$1 - \text{specificity} = 1 - \frac{\text{true negatives}}{\text{true negatives} + \text{false positives}} \quad (2.37)$$

Thus sensitivity is the percentage of true samples correctly identified, and specificity is the percentage of negative samples correctly identified. Sometimes what should be a positive match will end up being closer to a negative training point. This results in a Type I error where a positive sample is rejected. At other times a negative match will successfully pose as a target dot, resulting in a Type II error.

## 2.5 Summary

This chapter has described the classical theory of image formation as well as some of the recent developments in the treatment of lens distortion and non-central cameras. It provides the context and background theory required for discussion of the proposed rational function model for lens distortion, camera localization, and fiducial detection.

The spherical projection and cubic polynomial camera models set the stage for the distortion model presented in the next chapter. Nonlinear optimization and its application in bundle adjustment are invaluable aids to accurate camera calibration. They can be defined up to an arbitrary level of precision and careful formation. These concepts are integral to the distortion correction, camera calibration and camera localization methods presented in this thesis. The pattern classification concepts are used to construct a robust fiducial detector based on fast, efficient matching to example images of fiducials (see Chapter 7). Light attenuation is another use where an understanding of the physics of image formation points to a highly successful correction technique. In this case the theory will be used for the photometric stereo application detailed in Chapter 8.

With the physical basis of image formation firmly in hand, and a full toolbox of important techniques described, we now turn our attention to the subject matter of this thesis. First on the block is lens distortion, covered in three separate chapters: modelling describes the theory, calibration describes how it is applied in practice, and a third chapter evaluates various distortion correction strategies.

## Chapter 3

# Lens Distortion: Modelling

Geometric lens distortion refers to the deviation of a lens and camera system from the pinhole perspective projection model. For high quality lenses, the pinhole projection model accurately reflects the physics of the camera. However, video and consumer digital cameras often exhibit significant levels of nonlinear lens distortion. From §2.1.2 we see that perspective projection assumes  $\sin \alpha = \alpha$  for small  $\alpha$ , while any extensions to higher order terms encompass “geometric distortions”. This chapter reviews a number of models for lens distortion, and then introduces the Rational Function model, a general purpose model for representing the nonlinearity observed in many lenses.

### 3.1 Distortion Models

Parametric lens distortion models tend to decompose the effects into radial and tangential components (Slama 1980). The geometric distortion listed in the Seidel aberrations (§2.1.2) is purely a radial effect, and therefore many lens models account for radial distortion. Lenses are designed to be rotationally symmetric, and modern optics are remarkably close to this ideal. However, small errors in alignment and centering of multi-element lenses give rise to tangential distortion.

Both radial and tangential distortion are measured relative to a point called the *centre of distortion*. This point is invariant under both types of transformation. In the model of Tsai (1987) the centre of distortion corresponds with the principal point, and is assumed to be at the centre of the image. He claims that although the true distortion centre may deviate from the image centre by up to 10 pixels, this has a negligible effect on the calibration. A note in the appendix of Tsai (1987) calls this statement into question; later Lenz and Tsai (1988) stated that the observed deviation can be as high as 40 pixels, and that it is important to actually measure the true distortion centre for applications requiring 3D measurements. The distortion centre need not coincide with



the principal point, and any deviation is manifested as decentering distortion (Devernay and Faugeras 2001). Decentering distortion (Brown 1966) results when the centres of curvature of the optical surfaces in a lens are not collinear; it has both a radial and a tangential component. A thorough discussion of the different types of image centres and methods for measuring them are given in by Willson and Shafer (1994).

Many of the camera calibration techniques used within the computer vision community have been adopted from the field of photogrammetry. The Manual of Photogrammetry (Slama 1980) is an excellent reference on such matters, and presents (§4.8.8.2) the following model for radial and decentering distortion. The mapping from image pixels  $(i, j)$  to distortion corrected points  $(p, q)$  is given by:

$$\begin{aligned} p &= (1 + k_1 r_d^2 + k_2 r_d^4 + \dots) i_d + [p_1(r_d^2 + 2i_d^2) + 2p_2 i_d j_d][1 + p_3 r_d^2 + \dots] \\ q &= (1 + k_1 r_d^2 + k_2 r_d^4 + \dots) j_d + [2p_1 i_d j_d + p_2(r_d^2 + 2j_d^2)][1 + p_3 r_d^2 + \dots] \end{aligned} \quad (3.1)$$

where

$(i_c, j_c)$  is the distortion centre  
 $r_d = \sqrt{(i - i_c)^2 + a^2(j - j_c)^2}$   
 $(i_d, j_d) = (i - i_c, j - j_c)$  are the image coordinates relative to the distortion centre  
 $k_1, k_2$  are the radial distortion parameters, and  
 $p_1, p_2, p_3$  are the decentering distortion parameters.

This model has been used extensively (Bouguet 2003, Brown 1971, Heikkilä 2000, Weng et al. 1992). Tsai (1987) observed that, for industrial machine vision applications, one term of radial distortion is sufficient to model a lens. Others have included additional polynomial distortion terms (Hartley and Zisserman 2003).

The *division model* (Fitzgibbon 2001) is an alternative to the above distortion functions. It also includes only one distortion term, but takes a simpler form in homogeneous coordinates and therefore finds more application in multiple view geometry. Undistorted coordinates are given by:

$$\frac{1}{(1 + \lambda r_d^2)}(i_d, j_d). \quad (3.2)$$

More complex models are used in photogrammetry. One example is the Bicubic model (Kilpelä 1980):

$$\begin{aligned} p &= b_1 i_d^3 + b_2 i_d^2 j_d + b_3 i_d j_d^2 + b_4 j_d^3 + b_5 i_d^2 + b_6 i_d j_d + b_7 j_d^2 + b_8 i_d + b_9 j_d + b_{10} \\ q &= c_1 i_d^3 + c_2 i_d^2 j_d + c_3 i_d j_d^2 + c_4 j_d^3 + c_5 i_d^2 + c_6 i_d j_d + c_7 j_d^2 + c_8 i_d + c_9 j_d + c_{10} \end{aligned} \quad (3.3)$$

The Field of View (FOV) model (Devernay and Faugeras 2001) was developed for use with the high distortion levels evident in fish-eye lenses. A fish-eye lens is intended to cover a very wide field of view, and in order to meet this requirement a significant

divergence from the pinhole model is included in the design specification. This means that standard distortion models that are meant to account for slight deviations from pinhole are not suited to fisheye lenses. The FOV model assumes that the radius of an image point is proportional to the angle between the corresponding 3D point, the camera centre and the optical axis:

$$(p, q) = \frac{\tan(r_d \phi)}{2r_d \tan(\phi/2)} (i_d, j_d). \quad (3.4)$$

Furthermore, there are specific models for the catadioptric cameras described in §2.1.3.

### 3.1.1 Forward and Reverse Camera Models

Geometric distortion can be expressed either in distorted camera pixel coordinates, or in a world coordinate frame. The later is termed a *forward* distortion model, and it converts 3D world coordinates to distorted image coordinates. A *reverse* distortion model takes distorted image coordinates and computes the corresponding rays in 3D.

A forward distortion model is convenient for projecting known world points into an image: simply compute the point's perspective projection and then apply the distortion model. However, back-projecting the 3D ray corresponding to a given distorted pixel location (as required for triangulation) is more difficult. This task involves computing the inverse of the distortion model, after which triangulation from perspective images is routine. There is no analytic solution to the inverse mapping if both radial and tangential components are included (Heikkilä and Silvén 1997). The alternative to inverting the fifth order polynomial camera model (3.1) is to develop an approximate inverse (Heikkilä 2000, Mallon and Whelan 2004). Wei and Ma (1993) compute this inverse by determining a set of implicit parameters from a dense grid of features observed in the image. Heikkilä and Silvén (1997) adopted the the same approach except that the dense grid of features is generated using the forward model whose inverse is sought.

A reverse distortion model is ideally suited to rendering distortion corrected frames and to computing the line of sight for a given distorted image coordinate. However, projection is often more computationally expensive since it involves inverting the model. Bundle adjustment requires that many points have to be projected into the image plane in order to measure the error between each projected point and the corresponding measured image location. Under the proposed rational function model this can be done efficiently using the Sampson distance approximation rather than computing the actual projection. Another option is to make these measurements in the perspective frame obtained after distortion correction has been applied (provided the distortion coeffi-

cients are known—this doesn’t apply to point correspondences for auto-calibration). Projection into this frame is then a linear operation.

These differences are mentioned here to highlight that the Rational Function model is a reverse model, while the cubic rational polynomial camera (Hartley and Saxena 1997) is a forward model. Although the functional form of these two models is similar, they operate in opposing directions and are thus more easily adapted to different tasks.

## 3.2 Rational Function Model

In mathematics, a “rational function” refers to a quotient of polynomials. In this chapter we shall define a camera model that relates image locations to ray directions through a quotient of polynomials expressed in terms of the image coordinates.<sup>1</sup> The familiar pinhole camera is a special case of this rational function model where the polynomials are linear. More importantly, the use of quadratic polynomials permits the accurate modelling of non-linear lens distortion within a convenient mathematical framework.

### 3.2.1 General Mathematical Framework

As described in (Grossberg and Nayar 2001, Pless 2003, Sturm and Ramalingam 2004), calibration of a camera amounts to determining the 3D ray from which each pixel samples. For a central camera, this means finding the mapping from pixels  $(i, j)$  in  $\mathbb{R}^2$  to rays  $\mathbf{d}(i, j)$  in  $\mathbb{R}^3$ . This mapping takes the form of some vector of polynomials in  $(i, j)$  multiplied by a matrix of coefficients.

This process can be separated into two transformations. The first is a rotation  $\mathbf{R}$  and translation  $\mathbf{t}$  between the world and camera coordinate systems, referred to as the external calibration. The second is a projection from world points  $(x, y, z)$  in  $\mathbb{R}^3$  to image pixels  $(i, j)$  in  $\mathbb{R}^2$ . For a central pinhole camera the overall mapping is represented by

$$(i, j) = \pi \left( \mathbf{K} [\mathbf{R} \mid \mathbf{t}] (x, y, z, 1)^\top \right),$$

where the 3D-to-2D perspective projection function is defined as  $\pi(x, y, z) = (x/z, y/z)$ . The matrix  $\mathbf{K}$  represents the internal calibration parameters of the camera:

$$\mathbf{K} = \begin{bmatrix} f & s & u_0 \\ 0 & af & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad \left\{ \begin{array}{l} f \text{ is focal length;} \\ (u_0, v_0) \text{ is principal point;} \\ a \text{ is aspect ratio; } s \text{ is skew} \end{array} \right.$$

---

<sup>1</sup>Hartley and Saxena’s formulation sets out a quotient of cubic polynomials expressed in terms of the *world* coordinates. We will refer to this type of camera as a *Rational Polynomial Camera* to be consistent with the literature and to distinguish it from our inverse rational function model

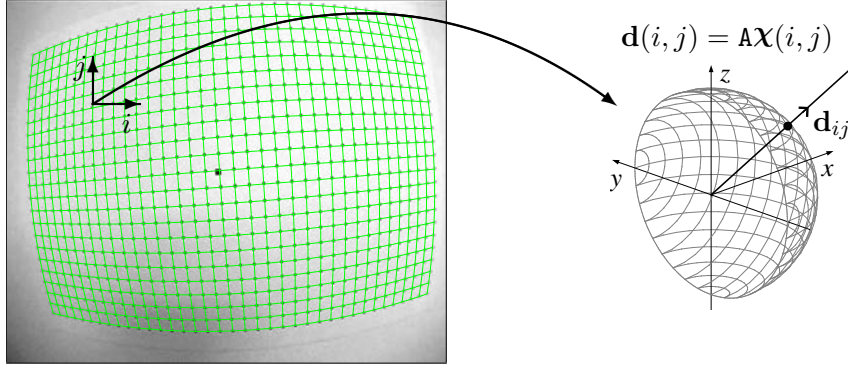


Figure 3.1: The task of distortion correction is to determine the mapping between image coordinates  $(i, j)$  and 3D rays  $\mathbf{d}_{ij}$ . The green lines superimposed on the image of the planar grid of calibration dots indicate the accuracy of the RF model.

By defining the camera coordinate system to be coincident with the camera center the pinhole projection reduces to

$$(i, j) = \pi \left( \mathbf{K} \mathbf{R} (x, y, z)^\top \right).$$

By aligning camera coordinates it is possible to set  $\mathbf{R} \rightarrow \text{Identity}$ . Thus, for a perspective camera, the mapping from image pixels to 3D rays  $\mathbf{d}(i, j)$  in  $\mathbb{R}^3$  can be expressed as:

$$\mathbf{d}(i, j) = \begin{pmatrix} B_{11}i + B_{12}j + B_{13}1 \\ B_{21}i + B_{22}j + B_{23}1 \\ B_{31}i + B_{32}j + B_{33}1 \end{pmatrix} = \mathbf{B} \begin{pmatrix} i \\ j \\ 1 \end{pmatrix}, \quad (3.5)$$

where the  $3 \times 3$  matrix  $\mathbf{B} = \mathbf{R}^\top \mathbf{K}^{-1}$ .

The pinhole camera model (3.5) maps image pixels to 3D rays via a vector of linear polynomials in  $(i, j)$ . Because it is linear in image coordinates it is unable to model non-linear distortion. The rational function model handles this lens distortion by permitting  $i$  and  $j$  to appear in higher order polynomials, in particular quadratic:

$$\mathbf{d}(i, j) = \begin{pmatrix} A_{11}i^2 + A_{12}ij + A_{13}j^2 + A_{14}i + A_{15}j + A_{16} \\ A_{21}i^2 + A_{22}ij + A_{23}j^2 + A_{24}i + A_{25}j + A_{26} \\ A_{31}i^2 + A_{32}ij + A_{33}j^2 + A_{34}i + A_{35}j + A_{36} \end{pmatrix}. \quad (3.6)$$

This model may be written as a linear combination of the distortion parameters, in a  $3 \times 6$  matrix  $\mathbf{A}$  (analogous to  $\mathbf{B}$  above), and a 6-vector  $\chi$  of monomials in  $i$  and  $j$ . Define  $\chi$  as the “Veronese lifting” (Sample and Kneebone 1998) of image point  $(i, j)$  to a six dimensional space

$$\chi(i, j) = [i^2 \quad ij \quad j^2 \quad i \quad j \quad 1]^\top \quad (3.7)$$

The imaging model may then be written

$$\mathbf{d}(i, j) = \mathbf{A} \chi(i, j) \quad (3.8)$$

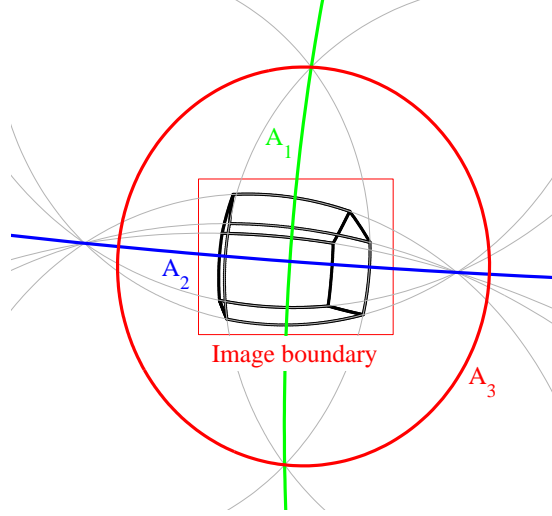


Figure 3.2: The three rows of the calibration matrix  $\mathbf{A}$  denote conics in the image plane, and correspond to the  $YZ, XZ$  and  $XY$  planes respectively. Conics corresponding to parallel lines intersect; lines which are also parallel to the image plane intersect on the conic  $A_3$ , the image plane horizon.

where  $\mathbf{d}$  is the ray direction along which pixel  $(i, j)$  samples (see Figure 3.1). Undistorted image coordinates  $(p, q)$  are computed by the perspective projection of  $\mathbf{d}$ :

$$(p, q) = \left( \frac{A_1^\top \mathbf{x}(i, j)}{A_3^\top \mathbf{x}(i, j)}, \frac{A_2^\top \mathbf{x}(i, j)}{A_3^\top \mathbf{x}(i, j)} \right) \quad (3.9)$$

where the rows of  $\mathbf{A}$  are denoted by  $A_{1..3}^\top$ . We see that the mapping  $(i, j) \rightarrow (p, q)$  is a quotient of polynomials, or *rational function* (Clapham 1996), in the image coordinates.

### 3.2.2 Physical Interpretation of $\mathbf{A}$

Each row in the calibration matrix  $\mathbf{A}$  may be identified with a conic in the image plane.

$$\mathbf{d}(i, j) = \begin{pmatrix} A_1^\top \mathbf{x}(i, j) \\ A_2^\top \mathbf{x}(i, j) \\ A_3^\top \mathbf{x}(i, j) \end{pmatrix} \quad (3.10)$$

Placing the first row (for example) in an expression of the form

$$A_{11}i^2 + A_{12}ij + A_{13}j^2 + A_{14}i + A_{15}j + A_{16} = 0 \quad (3.11)$$

yields the quadratic equation of a conic. Points described by this equation correspond to world rays where  $d_1 = 0$ , indicating that the conic  $A_1$  is the image of the world  $YZ$  plane. Similarly, rays derived from image points along the second row conic  $A_2$  have world coordinate  $d_2 = 0$  and represent the  $XZ$  plane. Points at infinity on the image plane have  $d_3 = 0$ , and are imaged onto the conic  $A_3$ , which is visible in the image only for cameras with a field of view greater than  $180^\circ$  (see Figure 3.2).

### 3.2.3 Back-projection and Projection

The RF model is defined as a reverse distortion model (see §3.1.1), so back-projecting an image point to a ray in 3D space is a simple matter of lifting the point to  $\mathbf{X}(i, j)$  and applying  $\mathbf{d}(i, j) = \mathbf{A}\mathbf{X}(i, j)$ . Often such reverse models are difficult to invert, but this section describes a straightforward method for projecting 3D points into the image plane.

To *project* a 3D point  $\mathbf{X}$  (in camera coordinates) into the image plane we must determine the  $(i, j)$  corresponding to the ray  $\mathbf{d} = -\mathbf{X}$ . As (3.8) is defined only up to scale, we must solve

$$\mathbf{d} = \lambda \mathbf{A}\mathbf{X}(i, j) \quad (3.12)$$

for  $i, j$ , and  $\lambda$ . Recall that each row  $A_{1..3}^\top$  of  $\mathbf{A}$  may be identified with a conic in the image plane  $A_i^\top \mathbf{X}(i, j) = 0$ . The skew-symmetric matrix corresponding to  $\mathbf{d}$  is

$$[\mathbf{d}]_\times = \begin{bmatrix} 0 & -d_3 & d_2 \\ d_3 & 0 & -d_1 \\ -d_2 & d_1 & 0 \end{bmatrix}.$$

This representation of the cross product ( $\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_\times \mathbf{b} = (\mathbf{a}^\top [\mathbf{b}]_\times)^\top$ ) can be used to eliminate  $\lambda$  from (3.12)

$$\begin{aligned} [\mathbf{d}]_\times \mathbf{d} &= \lambda [\mathbf{d}]_\times \mathbf{A}\mathbf{X} \\ \frac{1}{\lambda} 0 &= [\mathbf{d}]_\times \mathbf{A}\mathbf{X} \end{aligned}$$

By multiplying out the right hand side we obtain the points of intersection of the pair of conics

$$\begin{aligned} (d_1 A_3 - d_3 A_1)^\top \mathbf{X}(i, j) &= 0 \\ (d_2 A_3 - d_3 A_2)^\top \mathbf{X}(i, j) &= 0, \end{aligned} \quad (3.13)$$

yielding up to four possible image locations  $(i, j)$ . In practice there are only two intersections; one corresponds to the pixel which images ray  $\mathbf{d}$ , and the other to  $-\mathbf{d}$ . For cameras with viewing angles less than  $180^\circ$  the correct intersection is closest to the center of the viewable area (the boundary of which is given by the conic  $A_3$ ). More general cameras require a further check of the viewing quadrant of the point in order to determine the correct intersection.

Solving for the intersection of these two conics means that projection of a point involves the solution of a 4<sup>th</sup> order polynomial. A common application of projection is in bundle adjustment. Each purported 3D point is projected into the image plane so that the distances to each corresponding measured image location can be tallied and compared from one iteration to the next. The Sampson distance (2.4.1) to each conic can provide an approximation to the distance without having to compute

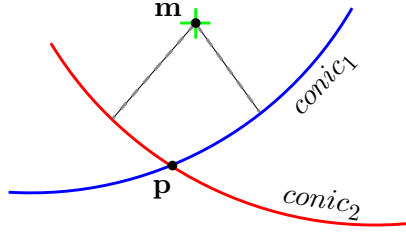


Figure 3.3: The distance from a measured image location  $\mathbf{m}$  to a projected point  $\mathbf{p}$  is required for bundle adjustment. Measuring the true distance requires expensive conic intersection to find  $\mathbf{p}$ . Instead, the sum of the Sampson distances from point  $\mathbf{m}$  to each conic can be used to efficiently approximate this error measure.

the intersection of the two conics (the projected point). This is illustrated in Figure 3.3. For a measured point  $\mathbf{m} = (i', j')$  and a projected point  $\mathbf{p} = (i, j)$  we wish to measure  $d(\mathbf{m}, \mathbf{p}) = \|\mathbf{m} - \mathbf{p}\|^2$ . Computing  $\mathbf{p}$  requires solving the polynomial, so use  $d(\mathbf{m}, \mathbf{p}) \approx d(\mathbf{m}, \text{conic}_1) + d(\mathbf{m}, \text{conic}_2)$ . Each  $d(\mathbf{m}, \text{conic})$  is then computed using the Sampson distance approximation. In this manner we obtain an efficient measure of the reprojection accuracy of the model.

### 3.2.4 Canonicalization of A

As mentioned above for the pinhole camera, we can recover the internal calibration by defining the world coordinate origin to be at the camera centre. The general RF model also calibrates only the intrinsic camera parameters, and so the overall camera calibration is recovered up to an unknown projective homography. The true camera intrinsics,  $\mathbf{A}_{\text{true}}$ , are related to the recovered calibration matrix  $\mathbf{A}$  by an unknown projective homography

$$\mathbf{A}_{\text{true}} = \mathbf{H}\mathbf{A}. \quad (3.14)$$

This homography is analogous to the unknown homography encountered when recovering pinhole projection matrices  $\mathbf{P}$  from a pinhole fundamental matrix. In some applications the choice of  $\mathbf{H}$  is unimportant. For example, if our goal was to produce distortion corrected pinhole images, then the unknown homography is irrelevant as the image corners should be mapped to the output corners of the image in order to make the best use of the available image resolution. On the other hand, for wide angle surveillance cameras an output homography can be used to render only a selected portion of the viewing area fronto-parallel. This avoids having to render the entire image area, and in particular the corners which are subject to very oblique perspective skewing. These examples illustrate that the unknown projective homography is not always a limitation.

In general, however, we would like to remove the ambiguity in  $\mathbf{A}$  as much as possible,

and this section discusses a canonical form to which an arbitrary  $\mathbf{A}$  can be transformed, without loss of generality. Because we are calibrating only the camera intrinsics, we are at liberty to define the camera coordinate system in a convenient manner. We shall therefore insist that the camera Z-axis passes through the pixel  $(i, j) = (0, 0)$ . This implies that  $\mathbf{A}\mathbf{x}(0, 0) \propto [0, 0, 1]^\top$ , and thus, as  $\mathbf{x}(0, 0) = [0, 0, 0, 0, 0, 1]^\top$ , that the last column of  $\mathbf{A}$  can be fixed to be of the form  $[0, 0, \cdot]^\top$ , where the dot denotes an arbitrary scalar. The two conics  $A_1$  and  $A_2$  correspond to the  $YZ$  and  $XZ$  planes (see §3.2.2). We require that the tangents to these conics be aligned with the  $i$  and  $j$  axes at the distortion centre. The normal to the conic  $ai^2 + bij + cj^2 + di + ej + f = 0$  is  $(2ai + bj + d, bi + 2cj + e)$ , and evaluated at the origin, the normal is  $(d, e)$ . In terms of  $\mathbf{A}$  we require that the normal to  $A_1$  is along  $(1, 0)$  and that the normal of  $A_2$  is along  $(0, 1)$ , fixing  $A_{14}, A_{15}, A_{24}$ , and  $A_{25}$ . Thus we have the canonical form of  $\mathbf{A}$ :

$$\mathbf{A} = \begin{bmatrix} \times & \times & \times & \times & 0 & 0 \\ \times & \times & \times & 0 & \times & 0 \\ \times & \times & \times & \times & \times & \times \end{bmatrix}.$$

We now present a method for transforming any  $\mathbf{A}$  into the above canonical form. The matrix of parameters  $\mathbf{A}$  is defined only up to a homography, so we are free to pre-multiply by a transformation matrix. By selecting this transformation to be the inverse of the last three columns of  $\mathbf{A}$  we obtain

$$\mathbf{A}_{\text{can}} = \begin{bmatrix} \times & \times & \times & 1 & 0 & 0 \\ \times & \times & \times & 0 & 1 & 0 \\ \times & \times & \times & 0 & 0 & 1 \end{bmatrix},$$

which is a specific case of the canonical form. This nine parameter expression of  $\mathbf{A}$  is the recommended form for general fitting where the underlying distortion model is completely unknown. It can cover the complete range of the full  $\mathbf{A}$ , but the reduced parameter count is more efficient for fitting. The next section examines additional simplifications that can be performed if the lens or distortion type is known in advance.

### 3.2.5 Parametrizations for Specific Lenses

Many lens types do not require the full generality of the RF model in order to achieve an accurate calibration. Trimming the parameter count to a level that is suited to the lens in question reduces the chances of over-fitting and stabilizes the fitting algorithms. This section describes reductions in the RF model suited for several common lens or camera types.



### Pinhole Camera

A pinhole camera does not exhibit any non-linear distortion, but for completeness we show that it can be modelled as a rational function:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & \frac{1}{f} & \frac{-s}{f^2 a} & \frac{uaf - sv}{f^2 a} \\ 0 & 0 & 0 & 0 & \frac{1}{fa} & \frac{-v}{fa} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.15)$$

where the parameters are defined as in (3.2.1) and the coordinate system is centred on the camera origin with  $\mathbf{R}$  equals the identity. Since  $\mathbf{A}$  is defined only up to a homography, it is possible to factor out the pinhole projection aspect of the camera model; this is done to help simplify the notation for the following distortion models.

### Division Model

The division model (Fitzgibbon 2001) given in (3.2) can be expressed as

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \lambda & 0 & \lambda & 0 & 0 & 1 \end{bmatrix} \quad (3.16)$$

where we are assuming that the distortion is centred at the origin and the aspect ratio is one. Closed form solutions for this model using the plumblines constraint are available (Strand and Hayman 2005).

### Reduced Rational Function Model

Observation of the distortion present in actual lenses led to the development of a reduced parametrization of the rational function model that can accurately represent more general distortion functions.

The general form of the rational function model images straight lines in the world as conics, but enforces no restrictions on the form of the conic. While fitting the model to noisy image data, it was observed that cases where the conics took the form of ellipses produced the most stable reconstructions. A parametrization that would constrain the conics to be ellipses was therefore developed. These ellipses would be circles except for the aspect ratio of the pixels. Parallel lines in the world produce ellipses which intersect at two points; the reduced model yields families of such ellipses. Many lenses can be assumed to have negligible skew at the centre of the image. Thus the conics representing the coordinate axes will intersect at the origin, be perpendicular to one another at the origin, and be centred on their respective axes. The overall diameter

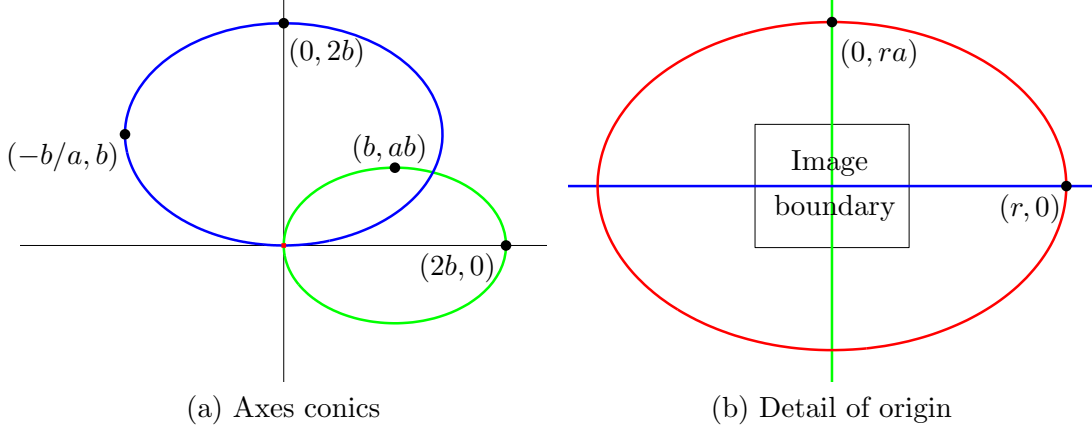


Figure 3.4: Conics pertaining to the three rows (shown in *green*, *blue* and *red*, respectively) in the reduced form of the rational function model. The images of the horizontal and vertical axes are each an ellipse with a “radius” large enough that the axis appears straight in the image. For this example,  $a = 0.7$ ,  $b = 2e7$ ,  $\phi = 0.5$  and  $r^2 = ((wa)^2 + h^2)/(4\phi^2 a^2)$ . The image origin has been set to the centre of the image to simplify the notation in this figure.

of these two axes conics is not significant for most applications; it shall be set to a value large enough to approximate straight lines within the image region. Finally, the parameter  $\phi$  encapsulates the overall curvature (or distortion level) of the lens. The resulting set of conics are illustrated in Figure 3.4.

The three conics that make up the rows in **A** are then

$$\begin{aligned}
 A_1 : \quad & \left(i - \left(b + \frac{w}{2}\right)\right)^2 + \frac{1}{a^2} \left(j - \frac{h}{2}\right)^2 = b^2 \quad (\text{image of the world } YZ \text{ plane}) \\
 A_2 : \quad & \left(i - \frac{w}{2}\right)^2 + \frac{1}{a^2} \left(j - \left(b + \frac{h}{2}\right)\right)^2 = b^2 \quad (\text{image of the world } XZ \text{ plane}) \\
 A_3 : \quad & \left(i - \frac{w}{2}\right)^2 + \frac{\left(j - \frac{h}{2}\right)^2}{a^2} = \frac{(wa)^2 + h^2}{4\phi^2 a^2} \quad (\text{image of the world } XY \text{ plane horizon})
 \end{aligned}$$

where  $(i, j)$  are image coordinates, with the image origin in the top-left;  $w, h$  are the width and height of the image in pixels and  $a$  is the pixel aspect ratio. This can be expressed as a calibration matrix **A**:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \frac{1}{a^2} & -2b - w & -\frac{h}{a^2} & \frac{1}{bw + \frac{w^2}{4} + \frac{h^2}{4a^2}} \\ 1 & 0 & \frac{1}{a^2} & -w & -\frac{2b + h}{a^2} & \frac{1}{bh + \frac{w^2}{4} + \frac{h^2}{4}} \\ 1 & 0 & \frac{1}{a^2} & -w & -\frac{h}{a^2} & \frac{((wa)^2 + h^2)(\phi^2 - 1)}{4\phi^2 a^2} \end{bmatrix} \quad (3.17)$$

Two important shape parameters for the lens are  $\phi$ , representing the field of view; and  $b$ , which represents the diameters of the first two conics. The image dimensions are

known, so this parametrization reduces the number of unknowns in  $\mathbf{A}$  from 14 (for the canonical form) to 3. The pixel aspect ratio is typically known, (*e.g.* for PAL cameras  $a = 576/720 \times 4/3 = 1.0667$ ), and the conic radius  $b$  can be set to a large number (*e.g.*  $2 \times 10^7$ ). The field of view parameter can also be reasonably guessed for most lenses (*e.g.*  $\phi = 1 \times 10^{-5}$  for pinhole, 0.5 for fish-eye). Note that as  $\phi \rightarrow 0$  the radius of the third conic becomes infinite. This is correct since this conic represents the image of the horizon, which cannot be viewed unless there is either some nonlinearity in the camera model or a focal length of zero. In practice a distortion free image can be represented by setting  $\phi$  to be very small.

This reduced rational function model is essentially a single parameter ( $\phi$ ) model similar to the FOV model proposed for fish-eye lenses. The algebraic form of the two models differs, but the rational function model can very closely approximate the FOV fit to certain lenses. Is there any advantage to using one model over the other, particularly for fish-eye lenses? One of the chief advantages of the rational function model is the full suite of available calibration methods. A single image linear calibration will be much more straightforward than the current FOV calibration methods (which require multiple images and nonlinear optimization procedures). Furthermore, if the RF model is allowed to deviate from the reduced parameterization, there is significantly more modelling power available than that offered by the FOV approach: the rational functions can fit many other lens types than just fish-eye.

### Para-catadioptric Camera Model

The rational function model can represent more general cameras in addition to pin-hole cameras with radial distortion. One example is the parabolic mirror catadioptric system (2.6). By simple algebraic manipulation we obtain

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 4f & 0 & -4c_x f \\ 0 & 0 & 0 & 0 & 4f & -4c_y f \\ 1 & 0 & 1 & -2c_x & -2c_y & c_x^2 + c_y^2 - 4f^2 \end{bmatrix}. \quad (3.18)$$

## 3.3 Two-view geometry

This section examines the relationships between two distorted images of a scene, and how these can be exploited to compute  $\mathbf{A}$ . This permits calibration of a camera without the need for a calibration grid, or even the camera; only two images are required.

A 3D point imaged in two views gives rise to a pair of point correspondences  $(i, j) \leftrightarrow (i', j')$  in the image. Their back-projected rays  $\mathbf{d}, \mathbf{d}'$  must intersect at the 3D point,

that is to say they must satisfy the Essential matrix constraint

$$\mathbf{d}^\top \mathbf{E} \mathbf{d}' = 0. \quad (3.19)$$

Substituting  $\mathbf{d} = \mathbf{A}\boldsymbol{\chi}$ , we obtain

$$\boldsymbol{\chi}^\top \mathbf{A}^\top \mathbf{E} \mathbf{A}' \boldsymbol{\chi}' = 0 \quad (3.20)$$

and writing  $\mathbf{G} = \mathbf{A}^\top \mathbf{E} \mathbf{A}'$  we obtain the constraint satisfied by the lifted 2D correspondences

$$\boldsymbol{\chi}^\top \mathbf{G} \boldsymbol{\chi}' = 0. \quad (3.21)$$

This “lifted fundamental matrix” is rank two and can be computed from 36 point correspondences in a manner analogous to the 8-point algorithm (Longuet-Higgins 1981) for projective views. We shall refer to this method for computing  $\mathbf{G}$  as the Direct Linear Transform (DLT) technique (described in detail in §2.4.3). Although the DLT permits  $\mathbf{G}$  to be computed in a strictly linear manner, greater accuracy is achieved through a nonlinear optimization using the Sampson distance approximation, as discussed in §5.4. The immediate consequence is that, because  $\mathbf{G}$  can be determined solely from image-plane correspondences, it is possible to remove distortion from image pairs for which no calibration is available. This depends upon two things: 1)  $\mathbf{G}$  must be stably estimated in the presence of image noise, and 2) we must have a method for recovering the calibration matrix  $\mathbf{A}$  from this  $\mathbf{G}$ . Both of these issues will be addressed in §4.3. The next section investigates the epipolar curves induced by the new model.

### 3.3.1 Epipolar curves

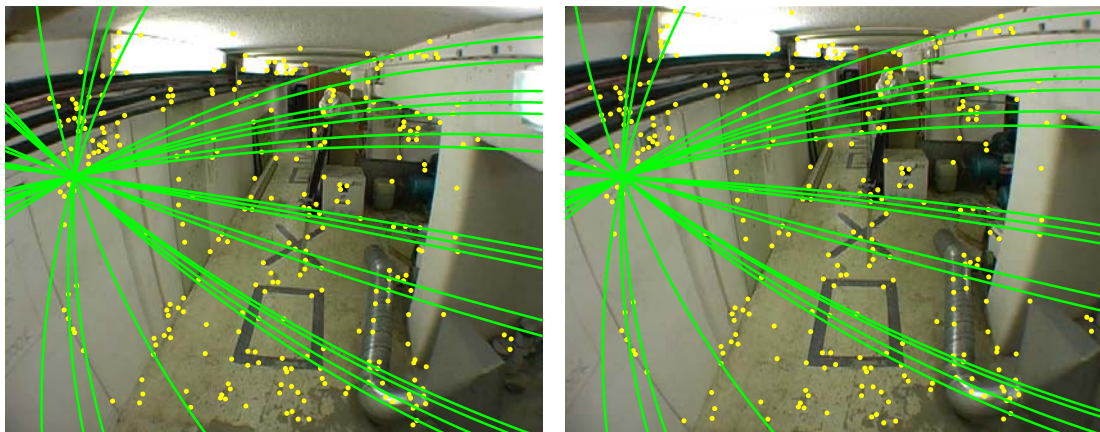
For a given pixel  $(i', j')$  in one of the images, its epipolar curve in the other image is easily computed, as the set of  $(i, j)$  for which  $\boldsymbol{\chi}(i, j)^\top \mathbf{G} \boldsymbol{\chi}' = 0$ . This is the conic with parameters  $\mathbf{G} \boldsymbol{\chi}'$ . Figure 3.5 shows epipolar curves computed from point correspondences on two real-world images. We note that all the curves intersect at two points, because  $\mathbf{G}$  is rank 2, so all possible epipolar conics are in the pencil of conics generated by any two columns of  $\mathbf{G}$ . The two intersection points in each image are the two intersections of the baseline with the viewing sphere of that image.

## 3.4 Summary

This chapter has presented a rational function model for lens distortion. The geometry of the model was outlined and equations for representing many existing camera models were provided.



Bridge of Sighs, Hertford College



Jenkin basement (revisited)

Figure 3.5: Epipolar curves recovered from the  $\mathbf{G}$  matrix computed using point correspondences. For the Bridge of Sighs images, 36 point correspondences were used. Note that the curves converge to the camera viewpoints on either side of the street. The Jenkin basement featured in some of the early structure-from-motion work. This fisheye pair contains a total of 300 point correspondences; some of the epipolar lines have been omitted for clarity.

## Chapter 4

# Lens Distortion: Calibration

A camera model is only as useful as the methods available for calibration. To this end we will present three separate calibration algorithms for the Rational Function model: one based on a planar calibration target, a plumbline method, and an autocalibration technique that uses only point correspondences between multiple images.

### 4.1 Linear Calibration from an Arbitrary Planar Grid

The simplest and most reliable of the calibration methods we present for the RF model involves the use of a planar calibration target. This target is photographed under any arbitrary orientation (it does not have to be fronto-parallel, as is sometimes impractically required for planar calibration) and all calibration equations are linear.

Prior to explaining the calibration algorithm, a few comments on planar calibration targets are in order. A planar target is used to easily establish known 3D positions. Calibration proceeds by establishing the correspondence between the physical location and a position in image coordinates. The advent of laser printers and the abundance of flat surfaces in a modern office or laboratory environments have made it inexpensive and precise to construct planar targets. Designing a custom calibration target provides the opportunity to use point markers that are easy to locate precisely. A checkerboard pattern is often used (*caveat emptor*: bias in corner or line detectors, aliasing, and lens distortion can significantly affect the accuracy of checkerboard corner detection), but any detectable features can be employed. I recommend small dots (see Figure 4.1) for reasons given in Chapter 7. Using a target imposes the constraint that we must have an image of a known object. While this is an easy task if one has access to the camera, it is not a practical method for archive footage (for example). However, when the camera to calibrate is available for testing the other advantages of planar calibration grids far outweigh this limitation.

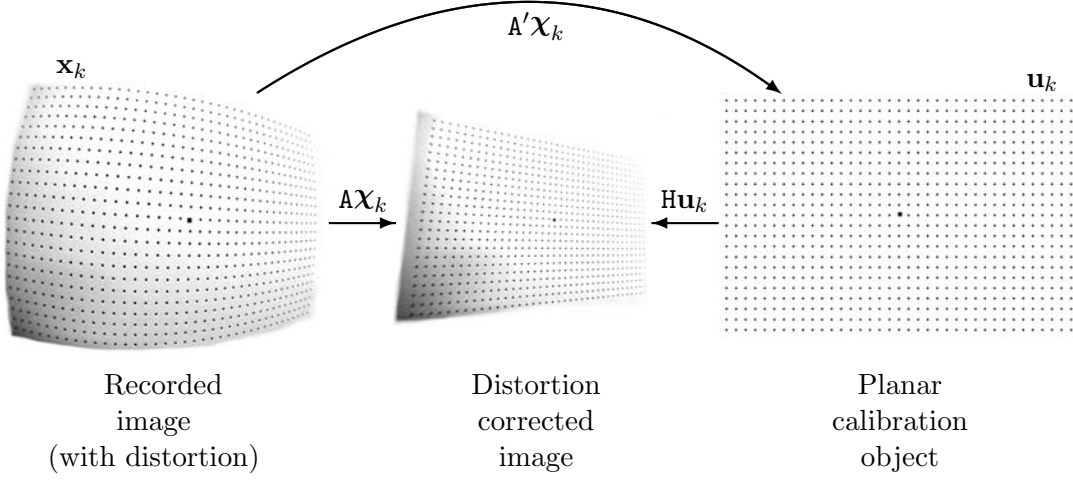


Figure 4.1: Relationships between calibration plane, image plane, and the distortion corrected image.

The grid is used to provide some number  $N$  of correspondences between (lifted) 2D points  $\mathbf{x}_k$  and 2D plane points  $\mathbf{p}_k$ . As the plane is in an unknown position and orientation, the mapping between plane and camera coordinates is an (unknown) planar homography  $\mathbf{H}$ . Thus each correspondence provides a constraint of the form

$$\mathbf{H}\mathbf{p}_k = \lambda_k \mathbf{A}\mathbf{x}_k \quad (4.1)$$

$$\implies \mathbf{p}_k = \lambda_k \mathbf{H}^{-1} \mathbf{A}\mathbf{x}_k \quad (4.2)$$

$$\implies \mathbf{p}_k = \lambda_k \mathbf{A}'\mathbf{x}_k. \quad (4.3)$$

Two constraint equations are obtained for each point correspondence, and a linear solution can be obtained through the use of the DLT (§2.4.3). For completeness, these steps are described here as well. As is always the case when using the DLT, data conditioning is essential. The quadratic terms in  $\mathbf{x}$  amplify any noise, and this propagates through the algorithm. Typical image dimensions are  $720 \times 576$  pixels, so the values of  $\mathbf{x}$  range in magnitude from  $10^5$  to 1. This large range can make the factorization unstable. The *condition number*, a measure of stability, of the factorization can be improved by scaling and translating the input data so that all image points have zero mean and standard deviation  $\sqrt{2}$ .

The lifted image point is given by  $\mathbf{x}(u, v) = [u^2 \ uv \ v^2 \ u \ v \ 1]^\top$ . We can set the  $z$ -component of  $\mathbf{p}_k \rightarrow 1$  simply by defining the calibration target to be the  $z = 1$  plane (recall that this is a 3D point on the planar calibration object). The world point is then given by  $\mathbf{p}_k = [x_k, y_k, 1]$ . Pre-multiplying both sides of (4.3) by the skew symmetric

matrix corresponding to  $\mathbf{p}_k$  eliminates the unknown scale parameter:

$$[\mathbf{p}_k]_{\times} \mathbf{p}_k = \lambda_k [\mathbf{p}_k]_{\times} \mathbf{A}' \boldsymbol{\chi}_k \quad (4.4)$$

$$\frac{1}{\lambda_k} \mathbf{0} = [\mathbf{p}_k]_{\times} \mathbf{A}' \boldsymbol{\chi}_k \quad (4.5)$$

Writing out both the skew symmetric matrix and the lifted components of  $\boldsymbol{\chi}_k$  yields

$$0 = \begin{bmatrix} 0 & -1 & y_k \\ 1 & 0 & -x_k \\ -y_k & x_k & 0 \end{bmatrix} \begin{bmatrix} A'^1 \\ A'^2 \\ A'^3 \end{bmatrix} \begin{pmatrix} u_k^2 \\ u_k v_k \\ v_k^2 \\ u_k \\ v_k \\ 1 \end{pmatrix}. \quad (4.6)$$

where  $A'^m$  denotes the rows in  $\mathbf{A}'$ . With a little re-arranging we obtain the following  $3 \times 18$  constraint matrix

$$\begin{bmatrix} \mathbf{0}^\top & -\boldsymbol{\chi}_k^\top & y_k \boldsymbol{\chi}_k^\top \\ \boldsymbol{\chi}_k^\top & \mathbf{0}^\top & -x_k \boldsymbol{\chi}_k^\top \\ -y_k \boldsymbol{\chi}_k^\top & x_k \boldsymbol{\chi}_k^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} A'^1 \\ A'^2 \\ A'^3 \end{pmatrix} = \mathbf{0}. \quad (4.7)$$

Each point correspondence yields a triplet of constraint equations of this form, and they are all stacked into a design matrix so that  $\mathbf{D}\mathbf{a}' = \mathbf{0}$ . The vector  $\mathbf{a}'$  is obtained through the SVD (for cases where greater than the minimal number of point correspondences are available)

$$(\mathbf{U}\mathbf{S}\mathbf{V}^\top) = \text{SVD}(\mathbf{D}) \quad (4.8)$$

$$\mathbf{a}' = \mathbf{V}^{18} \text{ or } \mathbf{V}(:, 18) \quad (4.9)$$

where  $\mathbf{V}^{18}$  denotes the last column in  $\mathbf{V}$ . This is the unit singular vector corresponding to the smallest singular value. The original matrix  $\mathbf{A}'$  is reconstructed from  $\mathbf{a}'$  as

$$\mathbf{A}' = \begin{bmatrix} a'_1 & a'_2 & a'_3 & a'_4 & a'_5 & a'_6 \\ a'_7 & a'_8 & a'_9 & a'_{10} & a'_{11} & a'_{12} \\ a'_{13} & a'_{14} & a'_{15} & a'_{16} & a'_{17} & a'_{18} \end{bmatrix}. \quad (4.10)$$

The solution obtained in this manner is correct up to an unknown projective homography

$$\mathbf{A}_{\text{true}} = \mathbf{K}\mathbf{A}'. \quad (4.11)$$

It is not essential to solve for this homography explicitly; the calibration matrix  $\mathbf{A}'$  is sufficient to remove the nonlinear distortion from the image. If one were to apply only this correction to an image, lines would appear straight but there would likely be evidence of skew, an unbalanced aspect ratio or some other perspective effects. The resulting image still follows the pinhole model, but it corresponds to an image plane



that is at some odd orientation relative to the optical axis. Generally, the image plane is expected to be perpendicular to the optical axis, the following paragraph describes a method for simulating this.

Compensating for the unknown projective homography is simple, the user merely applies whichever homography is required to bring the distortion corrected image into a suitable range for the current application. Often it is useful for the distortion corrected image to have the same overall dimensions as the original image. The projective matrix  $K$  that performs this rectification is computed using the corner coordinates of the original image  $\mathbf{x}_{corners}$ . The corners are pushed through the distortion correction process

$$\boldsymbol{\chi}_{corners} = \text{veronese}(\mathbf{x}_{corners}) \quad (4.12)$$

$$\mathbf{x}'_{corners} = \mathbf{A}\boldsymbol{\chi}_{corners}. \quad (4.13)$$

We can then compute the projective homography that satisfies

$$\mathbf{x}_{corners} = K\mathbf{x}'_{corners}. \quad (4.14)$$

This matrix is then applied to the distortion correction parameter matrix  $\mathbf{A}_{rectified} = K\mathbf{A}$ . This rectified set of distortion parameters can then be used to rectify distorted points or images into a coordinate system that fits within the original image bounds. Unless otherwise noted, all images included in this thesis that have been corrected by the rational function model have had this rectification applied.

## 4.2 Calibration by Plumb-line Constraints

In sequences where there are straight lines in the scene, it is desirable to be able to impose the constraint that the back-projections of their images are straight (Figure 4.2). This *plumbline* technique was first mooted by Brown (1971), and has been applied to various distortion models (Stein 1997, Swaminathan and Nayar 2000, Devernay and Faugeras 2001). We will show that the rational function model images straight lines as conics, and this permits an elegant factorization of the conics into the camera calibration and the equations of the straight lines.

A line in the scene forms a plane with the origin of camera coordinates, and is imaged to the set of  $\mathbf{d}$  in that plane. This yields the line equation  $\mathbf{l}^\top \mathbf{d} = 0$  which, in terms of image points  $(i, j)$  becomes

$$\mathbf{l}^\top \mathbf{A}\boldsymbol{\chi} = 0 \iff \boldsymbol{\theta}^\top \boldsymbol{\chi} = 0, \quad (4.15)$$



Figure 4.2: Plumblines use image curves corresponding to straight lines in the world to compute the distortion parameters. Here the detected edgels from 22 lines in a single frame of video have been used to calibrate the distortion. The unwarped image displays correct rectification; the lines are now straight.

where  $\boldsymbol{\theta} = \mathbf{A}^\top \mathbf{l} = (A_{xx}, A_{xy}, A_{yy}, A_x, A_y, A_0)^\top$  are the parameters of a conic in image coordinates  $(i, j)$  and  $\mathbf{X}$  is given by (3.7). Here we observe the important property that lines in the world are imaged as conics under the rational function model. The task of calibration is then to find an  $\mathbf{A}$  which will map these conics in the distorted image back to straight lines. Figures 4.3 to 4.5 describe the two different plumblines methods for finding  $\mathbf{A}$  that are detailed in subsequent sections.

#### 4.2.1 Linear Factorization Method

By fitting a conic to the image of the line, we obtain parameters  $\boldsymbol{\theta}$ , and thus the constraint

$$\boldsymbol{\theta} = \mathbf{A}^\top \mathbf{l}$$

for unknown  $\mathbf{A}$  and  $\mathbf{l}$ . The equality is exact (*i.e.* not just up to scale) as any scale factor is included in  $\mathbf{l}$ . Collecting  $L$  such constraints, we obtain

$$\underbrace{[\boldsymbol{\theta}_1 \mid \dots \mid \boldsymbol{\theta}_L]}_{6 \times L} = \underbrace{\mathbf{A}^\top}_{6 \times 3} \underbrace{[\mathbf{l}_1 \mid \dots \mid \mathbf{l}_L]}_{3 \times L} \quad (4.16)$$

$$\mathbf{C} = \mathbf{A}^\top \mathbf{L} \quad (4.17)$$

so the matrix of conic parameters  $\mathbf{C}$  is of rank no greater than 3. This matrix can be assembled directly from image measurements: fit conics to the detected edges of lines that are straight in the real world. From a knowledge of  $\mathbf{C}$ ,  $\mathbf{A}$  can be computed up to a homography by factorization: if

$$\mathbf{U}\mathbf{S}\mathbf{V}^\top = \mathbf{C}$$

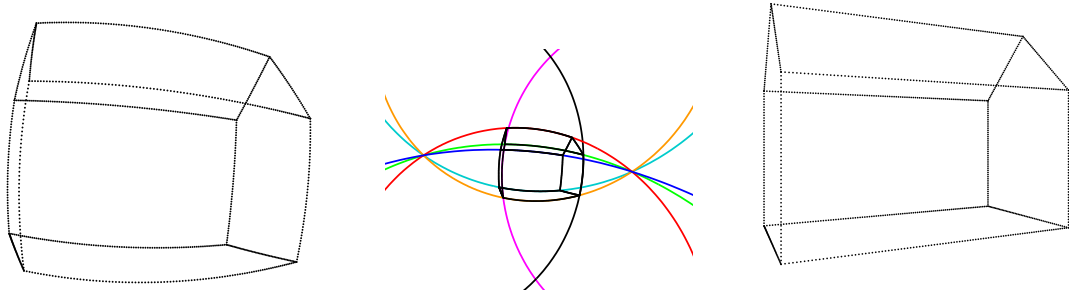


Figure 4.3: Plumline constraints on a single synthetic image. *Left* Input image *Centre* Fit conics to several of the lines. Note that conics corresponding to parallel lines (e.g. the horizontal ones) intersect. *Right* The input image rectified by a linear calibration from the fit conics. The noise free case can be rectified with the elegant linear factorization.

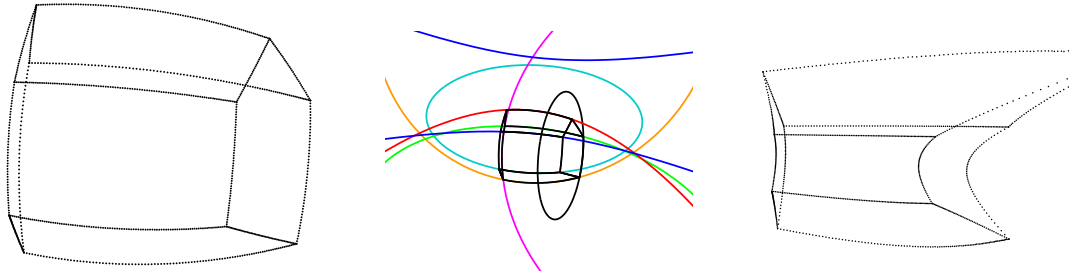


Figure 4.4: Linear plumline constraints on a single synthetic image including noise at 0.01 pixels RMS. *Left* Input image *Centre* Fit conics to several of the lines. The parallel conics no longer intersect. When fitting to such incomplete data, the conic parameters are highly unstable. *Right* The rectification fails, demonstrating that even at this artificially low noise level the linear solution begins to break down. The linear method minimizes the error in conic parameter space, which is not an appropriate error measure for rectification.

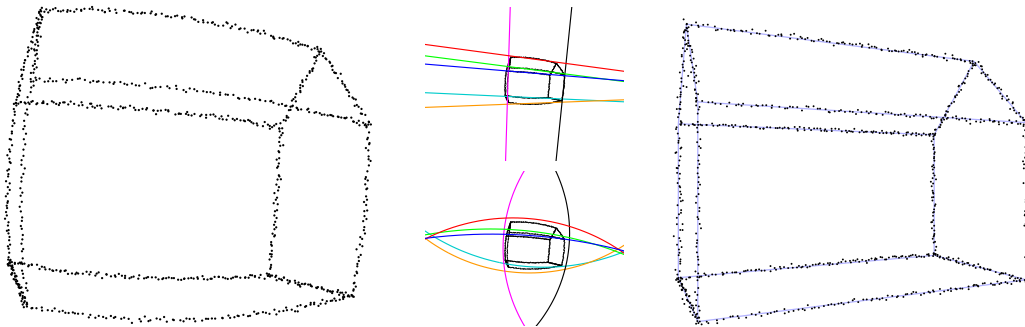


Figure 4.5: Nonlinear optimization of plumline constraints on a single synthetic image including noise at 2.0 pixels RMS. The optimization minimizes the distance between image points and their corresponding line projected as a conic into the image plane. *Left* Input image *Centre top* Straight lines fit to the point data for each line segment are used to initialize the world line positions. *Centre bottom* The initial line estimates projected into the image as conics based on the initialization of  $A$ . *Right* Rectification after optimizing for distortion coefficients and world line positions. The grey lines indicate the noise free edge locations.

is the SVD of  $\mathbf{C}$ , then

$$\mathbf{A} = \mathbf{S}_{(1:3,1:3)} \mathbf{U}_{(:,1:3)}^\top \quad (4.18)$$

is one member of the equivalence class of solutions.

The matrix  $\mathbf{C}$  will not be rank 3 if the conics were obtained by fitting to noisy image data. The above factorization truncates  $\mathbf{C}$  to rank 3 by taking only the first three singular values in (4.18). This finds the rank 3 matrix  $\hat{\mathbf{C}}$  that is nearest to  $\mathbf{C}$  by the Frobenius norm (square root of the sum of the squared entries). This is a minimization of the error in the conic parameter space. Figure 4.3 shows the results of an implementation on synthetic data, leading to an accurately rectified house in the absence of noise, but with reduced performance on more noisy data (Figure 4.4).

To quantify this numerically, the rectification error was measured for linear factorization of a synthetic house image with various levels of random noise added. The original image and seven selected lines are those shown in Figure 4.3. A conic was fit to the data for each line using the approximate Sampson distance (Taubin 1991). Data conditioning (as described in 4.1) was performed on the input points prior to fitting the conics. A calibration was also performed using the raw input data to compare the effects of this conditioning. The conic parameters were assembled into  $\mathbf{C}$  and the rational function distortion matrix  $\mathbf{A}$  computed via the truncated SVD. This set of distortion parameters was then used to unwarped the input images; this output should contain only straight lines. Evaluation thus consists of fitting straight lines to each set of data and reporting the perpendicular distance of each unwarped edgel to its line. The standard deviation of these residuals over all edgels (959 data points in total, 607 of which are included in the seven calibration lines) was reported (in Figure 4.6) as the error for that calibration run. A total of 21 noise levels ranging in amplitude from  $\sigma = 10^{-6}$  to  $\sigma = 3$  pixels were tested. As this is normally distributed random noise there will be a few data points which receive a perturbation that is much larger than the standard deviation. The position of these very noisy points along the length of each line can have a significant effect on the conic fitting. Fifty different sets of random noise were used to perform fifty calibration tests at each noise level. This way a single noise distribution cannot unduly influence the results. The mean of these trials was reported for each noise level, but the maximum and minimum are also plotted (as the solid lines in Figure 4.6) to indicate the range induced by the distribution of the noise.

In addition to the linear calibrations with and without conditioning, Figure 4.6 includes the results for a nonlinear optimization calibration (described in §4.2.2) and a noise free result, for comparison. The noise free ground truth baseline was established

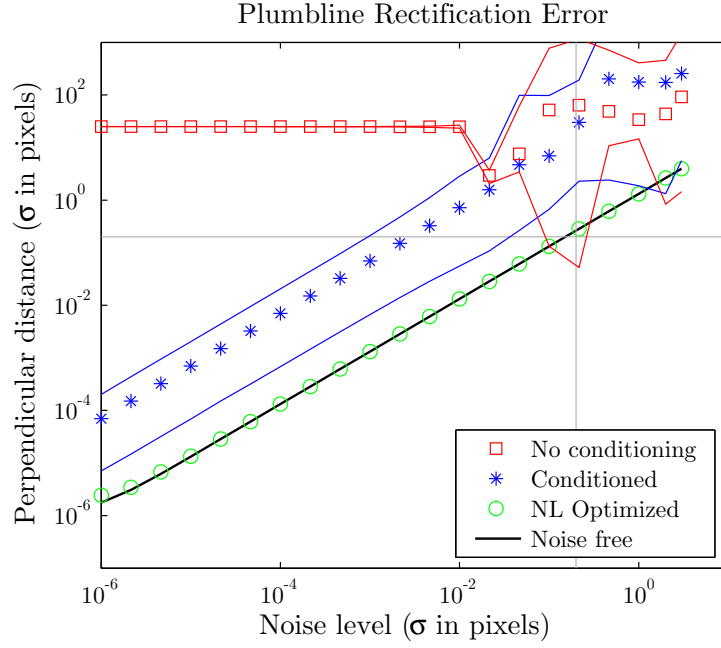


Figure 4.6: Performance of the linear plumblin factorization algorithm in the presence of image noise. The linear algorithm demands careful conditioning, yet still fails in the presence of realistic image noise. Errors measured perpendicular to straight lines fit to rectified image data. The solid lines denote the minimum and maximum error from 50 runs with random noise.

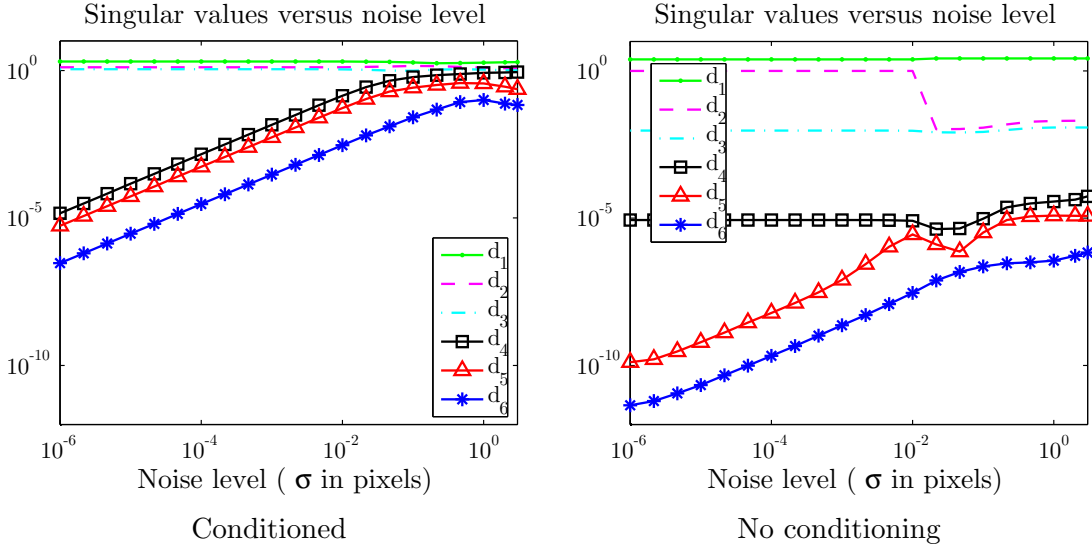


Figure 4.7: The singular values obtained in factoring the conic parameters to obtain **A**. The three smallest singular values should be zero if the conic parameter matrix **C** truly is rank 3. As the noise level increases this is no longer true, and the factorization algorithm fails. The solution is to perform an optimization using a parametrization that enforces the rank 3 constraint while minimizing image plane error.

by computing  $\mathbf{A}$  from the noise free data (via the linear factorization method with conditioning) and then using that calibration to rectify the noisy input data at each level of noise. This provides a measure of the best performance that could be achieved given the input data, and is plotted as the black line in Figure 4.6. Note that this line is essentially linear with a slope of 1. This demonstrates that a calibration does exist such that the noisy input data can be rectified to an error level that is directly proportional to the input noise. In the case of the ground truth line this calibration was artificially determined using the synthetic noise free input data, but it provides a realistic bound on what can be achieved.

The results of the conditioned linear calibration are parallel to the ground truth up to a noise level of  $\sim 10^{-1}$ , however the vertical offset indicates a higher error level. Even at extremely low noise levels the linear algorithm is unable to produce a calibration whose accuracy is on the order of the theoretical limit. This is due to errors in the conic fitting, which then propagate through the algorithm.

Conditioning of the input data is essential for the linear factorization method. The linear method uses the SVD to truncate  $\mathbf{C}$  to rank 3. The implied assumption is that the last three singular values are negligible and can therefore be discarded. Figure 4.7 plots the six singular values for both the conditioned and the non-conditioned cases. With conditioned data we observe that the three smallest singular values gradually increase with increasing noise level. At some noise level (certainly not higher than  $\sigma = 10^{-2}$  pixels) the assumption that  $d_{4:6}$  are insignificant relative to  $d_{1:3}$  is no longer valid. At this point the rank 3 truncation is discarding useful information from the matrix of conic parameters.

The singular values for the linear algorithm without data conditioning (Figure 4.7 *right*) do not exhibit the same regular behavior. Beyond the threshold around  $\sigma = 10^{-2}$ , the noise totally dominates the solution. The last three singular values  $d_{4:6}$  are no longer insignificant relative to  $d_{1:3}$ , and the matrix of conic parameters is far from rank 3. The Frobenius norm minimization implied in the SVD solution to (4.17) minimizes an error in the conic parameter space, but this does not necessarily leave us with meaningful conics in the image (see Figure 4.8).

Even with careful data conditioning, the linear algorithm fails at an unrealistically low noise level. A reasonable rule of thumb for the best performance that can be expected from an edge detector is 1/5 of a pixel of noise (indicated by the faint horizontal and vertical grey lines in Figure 4.6). The linear method with data conditioning falls short of this mark by at least a factor of 100. This instability stems from fitting conics to very short segments of a curve: there can be large variations in the actual parameter

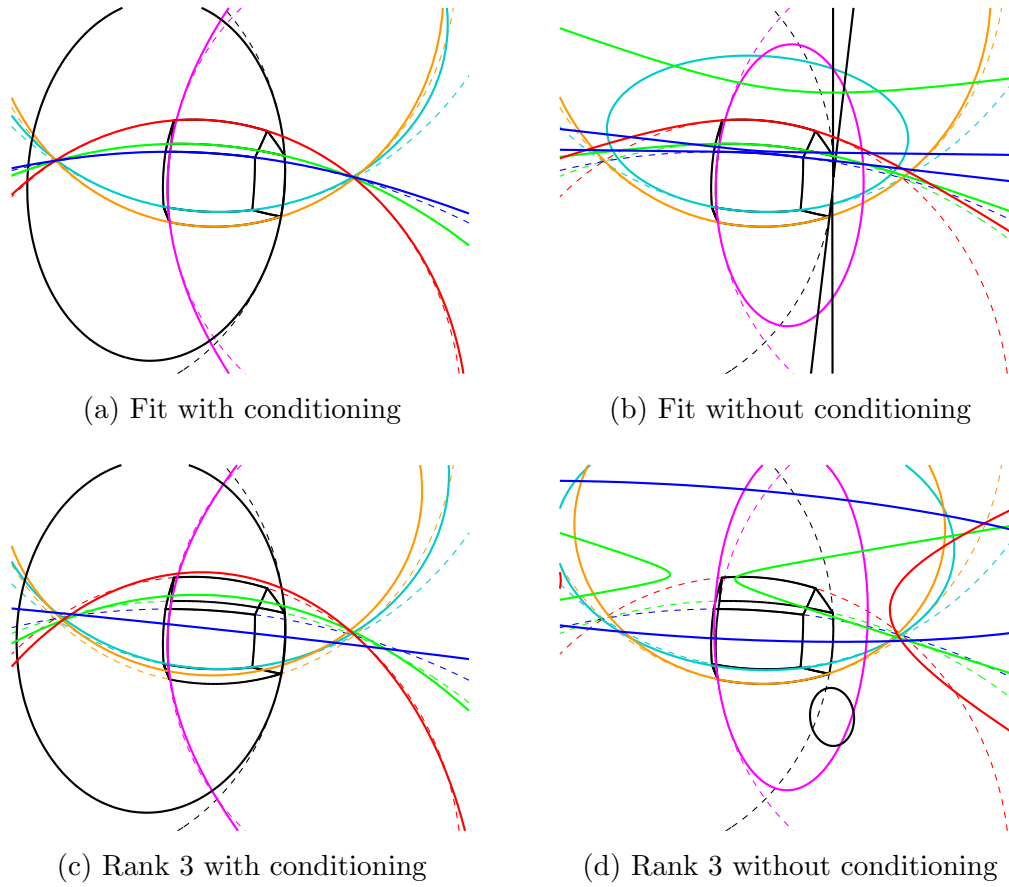


Figure 4.8: Data conditioning is essential for the linear plumblane method. (a) and (b) Conics fit to data from seven input lines for conditioned data and non-conditioned data, respectively. The dashed lines are conics fit to noise free image points, while the solid conics are fit to data with  $\sigma = 0.01$  pixels of noise. (c) and (d) The conics after enforcing the rank 3 constraint in the SVD. These are the conics that are used to calibrate the distortion. Note that without careful data conditioning the rank 3 truncation produces meaningless conic data; the curves no longer even approximately pass through their respective data points.

values representing wildly varying conics that are still very similar in the region of the data we are fitting to. The same underlying problem gives rise to the sensitivity to data conditioning. Conditioning helps to minimize the effects, but cannot correct the root cause. That would require that much more of the conic be visible in the image, something that is not possible with conventional or even fish-eye lenses.

What is needed is a means of ensuring that the fit conics are all ellipses. Or, expressed more strongly: all conics must be ellipses of the same family such that parallel lines produce intersecting ellipses. This constraint is the starting point for the reduced parametrization of §3.2.5. In the following section we describe a method for computing  $\mathbf{A}$  from noisy image data of straight lines in the real world by using this

reduced parametrization.

### 4.2.2 Optimization Method

The overall strategy for fitting  $\mathbf{A}$  from noisy image data is to run a non-linear optimization that finds the  $\mathbf{A}$  which minimizes the error between the image data and straight lines projected (as conics) into the distorted image (see Figure 4.9). This requires a reasonable initialization of  $\mathbf{A}$ , knowledge of straight lines in the real world, and an error measure in the image plane.

The overall parametrization is analogous to bundle adjustment (§2.3.2) and the ellipse fitting approach of (Gander et al. 1994): the world line directions and image projection parameters are *simultaneously* updated. The parameter vector is partitioned into two types of variables; coordinates describing the position and direction of straight lines in the real world, and the parameters of the camera model. In this case the camera model is the reduced parametrization (3.17) of the rational function model (§3.2.5). This model is very forgiving in the choice of initial parameters; the basin of convergence is wide and well-defined. We guessed that the field of view for our fish-eye lens is roughly  $90^\circ$  so  $\phi = 0.5$  using (3.17). The primary conic radius was set at  $b = 2e^7$ . The image dimensions are  $h = 576$ ,  $w = 720$  and  $a = 1.0667$ .

The world lines are 3D, but since they will always be projected into the image we can parameterize them as 2D lines in a plane parallel to the image plane. The lines are then represented using 2D homogeneous coordinates so that  $ax + by + c = 0$  is parametrized as

$$\mathbf{l} = (a, b, c)^\top. \quad (4.19)$$

The complete parameter vector for the plumblin optimization is therefore  $(a, b, \phi, \mathbf{l}_1 \dots \mathbf{l}_L)$  giving a total of  $3 + 3L$  variables, where  $L$  is the number of lines.

The method for initializing the straight lines  $\mathbf{L}$  in the world must still be discussed. It would not be practical to require that the true orientation of each straight line used for calibration be physically measured to initialize the algorithm. However, the optimization is based upon projecting the *world* lines into the image. To overcome this, we supply only a very rough approximation as the initial line positions and then allow the optimization to refine that estimate (recall that the line positions are included as parameter variables, not as supplied constants). The initial estimate is made by fitting straight lines to the detected image edgels, even though they lie along segments curved by the lens distortion. This is essentially an initialization that assumes no distortion, at least for the line fitting.<sup>1</sup>

---

<sup>1</sup>There is an inconsistency though, because the initial estimate for the *camera* parameters *does*



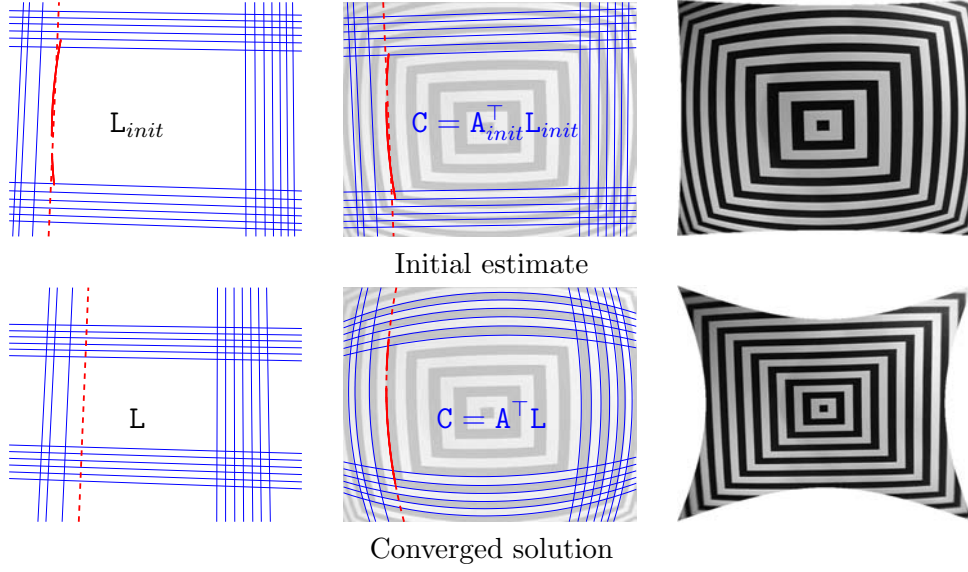


Figure 4.9: Non-linear optimization to fit  $\mathbf{A}$  to the distorted images of straight lines. *Top (left to right)* The initial lines  $\mathbf{L}$  are fit to edge data in the distorted image. These lines are projected into conics  $\mathbf{C}$  in the distorted image via  $\mathbf{A}$ . The image with distortion corrected via  $\mathbf{A}$ . *Bottom* The optimization varies  $\mathbf{A}$  and  $\mathbf{L}$  to minimize the error measured between the edgels and the projected conics  $\mathbf{C}$ .

We then project the lines into conics in the image using (4.17) and measure the Sampson distance (§2.4.1) from the conics to the detected edgels. The Sampson distance is a first order approximation to the distance from a point to a conic. The error function we minimize is then given by:

$$\epsilon(\mathbf{C}, \mathbf{L}) = \sum_{\ell=1}^L \sum_{k=1}^{n_l} \left[ \frac{\boldsymbol{\theta}_{\ell}^{\top} \boldsymbol{\chi}(i_{lk}, j_{lk})}{[(2A_{xx}i_{lk} + A_{xy}j_{lk} + A_x)^2 + (2A_{yy}j_{lk} + A_{xy}i_{lk} + A_y)^2]^{1/2}} \right]^2. \quad (4.20)$$

Figure 4.10 shows detailed pseudocode for the equation of the error. MATLAB's `lsqnonlin` was used to perform the minimization. The Levenberg-Marquardt algorithm computes the least squares error, and performs the squaring operation itself. For this reason it is important to supply the signed error (inside the square brackets in (4.20)). Then there is a zero crossing in the error space, which results in more stable and faster convergence. The coordinates of the detected edgels are static throughout the optimization, so their lifted coordinates  $\boldsymbol{\chi}(i_{lk}, j_{lk})$  can be pre-computed to lend additional computational efficiency. Finally, it is beneficial to normalize the parameter vector so that all entries are in the range  $[-1 \dots 1]$ . If the relative scales of the parameter variables vary widely the optimization can seem to favour certain variables, and not include distortion. This is resolved during the optimization, when the line parameters are updated.

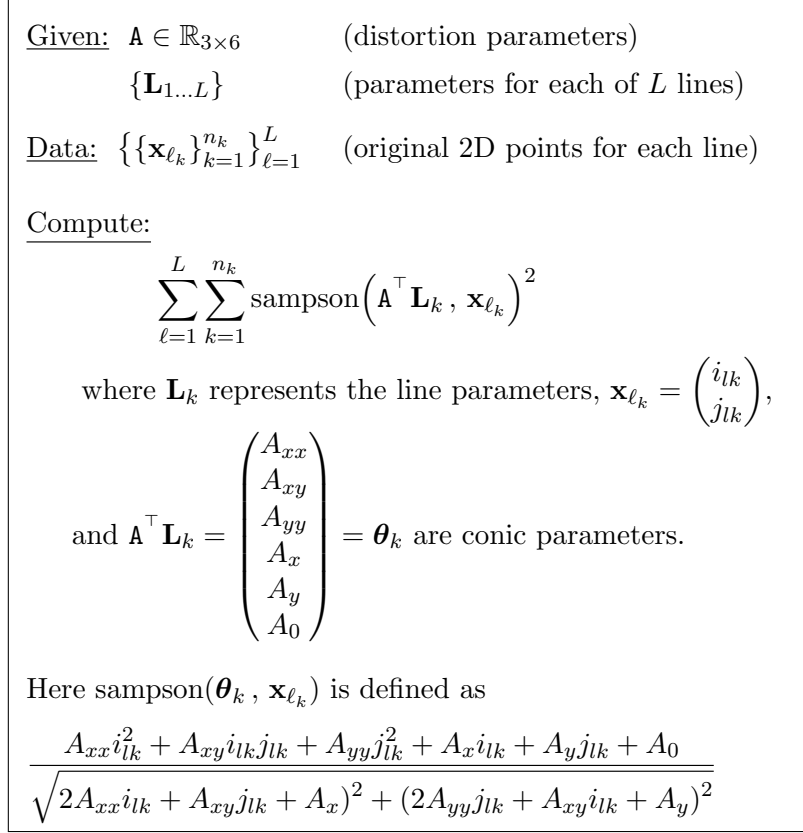


Figure 4.10: Pseudocode for computing the Sampson error measure between image points and plumblines projected as conics under the rational function model.

optimize others as full as it can when they are all more equally scaled.

Note that this optimization computes the error measured on detected edgels in the distorted image. That is, we compute the model that best fits the observed data in its least altered state. The detected edgels are not projected into some other viewing plane, nor are we trying to align the model to some curve fit. The model is evaluated based on the measured distance to each observation, *in situ*.

### 4.2.3 Condensed vs. Full Parametrization

After minimization using the reduced parametrization of §3.2.5, the full 18-parameter model can be fit nonlinearly. The complete model contains too many degrees of freedom to be fit directly via nonlinear optimization; even with multiple restarts from random initializations it converges to incorrect local minima. The restricted parameter space of the condensed model provides a robust constraint that provides easy initialization (reasonable guesses can be made since the parameters correspond to physically meaningful parameters) and stable convergence. This is shown in Figure 4.9 where the initialization parameters are far from the true values, yet the correct optimization result is reached

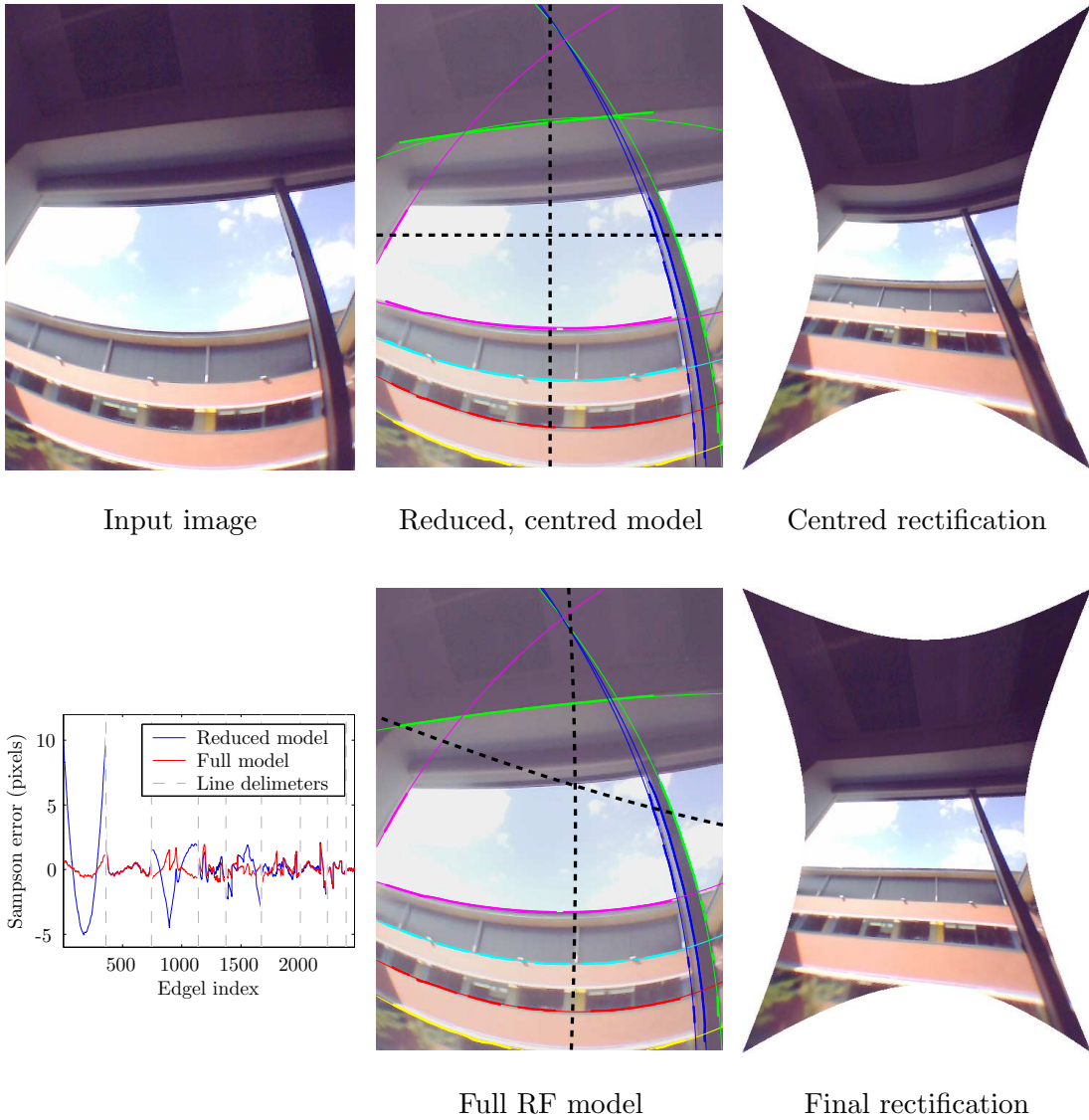


Figure 4.11: Decentred distortion can be handled by fitting the full RF model after initialization using the reduced parametrization. *Top row* The reduced model assumes the distortion centre is at the image centre. The dashed black lines denote the horizontal and vertical planes; their intersection is the distortion centre. The thick coloured lines are detected edgels in the image. The thin coloured lines are the conics fit to these edgels; these are the input data for calibration in both cases. The horizontal green line at the top of the window passes through the detected edgels, but does not match the curve of the window frame. Such mis-matched conics near the periphery of an image are one indication of a poor distortion fit. *Bottom row* Fitting the full RF model reduces the residuals from 1.98 to 0.53 pixels RMS and produces a better rectification; this is most evident at the top edge of the window. Note where the dashed black lines intersect that the distortion is not centred and that the fitted model requires significant skew in the image plane.

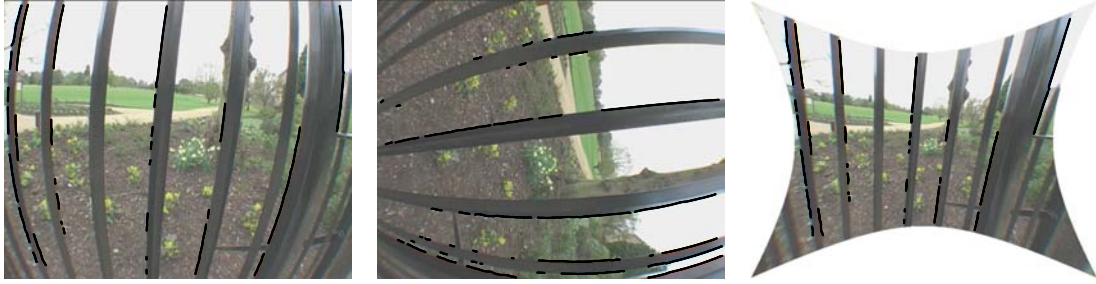


Figure 4.12: Calibration from multiple views. Lines observed in separate views can be combined in a single calibration because  $\mathbf{A}$  describes the *internal* properties of a camera. This rectification used only 14 lines (7 from each view), some of which are made up of a collection of very short edgel chains.

every time.

So do we really need the full modelling power of the 18 parameter RF model? In short, yes. The reduced model assumes that: the distortion centre is located at the centre of the image, there is no skew, and that the distortion is symmetric. These assumptions will not always hold, particularly for low cost lenses. Figure 4.11 gives an example of an image where the distortion centre has moved significantly from the image centre, most likely due to a misalignment of the lens relative to the CCD. This image is part of a sequence of photographs of the same scene; the rest of the images were calibrated accurately using the reduced parametrization, it was only this one that exhibited the displaced distortion centre. This dataset highlights another feature of optimization based plumblines method: it is possible to identify lines that may have been incorrectly labelled as straight. As illustrated in Figure 5.5, it is difficult to automatically detect lines that should be straight, particularly if the physical object is gently curved. The roofline visible through the window in Figure 4.11 is actually curved. This was realised solely by evaluating the error distribution across the fit line segments.

#### 4.2.4 Plumblines data from multiple views

The plumblines constraint obtained from edgels in one image is entirely independent of the camera's external calibration parameters, so repositioning the camera does not alter the *internal* calibration  $\mathbf{A}$ . This enables us to combine constraints  $\mathbf{C}$  obtained from  $M$  views in a single calibration:

$$[\mathbf{C}_1 \mid \dots \mid \mathbf{C}_M] = \mathbf{A}^\top [\mathbf{L}_1 \mid \dots \mid \mathbf{L}_M].$$

These views may be of different scenes or even calibration grids; the only requirement is that there be straight lines visible and that the same camera be used. Figure 4.12

shows a calibration obtained from two views of a wrought iron fence.

We have shown how the rational function model permits an elegant formulation of the plumblines constraint, and have demonstrated camera calibration from a single view of straight lines. The simple form of the model (world lines expressed as image conics) accurately and succinctly represents the underlying camera properties. The model can be fit to noisy data by a nonlinear optimization which minimizes a geometric quantity in the image plane. The linear factorization algorithm for calibration is elegant but its sensitivity to noise makes it effectively useless for real world problems. An area for future research would be to explore whether this deficiency can be somehow corrected.

### 4.3 Multiview Calibration from Epipolar Constraints

This section describes three methods for computing the lifted fundamental matrix  $\mathbf{G}$  from point correspondences. The first is a strictly linear algorithm based on the DLT which works in the absence of image noise. Two nonlinear optimization approaches are then described which use different parameterizations. One enforces only the rank 2 constraint on the fundamental matrix, while the second includes an explicit formulation of the calibration matrix  $\mathbf{A}$  but does not represent the full rational function model. A method for recovering  $\mathbf{A}$  from a computed  $\mathbf{G}$  in the absence of image noise is also described. Testing and evaluation of all the algorithms described here is included in §5.4.

#### 4.3.1 Linear method for $\mathbf{G}$

The Veronese lifting (see §3.2.1) of image points to a six-dimensional space encompasses the nonlinear aspect of the lens distortion and permits a linear solution for the fundamental matrix. Careful data conditioning is in general important for computing a fundamental matrix from noisy image correspondences (Hartley 1997), but it proves essential in this lifted case as the higher order terms amplify the noise. The steps of the linear algorithm are as follows.

1. Condition the input image data points so that all data lies within the range  $[-0.5 \dots 0.5]$ . This produces a conditioning matrix  $\mathbf{C}$  which conditions the homogeneous data as per

$$\mathbf{x}_c = \pi(\mathbf{C}\mathbf{x}) \quad (4.21)$$

where  $\mathbf{x}_c$  represents the conditioned input points  $\mathbf{x}$ . At this time we also produce the lifted conditioning matrix  $\zeta$  which can be used to condition and de-condition

lifted image points.

$$\mathbf{x}_c = \boldsymbol{\zeta} \mathbf{x} \quad (4.22)$$

If the elements of  $\mathbf{C}$  are

$$\mathbf{C} = \begin{bmatrix} c_1 & c_2 & c_3 \\ c_4 & c_5 & c_6 \\ c_7 & c_8 & c_9 \end{bmatrix}$$

then the elements of  $\boldsymbol{\zeta}$  are

$$\boldsymbol{\zeta} = \begin{bmatrix} c_1^2 & 0 & 0 & 2c_1c_3 & 0 & c_3^2 \\ 0 & c_1c_5 & 0 & c_1c_6 & c_3c_5 & c_3c_6 \\ 0 & 0 & c_5^2 & 0 & 2c_5c_6 & c_6^2 \\ 0 & 0 & 0 & c_1 & c_2 & c_3 \\ 0 & 0 & 0 & c_4 & c_5 & c_6 \\ 0 & 0 & 0 & c_7 & c_8 & c_9 \end{bmatrix}.$$

2. Lift the point correspondence data from each view according to the Veronese mapping

$$\mathbf{x} = \text{veronese}(\mathbf{x}_c) \quad (4.23)$$

3. Condition the lifted points so that each dimension of the data has RMS magnitude of  $\sqrt{2}$  and zero mean.

$$\mathbf{x}_{jc} = \mathbf{L}_j \mathbf{x}_j \quad (4.24)$$

For the  $i^{\text{th}}$  dimension of  $\mathbf{x}$  we compute the mean  $m_i$  and RMS magnitude  $s_i$  (about that mean) to produce the conditioning matrix

$$\mathbf{L}_j = \begin{bmatrix} 1/s_1 & 0 & 0 & 0 & 0 & -m_1/s_1 \\ 0 & 1/s_2 & 0 & 0 & 0 & -m_2/s_2 \\ 0 & 0 & 1/s_3 & 0 & 0 & -m_3/s_3 \\ 0 & 0 & 0 & 1/s_4 & 0 & -m_4/s_4 \\ 0 & 0 & 0 & 0 & 1/s_5 & -m_5/s_5 \\ 0 & 0 & 0 & 0 & 0 & -m_6/s_6 \end{bmatrix}. \quad (4.25)$$

This second stage of conditioning is somewhat redundant, but ensures that the linear algorithm receives data in a condition that is most likely to produce a stable output. The subscript  $j$  in (4.24) denotes the view; a different condition matrix is generated for each view.

4. Assemble the design matrix  $\mathbf{D}$  in a manner analogous to the DLT, except in six dimensions rather than three. This is most easily made precise with MATLAB's `meshgrid` and column indexing commands:

```
[ii,jj] = meshgrid([1:6]);           % indexing
D = chi_1t(:,ii(:)') .* chi_2t(:,jj(:)'); % design matrix
```

where `chi_1t` is a matrix of point data with each row corresponding to the transpose of a point  $\mathbf{X}_{1c}$ .

5. Compute the singular value decomposition of the design matrix

$$[\mathbf{U}_D \mathbf{S}_D \mathbf{V}_D] = \text{svd}(\mathbf{D}) \quad (4.26)$$

6. Assemble  $\mathbf{G}_{\text{proposed}}$  by reshaping the singular vector corresponding to the selected singular value. This is generally the smallest singular value, so we reshape the last column  $\mathbf{V}_D(:, 36)$ , but others may be tested for robustness to noise — see below for discussion.
7. Truncate  $\mathbf{G}$  to be rank 2 by computing  $[\mathbf{U}_G, \mathbf{S}_G, \mathbf{V}_G] = \text{svd}(\mathbf{G}_{\text{proposed}})$  and reconstructing  $\mathbf{G}_{\text{truncated}}$  using  $\mathbf{S}'_G$  where all but the first two diagonal entries in  $\mathbf{S}_G$  have been set to zero. The resulting  $\mathbf{G}_{\text{truncated}}$  is then scaled so that its Frobenius norm is 1.
8. De-condition the result to factor out the scaling and translations applied to the input data:

$$\mathbf{G} = \boldsymbol{\zeta}^\top \mathbf{L}_2^\top \mathbf{G}_{\text{truncated}} \mathbf{L}_1 \boldsymbol{\zeta} \quad (4.27)$$

This linear algorithm to compute  $\mathbf{G}$  is highly susceptible to image noise, even when two-stage conditioning is used. The truncation to enforce the rank 2 constraint (Step 7 above) introduces significant error because the truncation is rather arbitrary. It minimizes the error in the matrix norm space, but this has little or no bearing on the actual constraint. This is explored in §5.4, but we now turn our attention to a nonlinear method for computing  $\mathbf{G}$  that enforces the rank 2 constraint explicitly.

### 4.3.2 Rank 2 nonlinear optimization method for $\mathbf{G}$

The fundamental matrix should be rank 2. This section therefore describes a nonlinear optimization that seeks a rank 2  $\mathbf{G}$  that minimizes the Sampson error in both views. The parametrization defines a rank 2 matrix, so the solution obtained is guaranteed to meet the constraint; no truncation is required.

To ensure that the result was rank 2,  $\mathbf{G}$  was parameterized as a sum of two rank-one matrices thus:

$$\mathbf{G} = u_1 u_2^\top + u_3 u_4^\top, \quad u_{1..4} \in \mathbb{R}^6. \quad (4.28)$$

The optimization requires an initialization, and for this we shall use the output of the linear algorithm of the preceding section. MATLAB's `lsqnonlin` was used to perform the optimization (`fminsearch` was also tested but was found to provide inferior performance).

This 24 parameter representation of the lifted fundamental matrix (4.28) enforces the rank constraint, but does not expose a simple means for recovering the calibration matrix  $\mathbf{A}$ . Section 4.3.3 describes a method for performing this extraction in the absence of noise, but the parametrization of §4.3.4 is superior in that it defines a  $\mathbf{G}$  to both include  $\mathbf{A}$  and enforce the rank constraint. Prior to presenting these two methods we will describe the error measure used to evaluate  $\mathbf{G}$ .

### Measuring the error for $\mathbf{G}$

For a given  $\mathbf{G}$  and known point correspondences the error is computed using the Sampson distance approximation in each view. The error in the first view is the distance from a point in that view to the epipolar curve of the corresponding point in the other view. The epipolar curve is a conic  $\boldsymbol{\theta} = \boldsymbol{\chi}_2 \mathbf{G}$  and the distance to the point  $\boldsymbol{\chi}_1$  in the first view is given by (4.20). The error in the second view, for this same point correspondence, is computed in the same manner. The error measure used to evaluate  $\mathbf{G}$  obtained for a pair of views is

$$\text{error} = \frac{1}{n} \sum_{i=1}^n \sqrt{\frac{1}{2}(s_i^2 + s_i'^2)} \quad (4.29)$$

where  $s_i$  and  $s_i'$  are the Sampson distance approximations for point  $i$  in the two views respectively.

### 4.3.3 Recovery of $\mathbf{A}$ from $\mathbf{G}$

We now demonstrate that the distortion matrix  $\mathbf{A}$  can be recovered from the matrix  $\mathbf{G}$ , up to a projective homography. This  $\mathbf{A}$  can then be used to rectify the two images of the scene. Once rectified, the images and 2D correspondences can be passed through standard structure-from-motion algorithms for multiple views (based on the pinhole projection model, and also only determined up to a projective ambiguity) to recover 3D scene geometry.

Given  $\mathbf{G}$ , we wish to decompose it into factors  $\mathbf{A}'^\top \mathbf{F} \mathbf{A}$ , where  $\mathbf{F}$  is rank two. It is clear that any such factorization can be replaced by an equivalent factorization  $(\mathbf{H} \mathbf{A}')^\top (\mathbf{H}^{-\top} \mathbf{F} \mathbf{H}^{-1})(\mathbf{H} \mathbf{A})$  so we can at best expect to recover  $\mathbf{A}$  up to a premultiplying homography. By ensuring  $\mathbf{F}$  has the form of an essential matrix (*i.e.*  $[\mathbf{t}]_\times \mathbf{R}$  where  $\mathbf{R}$  is a rotation) this ambiguity can be reduced to a rotation of camera coordinates.

A method for computing  $\mathbf{A}$  by extracting its 3D orthogonal complement from the nullspace of  $\mathbf{G}$  is given in Appendix C.

Figure 5.12 shows the result of an implementation of this process on synthetic data generated using the real-world  $\mathbf{A}$  obtained from our fisheye lens. The recovered



$\mathbf{A}$  does indeed rectify the house, showing that the algorithm works in principle on noise free data. However, the assumption that the  $\mathbf{A}$  matrix is common to both views is not represented in the methods used to compute  $\mathbf{G}$ . Neither the linear nor the nonlinear methods described above enforce this constraint<sup>2</sup>. We shall now describe a parametrization for  $\mathbf{G}$  that corrects this deficiency.

#### 4.3.4 Parameterizing $\mathbf{G}$ using the reduced RF model

Here we describe a method for parameterizing  $\mathbf{G}$  (for nonlinear optimization) that uses few parameters and explicitly solves for  $\mathbf{A}$ . This speeds convergence, increases the robustness to image noise, and simplifies the initialization.

Section 4.3.2 described a nonlinear optimization to refine the estimate of  $\mathbf{G}$  given a set of lifted point correspondences. The lifted fundamental matrix  $\mathbf{G}$  is rank 2 and this trait is used to define the 24 parameter representation (sum of two rank one matrices). Once this  $\mathbf{G}$  has been obtained a further step (§4.3.3) is required to extract the calibration matrix  $\mathbf{A}$ . This process is cumbersome and error-prone in the presence of image noise (see §5.4).

An alternative parametrization of  $\mathbf{G}$  is suggested by (C.1) and the reduced RF model of §3.2.5. The reduced model for  $\mathbf{A}$  involves only two parameters, so it is possible to initialize the computation of  $\mathbf{G}$  without the linear algorithm of §4.3.1. Reasonable values for the aspect ratio  $a$  and field of view  $\phi$  can be chosen or guessed for most types of lenses. The initialization then consists of removing distortion using the initial  $\mathbf{A}$  and estimating a pinhole fundamental matrix  $\mathbf{F}$  using existing techniques (Longuet-Higgins 1981). The entire calibration involves the following steps

1. Initialize  $\mathbf{A}$  using a guessed aspect ratio  $a$  and field of view  $\phi$  according to (3.17), the reduced Rational Function model (see §3.2.5 for a detailed description). The constants are
  - $h$  = image height in pixels,
  - $w$  = image width in pixels,
  - $b = 20,000$ .

A transformation  $\mathbf{H}$  is applied so that the corrected image corners are mapped onto the original image corners.

$$\mathbf{A}_{\text{initial}} = \mathbf{H}\mathbf{A}_{\text{reduced}} \quad (4.30)$$

2. Remove the lens distortion (approximately) from the image using  $\mathbf{A}_{\text{initial}}$ . If the

---

<sup>2</sup>The described procedure solves for  $\mathbf{A}$  alone; repeating the calculations with  $\mathbf{G}^T$  yields a solution for  $\mathbf{A}'$ . If  $\mathbf{A}$  and  $\mathbf{A}'$  are the same (such as when analyzing two images taken by the same camera with fixed internal parameters) then both of these constraints can be used to recover  $\mathbf{A}$ . This was realized later on, and as such has not yet been implemented.

lifted image point data from the  $i^{\text{th}}$  view is denoted  $\mathbf{x}_i$  then

$$\mathbf{x}'_i = \pi(\mathbf{A}_{\text{initial}}\mathbf{x}_i) \quad (4.31)$$

is the distortion corrected point data for that view.

3. Condition this distortion corrected data using the  $3 \times 3$  transformation  $\mathbf{T}_i$  to ensure that each dimension has RMS magnitude  $\sqrt{2}$  and zero mean. The matrix  $\mathbf{T}_i$  is defined in a similar fashion to  $\mathbf{L}_j$  in (4.25).
4. Compute the standard fundamental matrix from the conditioned, distortion corrected points. This  $\mathbf{F}_{\text{initial}}$  is then converted to the parameterized form

$$\mathbf{F} = \begin{bmatrix} \times & \times \\ \times & \times \\ \times & \times \end{bmatrix} \begin{bmatrix} 1 & 0 & \times \\ 0 & 1 & \times \end{bmatrix} \quad (4.32)$$

where  $\times$  denotes a parameter entry in the matrix. This eight parameter form ensures that  $\mathbf{F}$  is rank two.

5. Perform nonlinear optimization with

$$\mathbf{G} = \mathbf{A}^\top \mathbf{T}_2^\top \mathbf{F} \mathbf{T}_1 \mathbf{A}. \quad (4.33)$$

The ten parameter optimization vector contains two parameters for  $\mathbf{A}$  and eight for  $\mathbf{F}$ . The conditioning transformations are passed in as constants, as are the image points.

There is now no need for a subsequent step to decompose  $\mathbf{G}$ , since  $\mathbf{A}$  is explicitly defined within the optimization. Not only does this approach simplify the algorithm, it also makes it more robust to noise (§5.4). The reduced parametrization constrains the form of  $\mathbf{G}$  so that the optimization's search space is reduced, and only includes valid calibration matrices.

These constraints represent a number of assumptions made about the cameras we are trying to solve for. The reduced RF model assumes a central camera with zero skew and a distortion centre at the image centre. While this is not strictly true in general (see §4.2.3 and Figure 4.11 for an example), it does provide a close approximation suitable for initialization. The second assumption is more likely to be true: that the camera parameters are the same for both views. Even if this is not known to be true it is easily tested for. The formulation of §4.3.2 is more general in that it permits different camera matrices, however these are needless degrees of freedom if one can determine beforehand that the cameras are the same. In a video sequence without zoom this will be the case, as it will for many other image sequences.

## 4.4 Summary

This chapter has presented three calibration techniques for the rational function lens distortion model, each using a different type of image information. Linear calibration using a planar calibration grid is the simplest, and in situations where the camera is available for offline calibration this provides a reliable calibration technique. Dot detection is easily automated so another key feature is that this calibration can be performed without any user intervention.

The plumline method based on straight lines in images relies on user input to identify which lines are truly straight in the real world. Although an elegant linear factorization solution exists, optimization is required to make the nonlinear plumline method a viable calibration technique in the presence of image noise.

The multiple view calibration method also requires nonlinear optimization to provide stable camera parameters in the presence of image noise. The reduced camera models have fewer parameters to vary and therefore constrain the optimization enough that a stable solution can be reliably found. This reduced solution can be used to initialize an optimization of the general model. With a good starting point the nonlinear search converges to the correct fit even with the full model.

Armed with a suitable optimization strategy, both the plumline and multiple view calibration methods yield very accurate camera parameters through a process that is simple yet robust enough for general application. We turn our attention to evaluating the performance and precision of each of these three calibration methods in the next chapter.

## Chapter 5

# Lens Distortion: Evaluation

In this chapter, both fisheye and standard lenses are calibrated using the RF model fitting algorithms described in Chapter 4. These results are compared with calibrations by existing methods. We shall see that the RF model accurately represents the distortion present in real world lenses, and provides a simplified and scalable family of calibration procedures.

### 5.1 Approximation of existing distortion models

This section investigates how accurately the RF model can approximate existing distortion models. If the distortion correction provided by an existing model is taken to be the gold standard, then this test will show how well the rational function model can match the performance of that algorithm.

Lens distortion is often approximated by the first few even terms of a radial Taylor series expansion (Hartley and Zisserman 2003, Heikkilä 2000, Tsai 1987), and this model has been shown to achieve high accuracy (Beyer 1992). We selected a fourth order polynomial and measured typical values for a lens with moderate distortion (Nikon Coolpix). The 4<sup>th</sup> order radial model was then used to generate a synthetic set of distorted image coordinates. Figure 5.1 shows the fitting error of several alternative models on this synthetic data. The rational function model provides the closest approximation to the 4<sup>th</sup> order radial model, with a maximum difference less than 0.25 pixels.

For fish-eye and other lenses with extreme distortion it may be necessary to include many higher order terms of the Taylor expansion. Devernay and Faugeras (2001) suggest that a more concise representation is obtained through parameterizing on the field of view (FOV), since these lenses are designed so that the image resolution is roughly proportional to the distance from the image centre.

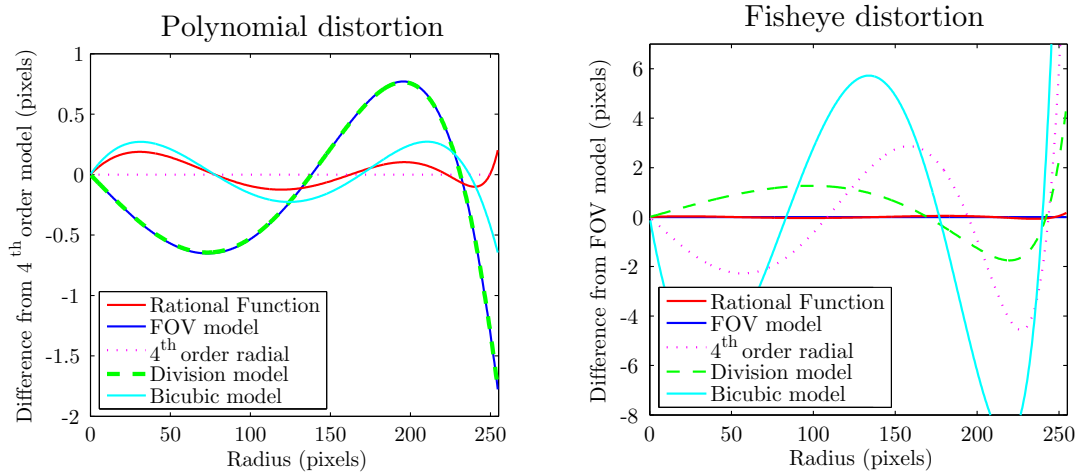


Figure 5.1: Synthetic comparison of the rational function model with existing lens distortion models. *Left* Moderate polynomial distortion corresponding to a standard consumer lens. All models are compared to the 4<sup>th</sup> order fit as it seems to be the most commonly used. Note that for low distortion the FOV model collapses to the division model. *Right* Fish-eye lens distortion is fit best by the FOV model, so it is taken as ground truth. The rational function model provides a very close approximation, and can be fit linearly.

Figure 5.1 also provides a comparison against this FOV model (synthetic data was generated based on the observed parameters of a Raynox 0.3X fish-eye adapter fitted to a Canon XM2 digital camcorder). Once again, the rational function model provides an extremely close approximation while the polynomial models vary widely.

Both of these tests compare the rational function model against other distortion correction methods. While this provides an indication of how well the new model will perform on data that the existing techniques model exactly, such an exact fit is never realized in an application. It is important to bear in mind that all of these models are merely approximations to a camera's true distortion function; we would like to select the model which most accurately fits this underlying (unknown, and potentially highly complex) function with the least number of parameters. The next section therefore compares each of the distortion models (including the rational function model) against real image data.

## 5.2 Planar Grid Calibration

This section compares the accuracy of removing lens distortion from an actual image of a calibration target using the various distortion models under consideration. The overall approach is to detect some positions within a distorted image, correct the dis-

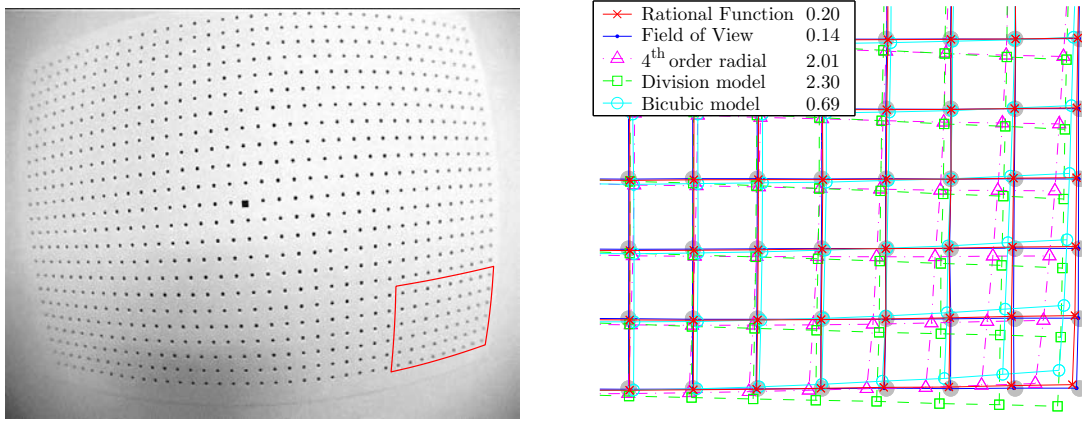


Figure 5.2: Comparison of distortion removal for the lower right corner of the grid in the image shown. For each of the comparison models, distortion parameters and a planar homography were fit via nonlinear optimization. The RMS errors (in mm) for each calibration method are listed in the legend; with no distortion correction the RMS error is 6.76 mm, and the overall dimensions of the target are  $277 \times 184$  mm. The RF model is comparable with the fish-eye specific FOV model, but permits linear estimation.

tortion, and then measure the error as the distance between the distortion corrected points and some known ground truth locations for the points. The challenge is in how to define the ground truth. Since all of the distortion correction techniques I study allow the production of an image that follows the pinhole projection model (refer to the introduction to this chapter for a discussion of this point), one might try to produce the pinhole image that corresponds to the distorted one to use as ground truth. This requires knowledge of the feature positions in the real world, internal camera parameters, and the camera position in the world coordinate frame. The first of these is easily obtained by using a planar calibration grid of known dimensions and defining the world coordinate system relative to that plane. The camera parameters and position, however, are in general not known *a priori* and are difficult to measure independently of lens distortion (Heikkilä 2000).

A camera’s internal and external parameters can all be treated as a single projective homography that maps planar world points to planar image points under the pinhole projection model (see §2.2 for further details). We therefore adopted a two-stage approach for obtaining the ground truth: the lens distortion was compensated for by the chosen model, and then a projective homography was used to map the distortion corrected points onto the planar locations of the calibration grid markers. These two stages are illustrated in Figure 4.1 (as it is the same as the process used for linear calibration of the rational function model). Error measurements can now be made in

Model	Equation
Rational Function	$(p, q) = \left( \frac{A_1^\top \mathbf{X}(i, j)}{A_3^\top \mathbf{X}(i, j)}, \frac{A_2^\top \mathbf{X}(i, j)}{A_3^\top \mathbf{X}(i, j)} \right)$
Field of View (3.4)	$(p, q) = \frac{\tan(r_d \phi)}{2r_d \tan(\omega/2)}(i_d, j_d)$
4 <sup>th</sup> order Radial (3.1)	$(p, q) = (1 + k_1 r_d^2 + k_2 r_d^4)(i_d, j_d)$
Division (3.2)	$(p, q) = \frac{1}{(1 + \lambda r_d^2)}(i_d, j_d)$
Bicubic (3.3)	$p = b_1 i_d^3 + b_2 i_d^2 j_d + b_3 i_d j_d^2 + b_4 j_d^3 + b_5 i_d^2 + b_6 i_d j_d + b_7 j_d^2 + b_8 i_d + b_9 j_d + b_{10}$ $q = c_1 i_d^3 + c_2 i_d^2 j_d + c_3 i_d j_d^2 + c_4 j_d^3 + c_5 i_d^2 + c_6 i_d j_d + c_7 j_d^2 + c_8 i_d + c_9 j_d + c_{10}$

Table 5.1: Summary of the distortion models used in planar calibration grid image correction tests. Aside from the rational function model, each of the comparison models, distortion parameters and a planar homography were fit via nonlinear optimization. The rational function model was fit linearly and includes the homography.

millimeters on the calibration grid.

Model specific calibration procedures are available for several of these distortion models, however for this test they were all calibrated by nonlinear optimization (see Figure 5.3 to confirm that global rather than local minima were achieved in all cases). This consistency ensures that it is the models that are compared and not the models coupled with their calibration methods. Although the latter would also be an interesting study, it is relegated to the realm of future work. The optimization was initialized to be a pinhole camera for each different model; this was found to yield the global minimum in each case. This optimal solution was tested by repeating the search both with random initialization values and by initializing with the parameters of the most accurate solutions found by any of the algorithms. In every trial the error either matched or was larger than that reached from initialization as a pinhole camera.

Figure 5.2 shows the input image for calibration and a plot of the distortion correction results for the lower right-hand portion of the calibration grid. This section of the grid is highlighted because it is near the corner of the image — the area where errors in distortion correction tend to be greatest. As described above, the ground truth data is the dot pattern on the planar calibration grid. These locations are plotted as grey circles in Figure 5.2 *right*. The distorted image locations  $(i, j)$  are mapped onto this grid by first removing lens distortion according to the chosen model

Model	RMS errors (mm)		$a$	$c_x$	$c_y$	Model parameters
	Raw	Transformed				
None		6.76				
Division	2.32	2.30	-1.05	-37.5	-101.2	$\lambda = -0.994$
Radial	2.31	2.01	1.07	6.9	-152.7	$k_1 = -0.115 \quad k_2 = -4.16$
Bi-cubic	0.69	0.69	-2.19	31.6	-158.0	<i>See below</i>
FOV	2.34	0.14	1.07	12.7	-84.4	$\phi = -1.77$
Rational	0.20	0.20	n/a	n/a	n/a	<i>See below</i>

Bi-cubic parameters:

$$\mathbf{b} = \begin{bmatrix} 1.69 & 14.10 & 7.89 & 54.36 & 0.94 & 2.59 & 7.48 & 1.15 & 9.36 & 1.32 \end{bmatrix}$$

$$\mathbf{c} = \begin{bmatrix} -10.70 & 7.12 & -45.44 & 27.18 & -1.42 & -2.90 & 1.99 & -6.71 & 4.70 & 1.49 \end{bmatrix}$$

Rational parameters:  $A_1^\top = \begin{bmatrix} 0.009 & -0.011 & -0.035 & 0.879 & -0.069 & 0.018 \end{bmatrix}$

$$A_2^\top = \begin{bmatrix} -0.005 & 0.050 & -0.014 & 0.068 & 0.816 & 0.103 \end{bmatrix}$$

$$A_3^\top = \begin{bmatrix} -1.122 & 0.005 & -0.983 & 0.107 & -0.282 & 1.000 \end{bmatrix}$$

Table 5.2: Table of residual errors and calibration parameters for planar grid distortion correction by various models.

$$(p, q, 1) = F(i, j) \quad (5.1)$$

and then applying the projective homography  $\mathbf{H}$

$$(x, y) = \pi \left( \mathbf{H} F(v; i, j)^\top \right). \quad (5.2)$$

which minimizes the difference between the transformed locations  $(x, y)$  and the coordinates of the dots printed on the calibration target. The distortion models  $F(v; i, j)$  (with  $v$  denoting the model parameters) are described in §3.2.5, but the pertinent equations are listed in Table 5.1 for easy reference. The distorted radius is given by  $r_d = \sqrt{(i - i_c)^2 + a^2(j - j_c)^2}$  where  $(i_c, j_c)$  is the distortion centre and  $a$  is the pixel aspect ratio. As discussed in §3.1, the distortion centre need not coincide with the principal point in an image. A reasonable initialization for these values is to set the distortion centre at the image centre, and the aspect ratio to one. However, this should only be viewed as an *initialization*; particularly for lenses with significant distortion these values lead to unacceptably large calibration errors. Table 5.2 lists the numerical results of calibration using each of the distortion models. The column of “raw” RMS errors are for a calibration centred at the image centre with an aspect ratio of one. The errors reported in the “transformed” column include the centre of distortion and aspect ratio in the distortion function optimization:

$$(x, y) = \pi \left( \mathbf{H} F(v; i - w/2 - c_x, (j - h/2 - c_y)/a)^\top \right). \quad (5.3)$$



Here  $w$  and  $h$  are the scaled width and height of the  $720 \times 576$  pixel image. The calibration grid measures  $277 \times 184$  mm overall, but both sets of coordinates were centred and scaled so that the horizontal axes' ranges were  $[-0.5 \dots 0.5]$ . The parameters in Table 5.2 are for the transformed input data and expressed in these scaled coordinates, with the exception of the distortion centre offsets. These are given in original image pixels to maintain a familiar frame of reference. The error measure was the RMS value of the distances between the 850 mapped image points and their true grid locations. These distances are measured in the plane of the calibration target, and are therefore expressed in millimeters.

The distortion correction results listed in Table 5.2 highlight a number of points regarding these models and their ability to fit wide angle distortion. The Division and Radial models have large error values, and incorporating the aspect ratio and distortion centre makes very little difference. These errors result from the model's inability to fit the underlying distortion function. As a point of reference, the error is 6.76 mm if only an alignment homography is used. This is the pinhole projection case where lens distortion is ignored. The Bi-cubic and Rational models also do not exhibit any change in error with the addition of the pre-transformation parameters because the model itself is compensating for the centre offset and aspect ratio. In this case, however, the errors are identical and much lower. Note that the Rational error is substantially lower than the Bi-cubic, even though the latter employs a larger number of parameters. The additional parameters and higher order polynomials of the bi-cubic model are insufficient to compensate for the lack of a rational function denominator. The FOV model is the only one that exhibits a significant change with the inclusion of aspect ratio and centering parameters. When these pre-transformation parameters are correct, the single parameter FOV model produces the best fit of all the models tested. This indicates that its underlying function of trigonometric relations models wide angle lens distortion quite well. However, its performance is highly dependent on the aspect ratio and distortion centre; if these are incorrect the performance degrades to the level of the worst models tested. In essence this is then a three parameter model with  $v = [a, c_x, c_y, \phi]$ . If only the aspect ratio is varied the error was found to be 2.31 mm, but if the distortion centre alone was varied the error was 0.54 mm. Thus for this model, the distortion centre is more important than the aspect ratio, but using both produces the best results. A further test was made using the aspect ratio and centre offsets obtained from the FOV optimization as the values for the other models. Fixing these values for the division and radial models (the only others where these values made a difference) was found to increase the error. This indicates that the optimizations are

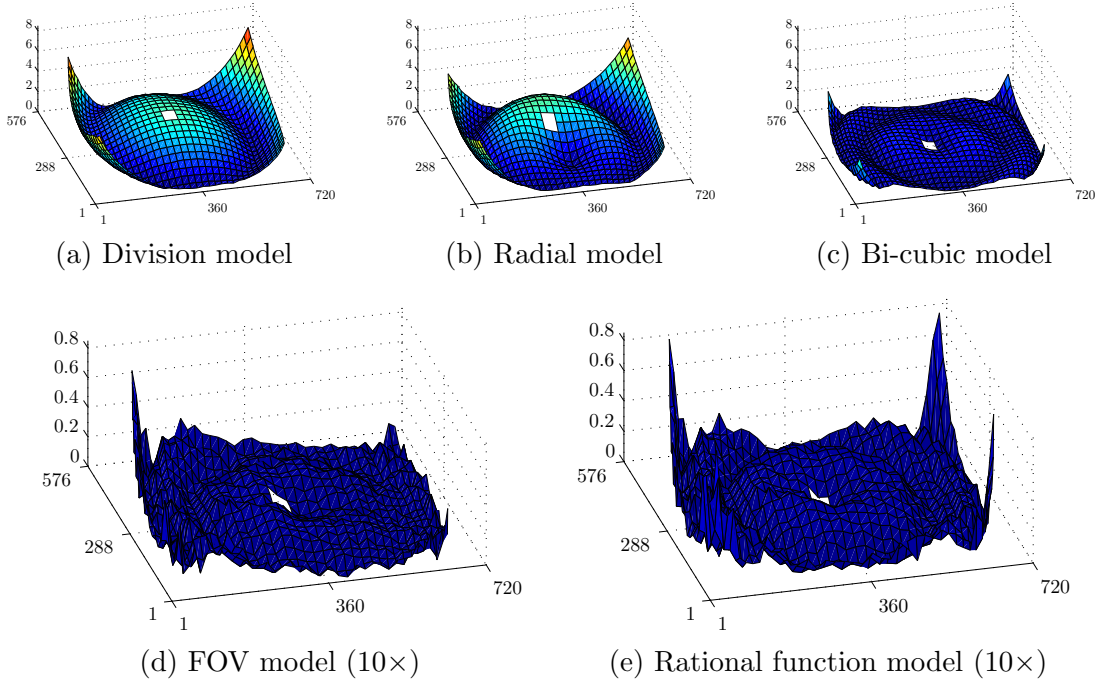


Figure 5.3: Residual errors after removing distortion from a wide angle image (Figure 5.2) of a planar calibration target. The projected error measured in the plane of the calibration object is plotted at each image pixel location within the convex hull of the detected dots. The rational function model nearly matches the performance of the fisheye specific FOV model, yet can be fit linearly. All dimensions are in pixels.

functioning correctly (finding the parameters to produce the lowest error) and that the problem lies with the form of the underlying model rather than with the parameters.

We will now discuss some of the details of the optimizations used to compare these models. All models were initialized with the pinhole perspective model; that is, distortion parameters were set to zero, no image centre offset and an aspect ratio of one. Each optimization converged to the results given in Figures 5.2 & 5.3 and Table 5.2. Each optimization was then repeated 100 times with random initial values to test if these results represent a global (as opposed to a local) minimum. In no case was a lower residual error obtained, and between 43% (for the FOV model) and 80% (for the Bi-cubic model) of the runs converged to the minimal error reached from the pinhole initialization.

The results plot in Figure 5.2 indicates the direction and relative magnitude of errors from the different distortion models, but it doesn't indicate the error distribution over the image area. For this we turn to the error plots of Figure 5.3. The  $x$  and  $y$  axes of each plot are the image dimensions in pixels, while the vertical axes indicates the error for that pixel location when it is projected into the plane of the calibration grid.

This error is measured in millimeters, the unit of measure for the calibration plane. Note that the FOV and RF plots have been scaled by a factor of 10 to show detail; the colormap is consistent with the other plots however, to facilitate comparison. The gap in each surface is from the missing data point where there is a small rectangle in the calibration grid. This rectangle is used in establishing the dot correspondences.

Note the predominant wave shape in the first three plots of Figure 5.3, which indicates a failure to accurately model the lens distortion. This shape is less pronounced in the bi-cubic plot as the higher order polynomials provide a better fit. Some artifacts remain however, which indicates that a still higher order polynomial model may perform better. By contrast, both the FOV and Rational models do not exhibit the same gross fitting error. Recall that the vertical scale on these last two plots is amplified by a factor of 10; the fitting error is almost on the order of the image noise, which has become evident in these plots. The FOV model is based on the fish-eye lens design parameters; this physical basis results in an excellent distortion fit. There is a slight bias from left to right, but very little residual curvature. The flexibility of the RF model yields a very close approximation to both the FOV solution and the true lens parameters, and can be fit linearly from a calibration grid. Here we have seen that the rational function model can be fit linearly in a single step, without the need to first estimate the aspect ratio and distortion centre, and that the residual errors are on the same order as the best lens-specific model available.

### 5.3 Plumblin Calibration

The noise sensitivity of the linear factorization was already explored in §4.2.1; careful conditioning of the input data helps somewhat, but the calibration still fails to rectify above a noise level of 0.01 pixels. For this reason the linear plumblin method is not recommended for use in practice. The recommended optimization method is examined later in this section, but first we shall describe the line detection process used to gather the input data to these plumblin methods.

**Line Detection for Plumblin Methods** The first task in finding straight lines is to detect curved line segments in the distorted image. This is accomplished semi-automatically as shown in Figure 5.4.

Edgels are detected to subpixel accuracy using the Canny edge detector (Canny 1986) with image smoothing  $\sigma = 1.6$  and edge magnitude threshold  $t = 6.0$ . This returns a set of locations where strong edges were located, but no relationships between the edgels are known. Delaunay triangulation is the first step in assembling the edgels

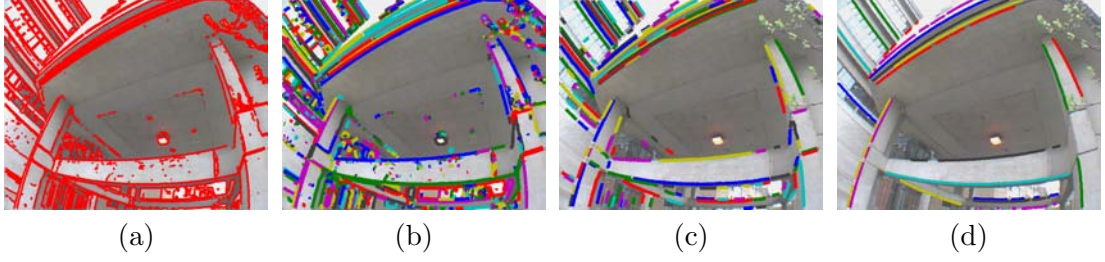


Figure 5.4: Steps used to detect plumblines in an image. (a) Subpixel Canny edge detection (b) link adjacent edges into chains (c) break the chains into linear sections by thresholding on curvature (d) manually identify sections that belong to individual lines that correspond to straight lines in the real world.

into linked chains. This triangulation links each edge to its nearest neighbours. Any triangle edges longer than some threshold (set to 3 pixels in our tests) are deleted. Any remaining junctions or branches are evaluated and only the longest possible chains are kept.

Since we are collecting edge segments as input data to a plumblines distortion correction method we cannot assume that the line segments will be linear. We can however, expect that any curvature induced by the lens distortion will be smooth. Thus we are seeking smoothly varying edge chains; which can be obtained by breaking the linked chains from above at every point where there is an abrupt change in edge orientation. If the orientation of a specific edgel is given by  $\phi(k)$  then a chain is broken at each point that

$$|\phi(k) - \phi(k + a)| + |\phi(k) - \phi(k - a)| > thresh \quad (5.4)$$

where the offset  $a = 5$  and  $thresh = 0.4$  radians.

The task of determining which curved image line segments correspond to straight lines in the real world scene was performed manually for the tests described here. Although it may be possible to partially automate this difficult task (Devernay and Faugeras 2001), the prior knowledge possessed by a human operator is too great to try reproducing it in digital logic for this task (refer to Figure 5.5 for an example). It takes only seconds for a user to view an image and select half a dozen lines which stand a good chance of arising from truly straight world features. These lines do not need to be continuous; it is advantageous to have data from long lines, but there may be large gaps between detected edgels if they are all identified as part of the same line.

Thus we have adjacent edgels linked into curved segments  $\mathbf{e} = e_1 \dots e_L$  that represent straight lines in the real world. We shall refer to the  $k^{th}$  edgel in linked segment  $\ell$  as  $\mathbf{e}_{\ell k} = (i_{\ell k}, j_{\ell k})$ .



Figure 5.5: An example of a common scene that contains many gently curved lines. A human operator can easily identify the edges of the television set as curved surfaces based on prior knowledge. It would take considerable effort to replicate that information in an automatic line detection scheme.

### 5.3.1 Reduced RF consistency

The first test of the RF plumblines technique compares the parameters obtained by fitting the reduced model to lines from different image sequences. This is the standard procedure whereby a camera is calibrated based on several images and then that calibration is applied to other images from the same camera. To examine the validity of this procedure, we performed separate calibrations on three different sets of images from a single camera and then compared the calibration parameters obtained from each image set.

The first image set used for calibration is actually a single image (see Figure 4.2 *left*) of a man-made building where 21 lines were detected. A total of 5805 edgels were available for this calibration. The second set is also only a single image, this time of a planar target chosen for plumblines calibration (see Figure 4.9). Twenty-three lines containing 10,716 edgels were detected in this set of concentric rectangles. Finally, two images of a wrought iron fence constitute the third set. These images contain 4574 edgels in seven lines from each image, and are shown in Figure 4.12.

The calibration parameters for the three image sets are summarized in Table 5.3. The aspect ratio and field of view parameters agree across these three sets. The low residual error indicates that the reduced parametrization fits this lens quite well. The initialization values were  $a = 1.07$ ,  $\phi = 0.5$ , and  $b$  was fixed at  $2 \times 10^7$ . All images are  $720 \times 576$ . It should be noted that in practice all the lines from all the image sets

Image set	Lines	Edgels	$a$	$\phi$	Residual
Tower	21	5805	1.03	0.68	0.42
Grid	23	10,716	1.04	0.63	0.36
Fence	14	4574	1.06	0.70	0.62

Table 5.3: Consistency of the reduced RF model. The residual errors are the RMS Sampson distances from the edgels to the corresponding conics, measured in pixels. The parameters for each set, estimated from only one or two images, display excellent agreement.

would subsequently be combined into a single calibration. Using all available input data will provide an increase in accuracy in the parameter estimation for this lens. The optimization framework is quite efficient, so although this combined calibration contains over 21,000 data points, the algorithm converges in under 200 *s* on a 1.6 GHz laptop.

The reduced parametrization is a stable framework in which to perform the optimization for calibration parameters. One measure of this robustness is the sensitivity to changes in initialization values. In Figure 4.9 the optimization was initialized with both  $\phi = 0.001$  (pinhole camera, shown) and  $\phi = 1$  ( $180^\circ$  field of view); each converged to the same result, implying a wide basin of convergence. A second measure is the ability to cope with noisy input data. Convergence was also verified on synthetic data with a noise level of 2 pixels (Figure 4.5).

### 5.3.2 RF vs. the Matlab Calibration Toolbox

The checkerboard image sequence shown in Figure 5.6 was used to compare the RF plumbline calibration with the MATLAB Camera Calibration Toolbox (Bouguet 2003). The Toolbox requires multiple images taken from different camera orientations; a total of 14 images were supplied as input. The corners between checkerboard squares are detected based on overall grid corners, spacing, grid dimensions, and distortion estimates supplied by the user. From these corners the intersections are detected then camera intrinsic and extrinsic parameters are computed through an iterative refinement procedure. Figure 5.6 shows the points used from one of the input images.

The distortion model includes radial, tangential and decentering terms as detailed in (3.1). The resulting calibration coefficients are listed in Table 5.4. The standard deviation of the reprojection error in each coordinate direction (reported as pixel error) is well below one pixel. This indicates that the model has accurately modeled the input data. However, the majority of the calibration points used by the toolbox were located in the central portion of the image and the many-parameter model diverges outside the region where calibration data is available. Figure 5.7 shows the same input image

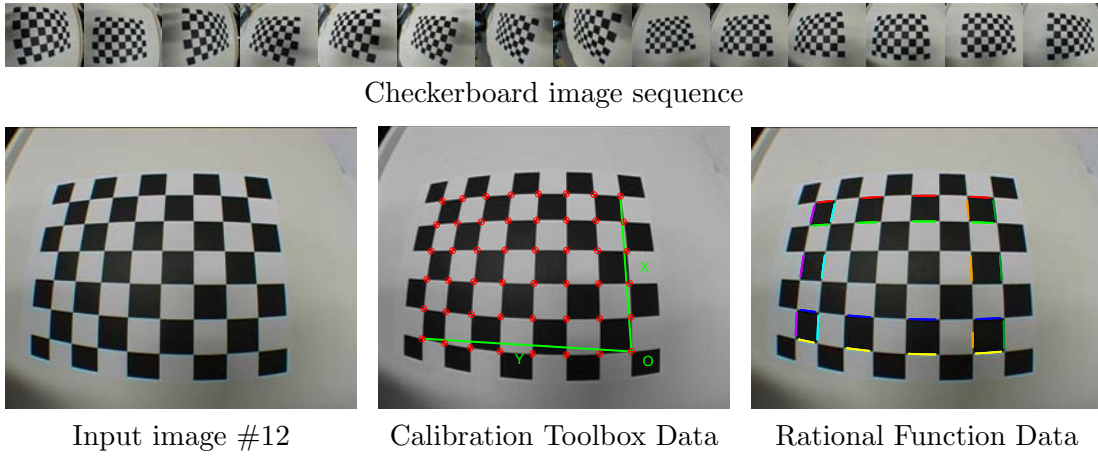


Figure 5.6: Rational function reduced parametrization compared with the MATLAB Calibration Toolbox (Bouguet 2003). Both models were calibrated from the checkerboard image sequence; 14 images were used by the toolbox, while the RF plumblime method was applied to eight lines from the single image shown.

rectified by the Toolbox model. Note that the interior squares of the checkerboard have straight edges, but the square in the lower left corner is far from aligned. This problem may be partially alleviated by using images where the calibration pattern occupies a larger portion of the image. With wide angle lenses such as was used in this example it is, however, quite difficult to obtain an in-focus image of a planar target that fills the view. An additional problem is that the Toolbox requires all the corners in an  $m \times n$  grid to be detected, something which is difficult as the grid is pushed further towards the perimeter of the image area.

To make a fair comparison between the Toolbox results and those obtained from the RF model, plumblines were only taken from within the image area of the corners used by the Toolbox. A sample set of detected lines are shown in Figure 5.6. Each

Radial Distortion:	$k_1 = -0.30884 \pm 0.0019$	$k_2 = 0.09597 \pm 0.0020$	
Tangential Distortion:	$p_1 = 0.00011 \pm 0.00018$	$p_2 = 0.00031 \pm 0.00024$	
	$x$	$y$	Units
Focal length:	$412.312 \pm 0.706$	$438.873 \pm 0.678$	pixels
Principal point:	$370.674 \pm 0.560$	$275.392 \pm 0.570$	pixels
Skew:	Not estimated		
Pixel error:	0.13192	0.11100	pixels

Table 5.4: Results from MATLAB Calibration Toolbox for fisheye images. The input images are shown in Figure 5.6 and the result of using these parameters to remove the distortion in an image is shown in Figure 5.7. The pixel error is the standard deviation of the reprojection error in the  $x$  and  $y$  directions respectively.



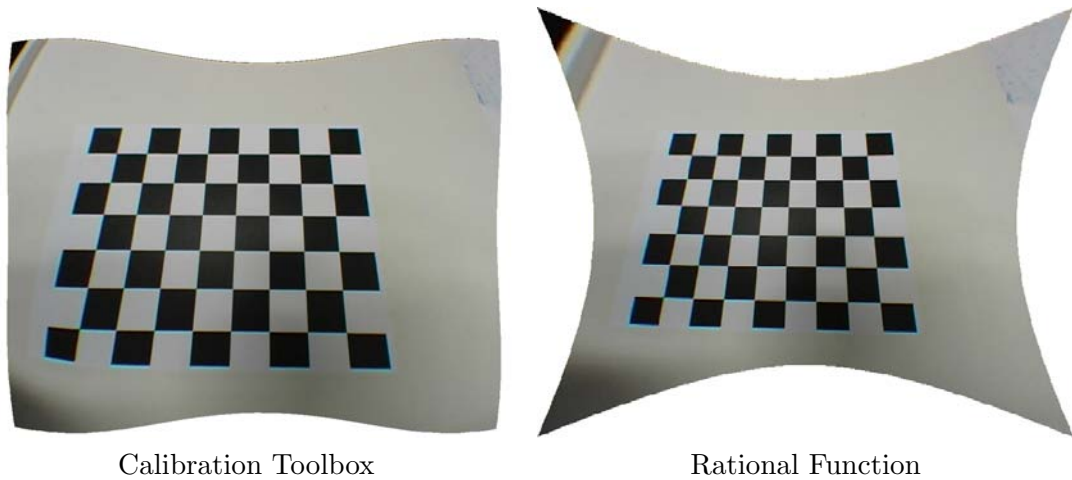


Figure 5.7: Distortion corrected input images. The toolbox implements a full radial, tangential and decentering model (3.1), yet it fails to rectify towards the image boundary. Distortion correction via the RF model does a much better job of generalizing to the entire image. Compare the squares in the lower left corner of each image.

line selects edgels from every other square in the checkerboard pattern; the reversing gradient direction between squares causes an offset (due to edge detection bias) that could introduce oscillations into the curve fit. The outer two lines all around the perimeter of the calibration region were selected for a total of eight lines per input image. Calibration was performed using lines from a single image, as well as using the lines from all 14 images.

A test image is needed to observe how well the distortion corrections generalize to other images taken with the same lens, and also to measure their ability to extrapolate to the image boundary. An image of concentric rectangles (Figure 5.8) can be used to measure the residual errors after removing the (predominantly radial) distortion. Define a set of ground truth rectangles based on the spacing and aspect ratio of the physical pattern. We then find the homography that maps the distortion-corrected edgels onto the ground truth rectangles by minimizing the perpendicular distance between each mapped edgel and its corresponding rectangle side. This was done via nonlinear optimization for each set of residuals. The result is a least squares optimal placement of the corrected data relative to the ground truth. This method could be extended to include weighting of the residuals based on distance from the distortion centre, for example. For our tests we are interested in the error distribution across the entire image equally, so no spatial weighting was applied.

The results of this alignment via projective homography are shown in Figure 5.8. The Calibration Toolbox image clearly shows the failure of the many parameter model to extrapolate to the image perimeter. Some curvature is evident even in the interior



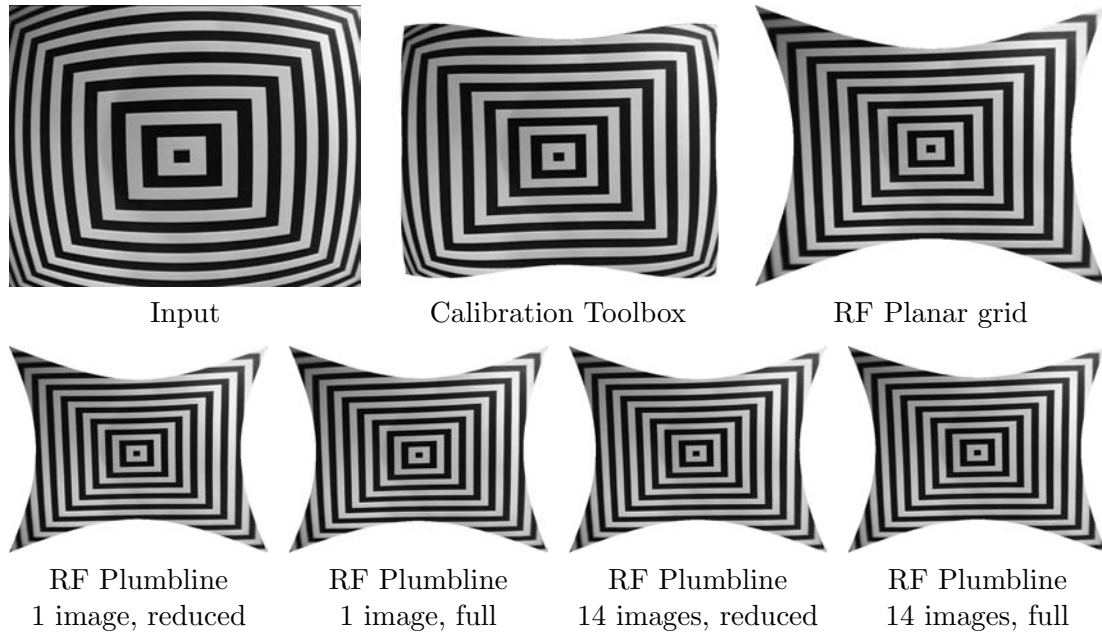


Figure 5.8: Generalization test: various calibration techniques were applied to data from the checkerboard image sequence 5.6. The concentric rectangles illustrate the ability of the calibration to rectify the entire image region. Qualitative inspection indicates that the Calibration Toolbox fails to rectify the outer portion of the image, but comparing the RF models requires the quantitative analysis of Figure 5.9.

rectangles. Compare this image with the rectification from the RF planar grid method. In the latter case there is little or no residual curvature evident in any of the lines. This calibration is a linear algorithm (§4.1), yet it removes distortion better than the iterative Toolbox model. The linear algorithm only needs a single image for calibration (Figure 5.2) while the Toolbox used 14 checkerboard images. To ensure that this is a fair comparison, a RF calibration was also performed using the checkerboard images. The checkerboard edges shown in Figure 5.6 were fed to the RF Plumblin method, and the resulting distortion model was used to correct the images shown in the bottom row of Figure 5.8. As noted in the caption, the calibration was performed using data from a single checkerboard image, as well as from all 14 images. Both the full and reduced parameterizations were used, but the results for all of these test cases are not visually different, so we turn our analysis to the residuals themselves.

Figure 5.9 contains visualizations of the spatial error distributions. Each mapped edgel is colour coded based on its residual error; edgels that are closer to the centre of the image than their ground truth line are shaded red, while those outside are blue. An edgel that coincides directly with its ground truth line (zero error) is shaded purple. The ground truth rectangles are shown in white for reference. The colour scale for each image is defined by three times the standard deviation of its residuals, so the

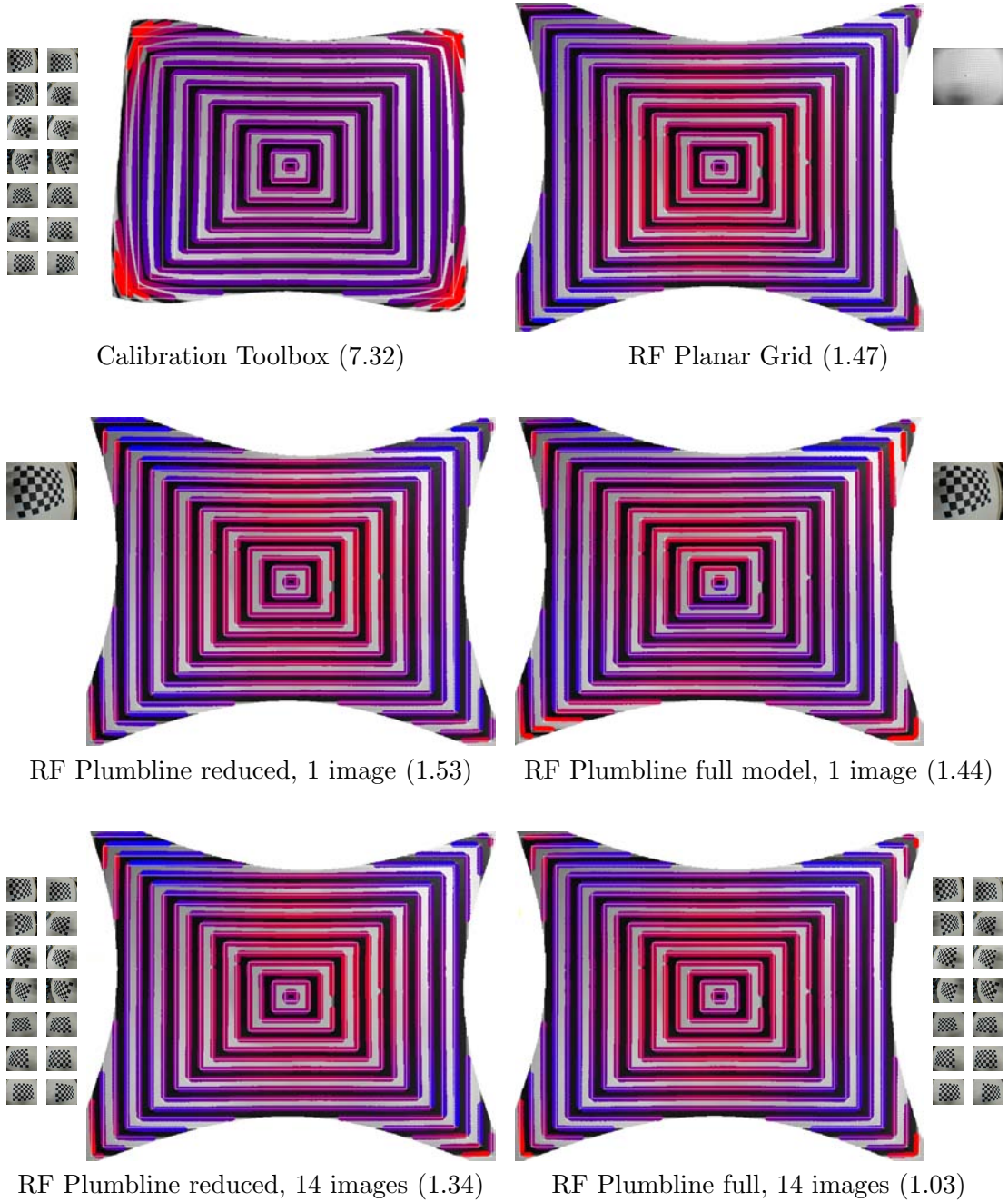


Figure 5.9: Extrapolation of calibration from checkerboard data in Figure 5.6. Residuals are measured relative to concentric rectangle ground truth (thin white lines) fit via homography; the colour coding conveys the spatial error distribution with red indicating residual errors inside the ground truth and blue being outside. The RMS pixel error for each calibration is given in parentheses.

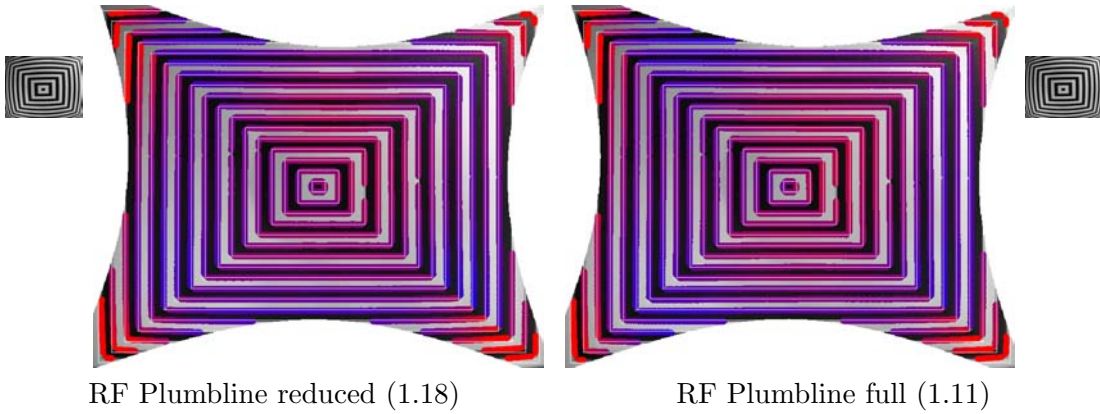


Figure 5.10: Rational Function plumblane calibration based on edgels in the concentric rectangle image to demonstrate the level of rectification possible by direct fit. The edgels used for this calibration are the ones used to measure the residuals in Figure 5.9. The RMS pixel error for each calibration is given in brackets, and the colour coding is the same as for Figure 5.9. Refer to the text for an explanation of the higher error (1.11 pixels) than observed in the previous figure (1.03 pixels)

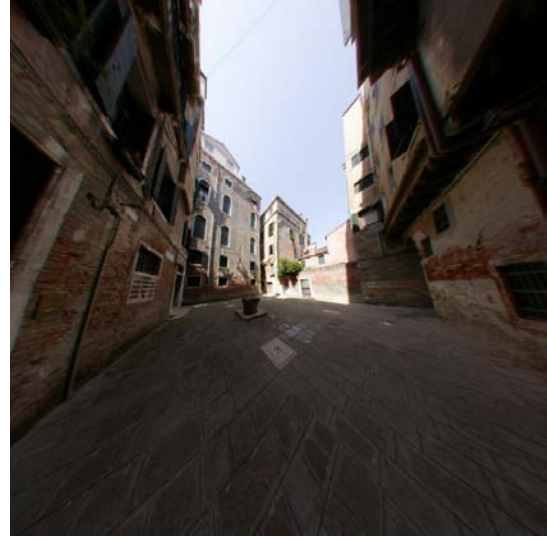
shading does not permit comparisons *between* images. The error magnitude between methods is evaluated by the RMS pixel error. The error on the Toolbox image is over four times greater than for any of the RF models. The results obtained using lines from a single image compares favorably with the RF planar grid results, particularly if the full model is used. Using data from additional lines (all 14 images) improves the result; if the full model is used the residuals are the lowest observed for calibrating this lens. This indicates that the RF model extrapolates well to the image boundaries, even if the full model is used.

The RF plumblane model was calibrated on the line data from the verification image (concentric rectangles) itself to evaluate the generalization of the calibration. The results (Figure 5.10) indicate a higher RMS error (1.11 pixels) than was obtained with the calibration on the checkerboard images (1.03 pixels). Although the checkerboard calibration has to extrapolate to the image boundary, and the images are from a different sequence, the additional data available in that set (from using eight lines from 14 images) provides such an advantage that the resulting calibration is better than that obtained by calibration on the test image itself.

These higher residuals are not the result of premature convergence of some local minima in the homography optimization. Instead, they are higher because the  $\mathbf{A}$  obtained from the plumblane calibration on the rectangular line data from this single image is not as accurate as the  $\mathbf{A}$  obtained by the plumblane method on data from all 14 checkerboard images.



Input image of a courtyard in Venice



Rectified image

Figure 5.11: A  $180^\circ$  field of view fisheye image of a Venetian courtyard corrected via plumline calibration of the reduced parametrization. Only the central portion of the rectified image is shown so that the highlighted buildings would be displayed at a higher resolution. Although there is some residual curvature evident at the vertical building edge in the centre of the image, the reduced model has done a remarkable job of correcting this complicated lens.

**Another fisheye lens** Figure 5.11 shows the result of calibrating the distortion from pinhole that is present in an image taken with a Sigma 8mm-f4-EX lens fitted to a Canon EOS-1D digital camera. The input image is one of thirteen images from the Venice Yard QY dataset recorded by Branislav Mićušik, although calibration lines were selected from several of the images for convenience. We observe that all of the foreground lines have been accurately rectified. There is a slight curve to the vertical edge of the building on the right in the centre of the image, but this is fairly minimal. One of the assumptions of the plumline method is that lines chosen for calibration represent straight line in the real world. With old buildings (particularly those built on wooden pilings in a lagoon, such as those in Venice) this may not be strictly true. In spite of this, the plumline calibration provides a highly accurate image rectification of this  $180^\circ$  field of view fisheye lens.

## 5.4 Multiview results

We present results on two experimental questions: how stable is the computation of  $\mathbf{G}$  as measured by the quality of epipolar curves; and how effective is rectification using the recovered  $\mathbf{A}$ . The performance of the linear algorithm and the rank two optimiza-

tion are examined first. Both of these methods use the entire modelling power of the rational function model, but make assumptions that cause them to be unstable in the presence of noise. This establishes the motivation for moving towards a more reduced parametrization that can provide auto-calibration from point correspondences in real images.

#### 5.4.1 Noise sensitivity of computing $\mathbf{G}$

Synthetic tests of the recovery of  $\mathbf{G}$  were carried out by distorting two images of a 3D model using an  $\mathbf{A}$  obtained from the calibration of an actual fisheye lens. The two images and a view of the original model are shown in Figure 5.12. The model contains 959 points which are imaged in each view, and make up a set of image point correspondences. Gaussian noise was added to these points at known amplitude levels to study the stability of the algorithms. The RMS value of the Sampson distances from the points to their epipolar conics was used as the error measure; this is the same error used in the nonlinear optimizations, as described in §4.3.2. The results are shown in Figure 5.13.

Ground truth was established by removing the distortion using the same  $\mathbf{A}_{\text{calibration}}$  used to introduce it, and then computing the pinhole fundamental matrix  $\mathbf{F}$  for the pair of views. The lifted fundamental matrix is reconstructed as

$$\mathbf{G}_{\text{ground}} = \mathbf{A}_{\text{calibration}}^{\top} \mathbf{F} \mathbf{A}_{\text{calibration}}. \quad (5.5)$$

This is the same procedure as the initialization of the reduced parametrization optimization described in §4.3.4, except that the initial  $\mathbf{A}$  is actually the correct one in this case. The ground truth establishes a lower bound (plotted as the dotted black line in Figure 5.13) for the amount of Sampson error that is introduced by the noise on the image points alone; any error above this level can be attributed to the calibration procedure. One other aspect of the results plot should be mentioned before we move onto

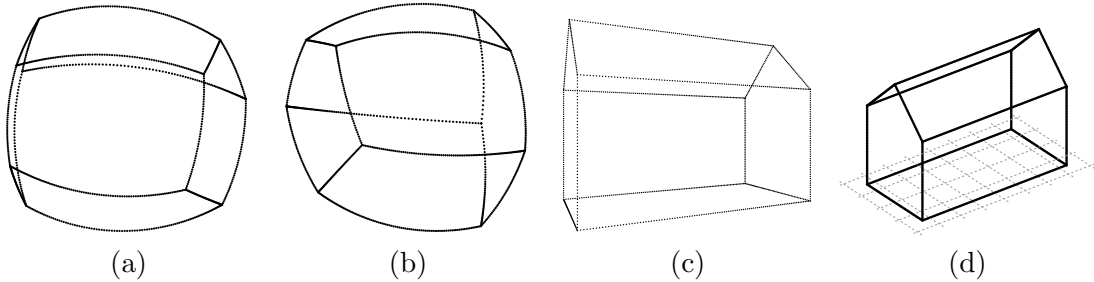


Figure 5.12: Rectification can be performed from two views and no additional information. (a),(b) two synthetic images of a house model (c) rectified structure computed from two views (d) original 3D model.

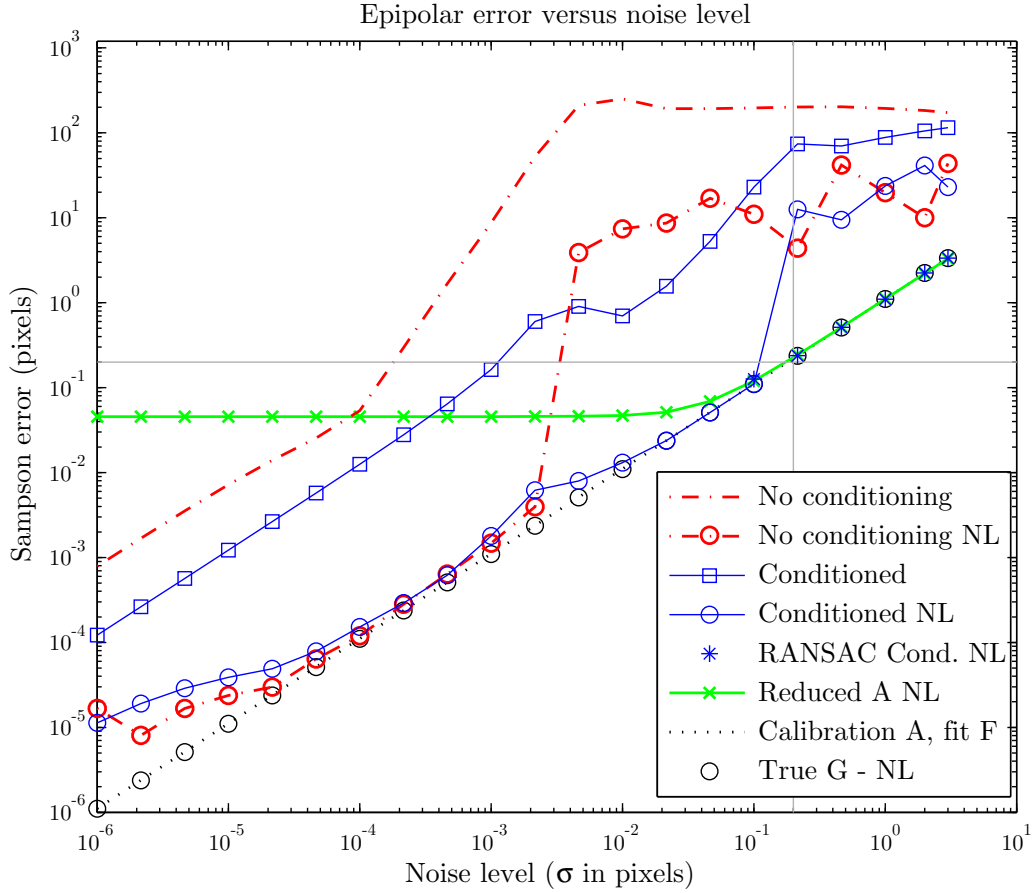


Figure 5.13: Sampson error for computing the fundamental matrix  $\mathbf{G}$  from synthetic data with various noise levels applied. Ground truth (black dotted line) was established using the calibration  $\mathbf{A}$  and then fitting  $\mathbf{F}$  to pinhole data. Data conditioning and nonlinear optimization (NL) reach the base error level for low noise. With a nonlinear RANSAC strategy (refer to text for details) the global Sampson minimum is found even for practical noise levels. Fitting  $\mathbf{G}$  parameterized by the reduced model for  $\mathbf{A}$  demonstrates modelling error at low noise, but is stable at higher levels. It also provides an explicit definition of  $\mathbf{A}$  within the fundamental matrix.

discussing the various calibration methods. Vertical and horizontal lines are included at 0.2 pixels. These mark a threshold of either realistic image noise or acceptable error, respectively. The actual values will vary from one application to the next, but this level provides a reference point for discussion.

The first calibration method listed in the legend of Figure 5.13 is the linear method without conditioning. This approach is not recommended, as the error crosses the 0.2 pixel threshold at a noise level barely above  $10^{-4}$  pixels. Note also that the error axis has a logarithmic scale, so the flat portion at the top of this curve represents a Sampson error of over 100 pixels. Clearly this is not an effective strategy for removing



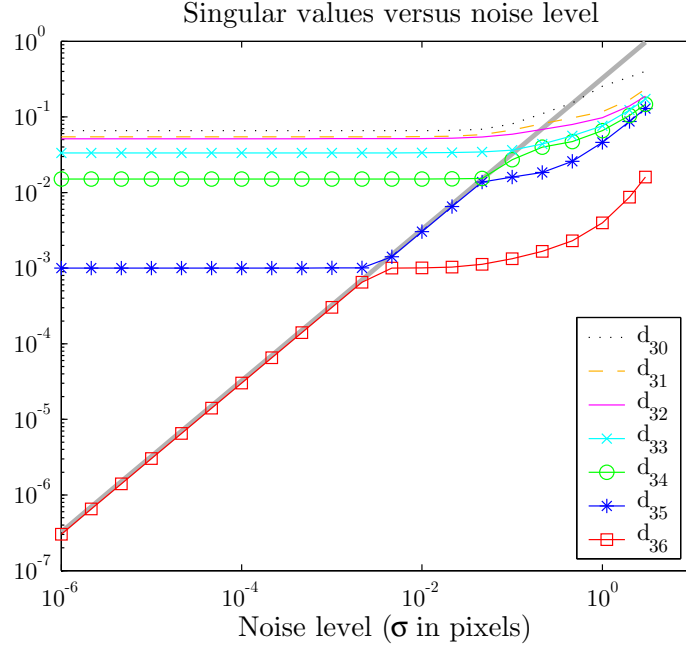


Figure 5.14: Singular values obtained for various noise levels. If the solution is always taken to be the unit singular vector corresponding to the smallest singular value then an incorrect choice is made after the transition at noise level  $10^{-3}$ . The correct singular value is the one which lies along the diagonal line.

lens distortion.

Conditioning the linear algorithm helps somewhat<sup>1</sup>, but the method still fails at the unacceptably low noise level of  $10^{-3}$  pixels. The reason for this failure is the rank truncation of the matrix; without truncation the errors closely follow the ground truth. However, we require that the fundamental matrix be rank two, so an alternative means of meeting this constraint is needed.

The nonlinear optimization of §4.3.2 parameterizes  $\mathbf{G}$  as a rank two matrix. The first test of this optimization aims to verify that there is a stable minimum in the optimization error space that corresponds to our goal. The algorithm was initialized with  $\mathbf{G}_{\text{ground}}$  and the results of the optimization are plotted as black circles in Figure 5.13. This curve exactly matches the ground truth, indicating that this is a stable minimum in the solution space.

When this optimization is initialized with the results of the linear algorithm the same minimum is reached for low noise levels. The slight increase over the ground truth error at very low noise levels is not significant as a noise level of  $10^{-5}$  pixels is not practically achievable. Just above  $10^{-3}$  pixels the result for the algorithm without

<sup>1</sup>Discussions of normalization for this type of linear algorithm (based on the Direct Linear Transform) can be found in Hartley (1997), Chojnacki and Brooks (2003) and Hartley and Zisserman (2003).

conditioning undergoes a dramatic increase. It is at this point that the image noise becomes significant: as with any nonlinear optimization, this one requires a reasonable initialization. The linear algorithm used for this initialization constructs  $\mathbf{G}$  from one of the singular vectors of the design matrix  $\mathbf{D}$  (refer to 2.4.3). The singular vector corresponding to the smallest singular value is chosen to provide the null vector. The smallest singular value should be much far less than the others and in the absence of image noise, this is the case. However, as shown in Figure 5.14, this singular value  $d_{36}$  increases linearly with noise. At approximately  $10^{-3}$  pixels of noise we observe that  $d_{36}$  intersects the path of  $d_{35}$ . At this point the noise level is the order of the solution data. Beyond this level the smallest singular value is no longer the correct one; the correct one to use continues to increase in proportion to the noise level. Although one might expect that a situation such as this would use a linear combination of the eigenvectors corresponding to the two smallest eigenvalues, in this case it was found that the correct answer is obtained by using the correct eigenvector on its own. This surprising result suggests that there is some structure to the problem which means that there should be a singular value at  $10^{-3}$ .

The slope of the line in Figure 5.14 and the transition noise levels aren't known ahead of time, so an exhaustive test procedure was employed to determine the correct singular value. Each singular vector was used, in turn, to generate  $\mathbf{G}_{\text{proposed}}$  and the Sampson error for that fundamental matrix and all image points was measured. The singular vector that produces the lowest error was then chosen. Note that Figure 5.14 is for conditioned data with the linear algorithm; without conditioning there is insufficient spread in the singular values for the noise to have a significant affect on their relative magnitude. This is reflected in Figure 5.13, where we see that the conditioned nonlinear result continues to follow the ground truth beyond  $10^{-3}$  pixels of noise (at least for several more data points) while the non-conditioned linear results degenerate.

The conditioned nonlinear algorithm (with singular value testing) still fails at a noise level of 0.2 pixels. Fortunately we still have one more trick up our sleeve. RANSAC is normally used to reject mismatched image point correspondences within the context of multiple view geometry. We can use it here to reject as outliers those points which have had large amounts of noise added. By throwing out the noisiest point correspondences we can remove the data which is causing the algorithm to converge on the wrong minimum.

The RANSAC strategy used here selects a random set of 40 point correspondences to use in computing the linear initialization for  $\mathbf{G}$ . This is done using conditioning and exhaustive singular value search. The initial  $\mathbf{G}$  is passed to a nonlinear optimization



that uses *all* of the point correspondences (not just the subset chosen for this RANSAC iteration) for refinement. This process was repeated for 500 iterations at every noise level. The results were evaluated on the Sampson error for the nonlinear  $\mathbf{G}$ ; the best iteration at each of the higher levels of noise is plotted as a blue asterisk in Figure 5.13. Selecting a subset of the points has produced linear initializations that fall within the basin of convergence of the nonlinear optimization. The error level has been reduced to that of the ground truth, but at significant computational cost. Each sample test involves a full nonlinear optimization, and 500 samples were tested for this analysis. This is not a practical approach for most applications.

We now present results of the nonlinear optimization parameterized by the reduced RF model, an approach that yields comparable results with much less computational effort. The reduced parametrization of §3.2.5 defines  $\mathbf{A}$  as three ellipses which describe the lens distortion. Once the distortion is removed the two views are related by a pinhole  $\mathbf{F}$  so  $\mathbf{G} = \mathbf{A}^\top \mathbf{F} \mathbf{A}$ . Using this parametrization in the nonlinear optimization (the algorithm is described in detail in §4.3.4) provides stable convergence at all measured levels of image noise (green curve in Figure 5.13). At very low noise levels this algorithm has a Sampson error that is considerably higher than the level for the other algorithms. The flat portion of the curve in Figure 5.13 indicates the level of modeling error associated with the reduced parametrization for  $\mathbf{A}$ . Constraining the model to use fewer parameters (this case used five: aspect ratio  $a$ , field of view  $\phi$ , radius  $b$ , and two ellipse centering terms to model skew and distortion) limits its ability to fit very general lens distortion. In practice this is not an issue because the model error is only significant below a noise level of 0.05 pixels, and even then the Sampson error that it introduces is 0.04 pixels in magnitude. So the error introduced is too small to be significant for most applications, and it is only observed at artificially low noise levels.

The first advantage of the reduced parametrization in multi-view calibration is increased stability in estimation of  $\mathbf{G}$  in the presence of image noise. We now turn our attention to the second advantage: it provides an explicit definition for  $\mathbf{A}$ . The previous algorithms rely on the technique of §4.3.3 to extract  $\mathbf{A}$  from a given  $\mathbf{G}$ . The rectified house in Figure 5.12c demonstrates that this extraction works, however that is a noise-free example. The algorithm is unstable at even low levels of image noise, as shown in Figure 5.15. That plot is really a continuation of the analysis and results for Figure 5.13, for a fundamental matrix  $\mathbf{G}$  (even with low Sampson error) is only as useful as the  $\mathbf{A}$  calibration that can be recovered from it. The fundamental matrix was computed by the most successful methods described above, and then an extracted  $\mathbf{A}$  was used to rectify the left house image (Figure 5.12a). A straight line was fit to

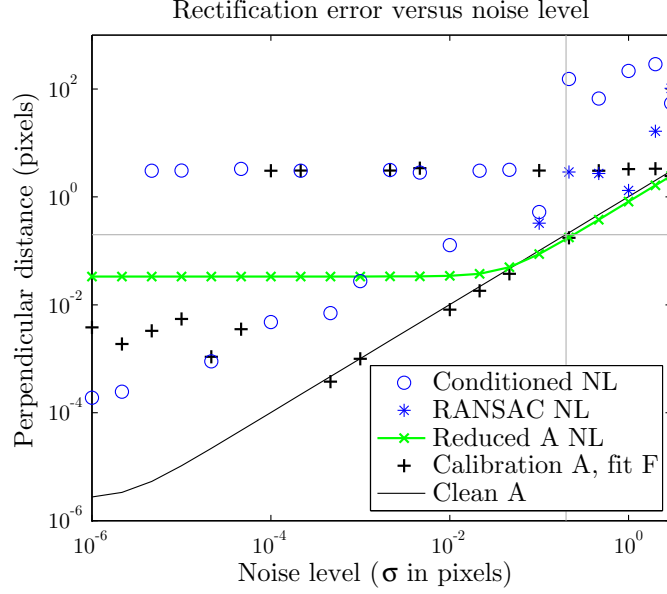


Figure 5.15: Error in removing distortion from synthetic images using  $\mathbf{A}$  recovered from correspondences in two views. An error baseline (clean  $\mathbf{A}$ ) was obtained by rectifying the noisy data with the  $\mathbf{A}$  used to generate the distorted views. The algorithm to extract  $\mathbf{A}$  from  $\mathbf{G}$  (used for the three datasets marked by points only) is highly unstable, whereas parameterizing  $\mathbf{G}$  by the reduced  $\mathbf{A}$  model provides stable recovery at all noise levels.

each line segment and the perpendicular distance from each rectified point to that line was measured. The RMS value over all 959 points is reported as the perpendicular error in Figure 5.15. A lower bound error was plotted by fitting  $\mathbf{A}_{\text{clean}}$  to image points without noise and then using that result to rectify the noisy image points. Although this provides a baseline, it isn't exactly the lowest error as adding noise to the data could alter it such that  $\mathbf{A}_{\text{clean}}$  is no longer optimal. This is likely the cause for the slight offset observed in the results for  $\mathbf{A}_{\text{clean}}$  in Figure 5.15 (solid black line). The calibration  $\mathbf{A}$  and pinhole fundamental matrix data (black crosses in Figure 5.15) was the ground truth for the Sampson error analysis, but here we see that even with such a reliable  $\mathbf{G}$  the decomposition of  $\mathbf{A}$  is not stable. At several noise levels the lower bound was reached, suggesting that further work may improve the performance of this technique. The general fundamental matrix optimization (conditioned and RANSAC) demonstrates similar behaviour, although the lowest observed errors are higher than in the calibrated  $\mathbf{A}$  case.

A more reliable approach is to optimize using the reduced parametrization. This provides  $\mathbf{A}$  without needing to decompose  $\mathbf{G}$ , a technique that is better conditioned because it constrains the optimization in a manner which matches the underlying model. These results are shown as the green curve in Figure 5.15. The flat portion up to a

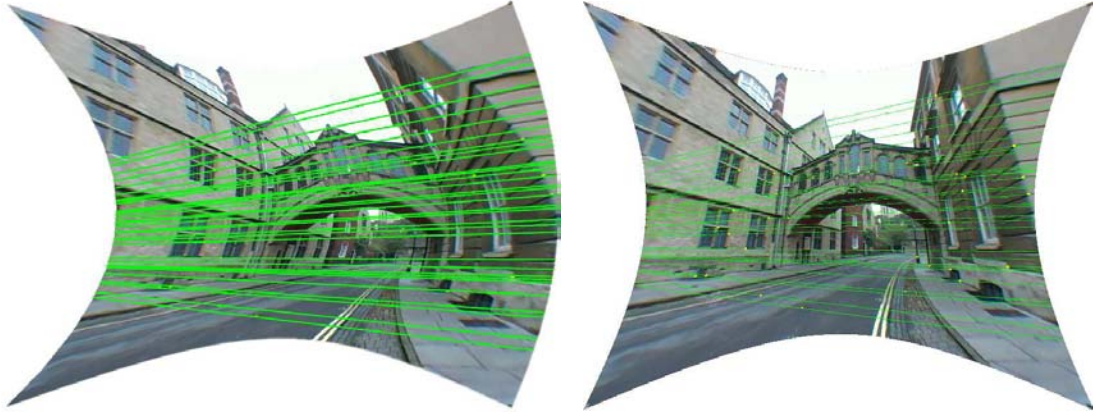


Figure 5.16: Images rectified using  $\mathbf{A}$  matrices recovered from  $\mathbf{G}$  using point correspondences alone. *Left* This reconstruction used the full 18 parameter  $\mathbf{A}$  which is too general for stable fitting from real data. Although the epipolar curves have correctly been straightened, the vertical lines are not straight. *Right* Using the reduced parametrization for  $\mathbf{A}$  results in a correct rectification.

noise level of 0.05 pixels of noise is at 0.04 pixels of error, and represents the error introduced by the simplifications of the reduced model for  $\mathbf{A}$ . As mentioned earlier in the discussion of the Sampson error, this error is only significant at impractically low levels of noise, and even then the error is small enough to be tolerated for most applications. The fact that the rectification error so closely matches the Sampson error reinforces the suitability of Sampson error as a metric for judging the performance of this type of image rectification. Returning to Figure 5.15, we see that the reduced parametrization closely follows the lower bound error for the last seven data points. This range of “reasonable” noise levels (from 0.05 to 3 pixels) corresponds to the practical performance band, and we see that the reduced parametrization provides a stable, low error solution.

This synthetic data analysis provides compelling quantitative reasons for using the reduced parametrization. Figure 5.16 gives a qualitative comparison of the two methods for parameterizing the nonlinear optimization. Although the full model/decompose  $\mathbf{A}$  method has straightened the epipolar curves (Figure 5.16 *left*), there are insufficient constraints to ensure that the vertical lines are straightened. By contrast, the reduced parametrization image (Figure 5.16 *right*) has had all lines rectified, including those which are perpendicular to the epipolar lines. It should be noted that no outliers were included in the sets of point correspondences, and the inclusion of a nonlinear optimization step means that any RANSAC based outlier rejection strategy will be computationally expensive. Likewise, detecting outliers where the required number of inliers is greater than 35 necessitates a very large number of RANSAC iterations.

### 5.4.2 Results on image sequences

We now present the results of multi-view calibration for a number of different video sequences. A digital camcorder equipped with a wide-angle lens was used to capture some real-world footage, and point correspondences were identified between each of two frames. The topic of establishing such wide baseline correspondences is beyond the scope of this work (the reader is referred to *e.g.* (Matas et al. 2002)). For these examples they were established either by: manually matching Harris corners; or else by rectifying the sequence (using an approximate distortion model) so that the pinhole methods of commercial structure-from-motion software (2d3 Ltd. 2003) could detect the correspondences, and then re-applying the distortion.

All attempts to correct the distortion using a general  $\mathbf{G}$  and decomposing  $\mathbf{A}$  failed (see Figure 5.16 *left* for an example that almost worked). By contrast, the reduced parametrization enabled each sequence to be corrected with very little computational time. Figures 5.17 to 5.20 show the calibration results for four different video sequences. All were recorded on a hand-held Canon XM2 digital camcorder fitted with a Raynox 0.3X fisheye converter. The caption for each figure lists the number of frames separating the two views (PAL video is 25 frames per second), the number of point correspondences used, the RMS Sampson error (the residuals of the optimization, and also a good indication of rectification precision), and the time required to perform the optimization. These times are for a straightforward Matlab implementation, and indicate order of magnitude rather than the fastest possible timings. Note in the epipolar curve figures that the curves for the majority of the established point correspondences have been omitted for clarity. The office sequence (Figure 5.17) is predominantly a laterally translating camera, as indicated by the horizontal epipolar lines. The Jenkin sequence (Figure 5.18) has a much shorter baseline (9 frames) and an almost forward translating camera (the epipole is visible in the image). Figure 5.19 represents the type of image that is difficult to calibrate by plumblines methods. There are very few long, straight lines and the vertical wall edge (in the centre of the image) appears to represent a line which perhaps should be straight, but in reality is curved. In addition, the large wall which occupies the right half of the image provides no line data at all, so the plumblines calibration would have to extrapolate to fill that portion of the frame. The large expanse of rubble course wall is actually an asset to multi-view calibration because it provides so many feature correspondences. The large number of correspondences results in this sequence having the lowest Sampson error of those tested. Finally, Figure 5.20 shows a correct rectification of the Bridge of Sighs. Note that all straight lines are now straight; the sides of the street are curved by construction.

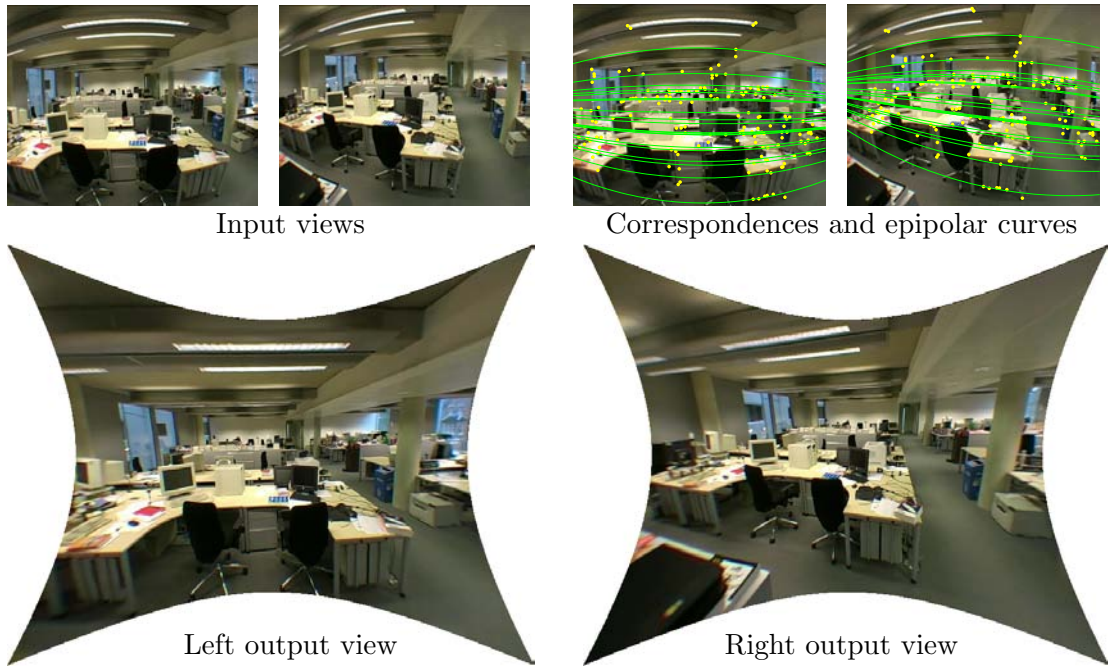


Figure 5.17: Distortion correction and epipolar geometry computed from point correspondences alone for two frames from an office sequence. These two images are 78 frames apart in the video sequence. (200 point correspondences, 0.48 pixels RMS Sampson error, 2.7 seconds)

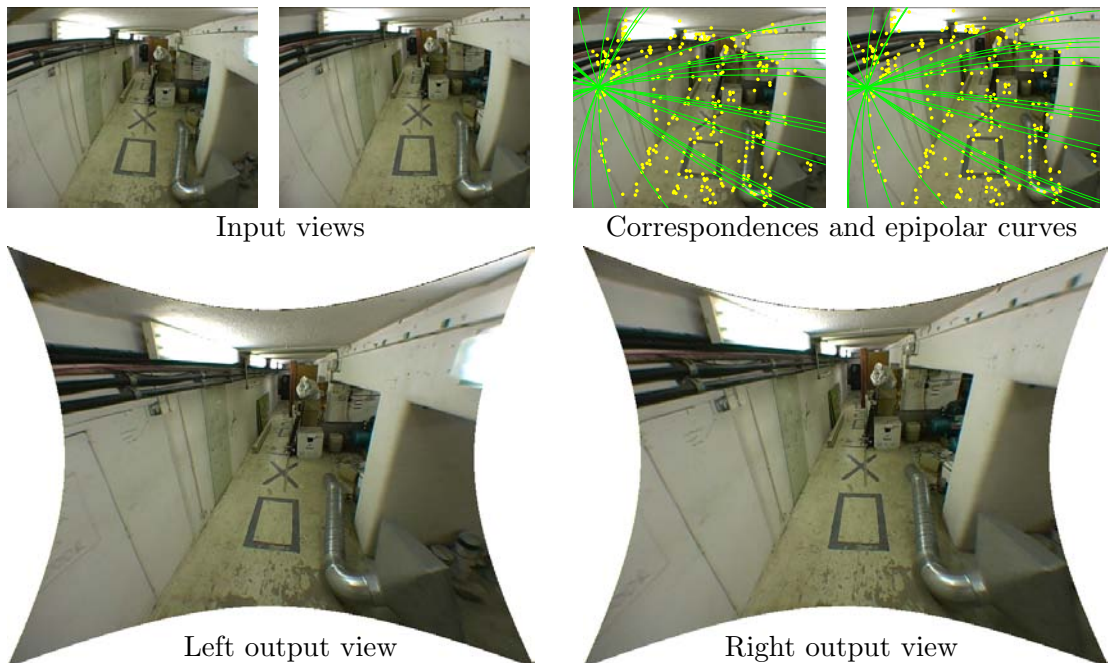


Figure 5.18: Distortion correction and epipolar geometry computed from point correspondences alone for two frames from a sequence of the Jenkin Building basement, University of Oxford. These two images are 9 frames apart in the video sequence. (300 point correspondences, 0.62 pixels RMS Sampson error, 1.6 seconds)



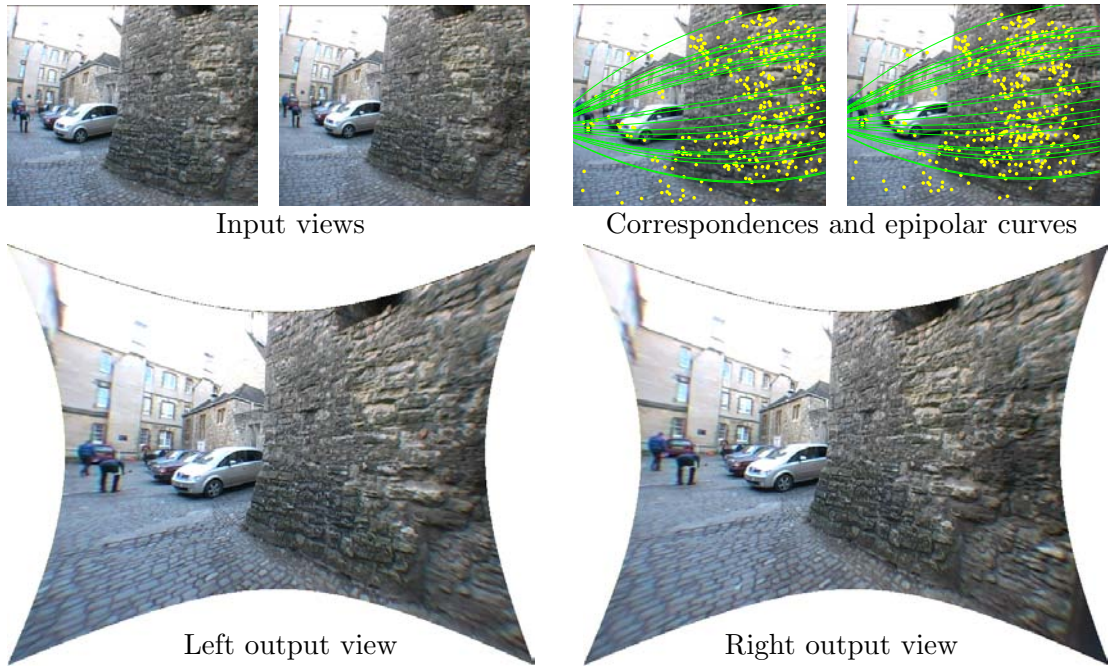


Figure 5.19: Distortion correction and epipolar geometry computed from point correspondences alone for two frames from a sequence of The Slype at New College. These two images are 9 frames apart in the video sequence. (880 point correspondences, 0.20 pixels RMS Sampson error, 7.3 seconds)

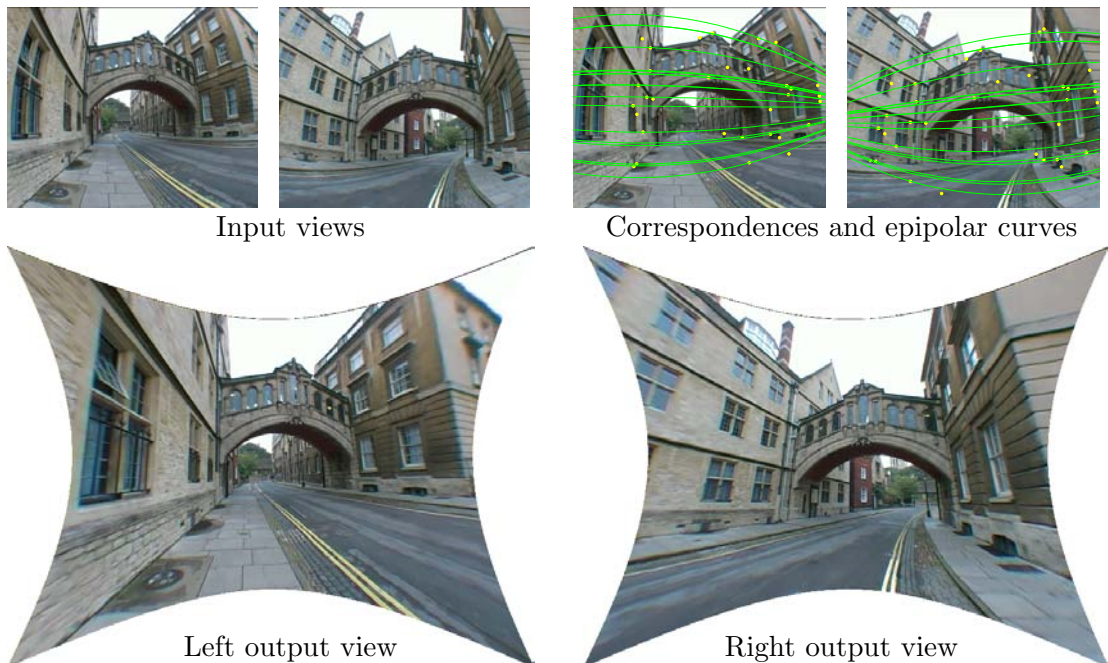
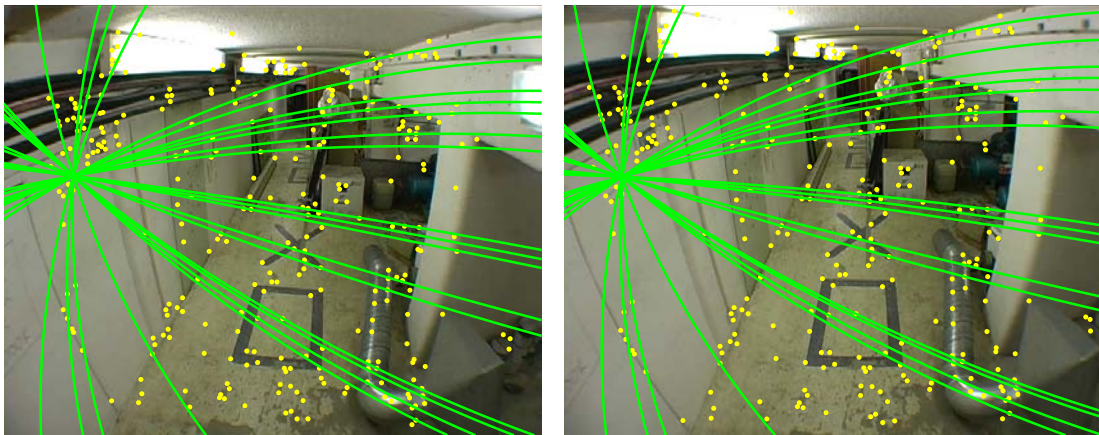


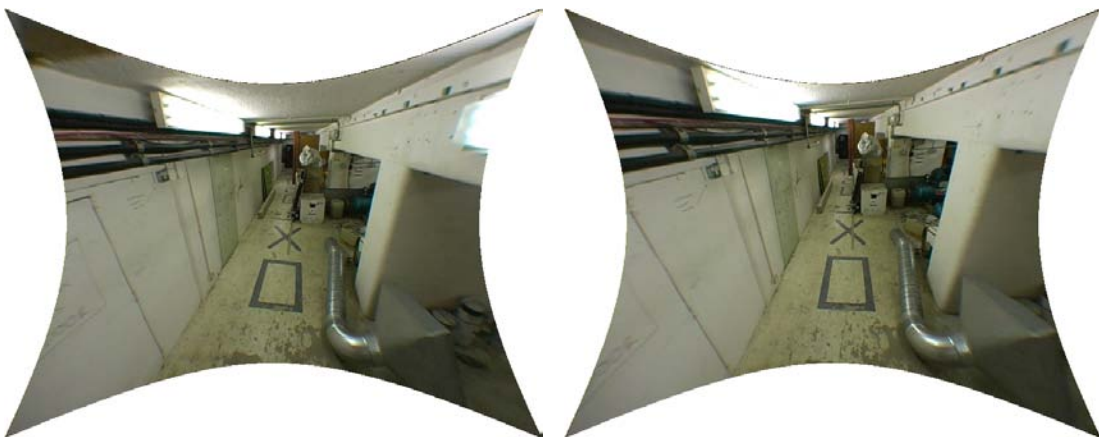
Figure 5.20: Distortion correction and epipolar geometry computed from point correspondences alone for two frames from a sequence of the Bridge of Sighs, Hertford College. These two images are 330 frames apart in the video sequence. (135 point correspondences, 0.67 pixels RMS Sampson error, 4.2 seconds)



Input views from the Jenkin sequence



Point correspondences and epipolar curves (some omitted for clarity)



Output views

Figure 5.21: Distortion correction and epipolar geometry computed from point correspondences alone. A total of 300 point correspondences were used to compute  $A$  and  $G$  for this pair of views using the reduced parametrization.

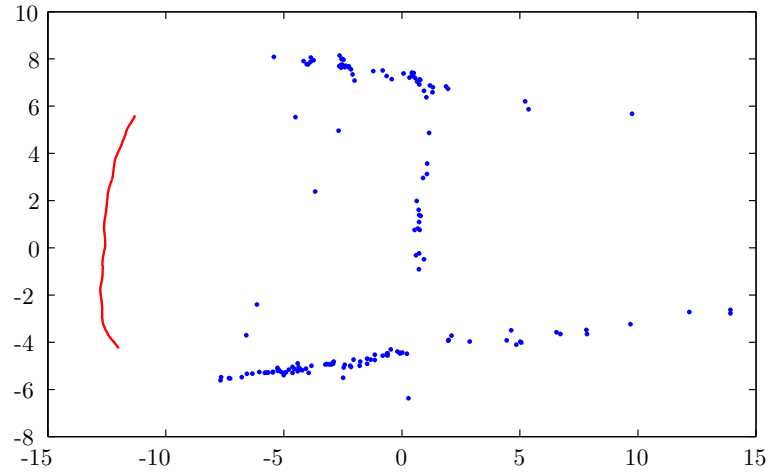


Figure 5.22: Top view of the reconstructed Bridge of Sighs showing the building walls (lines of blue points) and the camera path (in red). Note that the walls are straight, indicating that the rational function model has correctly removed the distortion. This reconstruction is only up to a projective homography; the reconstructed walls converge more than the physical ones. This can easily be rectified while establishing a metric coordinate frame. Although this is a non-metric reconstruction, the approximate scale is metres.

## 5.5 Three-Dimensional Reconstruction

Once the lens distortion has been corrected we are left with pinhole-equivalent images and it is possible to perform a 3D reconstruction using standard structure-from-motion techniques. This section describes such a reconstruction performed on the Bridge of Sighs sequence. First the image were distortion-corrected using an  $A$  from a planar grid calibration. This represents the easiest and most reliable method for calibration; either plumbline or multiview calibration could be used, with similar results. The commercial software package *boujou* (2d3 Ltd. 2003) was then used to: 1) track features through the sequence, 2) establish correspondences 3) reject outliers (incorrect matches between frames) 4) compute camera matrices for all frames, and 5) reconstruct the 3D position of each point via bundle adjustment. The resulting point positions and camera path are plotted in Figure 5.22. This top view shows the walls on either side of the street, and the front face of the bridge which joins them. The straight walls indicate that the distortion has effectively been removed from the input images; residual distortion tends to curve the reconstruction space. The walls do converge more than they should, this is a consequence of the solution being only up to a projective homography. Note that *boujou* returns a Euclidean reconstruction under the assumption that there is zero skew in the camera matrix, but with the general distortion model there is no guarantee that the skew will in fact be zero. As a result, the coordinate axes in the reconstruction space



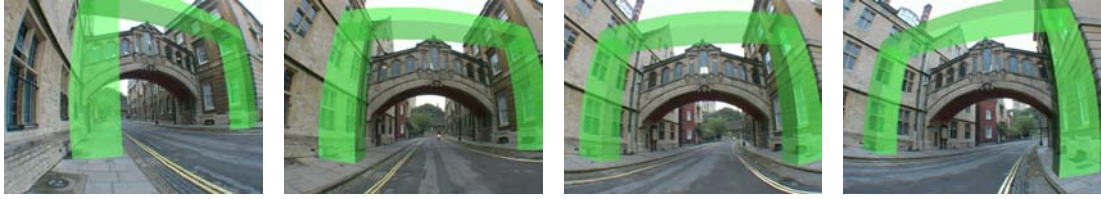


Figure 5.23: A virtual arch superimposed on the Bridge of Sighs sequence. The original sequence was distortion corrected, tracked, and reconstructed so that the arch could be defined relative to the moving scene. The inverse distortion model was then applied to warp the arch according to the distortion present in the original images.

are not necessarily orthogonal, but a 4D projective homography will correct this. Such a homography is required anyway to map the solution space onto a metric coordinate frame.

A further test of a 3D reconstruction is to insert a virtual object into the original sequence. This tests both the quality of the camera localization (poor localization results in object jitter between frames) and the effectiveness of the distortion modelling. The distortion model is used to remove the distortion (and thus influences the reconstruction, as described above) and the inverse distortion model is required to distort the virtual object for insertion into the original view. Figure 5.23 shows four frames from the Bridge of Sighs sequence with a virtual arch superimposed. The full sequence can be downloaded from

[http://www.robots.ox.ac.uk/~dclaus/thesis/rf\\_model.avi](http://www.robots.ox.ac.uk/~dclaus/thesis/rf_model.avi).

## 5.6 Summary

Here we have seen that the rational function model can be fit linearly in a single step, without the need to first estimate the aspect ratio and distortion centre; the residual errors are on the same order as the best lens-specific model available. The proposed rational function model is capable of accurately fitting existing distortion models: synthetic tests indicate that the proposed method is on par with existing models for approximating polynomial distortion.

Furthermore, both the existing and the proposed models were tested against real image data. Even when calibrated using nonlinear optimization including all model parameters plus optical centre and aspect ratio it is clear that not all models represent the imaging process with comparable accuracy. In our tests the general purpose rational function method performed as well as the lens-specific field of view model, yet the proposed method can be calibrated linearly for the planar calibration grid used in this example.

For the plumblin method of calibrating the rational function model we demonstrated that calibration for a given lens is consistent from one image sequence to the next (does not depend on input data), and is robust to changes in initialization values and image noise.

Comparison with MATLAB Camera Calibration Toolbox revealed that its model does not represent the distortion near the image periphery as accurately as the rational function model does. The RF model is also calibrated from a single image rather than the 14 used by the Toolbox.

Evaluating the various rational function models and calibration techniques indicates that the plumblin method is slightly more precise than the planar grid method. This is to be expected because the latter uses much more calibration input data. The reduced parametrization incurs a slight loss of accuracy over the full rational function model, but not enough to discourage its use for any applications.

The multiple view method for calibrating the rational function model demonstrates that it is possible to perform auto-calibration of fisheye camera parameters from point correspondences in real images. Nonlinear optimization using a reduced parametrization achieves stable convergence and provides an explicit definition of the model parameters.

Finally, image rectification results and a scene reconstruction from distorted input images demonstrate that these techniques are applicable to real world scenarios.

## Chapter 6

# Camera Localization

The previous three chapters discussed distortion and concluded that the Rational Function model does an excellent job of removing lens distortion from many video sequences. This holds true for a wide variety of lenses, and even for archive footage where neither knowledge of the lens nor calibration images are available. Thus it is possible to produce images that follow the pinhole lens model, and this chapter shall address the remaining components in camera calibration: computing the projection homography  $P$  and then decomposing it into the internal camera parameters and external pose.

The first step is to establish correspondences between world points and locations in an image. It is easy to define the world coordinates of features on a planar calibration grid (relative positions at least), however measuring the 3D position of general out of plane features proves to be a difficult and error-prone task. We advocate an optical technique for surveying the positions of points in a calibration environment. Markers placed in arbitrary locations are filmed with a video camera; the resulting images alone are sufficient to accurately determine the 3D locations of each marker in the environment. This is based on bundle adjusted structure from motion, and can also provide camera intrinsics for the camera used to perform the survey.

An environment that contains a set of easily recognizable features in known 3D positions can be used to calibrate both camera intrinsics and extrinsics. If this can be done quickly and reliably for every frame of video it can provide a basis for augmented reality; once the camera pose and projection are known it is a simple matter to insert scene registered virtual objects. We show that by computing the analytic derivatives of the reprojection function it is possible to perform nonlinear pose and focal length optimization in realtime. The resulting individual frame camera path achieves accuracy that is comparable to the path obtained through offline bundle adjustment over the entire sequence.

This chapter concludes with an evaluation of pose estimation methods for aug-

mented reality, but first we examine some of the components required to construct such an environment: homography calculation, optical surveying for point initialization, computing camera intrinsics and extrinsics, and pose estimation through fast nonlinear optimization.

## 6.1 World Points and Corresponding Image Locations

Many computer vision tasks require that geometric features in an image be reliably located. Sometimes the form of these geometric entities is important (such as finding straight lines for plumbline calibration methods), but often it is only required that the features be suitable for identification and localization. Identification requires that the selected features be distinctive and easily recognized. Precise localization ensures that the image coordinates returned for the point always refer to the same physical location on the feature. This can be a challenging requirement particularly under oblique viewing angles and in the presence of motion blur.

Interest operators aim to identify naturally occurring points that satisfy the above two criteria. Many interest operators have been proposed, including those of Moravec (1979), Harris and Stephens (1988), Deriche and Giraudon (1993), Smith and Brady (1997) and Kadir et al. (2004). It is possible to divide these into two broad categories: feature *detectors* and feature *descriptors*. We will classify as a detector any single low level image operation that makes no attempt to establish uniqueness (*e.g.* edge or corner detectors). Descriptors become well suited to establishing unique image correspondences by combining detectors, and also achieve some level of invariance to changes in viewpoint, scale or illumination (Tuytelaars and Van Gool 1999). One approach is to use moment invariants to describe features in an affine invariant way (Van Gool et al. 1996). One such descriptor is Scale Invariant Feature Transform (SIFT) keypoints (Lowe 1999) developed for image feature generation in object recognition applications. They are fully invariant to image translation, scaling and rotation, and partially invariant to changes in illumination or viewpoint. However, such invariance comes at a price: there is a tradeoff between invariance and localization precision. It is necessary to relax the registration criteria in order to ensure that a feature is identified under a wide variety of conditions.

Rather than relying on interest operators to identify naturally occurring landmarks, *fiducial detection* places markers of known geometry within the scene and employs a detector specifically designed to recognize them. A *fiducial* is a point, line etc. that is assumed as a fixed basis of comparison (Simpson and Weiner 1989). Fiducial detection is an important problem in real-world vision systems: the task of identifying the position

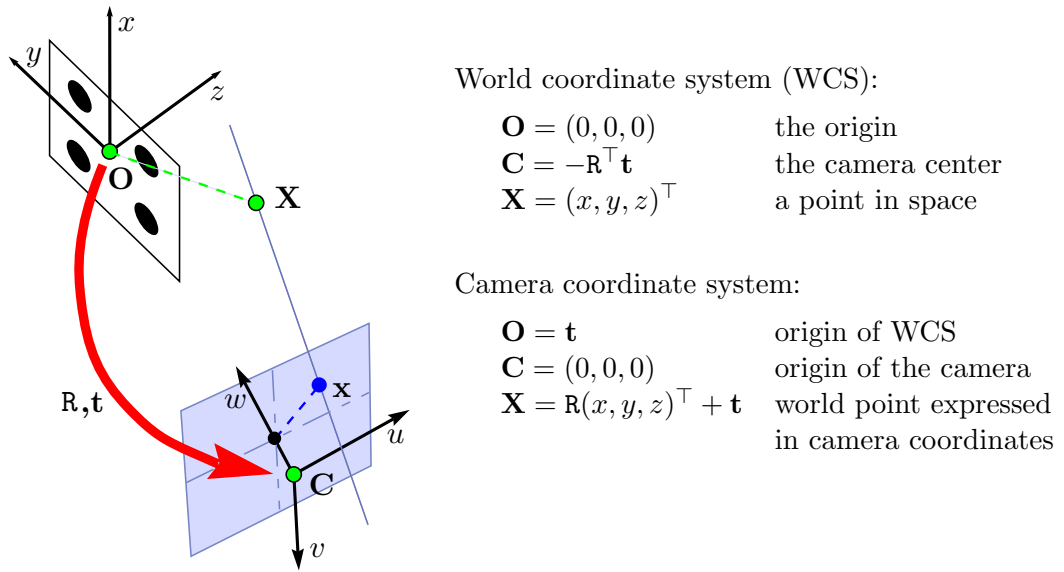


Figure 6.1: Transformations and relations between world and camera coordinate systems. The rotation matrix  $\mathbf{R}$  and vector  $\mathbf{t}$  are the extrinsic camera parameters which must be computed during camera calibration.

of a pre-defined target within a scene is central to augmented reality and many image registration tasks. Once again, we will defer the discussion of detecting and localizing fiducials to Chapter 7. This chapter assumes that, given an image with some markers present, we have an algorithm that can return the precise image coordinates and identity of each fiducial mark.

## 6.2 Initialization

There exist many methods for determining the internal calibration parameters of a camera given a set of image/world correspondences. Several of these were described in §2.2.1. The approach taken in this section is to compute an approximate initialization for both the internal and external camera parameters and then perform a nonlinear optimization to refine them both simultaneously. This is often implicitly included in a bundle adjustment stage. Simultaneous optimization is simpler and more accurate due to the high degree of dependency between the various camera parameters.

This section describes methods for computing the camera parameters from sets of image/world point correspondences where the points are either coplanar or in an arbitrary configuration. It is assumed that point data consisting of a set of 3D world points and their corresponding image locations is available. Refer to §6.4 for one method for obtaining such calibration input data.

The camera pose and calibration for a given image is represented as a  $3 \times 4$  projection matrix

$$P = K [R \mid \mathbf{t}] \quad (6.1)$$

where  $K$  represents the internal calibration parameters of the camera,  $R$  is a  $3 \times 3$  rotation matrix, and  $\mathbf{t}$  is the translation of the camera.

### 6.2.1 Camera Intrinsics

A camera's internal calibration  $K$  typically encompasses five parameters: focal length, aspect ratio, skew and principal point (as laid out in (2.3)). It is often useful to be able to supply initial values of  $K$  to a nonlinear optimization or even a full bundle adjustment. Fortunately, camera manufacture is a very precise art, and so these values can often be reliably estimated from several "rules of thumb".

The focal length for a standard lens (roughly corresponding to a 50 mm focal length lens on a 35 mm SLR system) will be on the order of 1000 pixels/m. A simple method for actually computing the focal length using the vanishing points of two parallel lines is given in (Simon et al. 2000). Additional computational methods for the focal length are described in (Hartley and Zisserman 2003). From §2.1.1 we see that  $a = 1.067$  for PAL video, otherwise set  $a = 1$ . The skew is generally zero, and approximating the principal point as the centre of the image is a reasonable starting point.

### 6.2.2 Extrinsics from coplanar data

An algorithm for computing the camera extrinsics from a set of coplanar world points was presented by Simon et al. (2000). Combined with an estimate of the intrinsic parameters, this can be used to initialize a nonlinear optimization for full camera calibration.

The transformation between image coordinates and planar points can be expressed as a homography. Given  $K$  (the internal parameters), this method recovers the rotation and translation between the camera coordinate system and the target plane from this homography. Details of the technique are provided in Appendix A.

This procedure provides an approximate calibration from planar data, however it is not very precise. Calibration from planar data is highly susceptible to image noise and feature localization error as these directly translate into camera parameter error. Also, truncating the rotation matrix columns to unit length and to be orthogonal introduces error. Table 6.1 gives the camera position accuracy and solution time for this method; a full explanation of the experimental procedure is provided in the next section.

Initialization method	Computation time (ms)	Camera centre RMS error (cm)
Planar target	3.1	69.6
DLT	3.5	3.2
POSIT	2.0	1.4

Table 6.1: Comparison of camera pose initialization methods. The planar target algorithm uses only four coplanar dots, while the other two were given twelve dot positions from widely spaced targets. The higher error for the planar case is partly due to this discrepancy in input data. The DLT method is the only one which also computes  $K$ ; otherwise the calibration matrix must be supplied to the algorithm. POSIT is marginally faster and more precise, and was generally used for initialization throughout the tests for this thesis.

### 6.2.3 Extrinsic from general point data

For more than five point correspondences which do not all lie on a single plane there are more general methods for recovering the camera calibration parameters. The Direct Linear Transform method is often used (refer to §2.4.3 and Appendix A for details). Another method for computing camera parameters from a set of world/image point correspondences is the POSIT algorithm (Dementhon and Davis 1995). It computes an initial affine camera and then upgrades it to full perspective through several iterative steps. This technique has a lower computational overhead, and is therefore preferable for many realtime applications. A downside is that this algorithm does not include the estimation of the camera intrinsics; these must be supplied.

Table 6.1 lists the convergence times and error for both of these methods, and for the planar calibration described in §6.2.2. The test sequence consists of 180 frames viewing three calibration targets with a mean distance to the camera of 3.36 m. Ground truth was established through full bundle adjustment by *boujou* (2d3 Ltd. 2003), which also furnished the intrinsic parameters required by the planar algorithm and by POSIT; the DLT method computes  $K$  for each frame. The error measure is the Euclidean distance from the ground truth camera centre to the centre computed for each frame by each method. This encompasses any error arising from both the translation  $\mathbf{t}$  and rotation  $\mathbf{R}$ . Table 6.1 gives the RMS value over the whole sequence. The computation time is the average for a single frame, all algorithms implemented in Matlab.

For most of the work listed in this thesis, POSIT was used to provide the initial estimate of camera pose with camera intrinsics estimated as per §6.2.1. When only coplanar calibration point data was available the planar target method was used. Although this method is less precise, the subsequent nonlinear optimization is still able to converge on the correct solution.

### 6.3 Optimization

The camera initialization methods described in the previous section provide an approximate estimate of a camera's pose and internal parameters, but additional accuracy can be obtained through nonlinear optimization. This section describes the optimization, details the use of analytic derivatives to speed convergence, and shows that this introduces very little computational overhead.

The pose optimization problem is to find  $\mathbf{K}$ ,  $\mathbf{R}$  and  $\mathbf{t}$  to minimize the reprojection error

$$\epsilon_2(\mathbf{K}, \mathbf{R}, \mathbf{t}) = \sum_j \|\mathbf{x}_j - \pi(\mathbf{K}(\mathbf{R}\mathbf{X}_j + \mathbf{t}))\| \quad (6.2)$$

for each frame  $j$ .

Estimates for the optimization variables are obtained by the methods of §6.2. The 2D marker positions  $\mathbf{x}_j$  and corresponding 3D locations  $\mathbf{X}_j$  are known, and constant throughout the optimization. The entire camera calibration matrix is not usually varied for every frame. The aspect ratio, skew and principal point will be constant for a given camera, so need not be estimated separately for every frame. Often the same is true for the focal length. In this case one can construct an optimization where  $\mathbf{K}$  is common over the whole sequence while  $\mathbf{R}$  and  $\mathbf{t}$  are fit to each frame. This approach is unsuitable for online operations (it requires the entire sequence at the outset), or for zooming lenses (changing focal length). In these cases  $\mathbf{K}$  is either fixed, or reduced to a variable focal length for optimization at each frame. A comparison between these approaches is given in §6.3.3; in describing the optimization we will assume  $\mathbf{K}$  is constant, for simplicity and maximum generality in describing the methods since constant parameters require special attention.

Thus the variable set is reduced to the camera extrinsics,  $\mathbf{R}$  and  $\mathbf{t}$  for each frame. The rotation matrix  $\mathbf{R}$  is an over-parametrization of a 3D rotation, and should be represented in a more concise format prior to optimizing. The work presented here uses a four element quaternion, but a three element Rodrigues vector would also work with a little extra attention to singularities. Refer to Appendix B for additional discussion of rotation parameterizations. The parameter vector for the optimization is therefore

$$[q_1 \ q_2 \ q_3 \ q_4 \ t_x \ t_y \ t_z]^\top. \quad (6.3)$$

These seven variables require at least four known image points (at two dimensions per point this gives eight equations for seven unknowns) to determine a solution. It is possible to perform the optimization on image data for a single planar four dot target, although greater precision is attained through the use of multiple targets.



### 6.3.1 Analytic Derivatives for Reprojection Error

The ability to compute the derivative of the cost function is central to nonlinear optimization (§2.3). Most implementations use finite differencing to approximate the derivative, but it is often possible to supply an analytic function for computing the exact derivative. This can speed convergence dramatically.

The Jacobian (§2.3.1) is the first derivative of a function, and is also used in the Levenberg-Marquardt algorithm to approximate the second derivative (§2.3.1). The  $i^{\text{th}}$  column of the finite difference Jacobian of a function  $F$  about point  $\mathbf{P}$  is given by

$$\mathbf{J}(:, i) = \frac{F(\mathbf{P} + \boldsymbol{\delta}_i) - F(\mathbf{P} - \boldsymbol{\delta}_i)}{2\epsilon} \quad (6.4)$$

where  $\boldsymbol{\delta}_i = \epsilon[0 \dots 1_i \dots 0]^T$  is a delta function with a one in the  $i^{\text{th}}$  row, multiplied by some small epsilon (typically  $\epsilon = 1 \times 10^{-4}$ ). Notice that this approximation can be generically implemented because it only requires the ability to evaluate the input function  $F$ , however, it assumes a linear function in the vicinity of the evaluation point. To correct this, we can evaluate the actual Jacobian rather than the finite difference approximation.

For a given point correspondence  $\mathbf{x} \leftrightarrow \mathbf{X}$  the reprojection error  $\mathbf{e}$  is

$$\mathbf{e} = \mathbf{x} - \pi(\mathbf{K}(\mathbf{R}\mathbf{X} + \mathbf{t})) \quad (6.5)$$

$$\begin{pmatrix} e_1 \\ e_2 \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} - \pi \left( \begin{bmatrix} f & s & u_0 \\ 0 & af & v_0 \\ 0 & 0 & 1 \end{bmatrix} \left\{ R(q_1, q_2, q_3, q_4) \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \right\} \right). \quad (6.6)$$

The Jacobian is made up of the derivatives with respect to each variable:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial e_1}{\partial q_1} & \frac{\partial e_1}{\partial q_2} & \frac{\partial e_1}{\partial q_3} & \frac{\partial e_1}{\partial q_4} & \frac{\partial e_1}{\partial t_x} & \frac{\partial e_1}{\partial t_y} & \frac{\partial e_1}{\partial t_z} \\ \frac{\partial e_2}{\partial q_1} & \frac{\partial e_2}{\partial q_2} & \frac{\partial e_2}{\partial q_3} & \frac{\partial e_2}{\partial q_4} & \frac{\partial e_2}{\partial t_x} & \frac{\partial e_2}{\partial t_y} & \frac{\partial e_2}{\partial t_z} \end{bmatrix} \quad (6.7)$$

Each of the  $n$  point correspondences gives rise to a  $2 \times 7$  section of the  $2n \times 7$  element Jacobian produced for a single frame optimization. If camera parameters are to be optimized over all frames then the overall dimensions of the Jacobian get quite large and sparse matrix techniques should be used. Otherwise, constructing the Jacobian proceeds in much the same manner, with some additional indexing considerations.

### 6.3.2 Timing

Providing an analytic derivative to the optimization routine speeds convergence to the point where it is practical to include this step in realtime camera tracking operations.

Pose algorithm	Time (ms/frame)
Initialization only:	0.4
Nonlinear, finite differencing:	11.1
Nonlinear, analytic derivative:	3.2

Table 6.2: Time to compute camera extrinsics (pose) per frame of video. Using an analytic derivative decreases the required computation time considerably. The resulting nonlinear optimization is fast enough to be included in realtime pose computation algorithms.

Table 6.2 gives timing results for computing camera pose with and without analytic derivatives and optimization. These results are average time per frame for the 182 frame sequence described in §6.5.2; timing values for the other sequences described in this chapter are comparable. The algorithms were implemented in C++ and ran on a 1GHz laptop. The nonlinear optimization with analytic derivatives requires less than three milliseconds (excluding the initialization time of 0.4 ms). For video streaming at roughly 30 frames per second this represents less than 3% of the available computing time for each frame. A realtime demonstration of this nonlinear pose optimization (together with the fiducial detection described in Chapter 7) was presented at the Workshop on Applications in Computer Vision in January 2005 (Claus and Fitzgibbon 2005c).

When provided with correct image/world correspondences and reasonable internal camera parameters this algorithm exhibits very stable convergence. Of the over two thousand frames computed for the tests in this chapter, plus the thousands of frames of the realtime demonstration, all converged to the correct pose. The only instances of failed convergence were when the correspondences were incorrectly established (usually an indexing problem) or when the incorrect camera calibration was supplied (*e.g.* using a wide angle calibration for a standard lens).

### 6.3.3 Zoom lenses

If the camera is changing focal length during tracking, this is also readily included in the optimization for pose, although with a reduction in position accuracy. To check the accuracy of this computation, we ran pose estimation including variable focal length on the fixed-focal-length ground-truth sequence of §6.5.2. The results are listed in Table 6.3. Comparison values were obtained from the MATLAB Camera Calibration Toolbox (Bouguet 2003), and *boujou*'s fully bundle-adjusted solution. The pose optimization results include the focal length obtained by optimizing over the entire sequence simultaneously (similar to bundle adjustment, but without including the world marker positions), and the mean of the focal lengths individually optimized for each frame. If

Method	$f$
MATLAB Calibration Toolbox	846.6
<i>boujou</i> calibration	848.7
Pose optimization - entire sequence	849.1
Pose optimization - individual frames	849.7

Table 6.3: Comparison of computed focal lengths, expressed in units of image pixels. The results agree to within the tolerances required for most applications; the individual frame optimization had  $\mu = 849.7$  and  $\sigma = 10.14$  pixels over 182 frames. Optimizing for the focal length at each individual frame is sufficiently accurate that this method can be used for realtime pose computation with variable focal length lenses.

the focal length is known to be fixed it can be obtained by optimizing over the entire sequence simultaneously; for realtime operations or variable focal length it is possible to estimate the focal length from individual frames.

## 6.4 Surveying Fiducial Positions Optically

All of the camera localization methods described in this chapter require images of markers for which accurate 3D coordinates are known. This section examines how one goes about obtaining that measurement data.

The key to accurate camera localization is accurate surveying of the markers in the environment; the resulting calibration will only be as precise as the input data allows. The markers should be well spaced throughout the workspace volume, which makes accurate measurement difficult. An environment with an abundance of flat surfaces helps somewhat, but ensuring that all coordinate measurements are at right angles is not a simple task (many walls, ceilings and floors are not precisely perpendicular to each other, particularly in older buildings). One solution is to construct a calibration object with markers at known positions. This is straightforward for planar calibration (targets can be produced on a laser printer and then mounted on any flat surface), but it is expensive, awkward and time consuming to produce the size of 3D target object required for calibrating wide field of view cameras.

Instead, we automate this difficult measurement task by using *boujou*, an off-the-shelf structure and motion system (2d3 Ltd. 2003). Such systems are frequently used for extremely demanding augmented reality tasks in cinema post-production, where virtual objects must be added to real-world footage with jitter of the order of half a pixel in a  $4096 \times 3312$  image, and drift over hundreds of frames of the order of a few pixels. However, because they depend on batch computation and bundle adjustment (Hartley and Zisserman 2003), they are unsuitable for online operation, although ideal for offline calibration and surveying.

To survey the environment, a video of the calibration workspace is captured, ensuring that each marker of interest is visible from at least two widely-spaced viewpoints. The fiducials are detected and their locations determined in each frame of the video sequence. These positions are fed to the structure and motion solver, which returns a single  $\mathbf{K}$  for the entire sequence, and per-frame rotation and translation estimates. We now turn our attention to the details of the structure from motion solution process. This section first describes the theory used by the SfM software package and then gives some practical details of the actual surveying procedure.

### 6.4.1 Structure from Motion

The field of computer vision that deals with the recovery of scene structure from camera motion (Structure from Motion, or SfM) has advanced greatly over the past decade. There are now commercial software packages available that include fast, robust implementations of the algorithms and user friendly interfaces. This is a tremendous boon to the researcher, as the package can be relied upon to perform feature detection and matching, bundle adjustment and scene reconstruction (triangulation). This frees one up to focus on other aspects of a project. In the context of camera localization, we use the software to survey the 3D locations of the fiducial markers.

In order to determine the position of a particular disc, we gather its 2D positions in every frame in which it was detected, as a list of  $(x, y, f)$  tuples, where  $(x, y)$  is the marker's position in frame  $f$ . The camera intrinsics are defined by a fixed aspect ratio (typically 1.0667), zero skew, fixed principal point (the centre of the image by default) and a variable focal length. Then the 3D point we require is that which minimizes the reprojection error (Hartley and Zisserman 2003) over all frames in the sequence

$$\epsilon(X) = \sum_{i=1}^{\text{num\_detections}} \left\| \begin{pmatrix} x_i \\ y_i \end{pmatrix} - \pi(\mathbf{K}(\mathbf{R}_{f_i}\mathbf{X} + \mathbf{t}_{f_i})) \right\|. \quad (6.8)$$

The variable parameters in this optimization are of two groups. The first group includes the focal length (although this can be optimized on a per-frame basis as well) and the point positions  $\mathbf{X}$ . These variables are common to the entire image sequence. The second group contains parameters that are unique to each frame: the rotation matrix  $\mathbf{R}_{f_i}$  and translation  $\mathbf{t}_{f_i}$ . Although this minimization has no closed-form solution, it is readily solved by initialization using one of the methods from §6.2 followed by nonlinear optimization of  $\epsilon(\mathbf{X})$ . This process is referred to as bundle adjustment (§2.3.2), which differs from the optimization of §6.3 by including the world points  $\mathbf{X}$  and simultaneously considering all frames. The §6.3 algorithm considers a single frame at a time (so it can

be used for online computation) and requires that the world marker positions be known. Here bundle adjustment is being used to *determine* those world positions.

### 6.4.2 Surveying procedure

We survey the fiducial positions by filming them, and computing their locations from the video. When recording the video it is important to have multiple targets in view so that their relative positions can be computed. The first step in processing the video is to detect the fiducials in each frame. The fiducial detection scheme (see Chapter 7) locates each fiducial with subpixel precision and identifies each marker so that correspondences between frames are already established. It is possible to use the feature detection and matching functions of the SfM package, but we require the image locations of the fiducial markers and have on hand a reliable method for finding them. Sometimes it is helpful to augment the fiducial feature set with detected features (particularly when few fiducial markers are used or visible), but this is not required. The next step is to remove any lens distortion. Depending on the lens used, this can be done using the built-in functions of the SfM software, but it is often necessary to use more sophisticated models as described in the previous chapters. The fiducial positions are then ready to be saved in a format that can be loaded into the SfM package.

If the SfM solver is fed precise and labelled image points it is a quick and straightforward optimization to compute the scene geometry (3D location of each fiducial marker) and camera parameters. However the solution is in an arbitrary space, so we must impose scale and coordinate system constraints to obtain measurements in a Euclidean space. The scale is set by assigning the (known) distance between the dots in each target. Likewise, any three dots in a single target form a right angle; provided all of the targets are not coplanar then any two are sufficient to establish a rectangular coordinate frame. If there is a target aligned with each of the three coordinate planes this simplifies the establishment of a reference frame considerably. Once the coordinate system has been defined we can export out the 3D fiducial positions and internal camera calibration parameters (if required). Note that in surveying the fiducials, we independently compute the 3D position of each of the four discs on each target. This allows for deviations from planarity in paper targets.

Table 6.4 gives the results of one such optical environment survey. The results are compared with target positions manually measured with a tape measure and plumb line. The calibration environment is pictured in Figure 6.13; note that making the measurements involved climbing on the desk while measuring around the shelving and electrical channels. This highlights some of the difficulties encountered when making

	Target 2			Target 3		
	$x$	$y$	$z$	$x$	$y$	$z$
Measured	1475	-99	704	321	-917	747
Computed	1474	-106	714	304	-923	752
Difference	1	7	-10	17	6	-5

Table 6.4: Comparison of surveyed target position coordinates obtained by manual measurement and computed by the optical method described in the text. Target 1 was used as the origin and to define the coordinate frame; all dimensions are given in mm. The minor differences are well within the 20 mm confidence level associated with manually recording these arbitrary 3D positions.

this type of 3D coordinate survey, and is one of the primary motivations for the optical technique described in this section.

## 6.5 Evaluation

This section presents the results of optimized camera localization with distortion corrected by the full rational function model. The main question to answer is “How precise can camera localization be and what are the important components in reaching that result?” We present results on raw image data, distortion corrected data, and results that include a skew parameter in the camera model.

Two separate experiments were performed to measure the accuracy of the camera localization methods described in this chapter. The principal difference between the two is the type of ground truth to which the localization results are compared: the first uses absolute coordinates from a physical measurement system, while the second compares to the Augmented Reality Toolkit (Kato and Billinghurst 1999). The ARToolkit is an existing optical camera localization technique that is readily available. Both of these methods are in turn compared to a fully bundle adjusted solution.

### 6.5.1 Coordinate Measurement System Ground Truth

The main challenge in evaluating a camera pose algorithm is establishing the ground truth coordinates for comparison. One approach is to constrain the camera motion to a particular path and then measure the perpendicular deviation. This can be done in a straight line (Wei and Ma 1994), or a circle by using a turntable (Mičušík 2004), but it severely limits the range of camera motion to be tested. We measure the absolute camera motion using the coordinate measurement system on a machine shop milling machine, and also by standard structure from motion techniques (as implemented in *boujou*) to provide the type of ground truth relevant for augmented reality applications.



Figure 6.2: Apparatus for camera pose ground truth from a milling machine coordinate measurement system. The camera was clamped in the mill’s vice, and then its pose was computed from the targets and compared with the positions displayed on the digital display.

The coordinate measurement system (CMS) on a milling machine was used to provide accurate and absolute ground truth camera motion. This type of CMS is used by machinists to indicate the motion of the milling machine table relative to the mill’s cutting tool. The Accurite III model used in this experiment provides digital  $x$  and  $y$  measurements to the nearest 10 microns. Camera motion was restricted to the  $z$  plane by fixing the  $z$ -axis of the mill’s table. The  $x$  and  $y$  axes on this particular mill are both fitted with constant velocity drives. These drives were used to provide a smooth path from which baseline data could be interpolated; the  $z$ -axis lacks such a drive so it was locked in place. The camera was securely clamped in the tabletop vice, so the camera viewing direction was also fixed. Thus the camera pose degrees of freedom were restricted to  $x$  and  $y$  translations. The experimental setup is shown in Figure 6.2.

**Procedure** The experimental procedure consisted of translating the camera while simultaneously viewing a set of fiducials and the output of the coordinate measurement system. This section describes the environment calibration, data recording, and post processing performed to evaluate the pose algorithms relative to the milling machine measurements.

**Initialization** The first task was to estimate the radial distortion for both the standard and fisheye lenses. An image of concentric planar rectangles was recorded with each lens, and the plumbline method used to calibrate the full rational function model as described in §4.2.

The environment for fiducial-based camera pose estimation was calibrated by surveying the target positions as described in §6.4. Four planar targets were placed at arbitrary locations chosen to be within the field of view for the camera while it was

Target dot positions (mm)							Camera intrinsics	
	$x$	$y$	$z$		$x$	$y$	$z$	Focal length (px)
Target 1	-53	-53	0	Target 3	726	46	-668	$f_{normal} = 801.5$
	-53	53	0		824	44	-698	$f_{fisheye} = 236.6$
	53	53	0		822	-60	-686	
	53	-53	0		725	-60	-660	Principal point (px)
Target 2	729	829	-1314	Target 4	90	256	1179	$p(x, y) = (360, 288)$
	731	934	-1314		161	251	1102	
	813	934	-1249		159	146	1107	Aspect ratio
	812	828	-1248		87	150	1183	$a = 1.0667$

Table 6.5: Initialization values obtained from milling machine calibration sequence. The camera intrinsics and target dot positions were used as inputs for the camera pose computation.

clamped to the milling machine. A calibration sequence was then recorded with the camera held freehand. All fiducial locations were then detected, and bundle adjustment was used to estimate the camera's intrinsic parameters, camera path (for the calibration sequence) and target positions. These target positions were computed in a coordinate frame centred on one of the targets. Later this coordinate system will be transformed to align it with the coordinate system of the milling machine measurement system, but for now this procedure provides the relative world positions of each fiducial dot and the camera intrinsics. This data, required as inputs to the pose algorithm, is summarized in Table 6.5.

**Capture** After initialization, the camera was clamped into the vice on the milling machine. The coordinate measurement system measures the motion of the mill's table; this rigid clamping ensures that the camera and mill table move together. The constant feed mechanism on the mill table was then used to translate the camera along three linear paths while video footage was recorded. This procedure was repeated for both a normal and a fisheye lens.

**Ground truth for each frame** As shown in the sample frames of Figure 6.4, the coordinate measurement system's digital display is visible in the captured video. The ground truth camera motion for each sequence was determined by manually recording the displayed  $x$  and  $y$  coordinates from approximately twenty frames distributed throughout the sequence. A function ( $y = f(x)$ ) was fit to this sparse data, and interpolated values computed for every frame. A second order polynomial function was used: the residual errors on a linear fit were found to be quadratic, implying that the underlying path was quadratic. The maximum residual was 0.38 mm, while the mean



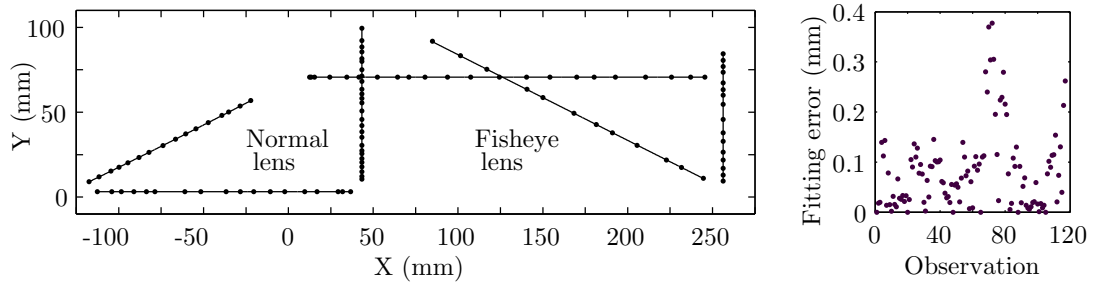


Figure 6.3: Ground truth data for the milling machine sequences. *Left* Camera paths for three sequences with each lens. The dots indicate data points recorded from the video and used to interpolate the position at each individual frame. *Right* The residual error for each observed point after fitting a smooth (quadratic) path.

was 0.08 mm. The ground truth paths and data points used for interpolation are shown in Figure 6.3.

**Fiducial detection and distortion correction** The fiducial detection and target identification code described in Chapter 7 was used to precisely locate the target dots in each frame. The position of each dot was adjusted to correct for lens distortion using the rational function model parameters computed in the initialization step. Note that it is not necessary to perform an expensive image warping to remove this distortion: the dots are detected in the original image (containing distortion) and then the dot coordinates alone need be corrected. Figure 6.4 shows an example frame from both lenses with the detected dot tracks overlaid.

Lens distortion causes a warping of the reconstructed coordinates space. Figure 6.5

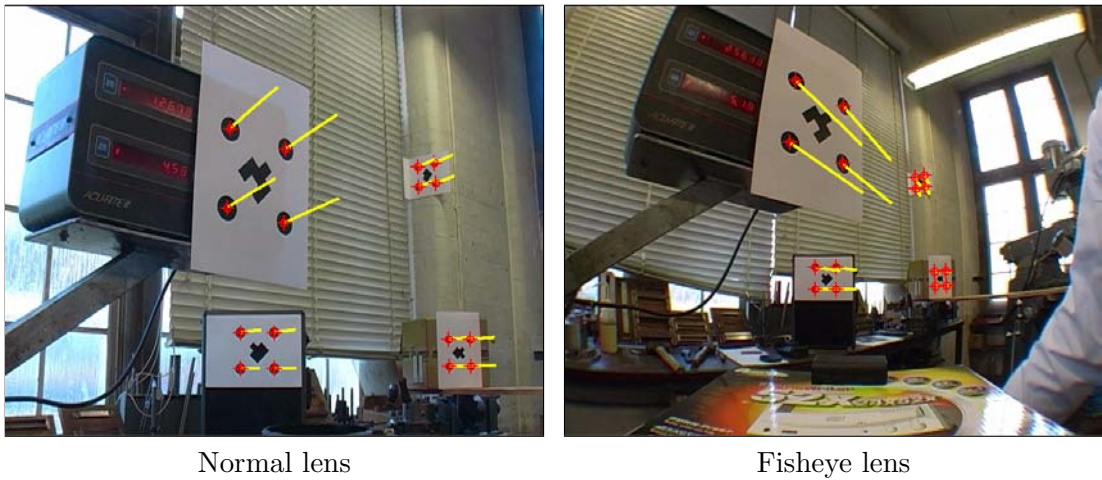


Figure 6.4: Sample frames from sequences using each lens. The yellow lines track the dot positions through the entire sequence. The dot positions were then corrected for lens distortion prior to computing the camera pose at each frame.

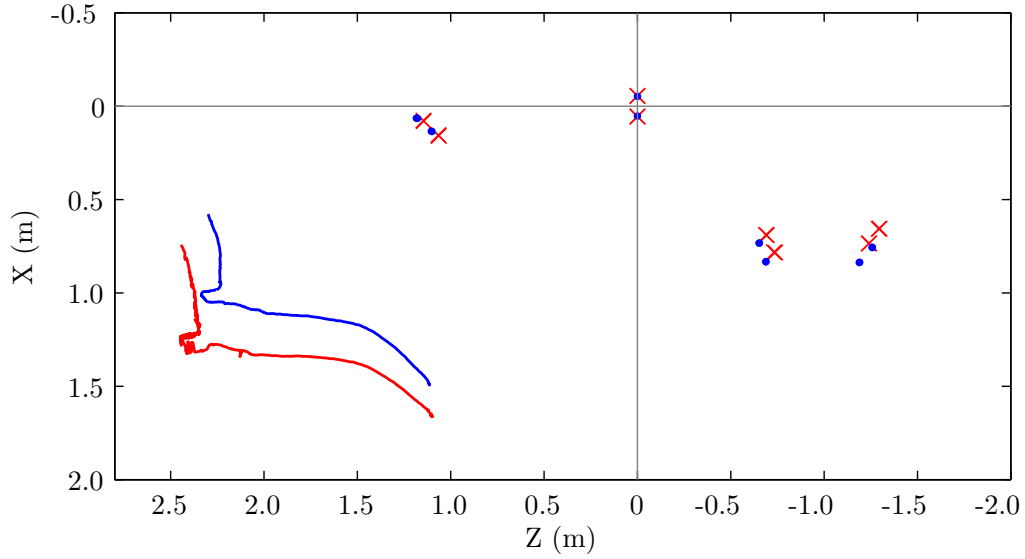


Figure 6.5: Target geometry and camera path reconstructed with lens distortion correction (blue) and with no correction (red). Failing to provide the reconstruction algorithm with input image data that has been corrected to follow the pinhole assumption results in an unstable camera track, and a warping of the solution coordinate space.

shows a plan view of the reconstructed target dot locations and camera positions for the freehand initialization sequence. The two coordinate systems were aligned based on the target centred at the origin. The sequence without distortion correction (shown in red) is less stable, and the target dot positions do not line up correctly. The jagged portions of the red camera path are due to oscillations between two possible camera paths. This sort of instability is often observed in sequences where lens distortion has not been adequately compensated for. Although it is possible to apply a smoothness prior to the camera track while performing the bundle adjustment, this does not correct the overall warping due to the lens distortion.

**Camera Pose** The camera pose was computed by the methods described in this chapter, and also by *boujou*. The algorithms used the same input dot positions, but *boujou* performs a full bundle adjustment over camera pose for every frame plus the world target dot positions. By comparing the results we see how well a single frame pose algorithm can compare with bundle adjustment (global optimization over the whole sequence).

**Coordinate System Transformations** Before the various camera pose solutions can be compared it is necessary to express them all in a single coordinate system. The milling machine coordinate frame was selected as the reference frame because it provides

the ground truth, and so that the measurements would be in units of millimeters. The rotation and translation between the mill coordinates and the targets attached to the walls was established by aligning the *boujou* track with the milling machine ground truth. This is a similarity transformation and was computed using the method of Arun et al. (1987). The rational function model includes a skew term in the solution, so a coordinate system transformation including skew was also tested. Finally, the distortion correction introduces an unknown projective homography into the final solution, so a full three dimensional projective homography was used to align the coordinate systems from any distortion corrected track. These alignment homographies were computed via nonlinear optimization initialized with just a rotation and translation. The ground truth track is planar, so the target dot positions were included in each optimization to prevent the  $z$ -dimension from being compressed into the plane.

**Results** The results of computing the camera locations for the milling machine ground truth experiment are summarized in Tables 6.6 and 6.7. The error was measured as the Euclidean distance between the computed camera centre and the measured position provided by the milling machine coordinate measurement system. This is measured after performing each type of coordinate alignment described above. For each lens and distortion correction we report the RMS, standard deviation and maximum error over all frames. The covariance  $\Sigma$  of the camera position estimate can be calculated from the Jacobian used in the nonlinear optimization

$$\Sigma = [\mathbf{J}^\top \mathbf{J}]^{-1}. \quad (6.9)$$

A transformation was applied to the solution space to bring it into a metric coordinate grid aligned with the milling machine. If the transformation applied to the optimization parameters  $\Theta$  is given by  $\Theta' = \mathbf{F}(\Theta)$ , then the covariance is transformed according to

$$\Sigma' = \nabla \mathbf{F} \Sigma \nabla \mathbf{F}^\top. \quad (6.10)$$

The  $\nabla \mathbf{F}$  term is the Jacobian of  $\mathbf{F}(\Theta)$  and was computed using finite differences. The covariance values reported in Table 6.8 are the diagonal elements corresponding to the three dimensions of the camera position.

Each sequence was tested with both distortion corrected by the rational function model and by processing the raw image coordinates. Each lens exhibited a significant error reduction when the distortion correction was included. The camera position error for the normal lens is less than 1 cm, even without correcting the distortion. However, even with such a low distortion lens there is a substantial improvement to

Distortion	Camera	Transformation	RMS	STD	MAX
Raw	Focal	Similarity	2.85	1.32	5.49
Raw	Full	Similarity	1.41	0.52	2.55
Corrected	Focal	Similarity	0.21	0.09	0.54
Corrected	Full	Similarity	0.22	0.09	0.57
Raw	Focal	R,t,s	0.38	0.16	0.82
Raw	Full	R,t,s	0.44	0.19	0.93
Corrected	Focal	R,t,s	0.21	0.08	0.57
Corrected	Full	R,t,s	0.21	0.08	0.57
Raw	Focal	Projective	0.15	0.05	0.37
Raw	Full	Projective	0.15	0.05	0.36
Corrected	Focal	Projective	0.16	0.07	0.48
Corrected	Full	Projective	0.15	0.07	0.47

Table 6.6: Camera localization results for a normal lens. For each set of distortion correction, camera model and coordinate alignment transformation we report the error (measured in mm) between the computed camera position and the ground truth for that frame. The tabulated statistics are for the entire 1447 frame sequence. Fiducial based camera localization with pose optimization can provide sub-millimeter precision. Refer to the text for additional discussion.

be realized by distortion modelling: the results here demonstrate that sub-millimeter position estimation is possible with a commercial video camera. Furthermore, the distortion correction models the fisheye lens so that the pose error for that sequence is less than a quarter of a millimeter.

For both lenses, the full camera model made little difference except on data where no distortion correction had been used, and only a similarity transformation was applied for coordinate alignment. In this case the extra camera parameters are modelling some of the nonlinearity due to the lens distortion.

For the normal lens, correcting the distortion yields a tenfold improvement for the focal length camera model. For both lenses, including the distortion correction removes the difference between the two camera models. When the rotation, translation and skew (R,t,s) transformation was used the focal-only camera model outperforms the full model. The full model is expected to always produce an equal or lower error level because the focal-only model is a reduced case of the more general full model. The discrepancy observed here is explained by the way in which the camera model was fit to the data. A subset of frames from the diagonal portion of each camera lens sequence was used to compute the camera model. Using fewer frames is more efficient (faster to compute) and also provides an indication of the generalization ability of the model. This is what we are observing here; the full camera model is over-fitting to the subset of frames, so its performance is worse on the complete sequence(s). The focal-only

Distortion	Camera	Transformation	RMS	STD	MAX
Raw	Focal	Similarity	6.10	2.39	10.82
Raw	Full	Similarity	5.79	2.11	10.69
Corrected	Focal	Similarity	1.68	0.85	3.76
Corrected	Full	Similarity	1.65	0.81	3.58
Raw	Focal	R,t,s	0.73	0.27	1.48
Raw	Full	R,t,s	0.91	0.32	1.75
Corrected	Focal	R,t,s	0.38	0.18	1.5
Corrected	Full	R,t,s	0.44	0.21	1.20
Raw	Focal	Projective	0.35	0.12	0.75
Raw	Full	Projective	0.35	0.12	0.77
Corrected	Focal	Projective	0.22	0.08	0.55
Corrected	Full	Projective	0.23	0.08	0.54

Table 6.7: Camera localization results for a fisheye lens. The camera position errors are measured in millimeters relative to the milling machine ground truth, and reported over the entire 1839 frame sequence. Even for a lens with such a high level of distortion it is possible to achieve camera position estimates that are accurate to within a quarter of a millimeter. Refer to the text for additional discussion.

model, with its single parameter, is less susceptible to over-fitting, and it follows that the complete sequence error is lower.

The progression of coordinate system transformations from similarity, to similarity plus skew, to full projective homography in 3D decreases the position error at each instance. This is to be expected, as the transformations become more powerful. The purpose of this transformation is to align the coordinate system in which the camera position has been solved with the ground truth coordinate system. If both were Euclid-

Lens Type	Distortion	Camera	X	Y	Z
Normal	Raw	Focal	6.92	0.56	0.72
	Raw	Full	7.25	0.57	0.74
	Corrected	Focal	7.81	0.61	0.78
	Corrected	Full	7.81	0.61	0.78
Fisheye	Raw	Focal	10.65	1.89	3.57
	Raw	Full	11.57	1.92	3.62
	Corrected	Focal	33.20	5.69	10.75
	Corrected	Full	32.85	5.67	10.84

Table 6.8: Covariance for the camera positions computed for both types of lens. The largest element is along the camera axis; this coincides approximately with the  $x$  direction for the milling machine coordinate system. The increase in covariance for the fisheye lens is a result of the targets occupying a much smaller portion of the viewing area. Correcting the distortion brings the targets into an even smaller region. Dimensions are in  $\text{mm}^2$ .

ean systems with orthogonal axes this would only require a similarity. However, the fiducial target surveying method using *boujou* was never constrained to have orthogonal axes so it may include significant 3D skew. Only the scale in each direction was set as part of the calibration procedure. In practice 3 targets should be set up at right angles to one another to aid in assigning an orthogonal coordinate system. The effect of this skew will be minimal for a normal, low distortion lens such as the one used for Table 6.6. The distortion corrected results are almost identical with and without the skew term added to the similarity transform. When the projective transformation is used the error drops from 0.21 mm to 0.15 mm, and we notice that the solution without distortion correction has a lower standard deviation. This is likely due to over-fitting to the specific camera path used in this experiment.

The fisheye sequence was recorded at roughly the same distance from the targets as the normal lens sequence. Because the targets were kept in the same locations, the image area occupied by the targets is much less for the fisheye lens sequence. This means that there is much less observed parallax in the wide angle shots, which in turn increases the numerical instability of the camera pose recovery. The principal advantage of a wide angle lens is that it can image a wider field of view. In this test we have restricted the utilized field of view (by not relocating the targets further apart) while introducing the lens distortion; this incorporates all the drawbacks without taking advantage of the enhancements. This causes the increase in covariance observed for the wide angle lens localization in Table 6.8. In spite of this, the results demonstrate that accurate pose estimation is possible for a wide angle lens.

The covariance was reported for only the projective coordinate system alignment. The other alignment methods are not significantly different in terms of the global coordinate system scaling, and so will have very similar effects on the computed covariance. Note that the standard deviation of the position error (reported in Tables 6.9 and 6.10) is always much lower than the covariance.

Tables 6.9 and 6.10 give the camera localization results from *boujou*'s fully bundle adjusted solution. These results agree with those obtained by optimizing for the camera parameters from the fiducial targets alone. This indicates that once the initial target survey is completed it is possible to achieve single-frame localization accuracy comparable to entire sequence bundle adjusted results. Single frame optimization can be done in realtime on streaming video, while bundle adjustment cannot.

The individual frame errors are plotted in Figures 6.6 to 6.9. These plots indicate the distribution of the errors both temporally and in each of the coordinate axes. In each figure the results for the proposed algorithm are shown in red, while the *boujou*

Distortion	Transformation	RMS	STD	MAX
Raw	Similarity	6.40	2.93	11.31
Corrected	Similarity	0.21	0.09	0.61
Raw	R,t,s	0.55	0.22	1.2
Corrected	R,t,s	0.21	0.09	0.61
Raw	Projective	0.17	0.07	0.71
Corrected	Projective	0.16	0.07	0.52

Table 6.9: Camera localization results for *boujou* with a low-distortion lens. Compare these fully bundle-adjusted errors with the camera-only optimization results in Table 6.6. All dimensions are in millimeters.

results are given in blue. The proposed algorithm includes the full camera matrix and a projective homography for the coordinate system alignment. These figures illustrate the errors on each frame of the sequence, and show the differences in solved camera path for the various methods.

Figure 6.6 shows the vertical error for the normal lens with and without distortion correction. The milling machine was constrained so that the camera could only move in the  $z = 0$  plane, so any deviation from this plane is an error in camera position. A front and right view are supplied for each set of data (refer to Figure 6.3 for a top view showing the camera paths for each lens). The results from the two methods are more correlated in the distortion corrected case, and the few outliers in the *boujou* solution without distortion correction have been fixed in the corrected case. Of interest is the overall accuracy of the camera localization methods: with twelve well localized image point correspondences it is possible to achieve millimeter level camera path data with a commercial video camera.

Figure 6.7 shows the error relative to milling machine ground truth along each of the coordinate axes. Again, the red curve gives the optimization localization results while the blue denotes the *boujou* solution. By comparing the top and bottom sets

Distortion	Transformation	RMS	STD	MAX
Raw	Similarity	7.16	3.15	14.99
Corrected	Similarity	2.77	1.55	6.61
Raw	R,t,s	0.46	0.19	1.54
Corrected	R,t,s	0.41	0.20	1.13
Raw	Projective	0.26	0.14	1.51
Corrected	Projective	0.20	0.08	0.49

Table 6.10: Camera localization results for *boujou* with a fisheye lens. Compare these fully bundle-adjusted errors with the camera-only optimization results in Table 6.7. All dimensions are in millimeters.

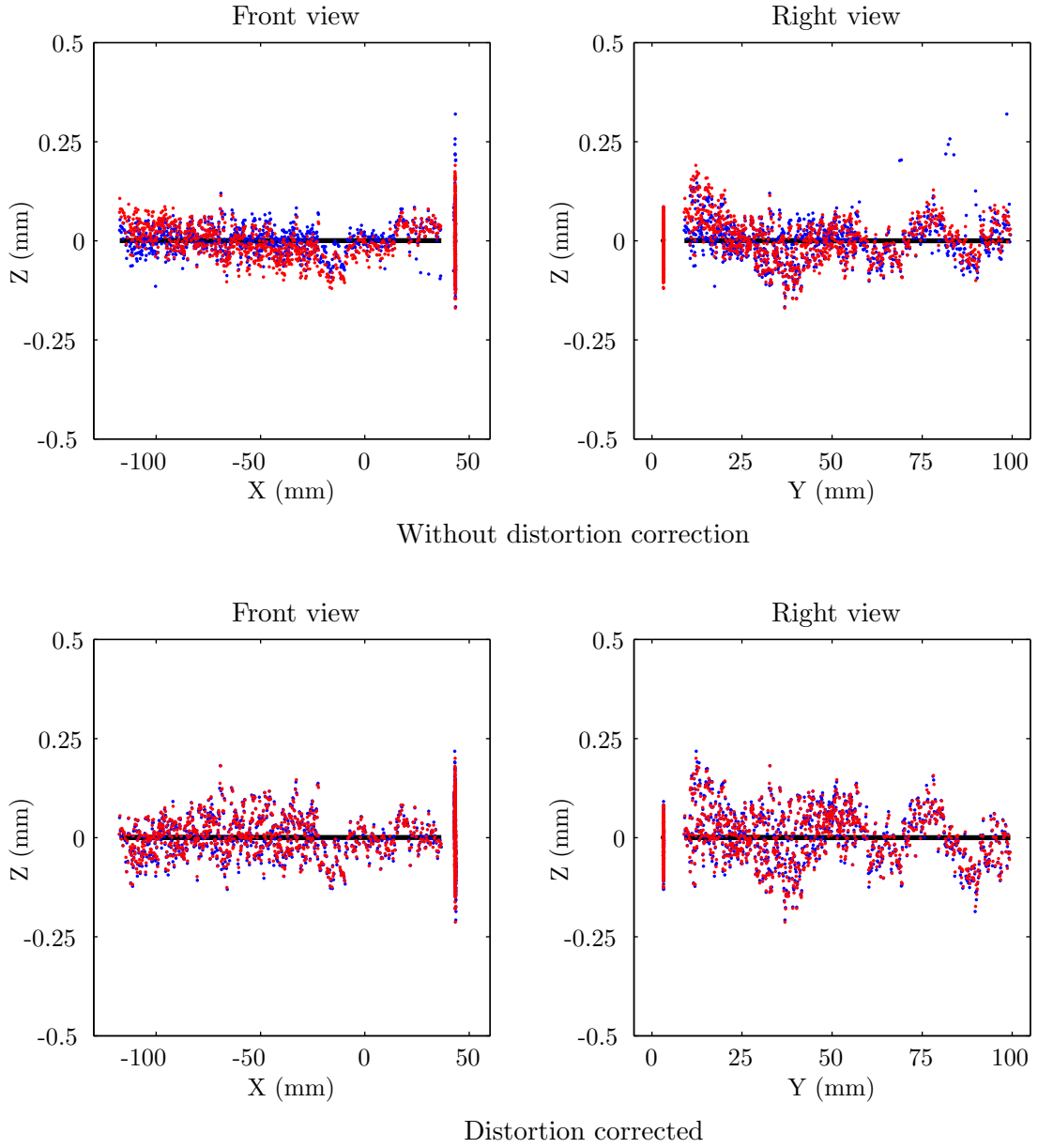


Figure 6.6: **Normal lens** Camera positions recovered by camera localization with a normal lens. The results from the proposed method are shown in *red*, the full bundle adjustment solution is in *blue*, and the ground truth is shown in *black*. The original sequence and the distortion corrected sequence were separately used to calibrate a full camera matrix including skew. A 4D projective homography was used to bring each camera path solution space into alignment with the ground truth.



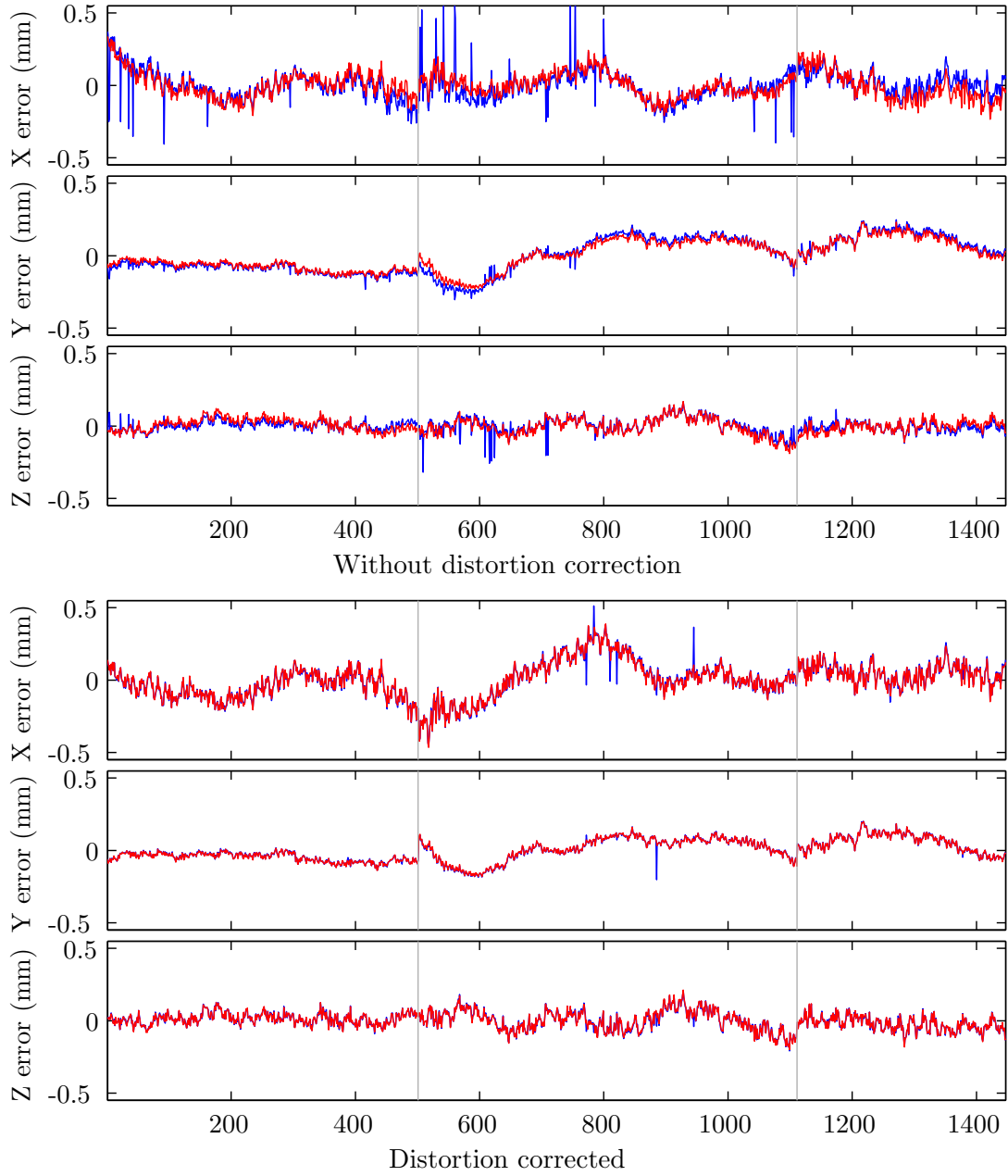


Figure 6.7: **Normal lens** Camera position errors relative to milling machine ground truth. The horizontal axis denotes the video frame, and the vertical grey lines delineate each distinct camera path as shown in Figure 6.3. The errors for the proposed method are shown in *red*, while the *boujou* errors are in *blue*. Note the decrease in incorrect positions for the *boujou* solution when distortion correction is used.

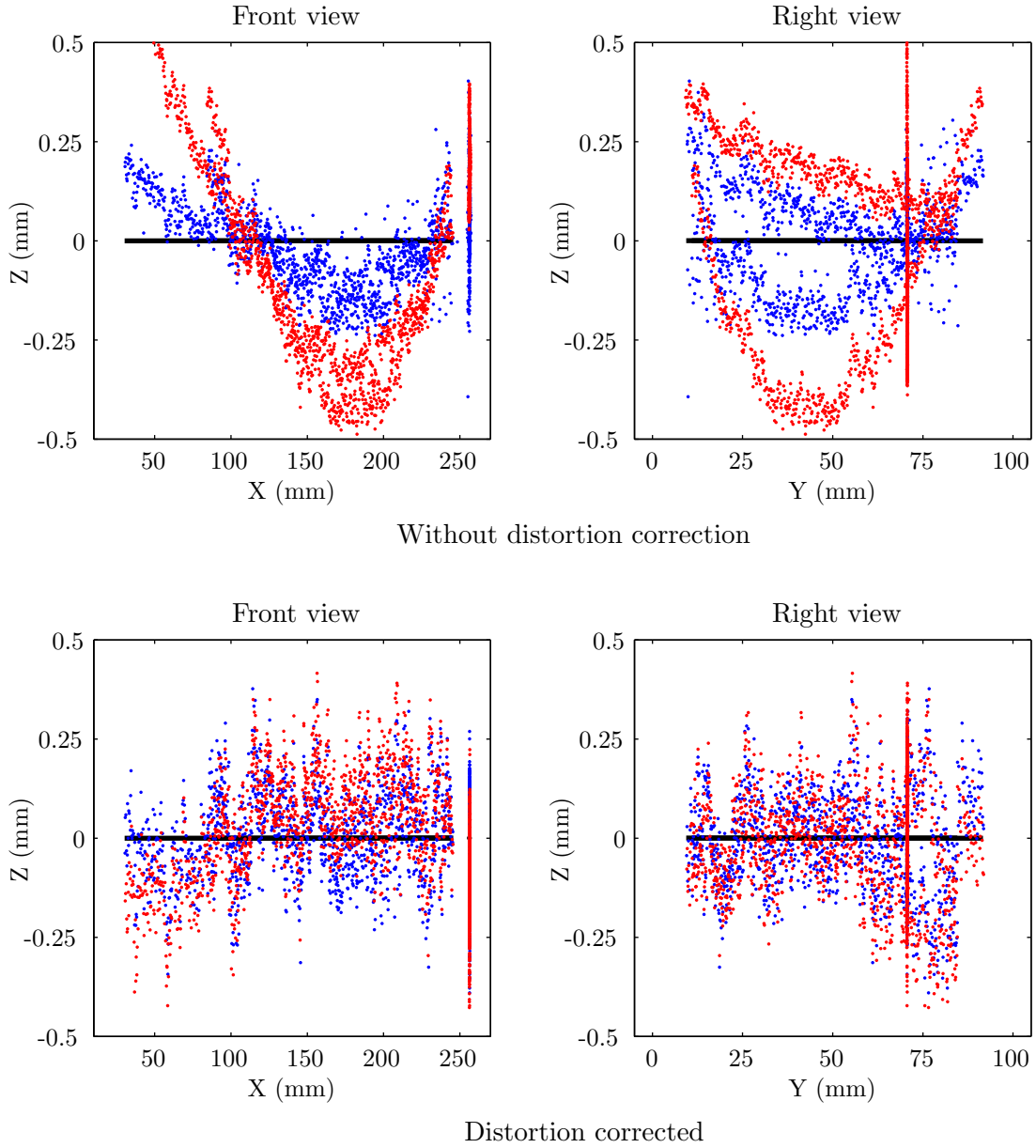


Figure 6.8: **Wide angle lens** Errors for wide angle lenses where the camera position was solved by single frame optimization based on the fiducials (in *red*), and by full bundle adjustment using *boujou* (in *blue*). Correcting the distortion removes the overall curvature from the solved camera path. Although it appears that correcting the distortion results in a larger standard deviation in errors, the plots on the next page illustrate that the bulk of the error has shifted into the z-axis.

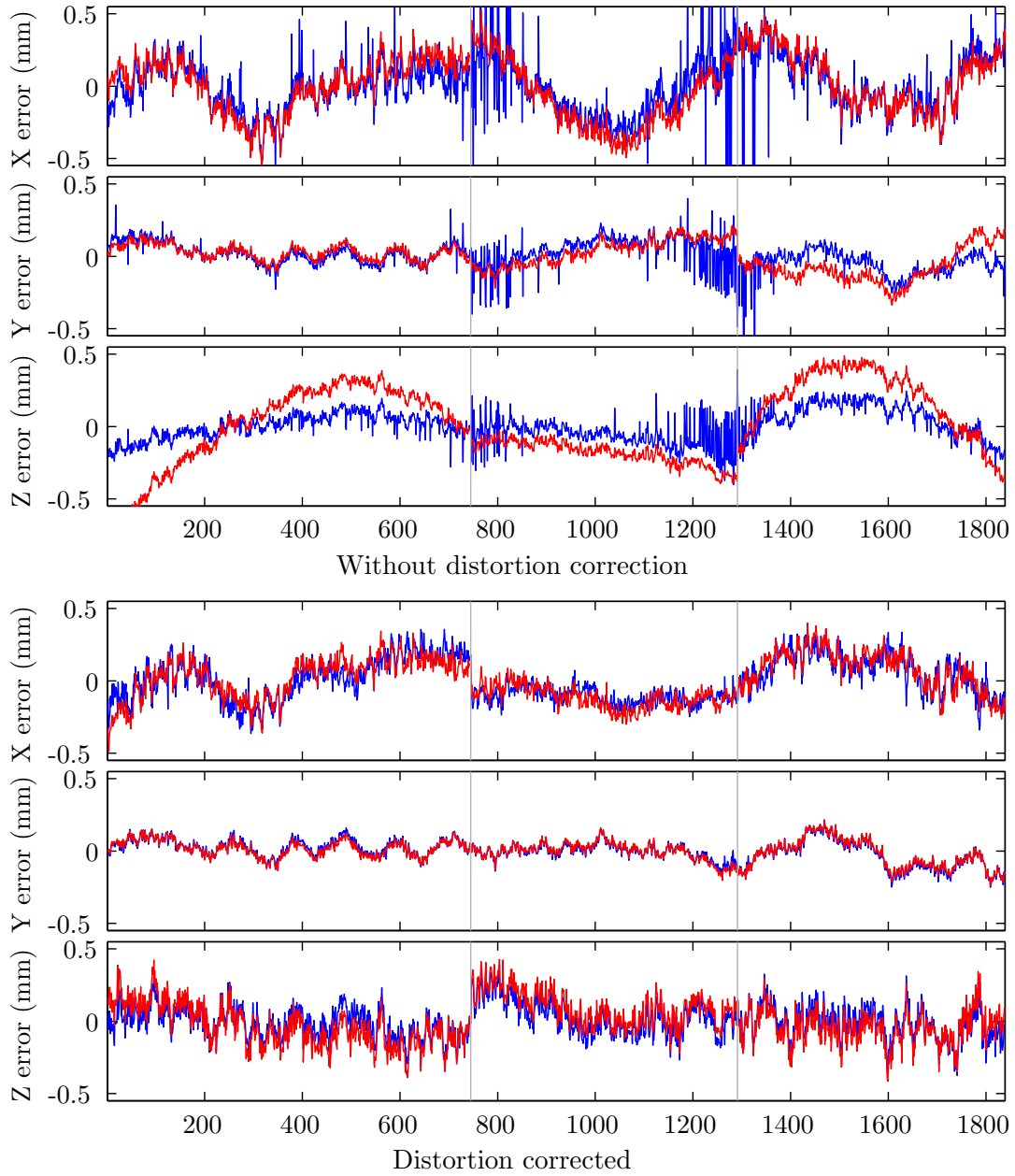


Figure 6.9: **Wide angle lens** Camera position errors relative to milling machine ground truth. Refer to Figure 6.7 for a description of the axes and legend. Modelling and removing the distortion corrects the overall curvature observed in the error plots without distortion correction. These results have had a projective homography applied to bring the camera paths into alignment with ground truth; without this coordinate system transformation the errors and overall curvature is much greater.

Lens	Distortion	Camera	RMS	STD	MAX
Normal	Raw	Focal	1.96	0.85	4.79
Normal	Raw	Full	0.55	0.28	1.64
Normal	Corrected	Focal	0.06	0.04	0.44
Normal	Corrected	Full	0.06	0.04	0.45
Fisheye	Raw	Focal	2.36	0.99	5.18
Fisheye	Raw	Full	1.08	0.67	3.44
Fisheye	Corrected	Focal	0.24	0.07	0.49
Fisheye	Corrected	Full	0.10	0.06	0.32

Table 6.11: Reprojection errors for the milling machine sequences. Modelling the lens distortion more accurately represents the imaging process, resulting in a lower reprojection error. The full camera model is only significant for the fisheye lens. All measurements are in pixels.

of plots we observe the improvements due to distortion correction. The upper set of plots are for positions recovered using a full camera projection matrix and per frame optimization but without correcting the lens distortion. A 4D projective homography has been used to bring the computed path into alignment with ground truth, so the variability observed is only due to pose error and not some global coordinate system misalignment (refer to Figure 6.5 for a depiction of the global misalignment caused by distortion). For the normal lens shown in Figure 6.7 the focal length is quite long and there is minimal distortion evident in the images. There is therefore very little difference between the two sets of plots for this lens. We do observe, however, a slight bias in the case without distortion correction. This is removed by modelling and removing the distortion. Another result of distortion in the images is that the *boujou* solution converges to an incorrect position more often. This can be seen as the spikes in the blue curve of Figure 6.7 *top*, which are far fewer in the distortion corrected bottom figure.

Figure 6.8 shows the vertical error for the fisheye lens. The solution space curvature induced by uncorrected lens distortion is very evident in the top set of plots, and absent in the lower set. On a lens with such extreme levels of distortion it is essential to model the distortion prior to computing camera pose. The position errors plotted in Figure 6.9 again demonstrate a reduction in both noise and overall curvature when distortion correction is included.

**Reprojection error** A secondary result to report when computing the camera location is the reprojection error. Using the camera parameters (including location) to reproject the known survey locations into the image provides a measure of how accurate

the camera parameter estimation is. It is expected that a model that includes additional parameters will be better able to reproduce the image; this experiment verifies that hypothesis and also provides a quantitative measure of the improvement realized through distortion correction.

Two different camera models were tested. One varied only the focal length, with all other internal camera parameters fixed as described in §6.2.1. The other model varied focal length, aspect ratio, principal point and skew. The reprojection errors for each camera model with both a normal and a fisheye lens are given in Table 6.11. Correcting the lens distortion significantly reduces the reprojection error, which shows (as expected) that the model more accurately represents the imaging process. For the normal lens, using the full camera model makes no appreciable difference, but for the fisheye lens the full model is necessary. This implies that the skew and principal point are parameters that should not be set to default values for this type of fisheye lens.

### 6.5.2 ARToolkit comparison

A second calibration environment was configured in order to compare the proposed fiducial-based localization system with an existing fiducial technique. The Augmented Reality Toolkit (ARToolkit) (Kato and Billinghurst 1999) is a set of C libraries for camera localization that can be freely downloaded from

[http://www.hitl.washington.edu/resarch/shared\\_space/download/](http://www.hitl.washington.edu/resarch/shared_space/download/).

It uses a global binarization threshold to detect square targets and then identify pre-defined patterns. Once these fiducials have been located it looks up the corresponding 3D location for each corner in a table which the user has defined. The set of image/world correspondences are used to compute the camera position to facilitate the introduction of virtual objects into video of a real scene (augmented reality).

Figure 6.10 shows the calibration environment for this comparison. A target was attached to each of the walls and the desk to establish an orthogonal coordinate system. This eliminates the need for skew or projective homographies to align the coordinate systems of the camera localization methods; both scale and orthogonality are imposed in the process of surveying with *boujou*. Target markers were produced for the ARToolkit system whose square edge length matched the spacing between dots on the proposed algorithm's targets. After performing the survey, a test sequence was recorded with the proposed markers in place. ARToolkit targets were then placed in the same physical locations for recording another test sequence. Placing the markers in the same locations ensures that the same geometric configuration is used by both solution methods, and requires the 3D coordinates to be surveyed only once. All image sequences had



Figure 6.10: Sample frame from the sequence used to survey the environment for comparison of the ARToolkit with the proposed method.

distortion corrected by calibrating the distortion and then warping the source images prior to fiducial detection. This procedure is time consuming, but it permits the same distortion correction to be used by each pose algorithm and *boujou*. Finally, *boujou* was used to establish baseline camera locations for each sequence. The *boujou* camera path is computed from tracks of a dense set of high quality features. This path produces augmented scenes which are visually accurate; since visual accuracy is the goal of augmented reality, this is taken to be the camera path against which others are compared. As *boujou*'s bundle adjustment chooses an arbitrary coordinate system, the two paths were aligned using a similarity transform, without skew or projection, before making the comparison.

The mean distance to the camera for the ARToolkit sequence was 3.33 m, which is on the edge of the operating range for the ARToolkit (Malbezin et al. 2002). Training patterns for each marker were recorded under the same lighting and viewpoint conditions as used in the test in order to maximize the marker recognition rate. However, the system was unable to reliably track multiple markers at this scale. The square border detection is successful for all three targets in 88% of the frames, but the pattern identification fails on most frames. In order to make a three-target comparison, we exported the image coordinates of every square pattern found and then manually established the marker identifications and world correspondences. This manual procedure yielded a listing of image coordinates that was then fed through the same nonlinear pose optimization as the proposed algorithm. As a result, the fiducial detection step is the only difference between the way the two camera paths under comparison were computed. It

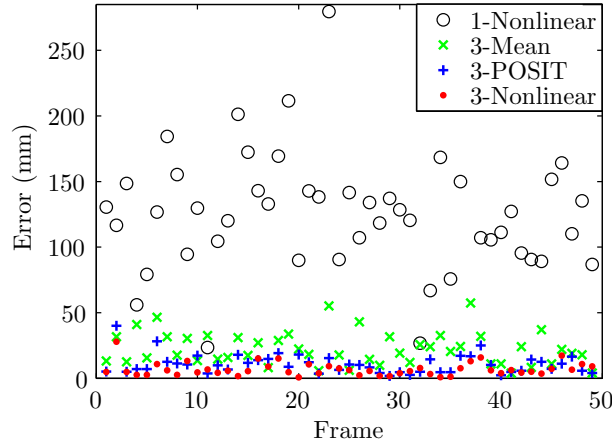


Figure 6.11: Errors in camera position computed by various optimizations on different numbers of targets. Plot shows absolute Euclidean camera position error for a sample set of 50 frames; the RMS errors over the entire sequence are given in Table 6.12.

should be noted that this pose optimization is more accurate than the pose computation method supplied with the ARToolkit; this change improves its results rather than degrading them.

The camera position errors measured between the computed path and the *boujou* solution are shown in Figure 6.11 and listed in Table 6.12. Several different pose estimation strategies were tested. First, both methods were used to compute the camera pose from a single target, the one on the left wall. This planar pose estimation task is more susceptible to fiducial detection error than the full 3D case, and this is reflected in the pose errors. The coordinates returned from the ARToolkit fiducial detection are less precise than the propose fiducial detection method, and thus the pose error is almost three times larger.

One method for obtaining a more accurate camera pose is to compute the average of the solutions obtained from several individual targets. The single target minimization was performed on each of the three targets, and then the mean of these three camera positions was reported. This does provide an improvement over the single target solution, but the iterative POSIT algorithm (see §6.2.3) does even better. Simultaneous optimization for camera pose based on the data from targets distributed in 3D space provides the most accurate camera pose of all the techniques tested. The errors reported in this section are higher than those from the milling machine sequence. This is likely due to the distortion correction methods used: the division model (see §3.1) is only a single parameter approximation, and the process of warping the image prior to fiducial detection introduced significant image noise.

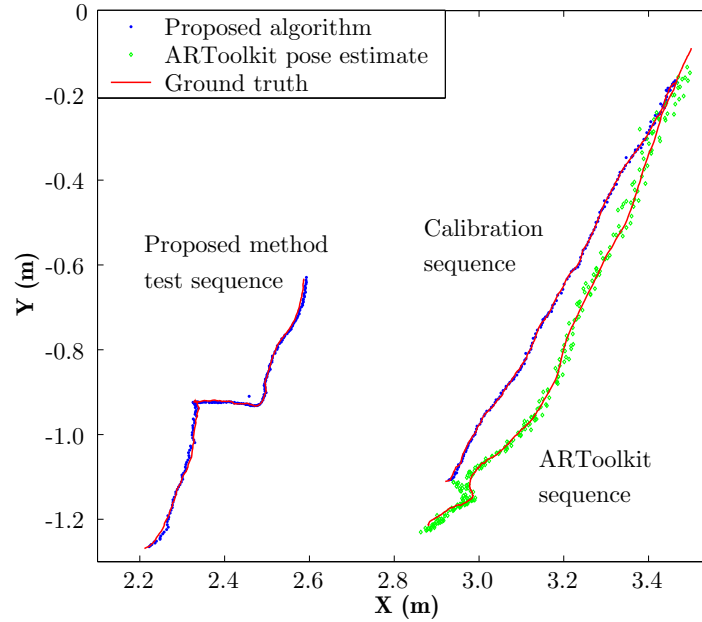


Figure 6.12: Camera tracks for three separate sequences in the same calibrated environment. The calibration sequence was used to survey the positions of the three targets. The other two sequences were used to test the camera localization algorithms. The proposed method (in *blue*) follows the baseline computed in *boujou* much closer than the ARToolkit technique (in *green*).

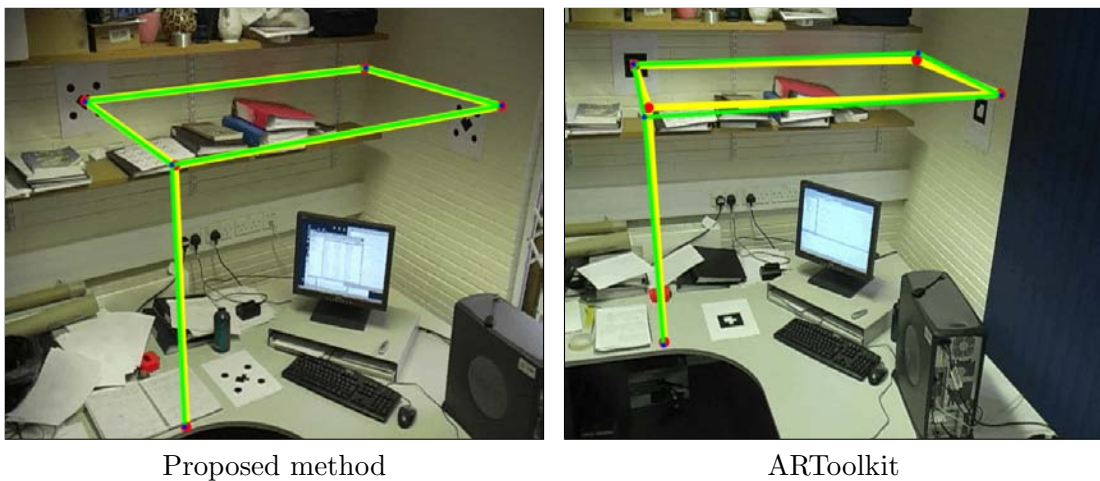


Figure 6.13: A virtual wireframe inserted into the scene using the camera position computed via the proposed method and the ARToolkit. Both are compared with the baseline solution.



Figure 6.12 shows a plan view of the computed camera paths. The coordinate system origin is at the centre of the target on the left wall. The camera path for the calibration sequence is also shown, with the pose estimation results from the proposed algorithm. The two test sequences provide an analysis of the generalization ability of the proposed optical marker surveying technique for camera localization; none of this data was used for calibration, so it is strictly a single-frame pose estimation algorithm.

A virtual 3D object has been superimposed on a sample frame from each sequence in Figure 6.13. The wireframe is aligned with the coordinate axes, and one end is attached to the origin on the left wall target. The image position and focal length are set by the camera parameters computed by either the proposed method or the ARToolkit. The yellow lines and red dots were rendered using the ground truth camera from *boujou*, while the green/blue set are from the to algorithms under test. Notice that all sets are well aligned with the target on the left wall; virtual objects tend to line up well in the vicinity of the reference points (fiducial targets in this case). However, further away from the targets, particularly out of the plane of the target, the alignment begins to break down. This misalignment is due to error in the camera parameters, and causes jitter and swaying for virtual objects inserted into video sequences.

## 6.6 Summary

Given a set of known world positions and their corresponding image locations, camera localization computes the pose of the camera when the image was recorded. This chapter has presented camera localization as a nonlinear optimization problem. Initialization of the camera parameters (internal and external, other than distortion which is

Number of Targets	Localization Method	RMS error (mm)
1	ARToolkit	360 <sup>†</sup>
1	Non-linear reprojection minimization	130
3	ARToolkit (refer to text)	43*
3	Mean of three positions	25
3	POSIT	14
3	Non-linear reprojection minimization	9.7

<sup>†</sup> Omitted 54% of frames where identification failed.

\* Omitted 12% of frames where border detection failed. Non-linear optimization added, manual target identification required.

Table 6.12: Camera position error relative to bundle-adjusted ground truth. Non-linear minimization of the reprojection errors from only twelve fiducial markers (three targets) provides centimeter level camera position accuracy.

handled beforehand as per the previous chapters) is handled by existing methods such as POSIT or the DLT. The optimization minimizes the distance between the detected image points and the location of the world points projected by the camera model. Depending on the sequence being solved for, these parameters may include focal length, aspect ratio, principal point, a rotation and translation. Computing the error function using analytic derivatives rather than finite differences speeds up convergence to the point where the optimization can be performed in realtime. It was shown that this method of computing the camera's intrinsic parameters such as focal length produces results that are comparable to dedicated calibration methods.

This chapter also presented an optical surveying procedure for determining the 3D position of the 3D markers. Structure from motion software is used to automate this difficult and error prone measurement task. This technique could also be applied to general position measurement tasks, and is an example of how a precisely calibrated consumer grade camera becomes a powerful yet inexpensive measurement instrument.

The final section of this chapter examined how precise such measurements can be. Camera localization results were compared with independent absolute measurements of the camera position. A normal lens corrected using the rational function model produced camera locations for each individual frame that are accurate to 0.15 mm RMS and 0.47 mm maximum. The results for a fisheye lens were 0.23 mm RMS and 0.54 mm maximum. This indicates that sub millimeter position accuracy is possible for single frame camera localization using nonlinear optimization with appropriate lens distortion correction. Moreover, these results are comparable to those obtained through bundle adjustment on the entire sequence; single frame methods suitable for realtime applications need not suffer in accuracy for lack of bundle adjustment.

## Chapter 7

# Fiducial Detection

The image position of four or more known points can be used to compute the camera position. Often these known points are special visual markers referred to as *fiducials* or *landmarks*. Fiducial detection is an important problem in real-world vision systems. It requires fast, accurate registration of unique landmarks under widely varying scene and lighting conditions. Numerous systems have been proposed which deal with various aspects of this task, all based around the concept of selecting some sort of feature or shape and then assembling a set of low level vision operators capable of identifying that feature. An example of an easily recognizable fiducial is the circle with alternating white and black quadrants used on crash test dummies. The operations used to detect the fiducials often include Canny edge detection, Hough transforms, and a collection of geometric heuristics (ellipse eccentricity, area vs. edge length, etc.). A good detector is then a product of careful engineering: selecting a recognizable feature, assembling the right mix of image operations, defining the identification logic, and tweaking all the parameters to tune the overall system. Such a system that demonstrates reliable performance on a wide variety of scenes has not yet been reported.

Figure 7.1 illustrates the difficulties inherent in a real-world solution of this problem, including background clutter, motion blur (Klein and Drummond 2002), large differences in scale, foreshortening, and the significant lighting changes between indoors and out. These difficulties mean that a reliable general-purpose solution calls for a new approach. In fact, this chapter shows how the power of machine learning techniques, for example as applied to the difficult problem of generic face detection (Viola and Jones 2001), can benefit even the most basic of computer vision tasks. Even these tasks become challenging when high reliability is included in the design constraints.

The problem addressed in this chapter is to design a planar pattern which can be reliably detected in real world scenes. One of the main challenges in fiducial detection is handling variations in scene lighting. Transitions from outdoors to indoors, backlit

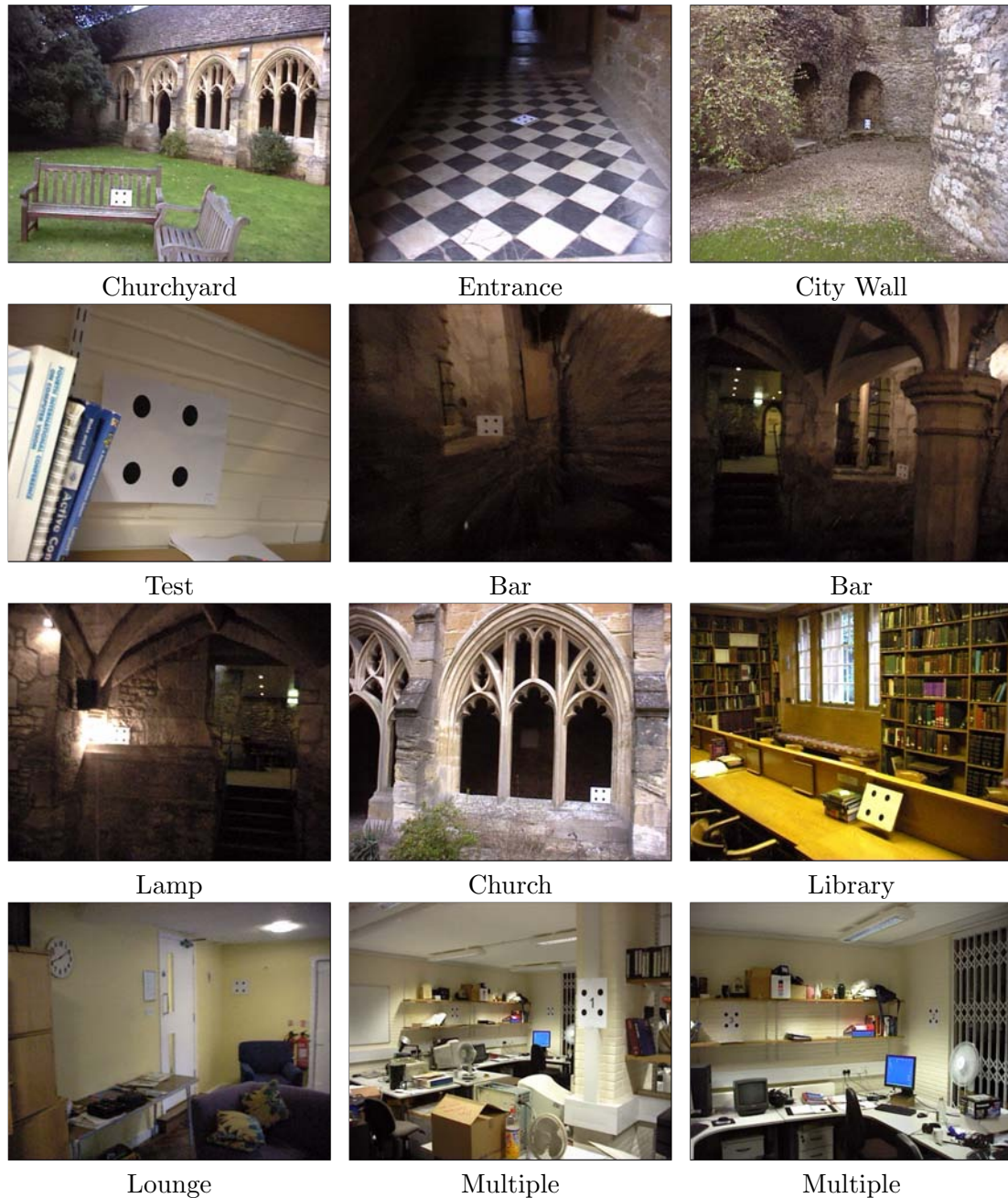


Figure 7.1: Sample frames from various test sequences. The task is to reliably detect the targets (four disks on a white background) which are visible in each image. It is a claim of this report that no technique currently in use is robust over a large range of scales, lighting and scene clutter. In real-world sequences, it is sometimes difficult even for humans to identify the target. The goal is to detect the target with high reliability in such images.

```

for input frame  $I$ 
  for  $scale = 1 : 4$ 
     $S = \text{resize}(I, 1/scale)$ 
    for  $i, j \in S$ 
       $\omega = \text{window}_{12 \times 12} \text{centered}(i, j)$ 
      if  $\text{classify\_fast}(\omega)$  then
         $\text{out}(i, j) = \text{classify\_full}(\omega)$ 
       $\text{dots}(scale) = \text{non\_maxima\_suppression}(\text{out})$ 
     $\text{targets} = \text{verify\_target}(\text{dots})$ 

```

Figure 7.2: Overall algorithm to locate fiducials. The key components are the cascaded classifiers *classify\_fast* and *classify\_full* which rapidly mark each pixel window as fiducial or non-fiducial.

objects and in-camera lighting all cause global thresholding algorithms to fail.

Existing systems (refer to (Claus 2004) for a review) all rely on transformations (such as adaptive thresholding) to produce invariance to some of the properties of real world scenes. However, lighting variation, scale changes and motion blur still affect performance. This chapter describes a new fiducial detection system which deals with these effects through machine learning, and concludes with a comparison of the learning-based and traditional approaches.

### Detection versus tracking

The system described here does not use any prediction of the fiducial locations; the entire frame is processed every time. This is in contrast to systems which only search the region located in the previous frame. This assumes that the target will only move a small amount between frames and causes the probability of tracking subsequent frames to depend on success in this frame. As a result, the probability of successfully tracking through to the end of a sequence is the product of the frame probabilities, and rapidly falls below the usable range. An inertial measurement unit can provide a motion prediction (Klein and Drummond 2002), but there is still the risk that the target will fall outside the predicted region. Another difficulty with such prediction strategies involves occlusion of the target. If a person (for example) walks in front of the camera, the target will not be visible for a number of frames. Algorithms based on search region prediction will fail if there was any camera movement during the period of occlusion. This work will therefore focus on the problem of detecting the target independently in each frame, without prior knowledge from the earlier frames.

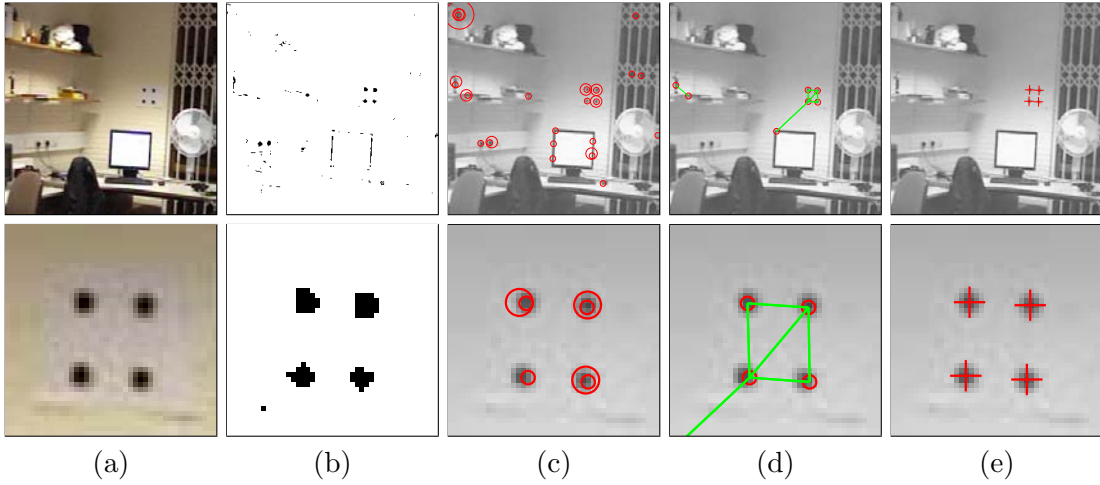


Figure 7.3: Overall algorithm to locate fiducials. (a) Input image, (b) output from the fast classifier stage, (c) output from the full classifier superimposed on the original image. Every pixel has now been labelled as fiducial or non-fiducial. The size of the circles indicates the scale at which that fiducial was detected. (d) The target verification step rejects non-target fiducials through photometric and geometric checks. (e) Fiducial coordinates computed to subpixel accuracy.

## 7.1 Strategy

The fiducial detection strategy adopted in this thesis is to collect a set of sample fiducial images under varying conditions, train a classifier on that set, and then classify a subwindow surrounding each pixel of every frame as either fiducial or not. There are a number of challenges, not least of which are speed and reliability.

Representative training samples were collected in the form of  $12 \times 12$  pixel images; larger fiducials are scaled down to fit this window. This training set is then used to classify subwindows as outlined in the algorithm of Figure 7.2 and shown in Figure 7.3.

The distance measure used to compare two images is the sum-of-squared-differences (SSD) between the intensity at each corresponding pair of pixels. This often-used method does not always give an accurate indication of proximity; two images misaligned by a single pixel may result in a very large discrepancy. For example, a checkerboard image with alternating white and black pixels will display maximal difference if offset by a single pixel, but the distance measure reports perfect alignment if offset by a multiple of two pixels. This is a synthetic example, however. Real images have smoothly varying intensity, so the SSD should provide a more representative measure of the difference in alignment between two images. To test this hypothesis, a sample dot was given a horizontal offset ranging from -3.5 to 3.5 pixels, and the SSD distance from the original image reported for each translation value. With no translation the difference between

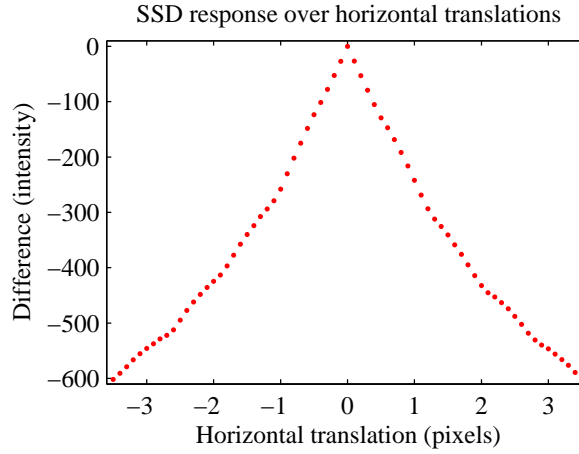


Figure 7.4: Sum of squared difference between a sample dot and the same dot translated from -3.5 to 3.5 pixels horizontally.

the two images is zero, and as the offset increases the difference grows. This difference can be interpreted as the alignment error due to the offset, and is given a negative sign to emphasize this. The results in Figure 7.4 demonstrate that there is a very sharp peak in the response when the samples are aligned, and the function drops off cleanly on either side. It follows that sliding a window over the image and measuring the SSD is an effective method for dot localization.

Each frame is subsampled (by half) four times to cover dots at multiple scales. For each location and scale, a centred  $12 \times 12$  subwindow is extracted and fed to the classifier. The classifier must be fast and reliable enough to perform half a million classifications per frame and still permit recognition of the target within the positive responses.

High efficiency is achieved through the use of a cascade of classifiers (Viola and Jones 2001). The first stage is a fast “ideal Bayes” lookup that compares the intensities of a pair of pixels directly with the distribution of positive and negative sample intensities for the same pair. If that stage returns positive then a more discriminating (and expensive) tuned nearest neighbour classifier is used. This yields the probability that a fiducial is present at each location within the frame; non-maxima suppression is used to isolate the peaks for subsequent target verification.

The target verification is also done in two stages. The first checks that the background between fiducials is uniform and that the separating distance falls within the acceptable range for the scale at which the fiducials were identified. The second step is to check that the geometry is consistent with the corners of a square under perspective transformation. The final task is to compute the weighted centroid of each fiducial

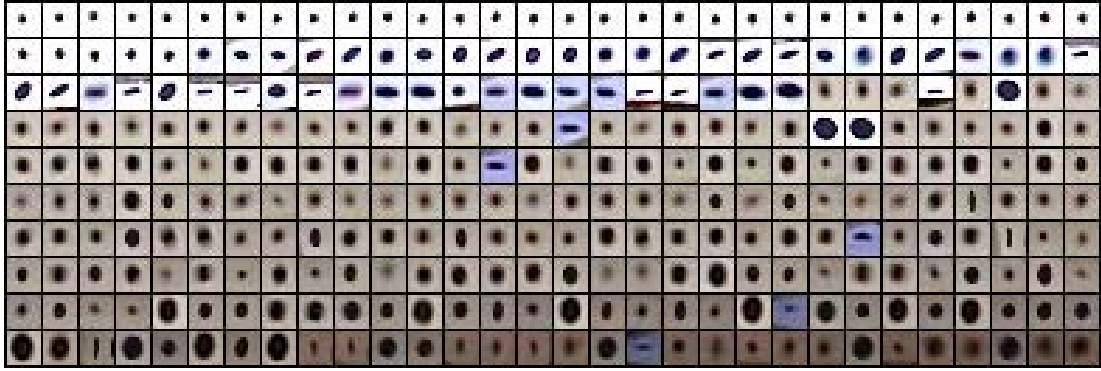


Figure 7.5: Representative samples of positive target images. Note the wide variety of positive images that are all examples of a black dot on a white background.

within the found target and report the coordinates.

The following section elaborates on this strategy; the fiducial design is presented first, followed by the selection of training data, and then each stage of the classification cascade is covered in detail.

### 7.1.1 Target Design

A circular fiducial has been selected because the centroid is easily and efficiently measured to sub-pixel accuracy. One drawback to a circular fiducial is that the centre of the original circle is lost under projective transformation (as described in §2.4.4).

Four known points are required to compute camera pose (a common use for fiducial detection) so four circles are arranged in a square pattern to form a target. This arrangement allows the original circle centres to be located, because the approximate projection can be determined and recursively updated. This centroid shift is only an issue in extreme projective cases, and the error resulting from the mismatched centres is in practice quite small (§7.3.3), so this scheme converges. Finally, the centre of the target may contain a bar-code or other marker to allow different targets to be distinguished.

### 7.1.2 Training Data

A subset of the positive training images is shown in Figure 7.5. These samples indicate the large variations that occur in real-world scenes. They were acquired from a series of training videos recorded under a variety of scene conditions. A dot was manually selected in one frame and then a simple tracking algorithm was used to locate the same dot in the subsequent frames. In this manner a large number of sample dots were obtained with minimal effort. The window size was set at  $12 \times 12$  pixels, which limited



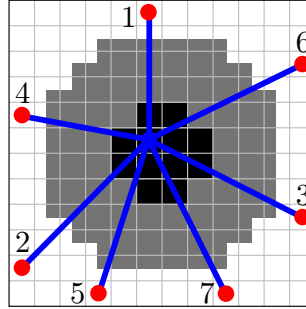


Figure 7.6: Geometry of the  $12 \times 12$  scanning window used to detect fiducials. The size of the detected fiducials ranges from 4-9 pixels in diameter, and is shaded in the figure for reference. The first classifier stage consists of seven sequential probes on pairs of inner and outer pixels, as shown. The second stage is a 144 dimensional tuned nearest neighbour classifier acting on all pixels in the window.

the sample dot size to between 4 and 9 pixels in diameter (see Figure 7.6); larger dots are scaled down by a factor of two until they fall within the specification. Samples were rotated and lightened or darkened to artificially increase the variation in the training set. This proves to be a more effective means of incorporating rotation and lighting invariance than *ad hoc* intensity normalization, as discussed in §7.3.3.

Every third image was designated as a training image, and the remainder form the test set. When we get to testing, it is important that the test and training sets not overlap: the training data should not include any samples from the video sequence being tested. This is accomplished through leave-one-out learning where any training samples from the video under examination are removed prior to generating the condensed nearest neighbour subset (§2.4.5).

### 7.1.3 Cascading Classifier

The target location problem here is firmly cast as one of statistical pattern classification. The criteria for choosing a classifier are speed and reliability: the four subsampled scales of a  $720 \times 576$  pixel video frame contain 522,216 subwindows requiring classification. Similar to (Viola and Jones 2001), this work adopts a system of two cascading probes:

- fast Bayes decision rule classification on sets of two pixels from every window in the frame
- slower, more specific nearest neighbour classifier on the subset passed by the first stage

The first stage of the cascade must run very efficiently, have a near-zero false negative rate (so that any true positives are not rejected prematurely) and pass a minimal

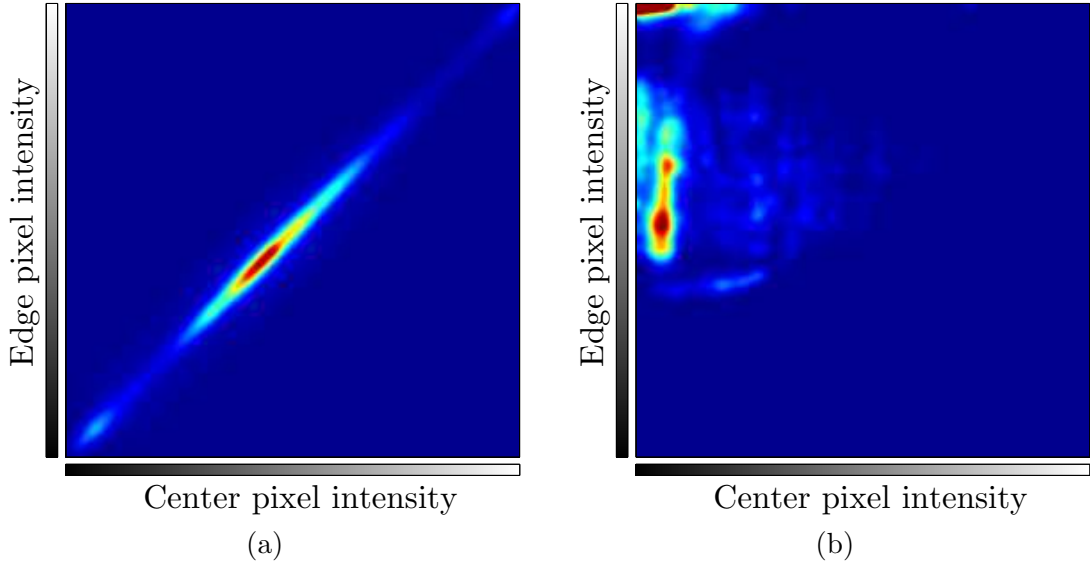


Figure 7.7: Distribution of (a) negative pairs, and (b) positive pairs used to construct the fast classifier.

number of false positives. The second stage should offer high classification accuracy, but may incur a higher computational cost.

#### 7.1.4 Cascade Stage One: Ideal Bayes

The first stage of the cascade constructs an ideal Bayes decision rule from the positive and negative training data distributions. Two pixels are selected within each subwindow: one at the centre of the dot and the other on the background. The distribution of the training data is shown in Figure 7.7. Notice in the negative distribution that nearby image pixels in natural scenes tend to have similar intensity values. These distributions were built into an ideal Bayes classifier as described in §2.4.5. The parameter  $\lambda$  (from Eqn 2.26) was varied to produce the ROC curve shown in Figure 7.8. A weighting of  $\lambda = e^{12}$  produces the decision boundary shown in Figure 7.8b, and corresponds to a sensitivity of 0.9965 and a specificity of 0.75. The specificity predicts the number of negatives that will be rejected by this stage of the classifier cascade.

The multiple application of the Bayes classification probe reduces the number of positives let through as the specificity is compounded. If an outer/inner pixel test is included from three outer edges then the predicted percentage of positives is  $0.25^3 = 1.56\%$  (assuming the subsequent probes are independent) where  $0.25 = 1 - \text{specificity}$ . The observed number of positives was found to be 1.5%, which agrees with the prediction. With all seven pairs applied the observed pass rate was 0.33%.

Using the same two dimensional training data but rotating the probe seven times

creates a pseudo-eight dimensional probe. Figure 7.6 shows the geometry of the probe, including the locations of the seven rotated probe pairs. A subwindow is marked as a possible fiducial if all seven intensity pairs all lie within the positive decision region. The outer edge pixels were selected to minimize the number of false positives based on the above empirical distributions. An early-bailout is implemented in practice so that a sample is rejected as soon as one probe fails; thus eliminating much unnecessary computation.

The first stage of the cascade seeks dark points surrounded by lighter backgrounds, and thus functions like a well-trained edge detector (refer to Figure 7.9c). Note however that the decision criteria is not simply

$$(edge - center) > threshold$$

as would be the case if the center was merely required to be darker than the outer edge. Instead, the decision surface in Figure 7.8b encodes the fact that {dark center, dark edge} are more likely to be background, and {light center, light edge} are rare in the positive examples. The simple thresholding strategy was compared against both the learnt decision surface and an artificially generated decision surface (based on what one would reasonably expect to differentiate between a dark dot and a light background). Slight changes to the learnt surface (such as removing islands) were found to improve the classifier, but wholesale modification did not; the thresholding method was also

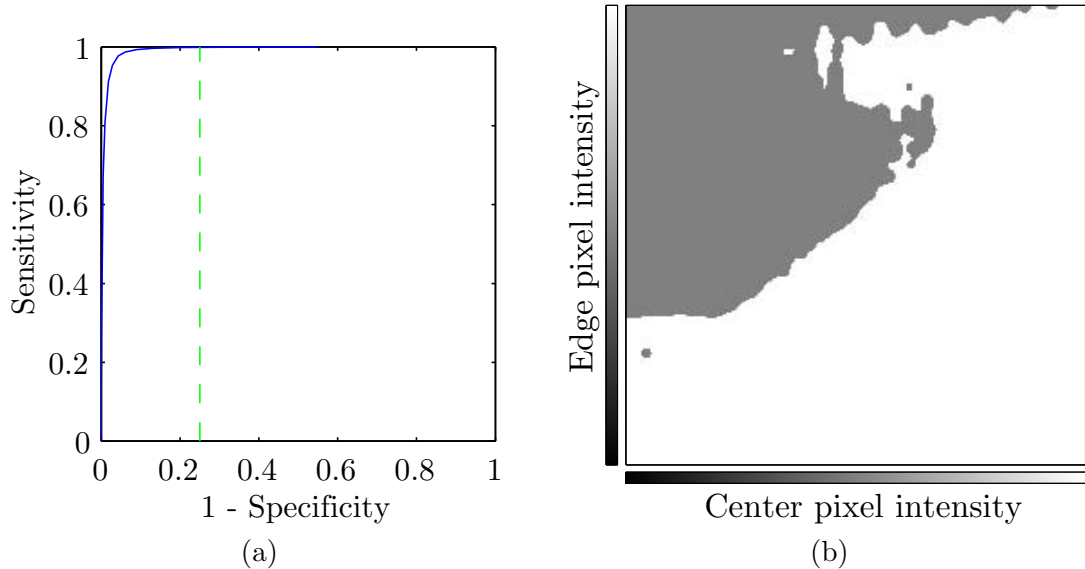


Figure 7.8: (a) The ROC curve used to determine the value of  $\lambda$  (indicated by the dashed green line). (b) The decision surface generated by  $\lambda = e^{12}$ , which corresponds to a 1-specificity value of 0.25. This is the optimal decision boundary given the costs of positive and negative errors.

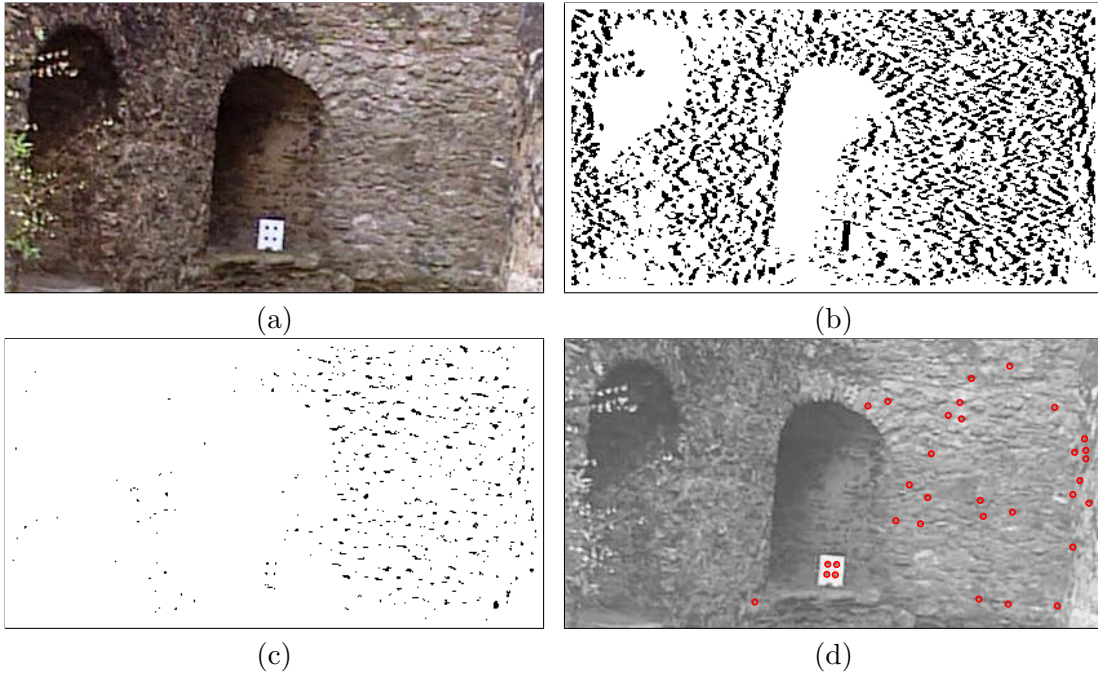


Figure 7.9: Sample output for the stages of the cascading fiducial classification algorithm (a) Original image of the Oxford city wall, also processed by the adaptive thresholding algorithm (see Figure 7.19). (b) A single pair in the fast classifier returns 1393 positive regions. (c) All seven rotated pairs in the fast classifier reduce the number of positive returns to 489. (d) The second classifier stage uses a parameterized nearest neighbour search and non-maxima suppression to select the most probable fiducials. The 32 positive responses are denoted by red circles.

found to be too simple of a decision surface.

### 7.1.5 Cascade Stage Two: Nearest Neighbour

The second stage of the classifier uses a nearest neighbour classifier to identify possible fiducials. Each  $12 \times 12$  window is re-shaped into a 144 dimensional vector of pixel intensities, and the nearest samples identified in this 144 dimensional space.

The nearest neighbour condensing algorithm (Hart 1968) was used to reduce the size of the training data sets. Testing showed that attempts to remove noisy points (such as edited nearest neighbour (Duda et al. 2001) dataset reduction) decreased the performance. The condensed nearest neighbour algorithm (§2.4.5) does not produce the globally minimal consistent subset of the original dataset; the choice of starting examples influences the final subset. The initial positive and negative samples should be representative of their representative classes in order to improve the generalization performance of the resulting subset. This was achieved through the use of a synthetic “ideal” fiducial image. All training data was compared to this image, and the training

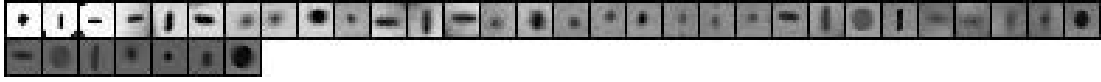


Figure 7.10: Positive training samples retained by the condensed nearest neighbour algorithm. These 37 samples were condensed from an initial training set of 8,506 samples.

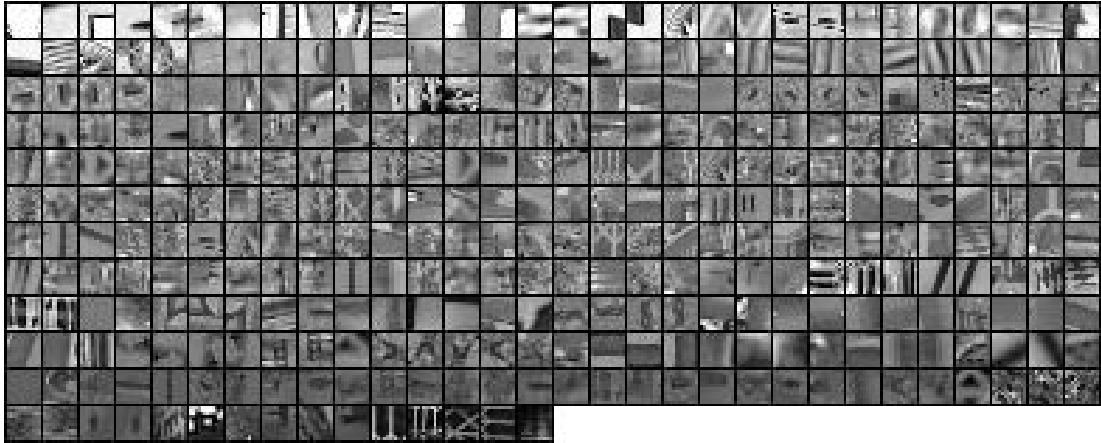


Figure 7.11: Negative training samples retained by the condensed nearest neighbour algorithm. These 345 samples were condensed from an initial training set of 19,052 samples (a compilation of the original samples plus warped and normalized copies).

examples with the shortest and longest distance were used as the positive and negative initial examples, respectively. The combined (test and training) data was condensed from 8,506 positive and 19,052 negative examples to 37 positive and 345 negative examples. The resulting subset is displayed in Figures 7.10 and 7.11.

The second classifier stage examines a window centered on each positive classification of the first stage. For each of these sample points the classifier computes the distance to the nearest positive and negative training examples, and returns the ratio  $\min(neg\_distances)/\min(pos\_distances)$ . A non-maxima suppression step then identifies the most ‘dot-like’ pixel within each region of positive classification. Figure 7.14 shows the results of this second classification stage.

### Lower Dimensional Probes

Several probe configurations (Figure 7.12) were initially proposed as a means of feature selection. The aim is to select a number of pixels from the  $12 \times 12$  subwindows that will permit classification by examining fewer dimensions. These were based on engineering intuition and a knowledge of the layout of the target. For instance, the target is a black dot with a white background, so some pixels should capture the black centre, while others should focus on the background. This implies that a probe with both

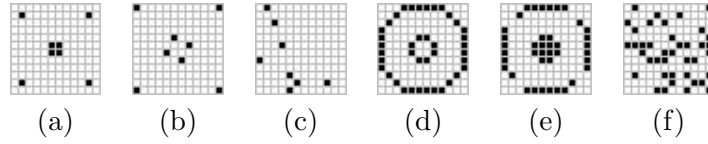


Figure 7.12: Test feature selection probes of various dimensions: (a) Initial 8D (b) Learnt 8D (c) Random 8D (d) Initial 40D (e) Learnt 40D (f) Random 40D For a given number of points, the learning process determines the best layout of query pixels.

centre and edge pixels should perform better, but the very corner pixels tend to pick up features that are off the target background.

In order to determine if these ‘designed’ probes are indeed the optimal configurations, a series of probes were assembled and ROC curves computed for each one. Eight-dimensional probes with four axes of symmetry, four interior and four exterior pixels were assembled. Forty-dimensional probes with twelve interior pixels and twenty-four exterior pixels were generated along similar lines. A set of random 8 and 40 dimensional probes were also generated. The top-performing designed probes and representative random probes are given in Figure 7.12. The ROC curves for each probe are given in Figure 7.13.

Classification with the feature selection probes resulted in a higher number of true target rejections (false negatives). The decrease in computational complexity was not sufficient to offset the drop in performance, so the nearest neighbour classification stage operates on the full 144 dimensional images.

## 7.2 Implementation

The implementation of the algorithm explained in §7.1 is straightforward for all but the target verification step. The cascading classifier identifies all dots within each video frame, but some false positives are found, in addition to the true target dots. Figure 7.14 shows a typical example of the classifier output, where the true positive responses are accompanied by a small number of false positives. These verifications steps are described in this section, but first we will address the issue of incorrect training data in the training sets.

### 7.2.1 Training Data Filtering

It is essential to ensure that the training data does not contain misleading data. For instance, negative examples that actually appear to be dots should not be included in the negative set as they will cause legitimate dots to be rejected. Likewise, positive examples where the background clutter encroaches at the edges will cause additional

false positives. One option is to model the positive training examples as a Gaussian distribution and reject the extreme outliers. However, the positive sample probability density function shown in Figure 7.7b demonstrates that the distribution is far from Gaussian. Instead, training data filtering was done manually: any dots with background clutter or improperly centred dots were rejected.

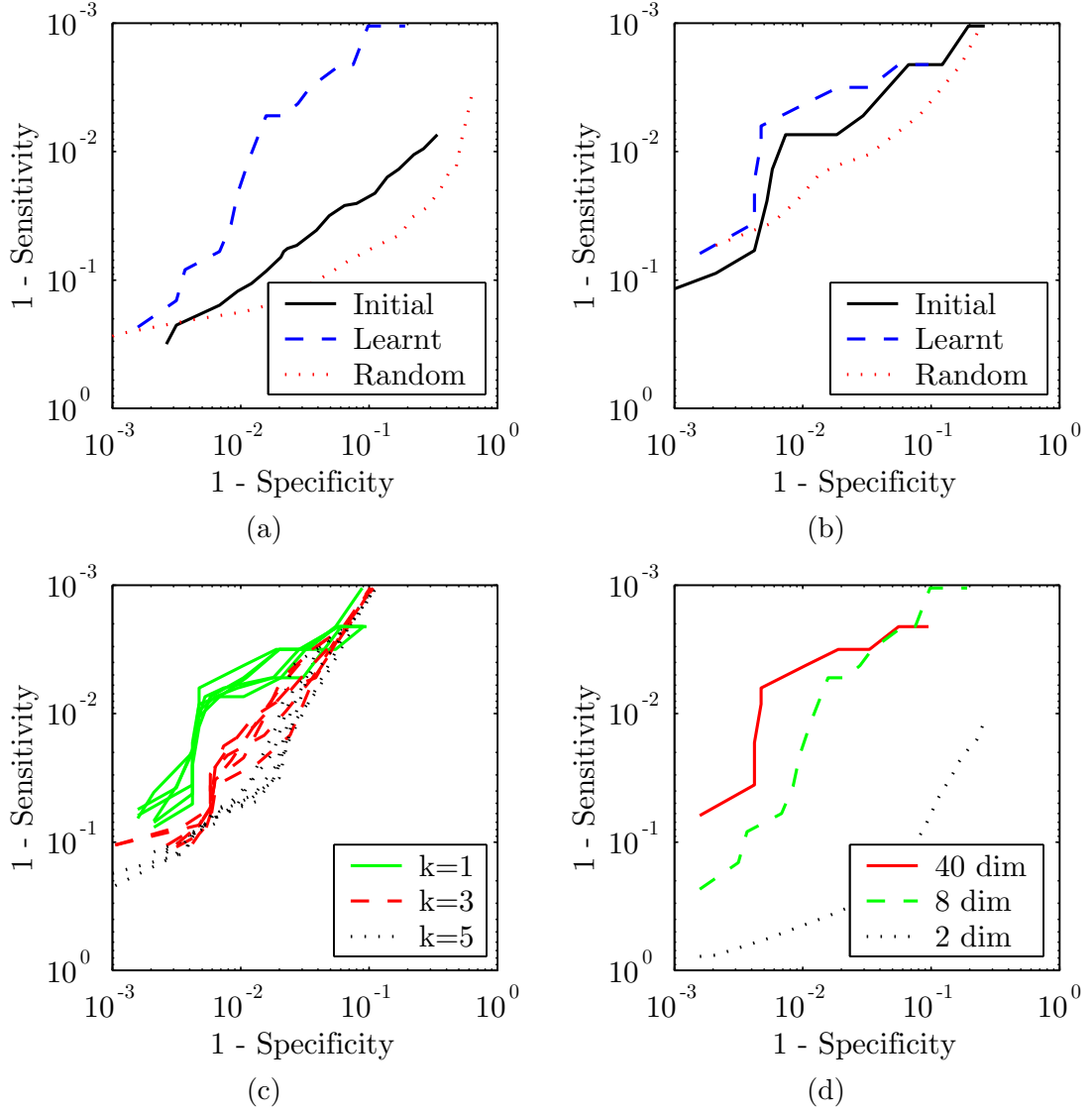


Figure 7.13: ROC curves for lower dimensional probes: (a) 8D probes (b) 40D probes (c) 1-NN outperforms  $k$ -NN for  $k > 1$  (d) Learnt probes for each dimension. As the probes have both a high sensitivity and a high specificity, the results are displayed on log-log scales with 1-sensitivity plotted on a reversed vertical axis.



Figure 7.14: Example output of the classification stages, at multiple scales, showing the true target fiducials identified along with a number of false positives. The target verification stage seeks to remove the remaining false positives.

### 7.2.2 Target verification

Target verification is merely used to identify the target amongst the positive classification responses; this section outlines one approach but there are any number of suitable techniques.

**Photometric Check** The Delaunay triangulation of all positive points is first computed to identify the lines connecting each positive classification with its neighbours. The Delaunay triangulation was used as a means to connect all of the points without introducing an unduly large computational load. A weighted average adaptive thresholding is used to segment the line into light and dark regions. This is essentially the adaptive thresholding algorithm described in §7.3.2 simplified to one dimension. The photometric test checks that there are two transitions, and that the centre region is over half the overall length. An example line check is shown in Figure 7.15 and detailed pseudocode is given in Figure 7.16. All lines that fail this test are removed; points that retain two or more connecting lines are passed to the geometric check.

**Geometric Check** The geometric check is used to identify groups of points whose positions are consistent with the layout of the target (*i.e.* fit the four corners of a square under perspective projection). The Delaunay triangulation is re-computed for the points that passed the photometric test, and all groups that cannot form the target at the current scale are removed (Figure 7.17). For each remaining group, all combinations of four points are tested. Three of the points are used to compute the transformation that maps them onto the corners of a unit right triangle. This transfor-



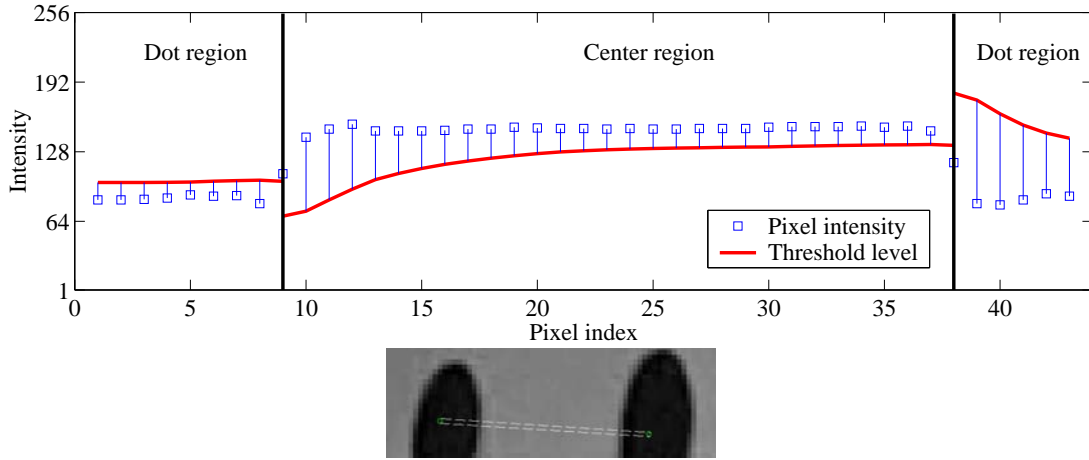


Figure 7.15: Example of the one-dimensional photometric check to verify that two proposed dots are joined by a smooth, light coloured background.

```

Inputs: image (current frame)
        endpoints (coordinates of the line)
        scale (current search scale as a fraction)
Initialization: offset = 20
                threshold_pos =  $0.8 * \text{offset}$  = 16
                threshold_neg =  $-5.0 * \text{offset}$  = -100
                line = get_line_pixels(image, endpoints)
                wgt_sum = line[0] * s (initialize the weighted sum)
                s =  $\max[\text{length}(\text{line})/8, 2]$ 
                num_transitions = 0
                region_sign = [1 -1 1]
if  $\text{length}(\text{line}) < 4/\text{scale}$  or  $\text{length}(\text{line}) > 60/\text{scale}$  then return false
if  $\text{standard\_deviation}(\text{line}) > 1$  then normalize(line,  $\mu = 128, \sigma = 30$ )
else return false
while i <  $\text{length}(\text{line})$  and num_transitions < 4 do
    if (line[i] - wgt_sum/s) < threshold_neg * region_sign[num_transitions] then
        return false (step too large in wrong direction)
    if (line[i] - wgt_sum/s) < threshold_pos * region_sign[num_transitions] then
        num_transitions = num_transitions + 1
        wgt_sum = wgt_sum * (1 - 1/s) + line[i]
    i = i + 1
loop
if num_transitions  $\neq$  2 or  $\text{length}(\text{center\_region}) < \text{length}(\text{line})/2$  then
    return false (line is not a match)
else return true

```

Figure 7.16: Algorithm for the photometric test used in target verification. Each pixel intensity is compared with a weighted average of the previous pixels; if the difference is greater than a set threshold (for the dot regions, less than for the center region) then a transition to the next region is triggered. For a line to pass it must have exactly three regions, with the lighter interior region being at least as long as the two dark outer regions combined.

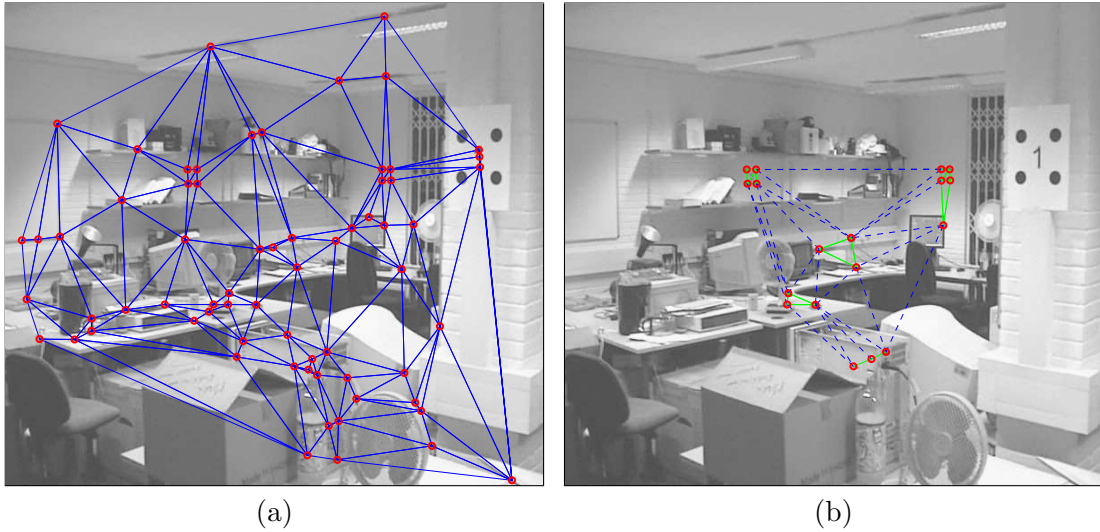


Figure 7.17: Target verification (a) All positive detections for a certain scale. The photometric test is performed on each Delaunay edge (shown in blue), and failed edges are deleted. Points that retain more than two edges are preserved. (b) Delaunay triangulation re-computed for those points that passed the photometric test. Edges that are too long for the current scale are removed (dashed blue lines), and once again only points with more than two edges are preserved. Groupings with fewer than four connected points cannot form a possible target, so these are also removed. The remaining groups are tested as described in §7.2.2.

mation is then applied to the fourth point, and the result plotted against an empirical distribution (see Figure 7.18). This empirical set of observations was generated from the ground truth coordinates of the training video sequences, and is distributed about the predicted fourth point (1,1) due to foreshortening in the image. The radius is determined by the amount of foreshortening, but is limited because the fiducials are

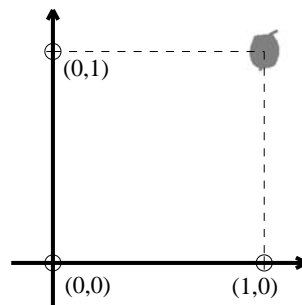


Figure 7.18: To check the geometry of a set of four prospective target dots, first compute the homography used to map three of the dots to the unit right triangle. The fourth dot is then plotted and compared with the empirical distribution for test target configurations (region shown in grey above). The small “tails” at the top and bottom of the distribution are due to foreshortened views within the training dataset.

only detected up to a foreshortening threshold. During testing, if the mapped point lies within the empirical region, then the group of points are considered to be a target match. The dot centres are then calculated by weighted centroid to achieve sub-pixel accuracy.

## 7.3 Evaluation

The intention of this work was to produce a fiducial detector which offered extremely high reliability in real-world problems. To evaluate this algorithm, a number of video sequences were captured with a DV camcorder and manually marked up to provide ground truth data. The sequences were chosen to include the high variability of input data under which the algorithm is expected to be used. It is important also to compare performance to a traditional “engineered” detector, and one such was implemented as described in the next section.

### 7.3.1 Engineered Detector

One important comparison for this work is how well it compares with traditional *ad hoc* approaches to fiducial detection. This section outlines a local implementation of such a system.

Each frame is converted to grayscale, binarized using adaptive thresholding as described in §7.3.2, and connected components used to identify continuous regions. The regions are split into scale bins based on area, and under or over-sized regions removed. Regions are then rejected if the ratio of the convex hull area and actual area is too low (region not entirely filled or boundary is not continually convex), or if they are too eccentric (if the ratio of axis lengths for an ellipse with the same second moments is too high).

### 7.3.2 Adaptive Thresholding

Many detection algorithms require a binary image, but changes in intensity make it difficult to select a global threshold value that will differentiate between foreground and background across an entire image. One solution to this is the adaptive thresholding proposed by Wellner (1993), which uses a weighted moving average of the previous pixels to set the threshold value for each pixel. Although this method is reasonably fast and produces adequate results, the averaging method and technique for initialization is somewhat *ad hoc*.

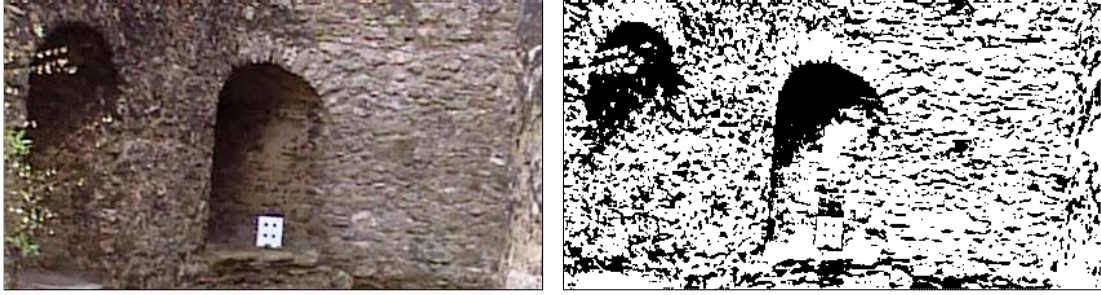


Figure 7.19: Adaptive thresholding an image of the Oxford city wall. Thresholding this natural scene produces 1269 regions from which to identify the target. In contrast, the cascading probe technique shown in Figure 7.9 returns a set of 32 possible target fiducials.

The threshold is defined as:

$$\frac{g(n)}{s} \frac{100 - T}{100} \quad (7.1)$$

$$g(n) = p(n) + \left(1 + \frac{1}{s}\right)g(n-1) \quad (7.2)$$

where:

- $p(n)$  is the pixel at position  $n$
- $g(n)$  is the weighted average of the previous pixels
- $s$  governs the number of previous pixels to include
- $T$  is the percentage to offset the threshold

This sort of moving average requires initialization, which can be set to the midrange value or the value of the first pixel. The rate at which the weighting falls off is set through the parameter  $s$ . For smaller values the effect of the initialization value drops off sooner.

### 7.3.3 Results

The fiducial detection system was tested on six video sequences containing indoor/outdoor lighting, motion blur and oblique camera angles. The reader is encouraged to view the video of detection results available from

<http://www.robots.ox.ac.uk/~dclaus/thesis/fiddetect.mpg>.

Ground truth target coordinates were manually recorded for each frame and compared with the results of three different detection systems: learnt classifier, learnt classifier with subwindow normalization, and the engineered detector described in §7.3.1. This provided a total of 3356 frames of video on which to test the algorithms. Table 7.1 lists the average number of positives found per frame. Each  $720 \times 576$  pixel frame requires 524,554 classifications; the fast classification stage returned just 0.33% of the

Sequence	True positives	Fast classifier	Full classifier	Normalized full classifier	Engineered detector
Church	4	5790	107	135	121
Lamp	4	560	23	30	220
Lounge	4	709	36	55	43
Bar	4	82	5	6	205
Multiple	7.3 <sup>†</sup>	2327	79	107	96
Library	4	1297	34	49	82
Average	-	1794	47	64	128

<sup>†</sup>The Multiple sequence contains between 1 and 3 targets per frame.

Table 7.1: Average number of positive fiducial classifications per frame. The full classifier is only applied to the positive results of the fast classifier. This cascade allows the learnt detector to run faster and return fewer false positives than the engineered detector.

subwindows as positive, allowing the classification system to process the average frame at four scales in 120 ms.

### Subwindow Normalization

Normalizing each subwindow to have mean 135 and variance 35 then classifying with normalized data increases the number of positives found. From Table 7.2 the false positive rate increased from 0.4% to 2.3% when normalization was incorporated. For the sequences studied here the normalized detector returned more false positives than even the engineered detector. The target verification stage must then examine a larger number of features; since this portion of the system is currently implemented in Matlab alone it causes the entire algorithm to run slower. This is added to the increased complexity of computing the normalization of each subwindow prior to classification. By contrast, appending normalized and rotated copies of the training data to the training set was found to increase the range of classification without significantly affecting the number of false positives or processing time. These modified copies of the training data were added prior to the condensing; after eliminating redundant data the set of positive/negative samples increased from 13/58 to 37/345. The success rate on the dimly lit bar sequence was increased from below 50% to 89.3% by including these normalized training samples.

### Detection Rate Comparisons

A perfect response from the fiducial classifier would return only 4 (or 7.3 for the multiple sequence) positives per frame. In practice it is necessary to accept additional

Sequence	Targets	Learnt detector		Normalized detector		Engineered detector	
		True	False	True	False	True	False
Church	300	98.3%	2.0%	99.3%	8.7%	46.3%	0.0%
Lamp	200	95.5%	0.5%	99.5%	3.0%	61.5%	0.0%
Lounge	400	98.8%	0.5%	96.5%	1.0%	96.5%	0.0%
Bar	975	89.3%	0.0%	91.3%	0.0%	65.7%	0.5%
Multiple	2100	95.2%	0.7%	93.5%	3.5%	83.0%	1.4%
Library	325	99.1%	0.6%	94.2%	0.0%	89.8%	5.2%
Summary	4300	94.7%	0.4%	94.0%	2.3%	77.3%	1.1%

Table 7.2: Success rate of target verification with various detectors. Normalizing each window prior to classification improves the success rate on some frames, but more false positive frames are introduced and the overall performance is worse. The engineered detector cannot achieve the same level of reliability as the learnt classifier.

(false) positives to keep the number of missed fiducials (false negatives) down. The photometric (§7.2.2) and geometric (§7.2.2) target tests can be used to reject false positives (results are given in Table 7.2) but there can be no recovery from a premature rejection of a true positive. In Table 7.2 we see that the number of false positives returned by the two stage learnt detector is the lowest of all the methods tested. The preprocessing normalization actually increases the number of false positives by bringing more background subwindows into the range where they appear as true fiducials. This requires additional processing time downstream to identify the true fiducials from a larger collection of candidates.

The performance of the overall target detector (fiducials plus target identification/verification) is listed in Table 7.2. The success rate for true detections is the percentage of the targets present in the video that were located by the algorithm. A correct identification requires all four dots be found and that their reported coordinates coincide with the ground truth location. The learnt detector on its own achieved the highest detection rate: 94.7%. The poor performance of the engineered detector results from missed detections on poorly lit or blurred frames. Altering the algorithm (by adjusting thresholds or adding image processing steps) to improve the performance on one scene was found to decrease the performance on some of the other scenes. The parameters of the engineered detector were manually tuned so that the reported results are the best observed on the collection of sequences. Evaluated on its own this is a reasonably good detector: the scenes used for testing here were purposely selected to demonstrate difficult aspects of fiducial detection.

The *reliability* of a detector depends on both the number of missed targets (false

negatives) and the mistaken identifications (false positives). We see in the summary line of Table 7.2 that the proposed detector without normalization provides the best results in both categories.

### Operating Range

The bounding criteria that determine the operating range of the target locating algorithm include: camera optical axis nearly perpendicular to target normal, motion blur, and small (far away) targets. As the camera is moved so that it shows the edge of the target, rather than head on, the target dots become ellipses rather than circles. If this is taken to the extreme the circles become lines and then vanish. The algorithm was found to detect targets at angles up to 75 degrees from the target normal. As the camera is panned or zoomed rapidly the image becomes blurred. A very fast blur would be a series of stripes, which can't be identified as a four dot target. A less severe case would enlarge the dots and the grey blurred edges could bias the location of the centres. The minimum dot that can be recognized is two by three pixels, which corresponds to a distance of approximately 10 m with a 50 mm lens.

## 7.4 Locating a Circle's Projected Centre

The fiducial detection algorithm described in this chapter provides an image location where a fiducial dot is located. This location is based on the pixel that was at the centre of the  $12 \times 12$  probe when the positive identification was made, and is therefore accurate to one pixel at best. Many applications require more precise localization than this.

The obvious choice is to threshold the image region containing the dot and then compute the weighted centre of the dot based on the pixel intensity values. This provides a reasonable measure of the dot's location, but foreshortening can cause the dot centre to shift in oblique views and other image effects can influence the thresholding. To overcome both of these potential drawbacks we collect four fiducials into a square target and compute the homography to warp their imaged locations into the corners of the unit square. Simultaneously computing the centre of four dots mitigates some of the problems with image noise, provides a grid for identifying the image regions where the dots should be, and allows the centre shift under perspective to be overcome. The following sections describe two approaches to computing this homography. The first is an efficient method that resolves most of the accuracy issues, while the second is a nonlinear optimization that precisely computes the dot positions.

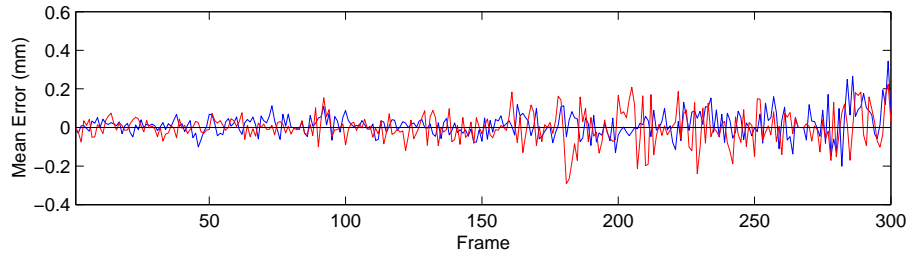


Figure 7.20: Errors in computed target coordinates for the church sequence. The horizontal error is shown in red, and the vertical in blue.

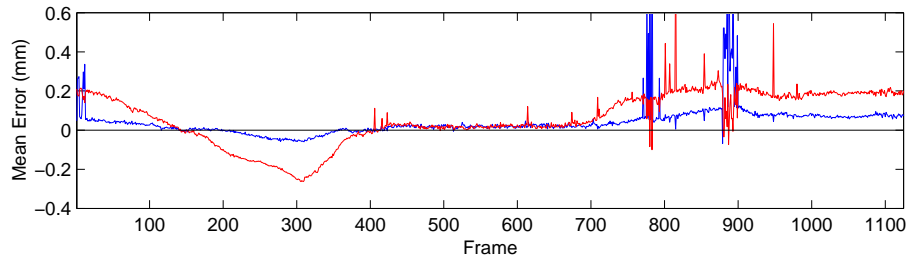


Figure 7.21: Errors in computed target coordinates for the “test” sequence (see Figure 7.1). The camera path is roughly one meter from the target, and begins with an oblique view on the right, progresses to the left and then returns to the right. The bias in error is due to the mismatch of the circle centres under the projective transformation experienced in oblique camera angles at such close range. The large oscillations around Frames 800-900 denote failed detections caused by camera angles outside the operating range.

#### 7.4.1 Homography from target coordinates

The registration accuracy can be evaluated by computing a planar homography from the screen coordinates (or image plane) to the plane that the target is in (e.g. the wall that it is taped to). Such a homography can be represented by a  $3 \times 3$  matrix, and computed from four corresponding point locations in both planes (such as the target centres).

Once the homography has been computed it can be used to warp the target into a plane that is fronto-parallel with the image plane. The accuracy of the homography (and by extension, the computed target coordinates) can be evaluated by measuring the offsets of the dot centres in the fronto-parallel image. Figures 7.20 and 7.21 show the error in each direction for the church sequence and test sequences respectively. The error is the mean between the four dots, so the absolute errors for specific dots could be higher. For these two figures the blue curve denotes vertical error and the red denotes the horizontal. The test sequence exhibits a cyclical bias caused by the circle centre mismatch under perspective transformation (refer to §7.1.1). The test sequence was



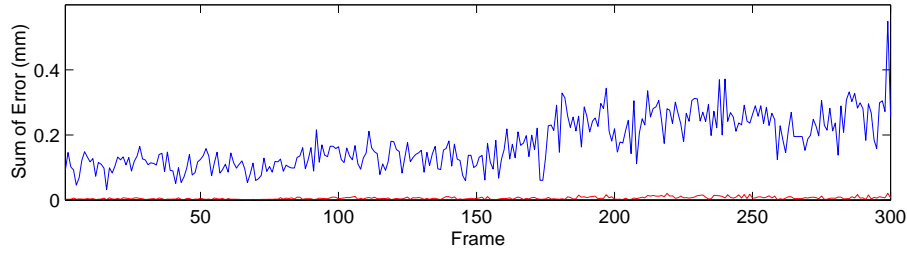


Figure 7.22: Single pass and double-pass error comparison for the “church” sequence (see Figure 7.1). The double pass method (red curve) computes the dot centres from the fronto-parallel target image obtained from the single pass homography. Note that this plot displays the sum of the error distance of all four dots in the target.

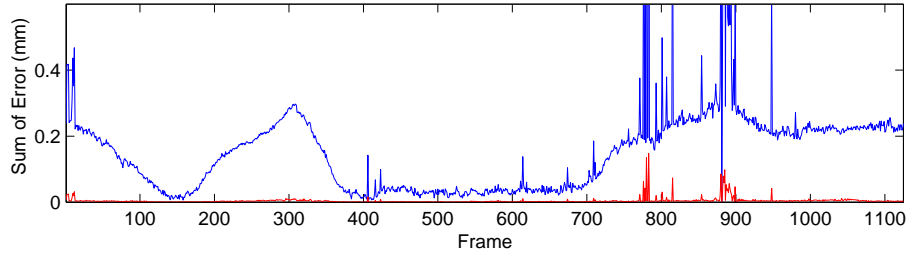


Figure 7.23: Single pass and double-pass error comparison for the “test” sequence (see Figure 7.1). Note that the second pass (red curve) has corrected the centre mismatch error observed in Figure 7.21.

filmed very close up, and alternates between very oblique views on the right, left and finally the right side of the target. Note that the horizontal error (red curve) oscillates accordingly.

The offset in computed dot centres is due to foreshortening, but the error is small enough to be insignificant for most applications. Where this level of error is relevant (computing camera pose, for example), accurate coordinates can be computed, in a second pass, from the fronto-parallel image. This is done by first computing approximate dot centre locations from the original image. These locations are used to compute the homography that will map the four dots to the unit square; this homography is then used to warp the image. This warped image is approximately a fronto-parallel view. The dot centres are then detected a second time in this newly warped image. Again, the homography to warp the detected dot centre locations onto the unit square is computed. This second homography should induce only a slight transformation as it merely corrects for the error in the first dot detection. The overall homography is then produced by concatenating the homographies computed in the first and second stages. Figures 7.22 and 7.23 compare the error between the target coordinates computed directly and the error when a second pass is used. These figures show the sum of the

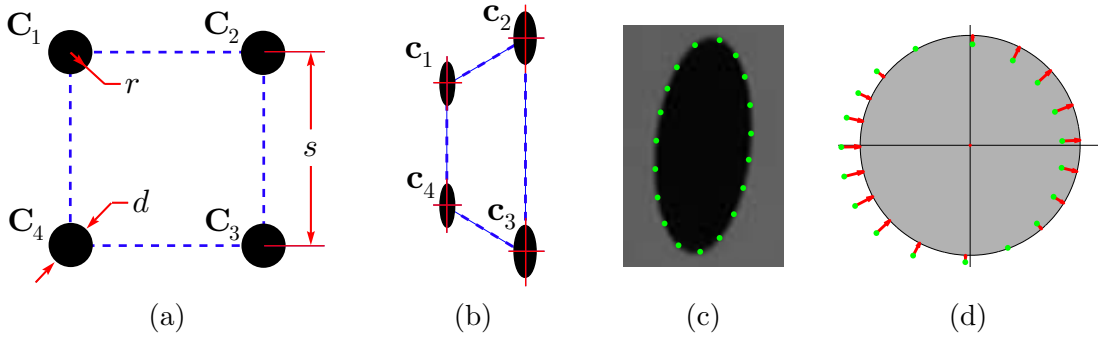


Figure 7.24: Finding ellipse centres through homography optimization. (a) Target consisting of four circular dots with known spacing and diameter. (b) A perspective image of the target. (c) The true centre of a circle under perspective projection is found by transforming the detected edgels (green points) into the fronto-parallel plane (right view). (d) A non-linear optimization using the errors relative to the true circle (red arrows) then yields the refined transformation. The original centre is found by back-projecting the true circle centre into the image using this refined transformation.

error distance for each of the four dots.

The inverse homography can also be used to map an image into the same perspective as the target. This has the effect of “pasting the image on the wall”. Each frame will have a slightly different homography based on the computed dot centres for that view. Any inaccuracies in the computed centres will cause the image to shift slightly on the wall. This motion is noticeable and provides a qualitative measure of the target location algorithm’s performance.

#### 7.4.2 Homography Calculation from Four Circles

This section describes a non-linear method for finding the centres of four coplanar circles by optimizing a planar homography. The four-fiducial target used throughout this chapter and thesis is an example of such a set of coplanar circles. The centres are computed by optimizing the homography that warps the detected edgels of the imaged circles onto the perimeter of the true dots. This approach yields the precise centres (irrespective of parallax) and the homography to warp the target fronto-parallel. The homography can be used to initialize a pose computation (refer to §6.2.2), but the precise centres are required for any further processing based on the fiducials. The optimization presented here is a continuation of the double-pass approach described in the previous section; here we use nonlinear optimization to iterate until convergence. A fiducial pattern consisting of a set of four circular dots is sufficient to constrain the homography (up to a forward/backward ambiguity that is trivially resolved for a camera with less than  $180^\circ$  field of view).

Our task is two-fold: find the homography that will warp the imaged dots fronto-parallel, and locate the image projections of the dot centres. The problem is illustrated in Figure 7.24. We begin with a planar target of known dimensions: each dot has diameter  $d$  and is spaced a distance  $s$  from its neighbours. This analysis assumes that the fiducials have been detected, so their centres are already approximately known. These initial estimates of the centres will be denoted  $\mathbf{c}'_{1\dots 4}$ . As discussed in §2.4.4, the centre of a circle is not projectively invariant (see Figure 2.14), so the weighted centroid method for computing the centre does not provide the true centre. Using all four dots simultaneously, and basing the alignment on the perimeter rather than an assumed centre, we are able to overcome this projective problem.

For each dot, a set of edgels  $\mathbf{e}$  are found by extracting a region containing the dot and performing Canny edge detection (Canny 1986) (Figure 7.24c). These edgels are transformed into the fronto-parallel plane via an initial homography  $\mathbf{H}_0$  computed from the estimated dot centres in the image and the true dot centres in the fronto-parallel view (denoted by  $\mathbf{C}_{1\dots 4}$ ) as follows

$$[\mathbf{C}_1 | \dots | \mathbf{C}_4] = \mathbf{H}_0 [\mathbf{c}'_1 | \dots | \mathbf{c}'_4]. \quad (7.3)$$

This initial homography is fed to the optimization, along with the complete set of edgels and dimensions of the target. The set of edgels detected for the  $j^{th}$  dot  $\mathcal{E}_j = \{\mathbf{e}_j^1 \dots \mathbf{e}_j^{m_j}\}$  are individually transformed into the fronto-parallel view by  $\mathbf{E}' = \mathbf{H}\mathbf{e}$ , where  $\mathbf{H}$  is the homography being optimized. The error for each edgel is computed by subtracting the dot radius  $r$ :

$$\epsilon(\mathbf{H}) = \sum_{j=1}^4 \sum_{k=1}^{m_j} \left\| \sqrt{\|\mathbf{H}\mathbf{e}_j^k - \mathbf{C}_j\|^2} - r \right\|^2. \quad (7.4)$$

This error is not measured in the image plane, but rather in a world coordinate frame that is approximately aligned with the target plane. This is not treating the edge detection errors exactly correctly, but it yields an approximation that is reasonable.

The ellipse centres in the image are the back-projection of the known fronto-parallel centres using the optimized homography

$$\mathbf{c} = \mathbf{H}^{-1}\mathbf{C}. \quad (7.5)$$

## 7.5 Target Identification

The task of identifying which target has been found is again handled through a nearest neighbour (sum of squared differences) match with a set of trained patterns (Figure 7.25). These patterns are easily recorded using the detection algorithm itself. The

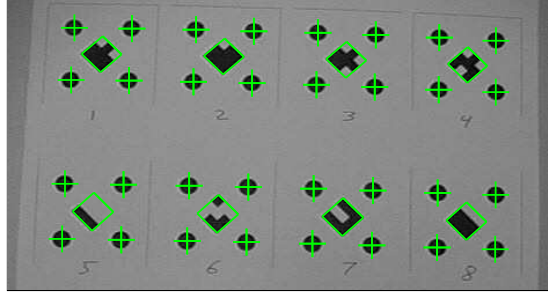


Figure 7.25: Multiple fiducials printed out on a single sheet of paper. The white overlay shows the detected marker positions and the squares bounding the marker identification codes. The target detection and assignment of the identification patterns is performed automatically from a single frame of video.

target dots are automatically located, and the registration pattern is recorded for each target found. In our tests, the patterns consist of non-symmetric binary patterns in a  $3 \times 3$  grid, but in general any pattern could be used. A normalization step is used to handle differences in lighting conditions. A higher degree of robustness can be obtained by recording training images under a wide variety of lighting conditions, however in this case we want to minimize the amount of storage and computational overhead associated with the target recognition. Therefore a single image is recorded at the expense of some robustness.

## 7.6 Fiducial Localization Trials

This section describes an experiment to measure the precision of fiducial localization. We recorded a video of fiducials moving along a circular path and measure the deviation in the computed location. The circular paths were generated by placing a four-fiducial (dot) target on a turntable and filming it from directly above. As shown in the sample frame of Figure 7.26, the target was offset from the centre of the turntable so that each fiducial would trace out a different radius. The camera's viewing axis was aligned with the turntable's axis of rotation to produce a nearly circular rather than an elliptical path. The turntable was approximately centred in the camera's field of view so that any radial lens distortion effects would be mitigated. Lens distortion was corrected (using the division model described in §3.1) as a precautionary measure.

A 524 frame sequence was recorded, totalling 3.3 complete rotations. The ground truth paths were determined by fitting a circle to the path of each fiducial. This was done after the pixel aspect ratio was computed (from the axis ratio of the elliptical path) and factored out. The radial and tangential errors were computed by comparing each fiducial dot localization result with the location predicted by the circular path.

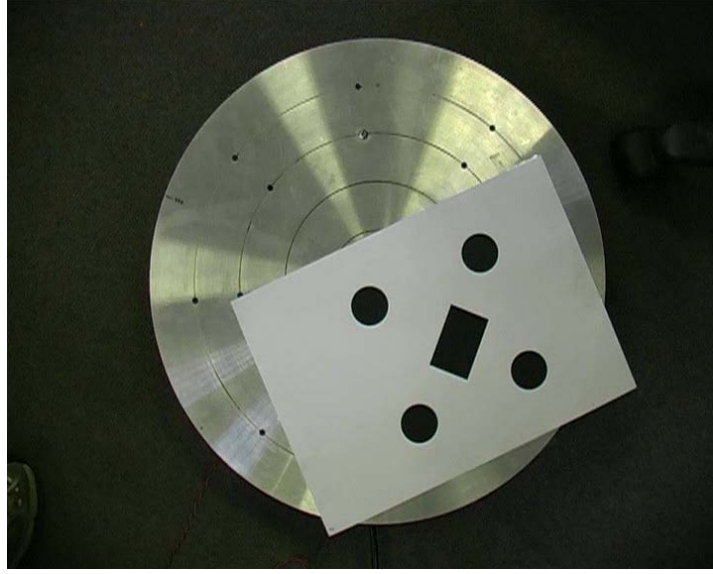


Figure 7.26: A target consisting of four fiducials was placed on a turntable to measure the fiducial localization accuracy. Each dot traces out a circle, and the position error for the dots detected in each frame is measured relative to this known path.

	Radius	Radial Errors			Tangential Errors		
		Mean	Max	Std. Dev	Mean	Max	Std. Dev
Dot 1	187.0	0.04	0.18	0.03	0.11	0.46	0.09
Dot 2	151.8	0.04	0.25	0.04	0.11	0.44	0.09
Dot 3	113.2	0.03	0.15	0.03	0.12	0.52	0.09
Dot 4	28.0	0.02	0.08	0.02	0.15	0.91	0.13

Table 7.3: Errors measured relative to the circular path for each fiducial dot. The computed values are averaged over all 524 frames in the sequence; all measurements are in pixels. The tangential errors are elevated due to sampling rate issues (refer to text). The radial errors indicate that the proposed method can provide very precise marker localization.

The results are tabulated in Table 7.3 and plotted in Figure 7.27. Note that the plotted errors denote the radial error only and have been amplified by a factor of 100 so that they are visible. Observe that the errors are predominantly noise; there is little or no systematic bias which would indicate a problem with the fiducial localization algorithm.

From Table 7.3 we see that the mean radial errors are below five hundredths of a pixel. This level of precision is more than adequate for most computer vision tasks. For comparison, the corners of the target identification pattern were tracked in *boujou* (which uses a Harris corner type of detector) and the mean radial error was 0.07 pixels.

The tangential errors are slightly higher than the radial ones due to the difference in operation frequency between the turntable and the camera. The turntable uses a

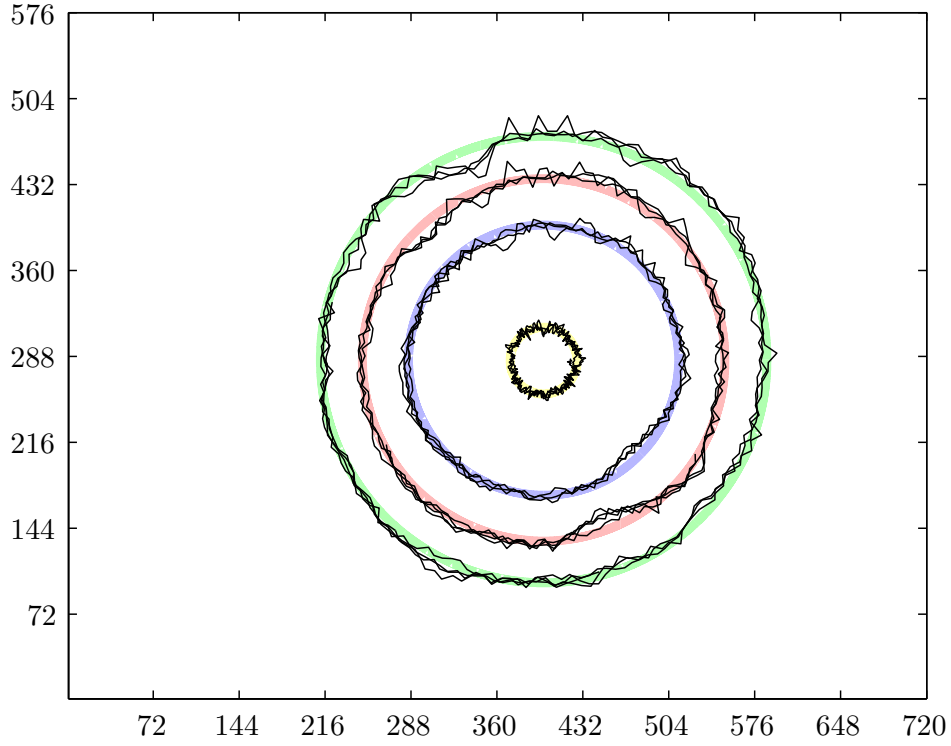


Figure 7.27: Fiducial localization results with radial errors amplified by a factor of 100. By exaggerating the error we see that there is little systematic error; the position errors reported in Table 7.3 are predominantly due to noise rather than bias in the localization scheme.

stepper motor to advance in discrete intervals. Because this timing does not match the frame rate of the camera the turntable will not rotate an equal amount between frames. The ground truth data assumes a constant angular velocity; any deviation from this is seen as tangential position error. As a result, the radial error is more representative of the error level that can be expected.

In summary, the proposed fiducial detection method can provide localization accuracy on the order of  $\frac{1}{20^{th}}$  of a pixel, which is half the error level from a standard feature detector for this test sequence.

## 7.7 Summary

This chapter has presented a fiducial detection scheme that uses example based learning to handle lighting variation, scale changes and motion blur. Engineering a fiducial detector by assembling image operations produces a narrowly focused algorithm that is difficult to adapt to new scene challenges. By contrast, an exemplar based method

is retrained simply by including relevant image samples.

The presented method tests a  $12 \times 12$  pixel window surrounding very pixel in the image; a two stage classifier is required to increase its efficiency. Once fiducial candidates have been identified a series of *ad-hoc* tests are run to identify sets of hits that constitute targets.

Evaluation on a varied collection of image video sequences demonstrated that the learnt detector returns fewer false positives than an engineered one, while simultaneously identifying fiducials over a wider operating range. Adding image processing steps such as normalization to the learnt detector actually decreases the performance because more false positives are returned (0.4% false positives without normalization versus 2.3% with the extra processing step).

Testing on a collection of videos containing fiducial targets in different scenes showed that the learnt detector outperforms the engineered detector in terms of percentage of targets found and number of objects incorrectly identified as targets. Adjusting the engineered detector to improve its performance on one scene caused a corresponding drop in performance on another. By contrast, the exemplar based algorithm can handle new scene challenges simply by including relevant example images. There was little or no observed increase in false detections from using this expanded training set. An image normalization step prior to running the classification algorithm was also tested and found to degrade the overall performance. Including relevant training data is a more reliable and efficient method for improving the performance of the detector than adding image processing steps to deal with specific input scenarios.

Any discussion of finding centres of circles where oblique views are possible should address centre shift. This chapter measured the reprojection error in the plane for compensated centres and for a single step correction. Although this foreshortening induced error is generally small enough to be insignificant the single stage correction drastically reduces it.

An iterative optimization technique is also presented for precisely computing both the centres of four circles and the homography that warps them to a unit square. This technique definitively solves the foreshortening issue, but is more important where extreme accuracy in target location is required.

Finally, turntable based fiducial detection trials indicate that the proposed dot detector computes the location of the dot centres to within 0.05 pixels of the true location. This is without the homography optimization.

In summary, this exemplar based fiducial detection algorithm is fast enough for real-time implementation, robust enough for use in a wide variety of real life environments,

and precise enough for applications requiring accurate localization. This precision is crucial to the photometric stereo application described in the next chapter; the accurate image registration and camera localization depend upon this type of fiducial detection.



## Chapter 8

# Photometric Stereo Application

This final chapter describes an application of the various metrology strategies presented in the preceding chapters. This particular system relies on accurate camera localization, fiducial detection and distortion correction to produce high resolution 3D texture models of real world objects.

The capture of real-world data is becoming increasingly important for accurate computer modeling of real-world objects. Three dimensional object scanners are used as part of the rapid prototyping process — an artist’s model can be scanned then directly converted into a CAD object and even printed in 3D. The film and media industries require texture maps for accurate rendering of synthetic objects. In the past these textures have either been drawn manually or generated procedurally based on the behavior of that particular class of materials. A simple method for creating these textures by recording real-world surfaces has the potential to significantly alter computer graphics rendering. A third application for real-world object capture is the recording of historical artifacts. Museums could potentially showcase a larger portion of their collections through media based presentation, while researchers could make use of the data sharing, searching and analysis tools available for such digital data.

Just as there are many uses for digital models of real-world objects, there are also many different technologies available for capturing that data. Here we will focus on photometric stereo: using controlled lighting to illuminate an object from several viewpoints and then inverting the illumination equations to recover the object’s shape. This photometric stereo application was a joint project with James Paterson. It was presented at Eurographics 2005 (Paterson et al. 2005) and comprises a portion of UK patent application 0608841.3. This thesis describes only the portion of the project which was my work.

The image of a 3D object depends upon the object’s shape, its reflectance properties and how it is lit. By controlling the lighting and making assumptions about the

reflectance properties it is possible to compute the shape. This is the basis for photometric stereo (Woodham 1980), which is usually recorded with a single camera in a fixed position, multiple lights in known locations and the assumption that the object has Lambertian reflectance properties. If the camera and object are fixed relative to one another then a given pixel in any of the images will always correspond to the same point on the object. Under the Lambertian assumption, the observed brightness at that pixel is given by

$$I = \mathbf{n} \cdot \mathbf{L} \quad (8.1)$$

where  $\mathbf{n}$  is the surface normal and  $\mathbf{L}$  is the light direction. Collecting images from at least three distinct light directions allows  $\mathbf{n}$  to be solved for at every pixel.

This simple explanation is complicated by (among other things) the fact that very few materials are purely Lambertian. A discussion of other reflectance models and the details of a photometric stereo system are beyond the scope of this thesis, so the interested reader is referred to Paterson (2005) for details on these aspects of the project.

There are three major steps to our reconstruction process: 1) transform all input images so we have pixel-for-pixel alignment, 2) use photometric stereo techniques to compute the surface normal and albedo at each pixel, and 3) integrate the surface normals to recover the 3D shape of the test sample. Here we will only describe the first and third steps. The transformation to align each pixel is facilitated by a fiducial based target. Both the camera and light position are recovered through nonlinear optimization.

## 8.1 Motivation

The novel approach taken by our photometric stereo system is to permit the camera to move relative to the (approximately planar) object being sampled. A target with fiducials is attached to the object and the camera localization techniques described in this thesis are used to compute the viewing position for each image. A known camera position permits the re-rendering of the simulated image that would be recorded by a camera directly in front of the object (fronto-parallel). There are two advantages to the moving camera photometric stereo setup which make the extra work required to re-render aligned fronto-parallel images worthwhile. The first is that it simplifies the lighting apparatus; we use a single light source that is rigidly attached to the camera. As the camera is moved (rotated) between shots the light is in a different position relative to the object for each image. Thus there is only one light whose position must be determined during calibration. The second advantage to a moving camera is

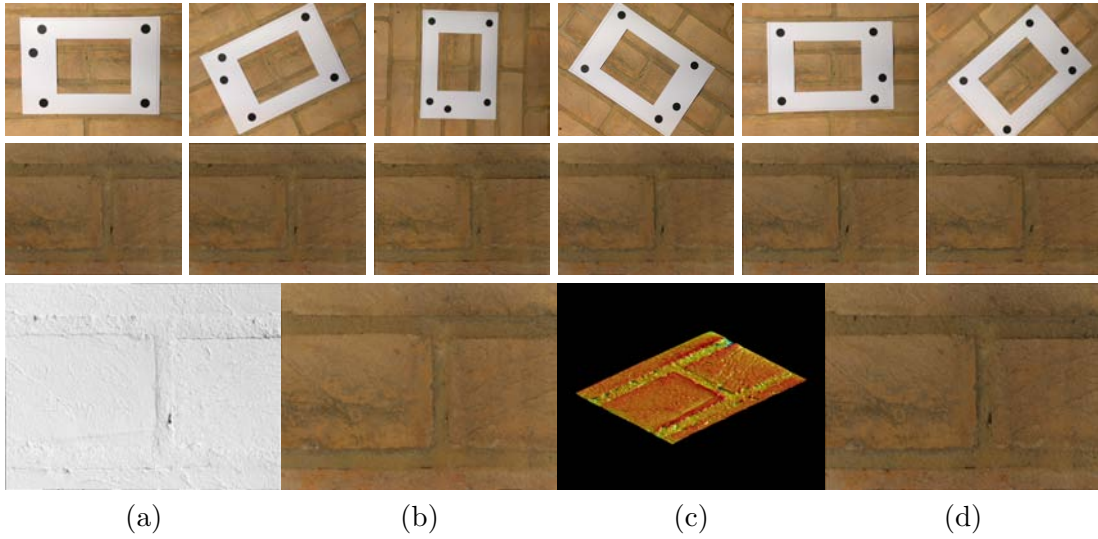


Figure 8.1: Overview of the photometric stereo 3D reconstruction process. *Top row:* Input images showing the calibration target stuck to an exterior wall. These images are captured using a camera with flash. *Second row:* Images rectified to front-parallel view. This chapter describes the processes required to obtain these views. *Final row:* Photometric stereo reconstruction results shown as (a) the computed normal map  $\mathbf{n}(x, y)$ , (b) the recovered albedo, a part of  $\mathbf{a}(x, y)$ , (c) 3D geometry depicted by a colour-coded height field  $z(x, y)$ , and (d) a resynthesized image.

that views which are not fronto-parallel permit the computation of an object's depth. Traditional photometric stereo with a single fixed camera solves for the surface normal at every point on the object, but there is still a depth ambiguity that cannot be resolved from a single viewpoint. By allowing the camera to move we introduce parallax that can be used to establish a depth scale. This scale is then used to constrain the surface integration so that an accurate representation of a 3D surface can be recovered.

This chapter presents an application of the camera localization and fiducial detection techniques detailed in Chapters 6 & 7. We shall describe a photometric stereo system that uses simple, standard equipment to produce high resolution models of 3D geometry and reflectance. Precise camera localization under varying light and camera directions permits a single camera and flash to provide the input images required for traditional photometric stereo techniques. Careful engineering of all aspects of the processing pipeline (in particular compensation for flash attenuation) yields very high resolution surface geometry.

## 8.2 Overview

Photometric stereo requires images with pixel for pixel correspondence, but different (known) lighting directions. Traditionally the correspondence is achieved by using a

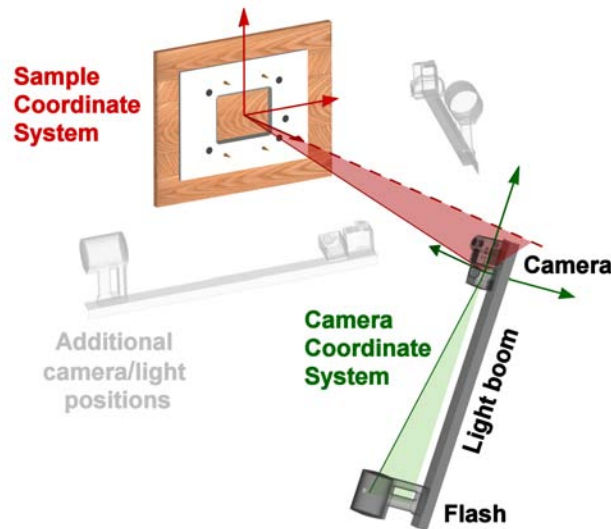


Figure 8.2: Photometric stereo system schematic. The camera is moved relative to the surface to be sampled, generating multiple views of the surface with different lighting directions. Photometric stereo is then used to recover surface geometry directly from the camera image. The flash is a conventional camera flash, and does not project structured light.

single stationary camera while several lights are permanently mounted in some sort of frame. Then the light and camera positions are measured relative to the object in a global 3D coordinate system. By contrast, the system detailed here employs a single handheld camera with an attached flash (see Figures 8.2 and 8.3). This rig is free to move arbitrarily around the sample. A specially designed target plate is used to calibrate the light position relative to the camera, and then to compute the camera position relative to the sample. Once the geometry is known, it is possible to establish pixel for pixel correspondences between all the images. Standard photometric stereo techniques are then used to compute surface normals. Due to the precision built into the system, these normal measurements are largely free from bias. The non-static viewpoint yields a depth constraint which can be applied while integrating the surface normals to produce a reasonable 3D surface. For samples which are far from planar this surface can be used to incorporate parallax correction in the task of producing fronto-parallel views. This entire process is outlined in Figure 8.4 and detailed in the next few sections.

The system (shown in Figure 8.2) uses a five-fiducial target to compute camera pose and thus rectify all images. A commercial flash gun is used to provide illumination which renders the ambient lighting insignificant. Four cones on the target cast shadows which are used to compute the light position during calibration.

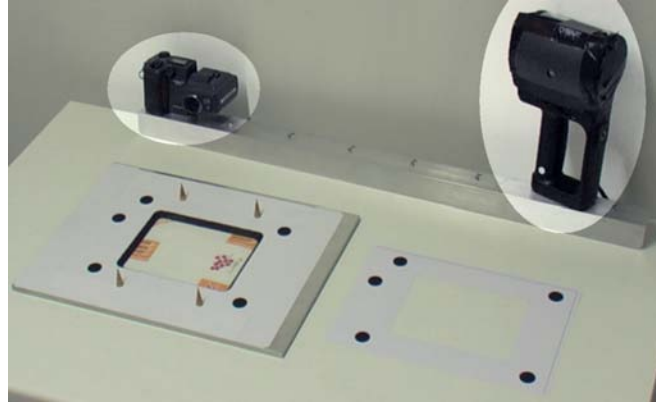


Figure 8.3: The apparatus used for the photometric stereo image capture. The camera and flash (highlighted) are rigidly attached to a boom so that they move together. Two types of targets were used: one is attached to a metal plate with cones, and the other is merely a laser printed sheet of paper with a window cut out of the middle. The cones are used to solve for the light position relative to the camera (refer to text).

### 8.2.1 Lens Distortion Correction

Prior to performing any other processing, the input images were corrected for lens distortion. For this sample application, the lens distortion model proposed by Heikkilä as implemented in the MATLAB Camera Calibration Toolbox ((Bouguet 2003)) was used with only the second order radial term.<sup>1</sup> This model is sufficient to handle the low levels of distortion present in these images taken at moderate levels of zoom. The code is freely available and easy to use; other researchers can therefore readily compare the results.

Focal length:	$f_x$	7698.681	$\pm 17.899$	pixels
	$f_y$	7685.665	$\pm 16.971$	pixels
Image dimensions:	$width$	2048	pixels	
	$height$	1536	pixels	
Principal point:	$c_x$	1032.262	$\pm 2.194$	pixels
	$c_y$	778.622	$\pm 2.828$	pixels
Distortion:	$k$	0.970	$\pm 0.014$	
Pixel error:	$e_x$	0.307		
	$e_y$	0.239		

Table 8.1: Calibration values for radial distortion correction. These sample values were output by the MATLAB Camera Calibration Toolbox; note that only the second order radial distortion term and principal point coefficients were used.

<sup>1</sup>The rational function model for lens distortion was not used in this work because it had not yet been fully formulated.

---

**Offline calibration**

---

1. Radial distortion
  - Model the radial lens distortion for this specific camera/lens combination
2. Light position
  - Compute the light position relative to the camera using shadows cast by the cones on the target plate. This can be done online if the target plate with cones is used rather than the paper target.

---

**Online operation**

---

1. Camera localization
    - detect dots and use a homography optimization §7.4.2 to locate centres
    - (optional: if using target plate for online light position calibration) detect cone shadow tips
    - optimize for camera pose (and optionally, light displacement)
  2. Light attenuation
    - sample the intensity on the white target periphery and use this to model the light attenuation
    - use the known light and camera positions to correct for light fall-off
  3. Warp images
    - use the known camera pose to warp a fronto-parallel image (this is a planar warping as the height  $z = 0$ )
  4. Photometric stereo
    - compute the surface normal at each pixel
  5. Surface integration
    - triangulate several user-selected point correspondences across multiple views to use as constraints
    - integrate the surface normals to create a 3D surface, subject to the above constraints
  6. Parallax correction
    - for highly non-planar samples the height map is used to correct for parallax and occlusion when warping the fronto-parallel views
  7. Iterate steps 4–6 as necessary
- 

Figure 8.4: Photometric stereo system overview. A single camera with an attached flash is free to undergo arbitrary motion relative to the sample due to precise camera localization based on a target plate of known dimensions.

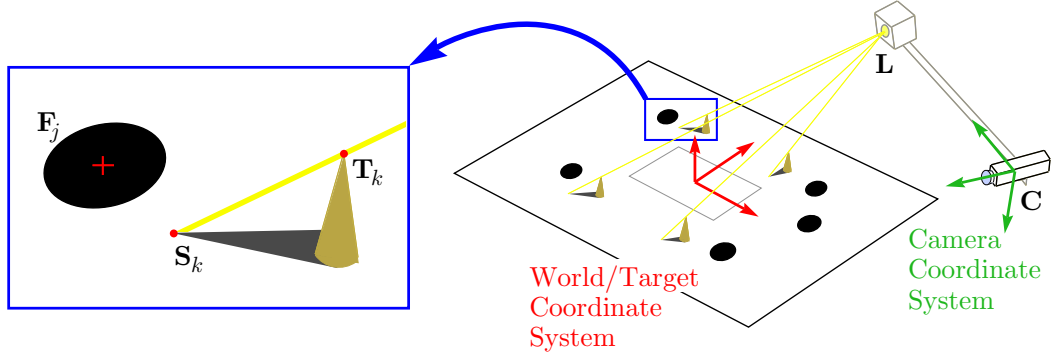


Figure 8.5: Light position calibration geometry. The position of the light relative to the camera is established using the images of the cone shadows and fiducial dots on the target plate.

### 8.2.2 Light position

The second offline task is to calibrate the light/camera boom. We measure the 3D position of the light relative to the camera centre using the target plate with attached cones shown in Figure 8.5. The light is rigidly attached to the camera, so once its position in camera coordinates is known we are only required to compute the camera position for each input image. Since the camera pose can be recovered from the planar dots alone the plate target with cones is only needed for the initial calibration (though it is also useful for keeping the fiducial target planar while sampling non-planar objects). The general calibration procedure is to locate the camera using the fiducials and then to determine the light position by intersecting the rays from each cone tip and shadow. Both the light and camera positions are thus defined relative to the known target geometry, and their orientation relative to one another is easily computed.

The plate consists of a set of fiducials  $\mathbf{F}_{1...5}$  and cone tips  $\mathbf{T}_{1...4}$  with known coordinates in the world coordinate system. The light and camera are rigidly attached to the boom, and then a set of  $N$  images are recorded from different (arbitrary) camera/light positions. In each image  $I_i$  we observe cone tip shadows at pixel locations  $\mathbf{s}_{1...4}$ . For light position  $\mathbf{L}^i$  (in world coordinates) the preimage of shadow tip  $\mathbf{s}_k^i$  is the intersection of the line  $\lambda \mathbf{T}_k^i + (1 - \lambda) \mathbf{L}^i$  with the world  $XY$ -plane. This plane corresponds to the front face of the target plate,  $\mathbf{n} = [0 \ 0 \ 1]^\top$ , and yields the constraint

$$\mathbf{n}^\top (\lambda \mathbf{T}_k^i + (1 - \lambda) \mathbf{L}^i) = \lambda \mathbf{n}^\top (\mathbf{T}_k^i - \mathbf{L}^i) + \mathbf{n}^\top \mathbf{L}^i = 0 \quad (8.2)$$

$$\Rightarrow \lambda = \frac{\mathbf{n}^\top \mathbf{L}^i}{\mathbf{n}^\top (\mathbf{L}^i - \mathbf{T}_k^i)}. \quad (8.3)$$

The world position of the shadow tip can therefore be expressed as

$$\mathbf{S}_k^i = \left( \frac{\mathbf{n}^\top \mathbf{L}^i}{\mathbf{n}^\top (\mathbf{L}^i - \mathbf{T}_k^i)} \right) \mathbf{T}_k^i + \left( 1 - \frac{\mathbf{n}^\top \mathbf{L}^i}{\mathbf{n}^\top (\mathbf{L}^i - \mathbf{T}_k^i)} \right) \mathbf{L}^i.$$

The world coordinates of the shadow tips are given by

$$[\mathbf{S}_1 | \dots | \mathbf{S}_4] = \mathbf{H}[\mathbf{s}_1 | \dots | \mathbf{s}_4] \quad (8.4)$$

where the planar homography  $\mathbf{H}$  that maps the image plane onto the world plane is computed as in §7.4.2 from the observed fiducial centres  $\mathbf{f}_{1\dots 5}$ . The light position for each view can be computed as the closest point to all four lines  $\ell_k^i = \gamma_k \mathbf{T}_k^i + (1 - \gamma_k) \mathbf{S}_k^i$  where  $k = 1 \dots 4$ . This method provides the 3D light positions for each view, and is used to initialize the light position.

For each image, we seek the rotation  $\mathbf{R}^i$  and translation  $\mathbf{t}^i$  that will transform world reference points into the coordinate frame centred in the camera. This transformation can also be used to express the light position  $\mathbf{L}^i$  for each view as a constant displacement from the camera

$$\mathbf{L}_c = \mathbf{R}^i \mathbf{L}^i + \mathbf{t}^i. \quad (8.5)$$

This constant light position can be included in the nonlinear optimization for camera pose, and helps to constrain the overall solution.

The camera and light positions are then simultaneously optimized over all  $N$  images by minimizing

$$\begin{aligned} \epsilon(\mathbf{R}_1, \mathbf{t}_1, \mathbf{L}_1, \dots, \mathbf{R}_N, \mathbf{t}_N, \mathbf{L}_N) &= \sum_{i=1}^N \underbrace{\sum_{j=1}^5 \left\| \mathbf{f}_j^i - \pi \left( \mathbf{H}^{-1} \{ \mathbf{R}^i \mathbf{F}_j^i + \mathbf{t}^i \} \right) \right\|^2}_{\text{fiducial residuals}} \\ &\quad + \underbrace{\sum_{k=1}^4 \left\| \mathbf{s}_k^i - \pi \left( \mathbf{H}^{-1} \{ \mathbf{R}^i \mathbf{S}_k^i + \mathbf{t}^i \} \right) \right\|^2}_{\text{shadow residuals}}. \end{aligned} \quad (8.6)$$

Recall that the operator  $\pi$  computes non-homogeneous coordinates as described in §1.3. The inverse homography  $\mathbf{H}^{-1}$  is used to express world points in image coordinates, which can then be compared to the measured location (either of a fiducial centre or a shadow tip) to produce individual components of the error function.

By substituting (8.4) and incorporating  $\mathbf{L}^i = \mathbf{R}^{i\top} (\mathbf{L}_c - \mathbf{t}^i)$  we obtain an error function that depends only on the camera pose for each frame and the overall camera/light



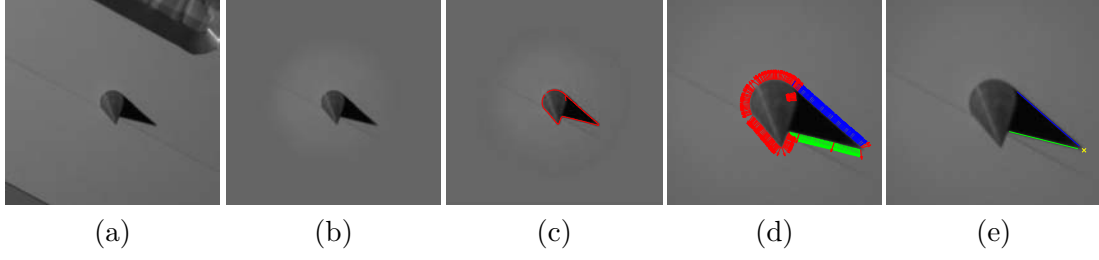


Figure 8.6: Cone tip shadow detection. (a) Region of the cone extracted from the input image (b) Background masked off (c) Detected edgels (d) Edgels classified into two line segments based on edge orientation (e) Shadow tip found by intersecting lines fitted to detected edgels.

displacement.

$$\begin{aligned}
& \epsilon(\mathbf{R}_1, \mathbf{t}_1, \dots, \mathbf{R}_N, \mathbf{t}_N, \mathbf{L}_c) \\
&= \sum_{i=1}^N \sum_{j=1}^5 \left\| \mathbf{f}_j^i - \pi \left( \mathbf{H}^{-1} \{ \mathbf{R}^i \mathbf{F}_j^i + \mathbf{t}^i \} \right) \right\|^2 \\
&\quad + \sum_{k=1}^4 \left\| \mathbf{s}_k^i - \pi \left( \mathbf{H}^{-1} \left\{ \frac{\mathbf{R}^i \mathbf{n}^\top \mathbf{R}^{i\top} (\mathbf{L}_c - \mathbf{t}^i) [\mathbf{T}_k^i - \mathbf{R}^{i\top} (\mathbf{L}_c - \mathbf{t}^i)]}{\mathbf{n}^\top [\mathbf{R}^{i\top} (\mathbf{L}_c - \mathbf{t}^i) - \mathbf{T}_k^i]} + \mathbf{L}_c \right\} \right) \right\|^2.
\end{aligned} \tag{8.7}$$

### 8.2.3 Cone Detection

The tip of each cone's shadow ( $\mathbf{s}_k^i$ ) is located automatically using edge detection and line fitting as illustrated in Figure 8.6. First, the image region containing a cone and its shadow (this region is known because the fiducial locations gave the homography to transform the reference plane into the image) are extracted and converted to greyscale. Then the background is masked off using a circular mask of radius equal to the cone height. This ensures that the shadow tip will fall within the mask window for light angles up to  $45^\circ$  from the target plane normal. The light position can still be recovered for angles greater than this design limit; the tip is then merely outside the region used for edge detection. One could enlarge the region, but this introduces additional background clutter which must then be differentiated from the shadow and rejected. The masked region is set to the mean image region intensity, and a smooth fade to transparent is used so that artificial edges are not introduced (Figure 8.6b).

Edge detection and line fitting is performed as an iterative process whose stopping condition is a pair of lines supported by some percentage of the detected edges. This follows from the assumption that the shadow will give rise to the strongest linear edges in the masked region. Sub-pixel Canny edge detection (Canny 1986) with smoothing  $\sigma$  and edge threshold  $t$  is used to identify the position and orientation of edgels within

the masked greyscale image:

$$E(\sigma, t) = \{\mathbf{e}_1 \dots \mathbf{e}_m\} \quad \text{where } \mathbf{e}_j = [e_x \ e_y \ e_\phi]^\top. \quad (8.8)$$

All detected edgels are then classified based on their orientation. The two dominant directions are identified by sorting all edgels into bins and selecting the two with the highest frequency. All remaining edgels are classified by their proximity to either of these directions, with an absolute distance cutoff. Figure 8.6d shows the classified edgels with their orientations. If the majority of the edgels are classified as the dominant lines, then these lines are intersected to compute the tip of the shadow. If the classification failed to identify two dominant lines within the detected edgels, then the edge detection parameters  $\sigma$  and  $t$  are relaxed (so that some weak edges are rejected) and the detect/classify procedure is repeated.

#### 8.2.4 Camera pose

The camera pose is estimated from the imaged positions of five black dots on the target plate as described in Chapter 6. Camera viewpoint and lighting direction are varied as part of the photometric stereo requirement, making this an ideal application for robust exemplar based fiducial detection.

A semi-automatic approach was taken to processing the input images for camera pose estimation. This simplified the amount of special case code that had to be written, and gave the user the opportunity for hands-on verification of the algorithms. The variables that the user could tune were: search scales, nearest neighbour weighting threshold, window size for non-maxima suppression, and the background area to mask off. The automatically detected dot locations are superimposed on the image and presented to the user for approval. The cones and their shadows tended to be identified as possible fiducials, which is not unreasonable since they do make up a black blob on a white background. Example images of the cones could have been added to the set of negative exemplars to prevent these false identifications, but this was not done. Instead, the dot detection criteria were relaxed to purposely *include* both the cones and the dots. If the algorithm failed to correctly identify all five dots and four cones, the above values could be altered until all were located (see Figure 8.7 for an example). Such tweaking was only required for a small set of extremely oblique viewing angles or poorly lit views.

The orientation of the target pattern was determined based on the position of the fifth dot. As all the dots are identical, their proximity to one another was used to identify the one that wasn't on a corner. This was done by computing the convex hull

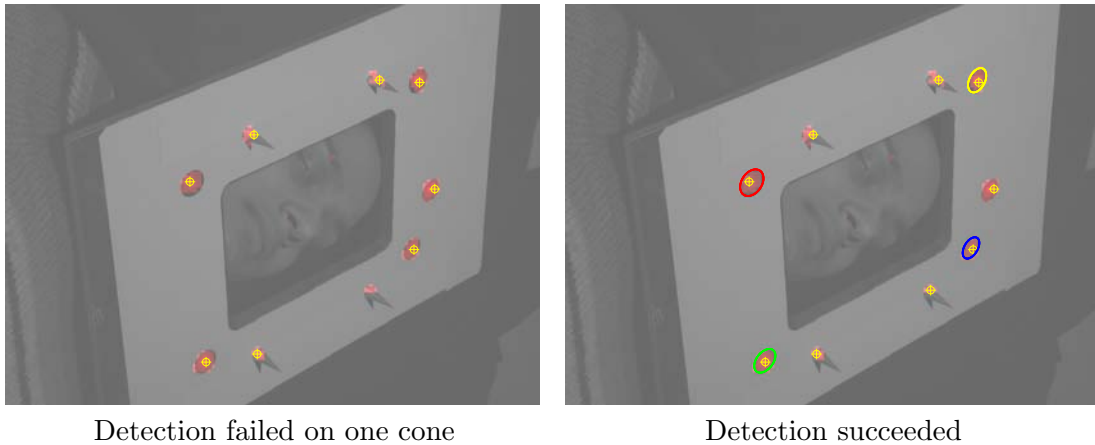


Figure 8.7: An example of an oblique view of the target where the fiducial detection failed to locate one of the cones. Regions which might contain a fiducial are highlighted in red; darker tones indicate a stronger response. The yellow markers indicate regions that are above the threshold. In the left image one of the cones is not identified, but by increasing the threshold from 0.85 to 1.35 (right image) we are able to detect all the fiducials and properly label the four corners of the target.

of all detected fiducials and then measuring the distances between adjacent vertices. The fiducial with the two nearest neighbours corresponds to the corner immediately after (in a clockwise sense) the fifth dot (recall that the cones are included in the fiducial set). Although the proximity test will fail for extreme perspective views, this straightforward algorithm worked quite well in practice. Another approach would be to test all the permutations of the detected points by computing the homography to map each ordering onto the reference positions. The permutation that yields the lowest residual error would be the correct orientation.

Once the approximate dot centres and their image-world correspondence were established, the homography based circle centre finding optimization of §7.4.2 was used to refine the centre positions. These image positions were then fed into the camera pose optimization of §6.3. If the target plate with the cones was used, then the cone shadow tips were also fed into the optimization and both camera and light positions were simultaneously estimated. In either case, the output of the optimization is the homography which maps the imaged target plane to be fronto-parallel. Applying this homography to the image results in a view of the target and sample where both are at specified image positions. Once each input image has been fit to this known template it is a straightforward matter to locate the regions where the cone shadows will be, and to mask off the matt target background.

The camera pose optimization includes the camera internal calibration matrix  $K$ . As outlined in §2.1.1,  $K$  is parameterized by focal length  $f$ , aspect ratio  $a$  and principal

point  $(c_u, c_v)$ . Table 8.3 lists the results of the optimization for each of the sequences. These are grouped into sets that were recorded in separate sessions; the light position and camera parameters were kept constant through each set. Note that the parameters for each set display a high degree of agreement. The primary exceptions to this are the focal length and the  $z$ -component of the light position. The planar target geometry produces an optimization that is poorly conditioned to solve for focal length. As a result we expect to see some variability in the focal length, which corresponds to a range of  $z$ -values for the position of the approximately fronto-parallel camera. By contrast, the ray intersection for light position produces a strong constraint and a precise 3D measurement of the light location. This means that if we consider the solution for the camera and light positions in the same global coordinate frame, the light is fixed but there is some “play” in the camera position. Because we solve for the light position *relative* to the camera, this play is taken up in the  $z$ -component of the light position. Thus there is some correlation between the focal length and the  $z$ -component of the light position.

One means for evaluating the calibration via optimization is to compare it with the results from a dedicated calibration package. Images of the MATLAB Calibration Toolbox checkerboard were already recorded for distortion correction, so it was also used to provide focal length and principal point comparison values. The results shown in Table 8.4 show that the calibration methods produce comparable results.

Included in the camera pose is a rotation matrix  $\mathbf{R}$  which is parameterized by a 3-element Rodrigues vector measured relative to the initial estimate (refer to Appendix B for a description of this parametrization and the problems associated with singularities in rotation matrices). The overall parameter vector for the optimization is therefore

$$\mathbf{v} = \left[ \underbrace{\mathbf{K}}_4 \underbrace{\mathbf{L}_c}_3 \underbrace{\mathbf{R}_1 \dots \mathbf{R}_N}_{3N} \underbrace{\mathbf{t}_1 \dots \mathbf{t}_n}_{3N} \right]. \quad (8.9)$$

For a typical set of  $N = 10$  images this optimization is on  $7 + 6N = 67$  parameters which is easy and quick to converge. The result of this optimization is the homography required to warp all images fronto-parallel and thus achieve pixel for pixel correspondence between all images in each sequence.

### 8.2.5 Light Attenuation

A brief review of the physics of image formation will prove helpful in discussing the light fall-off correction implemented for the photometric stereo system §8.2. Let us assume that light emanates from a point source with equal power in all directions. Some of this light falls upon an object’s surface. Neglecting absorption and transmission, this

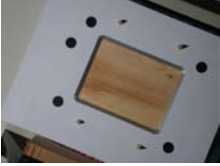
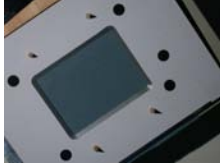
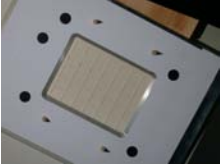
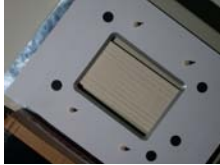


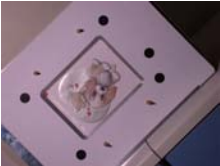
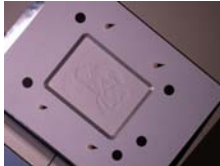

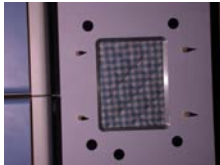
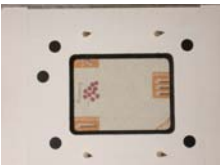

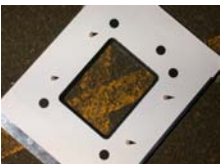


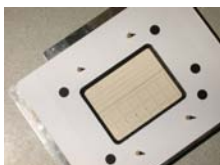


	<b>Wood</b> 10 images Set A	Light: Box Target: Plate	
	<b>Towel</b> 8 images Set A	Light: Box Target: Plate	
	<b>Tile A</b> 10 images Set A	Light: Box Target: Plate	
	<b>Tile B</b> 9 images Set A	Light: Box Target: Plate	
	<b>Polystyrene</b> 11 images Set B	Light: Box Target: Plate	
	<b>Bounty</b> 8 images Set B	Light: Box Target: Plate	
	<b>Card front</b> 8 images Set B	Light: Box Target: Plate	
	<b>Card back</b> 7 images Set B	Light: Box Target: Plate	
	<b>Drink can</b> 11 images Set B	Light: Box Target: Plate	
	<b>Glove</b> 14 images Set B	Light: Box Target: Plate	
	<b>Towel</b> 4 images Set C	Light: Flash Target: Plate	
	<b>Towel</b> 4 images Set C	Light: Flash Target: Paper	
	<b>Asphalt</b> 8 images Set D	Light: Flash Target: Plate	
	<b>Bricks</b> 8 images Set D	Light: Flash Target: Paper	
	<b>Hand</b> 11 images Set D	Light: Flash Target: Plate	
	<b>Tiles</b> 5 images Set D	Light: Flash Target: Plate	
	<b>Tiles</b> 6 images Set D	Light: Flash Target: Paper	
	<b>Face</b> 10 images Set E	Light: Box Target: Plate	

Table 8.2: Description of the sequences used in the photometric stereo application.

Sequence	Set	Focal length	Aspect ratio	Principal pt		Light position		
				$c_u$	$c_v$	$x$	$y$	$z$
Tile A	A	5914	0.9990	1040	748	-538	-23	-5
Tile B	A	5926	0.9993	1031	808	-536	-24	-6
Wood	A	6146	0.9990	1044	798	-538	-20	37
Towel	A	6870	0.9993	1080	859	-542	-24	174
Polystyrene	B	5286	1.0000	1026	777	-540	-14	46
Bounty	B	5065	0.9999	1023	745	-537	-11	-20
Card front	B	4994	0.9995	990	764	-534	-16	-39
Card back	B	5101	0.9997	994	756	-528	-13	-7
Drink can	B	5346	0.9999	1053	763	-538	-12	44
Glove	B	5329	0.9999	1023	763	-541	-13	40
Towel - plate	C	7763	1.0006	1033	795	-552	-162	32
Towel - paper	C	7582	0.9999	1023	786	-	-	-
Asphalt	D	5400	0.9994	1034	767	-544	-151	39
Bricks	D	5785	1.0001	1061	787	-	-	-
Hand	D	5445	1.0008	1059	814	-546	-164	58
Tiles - plate	D	5643	0.9982	1010	704	-537	-143	94
Tiles - paper	D	6419	1.0000	1105	661	-	-	-
Face	E	5085	1.0000	1076	806	528	-36	37

Table 8.3: Results of camera calibration optimization for sequences used in the photometric stereo application. All measurements are given in pixels, except for the light positions which are in *mm* relative to the camera centre. Sequences that used the paper calibration target do not permit the independent solution of light position, so no values are given. Refer to Table 8.2 for a listing of which target and light source were used for each image sequence.

Set	Focal length			$\mathbf{c}_u$			$\mathbf{c}_v$		
	TBX	OPT	Diff.	TBX	OPT	Diff.	TBX	OPT	Diff.
A	6104	6214 (450)	1.8%	1019	1049 (22)	2.9%	823	803 (45)	2.4%
B	5313	5193 (173)	2.3%	1033	1015 (29)	1.7%	765	762 (4)	0.5%
C	7699	7672 (128)	0.3%	1032	1028 (7)	0.4%	779	790 (6)	1.5%
D	5449	5738 (411)	5.3%	1031	1054 (36)	2.2%	757	747 (63)	1.4%
E	5036	5085 -	1.0%	1033	1076 -	4.2%	760	806 -	6.0%

Table 8.4: Comparison of camera calibration parameters obtained from the MATLAB Camera Calibration Toolbox and the non-linear optimization on image locations of the fiducials. For each set of calibration images, TBX denotes the value obtained by the Toolbox calibration, OPT lists the mean value and standard deviation (in brackets) across all sequences. The low percent difference values indicate that optimization of the camera pose from a small set of fiducials provides comparable camera calibration to a dedicated calibration package.

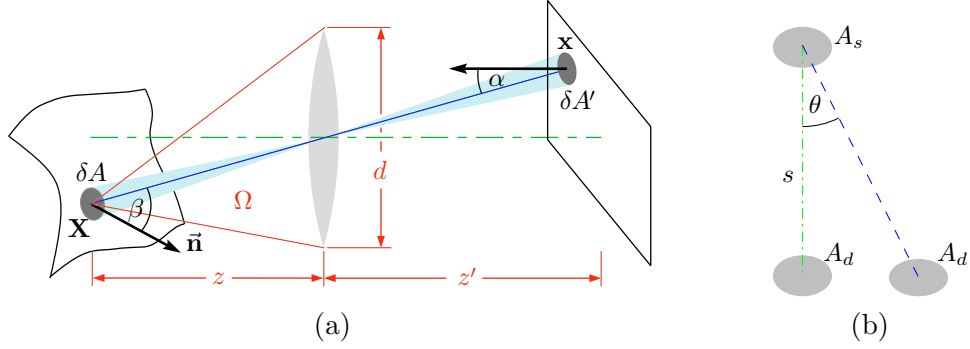


Figure 8.8: Off-axis irradiance. (a) The image irradiance viewed through a thin lens falls off as per  $\cos^4 \alpha$  due to foreshortening of the projected areas. (b) The same  $\cos^4 \theta$  decrease is observed for a circular source and detector pair offset by an angle  $\theta$ ; this is purely a geometric effect and not due to the thin lens.

light will then be reflected and we can say that the patch  $\delta A$  centred at point  $\mathbf{X}$  has radiance  $L$ . The photometric stereo derivation requires that the incident radiance  $L(x, y)$  is known for every point on the object's surface. A number of photometric stereo implementations achieve this by assuming a point light source at infinity and negligible attenuation. As shown in Figure 8.12, however, the observed irradiance does not necessarily match this assumption. This section will first explain several of the factors which contribute to the attenuation, and then describe an empirical method for correcting it.

The off-axis irradiance falls off according to a cosine-to-the-fourth term for a thin lens observing a Lambertian surface (Zalewski 1995, Forsyth and Ponce 2003). Consider the surface area  $\delta A$  centred at point  $\mathbf{X}$  and imaged by a lens of diameter  $d$  onto the image plane area  $\delta A'$  centred at  $\mathbf{x}$  (Figure 8.8a). The two areas are related by  $\delta A z'^2 \cos \beta = \delta A' z^2 \cos \alpha$ . The solid angle of the lens is given by  $\Omega = \frac{\pi}{4} \left(\frac{d}{z}\right)^2 \cos^3 \alpha$  and the emitted power from  $\delta A$  that reaches the lens is  $\delta P = L \Omega \delta A \cos \beta$ . From these definitions, the image irradiance is then

$$E = \frac{\delta P}{\delta A'} = \frac{\pi}{4} \left(\frac{d}{z'}\right)^2 L \cos^4 \alpha.$$

Note that for points further from the centre of the image the  $\cos^4 \alpha$  term attenuates the observed irradiance. This is due to foreshortening of the projected areas and an increase in the overall distance between source and detector (as per the inverse square law). To see this more clearly, consider a circular source and detector pair as shown in Figure 8.8b. The irradiance observed by the two identical detectors are

$$E_{aligned} = \frac{L A_s A_d}{s^2} \quad E_{off-axis} = \frac{L A_s A_d}{s^2} \cos^4 \alpha.$$

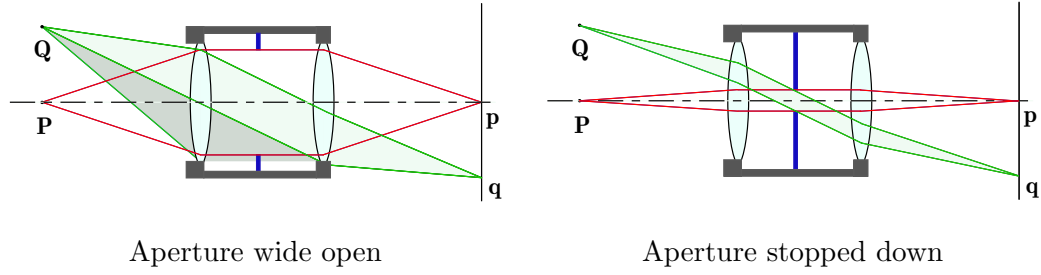


Figure 8.9: Lens vignetting results when the physical components of the lens body and the aperture obscure light passing through the lens. For the on-axis point  $\mathbf{P}$  all light reaches the image at  $\mathbf{p}$ . When the aperture is wide open, some of the light from  $\mathbf{Q}$  is occluded by the lens body so point  $\mathbf{q}$  is not as bright. This causes the outer edges of the image to appear darker, but the effect is not as pronounced with a stopped down lens.

The projected areas are  $A_s \cos \alpha$  and  $A_d \cos \alpha$ , while the distance from the source to the detector is increased by  $1/\cos \alpha$ .

Vignetting is another effect which contributes to a decrease in irradiance towards the edges of an image. As shown in Figure 8.9, light originating from off-axis points may be partially occluded by components of the lens body, including the aperture. The amount of attenuation is proportional to the angle of the source relative to the optic axis. Reducing the size of the aperture (stopping down the lens) increases the diameter in the image plane where this effect sets in (Figure 8.9b).

### 8.2.6 Light Fall-off Correction

Prior to applying the photometric stereo algorithms and solving for the surface normals it is important to apply radiometric correction to the aligned input images. Correcting for radiometric attenuation reduces the amount of bias in the recovered surface normals. When these normals are integrated to form a 3D surface, the bias is manifested as an error in overall curvature (low frequency spatial variation). Figure 8.10 demonstrates the dramatic reduction in reconstruction error that results from careful correction for radiometric fall-off.

Vignetting and cosine fall-off (refer to §8.2.5) are observed image intensity variations caused by the camera and optics. The light source employed by the photometric system may also introduce a varying intensity caused by non-uniform light dispersion. An ideal light source would originate from a point and irradiate equal power in all directions. Any actual flash or light box will have a finite size and some sort of light dispersion pattern. For this work the fall-off observed in the target plane is measured in order to compute the correction terms. While it may be possible to measure the radiance



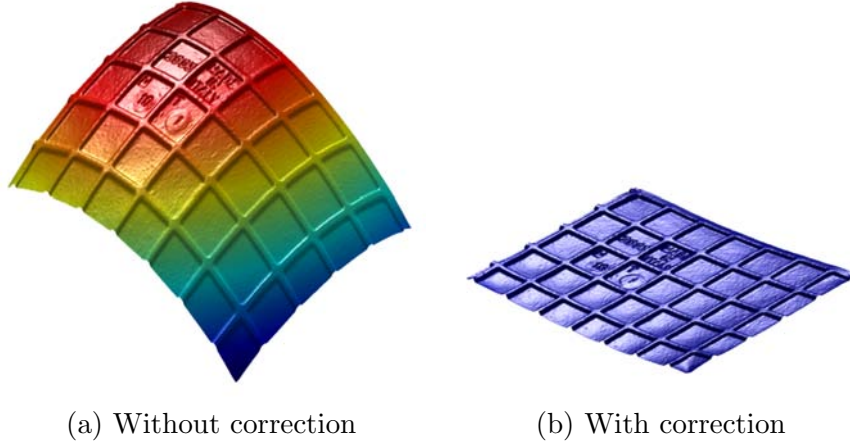


Figure 8.10: Radiometric correction greatly improves the reconstruction of a planar surface (Tile from Figure 8.12). Correcting for light attenuation in each input image reduces the bias in the recovered surface normals. This bias manifests as low frequency errors in the integrated surface reconstruction. The reconstruction with corrections is much closer to the true shape of the flat tile. *Colour indicates height; the same scale is used for both surfaces.*

at many points in space and build up a volumetric representation for a specific light source, this would be difficult and does not account for possible time-varying changes in the light properties.

Photometric stereo is only concerned with the light radiance in the sample area as observed by the camera during the instant the image was recorded. The target surrounds the sample area with a border of known photometric properties. By observing the irradiance observed on this border it is possible to estimate the corrections required to compensate for all attenuation effects: cosine fall-off, vignetting, spatial or temporal variations of the light and any others. This is similar to the approach of Marschner et al. (1999), who record calibration images of a flat white plane to calculate the compensation. By contrast, we sample points on the target background to measure the radiometric properties of each image and provide specific compensation.

A rough paper target will follow the Lambertian model well enough that the imaged light intensity at each location on the matt background is given by  $I(x, y) = L(x, y)\rho_0\mathbf{n} \cdot \mathbf{l}_i$  where  $\rho_0$  is the surface albedo, and  $\mathbf{n} = [0, 0, 1]^\top$ . Then, for image  $i$  we have

$$I_i(x, y) = L_i(x, y)\rho_0\mathbf{n}(x, y)^\top \mathbf{l}_i(x, y) \quad (8.10)$$

at each point on the target background (outside the sample window). We assume that attenuation effects vary slowly over the image, and fit a smooth function  $A_i(x, y)$  to  $I_i(x, y)/(\mathbf{n}(x, y)^\top \mathbf{l}_i(x, y))$  to determine  $L_i(x, y) = A_i(x, y)/\rho_0$ .

We sample the intensity at a grid of locations on the background, masking off areas

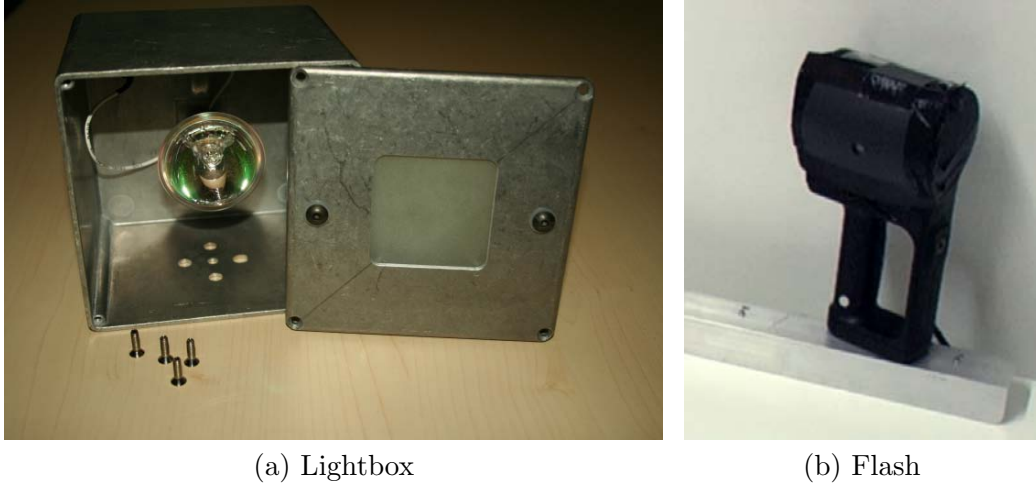


Figure 8.11: The two different types of light source used for the photometric stereo application. The lightbox contains a halogen bulb and is fitted with frosted glass to act as a diffuser. The flash is covered by a piece of plastic with a small aperture cut in it to approximate a point light source.

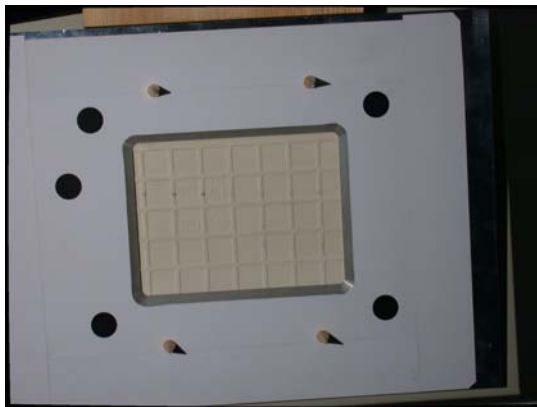
where the cone shadows or sample window may interfere (see Figures 8.12c and 8.13c), and then use these values to fit an analytic function. This function then provides radiance estimates within the sample region. Since the ambient light is assumed negligible, the scale on  $L$ , and hence  $\rho_0$ , is arbitrary; we specify it by selecting an intensity value for the background that will maximize the dynamic range of the images.

The form of the analytic function depends on the properties of the light source used. We used two light sources for this work: 1) a special light box, and 2) a commercial flash gun fitted with a small aperture to approximate a pinhole light source. Both of these are pictured in Figure 8.11. The light box constructed for these experiments consists of a halogen bulb fitted with sandblasted glass to act as a diffusion plate. The “frosted” glass is quite effective at providing a smooth spatial distribution, so a biquadratic function

$$A_i(x, y) = a_1x^2 + a_2xy + a_3y^2 + a_4x + a_5y + a_6 \quad (8.11)$$

provides a good fit. The commercial flash unit used in our experiments employs a curved piece of plastic formed with horizontal grooves to achieve diffusion. This is effective when the entire flash is uncovered, but we fit an opaque masking panel over most of the flash area to closer approximate a point light source. The result for the diffusion is that the few remaining visible grooves induce a saw-tooth intensity pattern (see Figure 8.13d). Therefore a piecewise linear function was superimposed over the biquadratic function when correcting for images recorded with the flash.

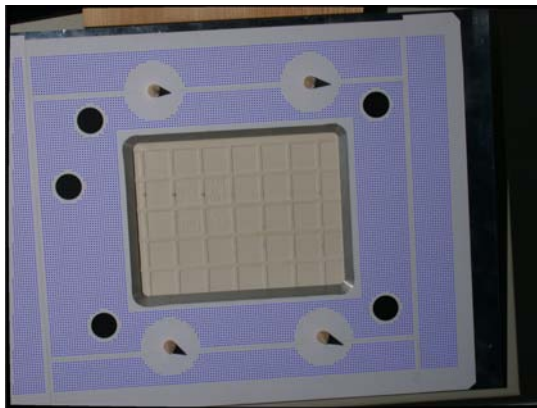
This lighting fall-off correction was performed separately on each colour channel.



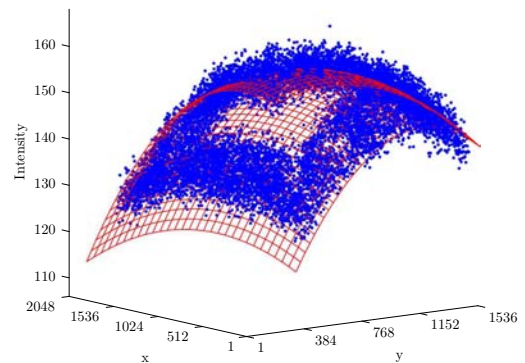
(a) Input image



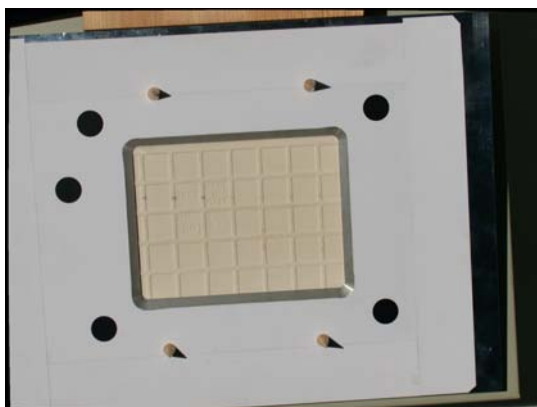
(b) Input image, normalized for emphasis



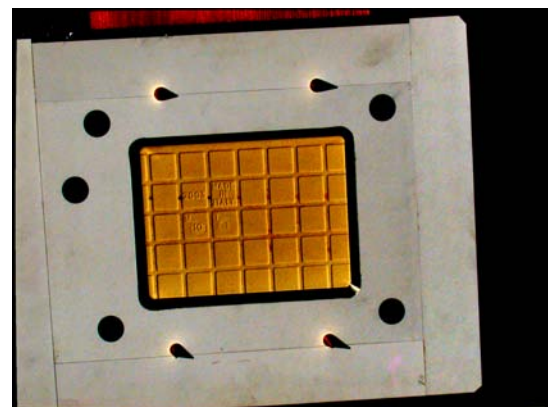
(c) Sample points on matte background



(d) Bi-quadratic fit to sample intensities

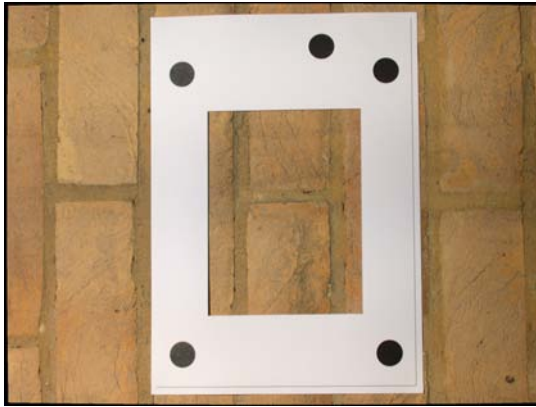


(e) Corrected image

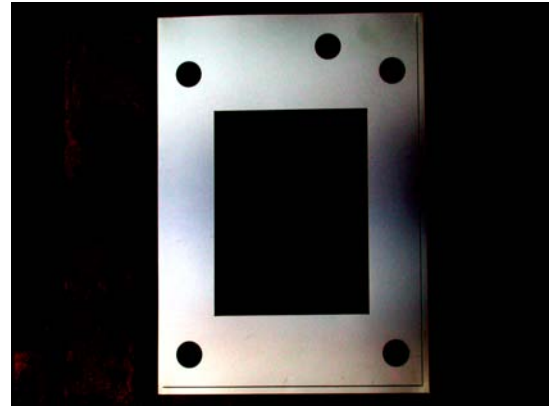


(f) Corrected image, normalized for emphasis

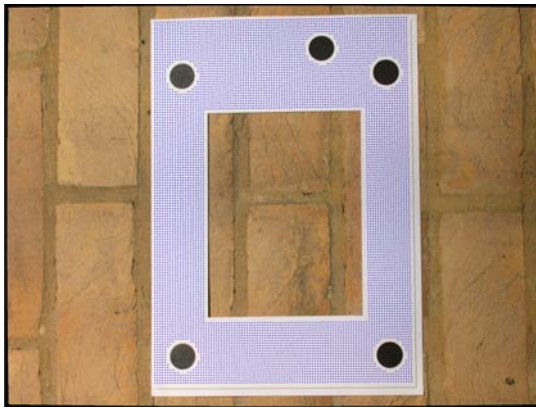
Figure 8.12: Radiometric calibration is performed on each sample image to compensate for light fall-off across the target area. Points on the target background are sampled (with non-white areas masked off automatically based on the known geometry), and a biquadratic fit to the intensity values. The original image is then divided through by this mask.



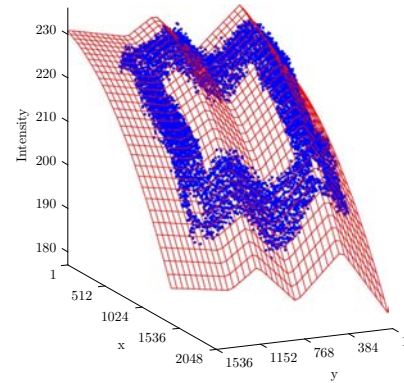
(a) Input image



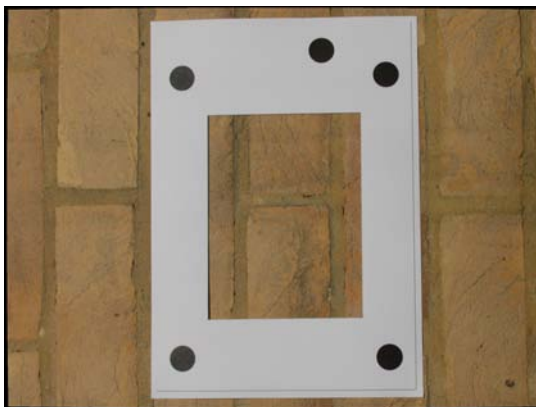
(b) Input image, normalized for emphasis



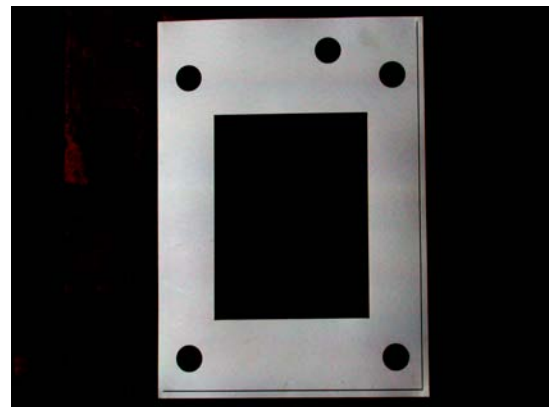
(c) Sample points on matte background



(d) Piecewise linear fit to sample intensities



(e) Corrected image



(f) Corrected image, normalized for emphasis

Figure 8.13: Radiometric compensation for images illuminated with a commercial flash gun. A small aperture is used to restrict the light to approximate a point light source. This negates the effects of the diffusion grooving, so the illumination varies spatially. The biquadratic function described in Figure 8.12 is modulated by a piecewise linear function to better approximate the observed intensity pattern.

This provides white balance and intensity normalization such that the white card of the background is imaged as true white with intensity determined by  $L_{bg}$ . Indexing the colour channels by  $k = [R, G, B]$  the light intensity is given by

$$L_{ik}(x, y) = I_{ik}(x, y) / (\mathbf{l}_z A_{ik}(x, y)). \quad (8.12)$$

This radiometrically corrected intensity value signifies the end of the pre-processing required for the photometric stereo application. We have previously completed the geometric alignment: each input image is aligned to pixel for pixel correspondence and we know the light and camera positions for each view. The only remaining unknown from (8.1) is the surface normal. Standard photometric stereo techniques can be used to solve for this normal at each pixel given three or more views of that pixel that are lit from different directions. Details of the techniques used for this project are provided in Paterson's thesis (Paterson 2005).

### 8.2.7 Surface Integration

We now turn our attention to the task of constructing a 3D surface based on the surface normals returned by the photometric stereo algorithms. The normal map alone is useful for applications such as graphics rendering, which can give the appearance of microscale texture by using the normal map to modulate the reflected light. However it is often desirable to have an actual representation of the 3D surface, and for that it is necessary to integrate the surface normals.

The problem posed by surface integration is the reconstruction of a surface  $z(x, y)$  given only the gradient information  $\frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}$ . Surface integration is the task of computing the 3D surface that is most consistent with the surface normals  $\mathbf{n}(x, y)$  output by the photometric stereo algorithm. The normal map corresponds to a gradient field defined on a regular (pixel) grid. The task (illustrated in Figure 8.14) is to find, via discrete integration, the function  $z(x, y)$  which could have given rise to these computed derivatives.

A number of surface integration techniques have been proposed (refer to (Klette and Schlüns 1996) for a summary and comparison). The methods can be broadly divided into local integration along paths (Coleman and Jain 1982) and global minimization techniques. While local integration is computationally very efficient, it is also susceptible to drift caused by errors or noise in the normal map. For this reason global minimization (Horn and Brooks 1986) was selected for inclusion in our system. The Fourier based integration of Frankot and Chellappa (1988) was found to produce a very similar surface in less than  $1/200^{th}$  of the time.



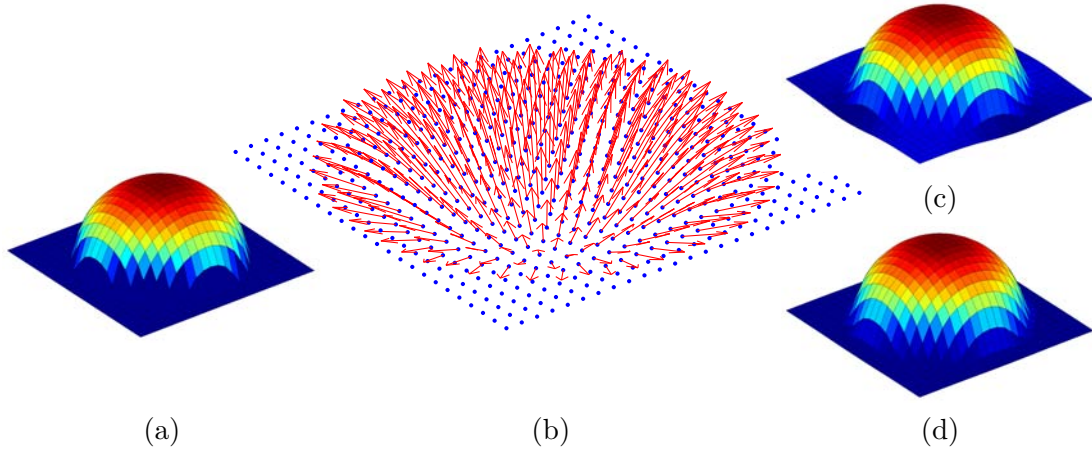


Figure 8.14: Surface integration from a normal map. (a) The true surface (b) The discrete set of surface normals which are integrated to reconstruct the original height map  $z(x, y)$ . *Vertical normals omitted for clarity.* (c) Surface produced by quadratic optimization (d) Surface produced by converting to the Fourier domain. Note the differences at the discontinuity between the plane and the sphere.

One of the advantages of the quadratic optimization approach used here is the ease with which boundary conditions are included. When integrating surface normals, overall vertical ( $z$ ) translation is unconstrained, so additional interpolation constraints are introduced to establish the correct scale. Our photometric stereo system includes multiple viewpoints, which permits the application of multiview geometry techniques to recover the heights at a sparse set of points on the object. These keypoints are established from user-selected point correspondences in the input image set. The process of identifying these depth keypoint correspondences could be automated (especially since the rectified images are available for initialization) but this was not done for the current implementation since selecting the points is a half-minute task. The ability to measure the depth scale is a significant advantage over single-viewpoint photometric stereo methods.

These depth constraints cannot be relied upon to correct for low frequency errors in the surface caused by bias in the underlying normals. The pre-processing steps such as lens distortion correction and radiometric compensation correct for the most significant causes of this bias in the normals, but an effective method for removing the low frequency drift in the output surface would mitigate the effects of the other un-modeled causes. This is an area for continued investigation. Although the output surface is subject to some drift, the surface normal map obtained from photometric stereo tends to be fairly free of high frequency noise. In the case where greater than three input views are used the system is over constrained and a least squares optimal

solution is computed. This has the effect of averaging out much of the high frequency noise. For three view reconstruction noise is more of an issue, and noise reducing integration methods such as non-linear 2D leap-frog (Noakes and Kozera 2003) may produce a better surface.

### 8.2.8 Parallax Correction

The homography used to warp the input images fronto-parallel (described in §8.2.4) assumes that the sample object is planar. A planar object does not induce parallax in images produced from different viewpoints so a planar homography is all that is required to fully align the images. Sample objects which are not planar (or even approximately planar as is the case for many textured surfaces) will not be accurately aligned by a simple homography due to motion parallax. The difference between such images aligned by a planar homography is a function of the distance from each object point to the camera: closer points move further in the images than more distant points. If the camera position and surface geometry are known it is possible to predict (and therefore correct for) these parallax effects. In this application the camera location is known for each image, and an approximation of the object's geometry can be obtained using the surface integration described in the previous section. That integrated shape is only an approximation because it was produced using a planar assumption, but it provides a starting point. Parallax correction proceeds by iteratively estimating the object's shape, using that estimate to correct for parallax in the input images, and then re-estimating the object's shape using the corrected input images. In practice one or two iteration are sufficient to produce a high-fidelity surface representation.

Parallax correction serves two purposes in this application. These are illustrated in Figures 8.16 and 8.17. The first is to ensure proper alignment when rendering the fronto-parallel views by reprojecting from the correct height. This reduces crosstalk at abrupt changes in surface albedo. The second purpose is to correctly model occluded pixels. A portion of the object that is hidden in an oblique view should not be rendered in the fronto-parallel view corresponding to that input image. Pixels that are not visible cannot contribute to the photometric solution. Without this occlusion handling the hidden pixels are shaded according to whichever pixels cause the occlusion. This in turn affects the recovered surface normal at those points.

Figure 8.16 shows how rendering based on the correct height at each point on the object reduces crosstalk in the recovered albedo. We show a small one dimensional slice of an object; the slice is flat but includes a step change in surface colour from blue to red. The object has been photographed twice: once from the left and once from

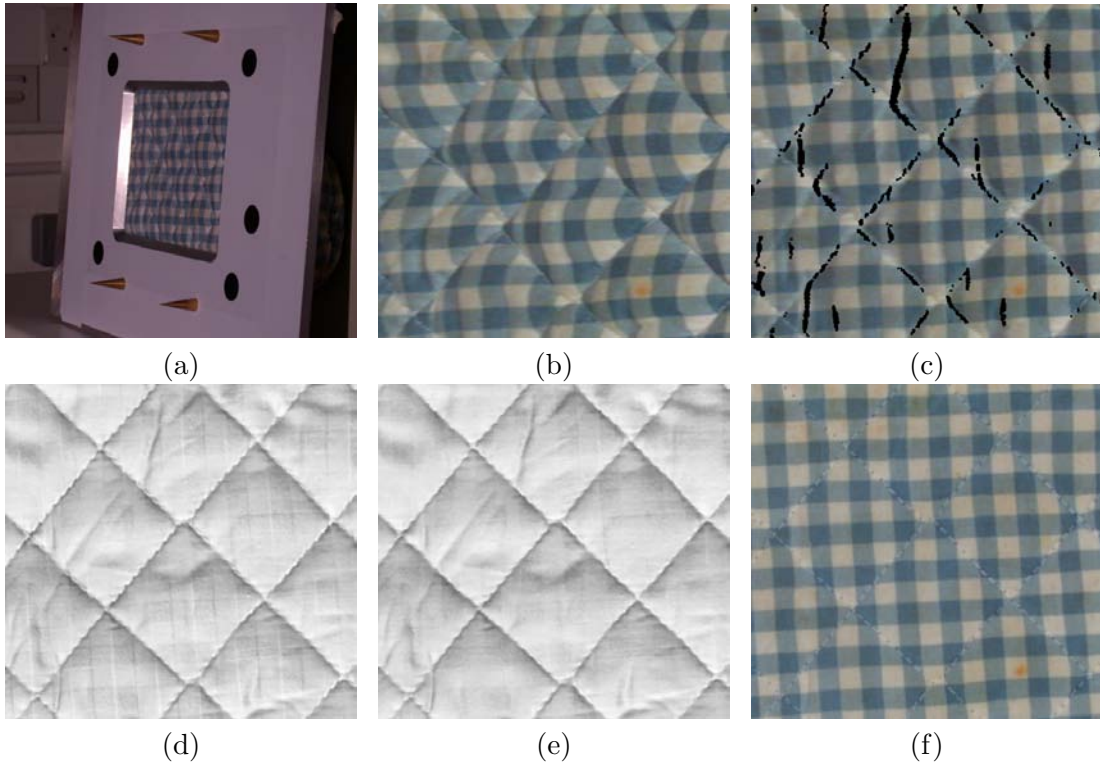


Figure 8.15: Parallax correction of a sample photographed from a very oblique angle. (a) Original image (b) First iteration,  $z = 0$ . Note the undulating vertical lines. (c) Second iteration, lines have been straightened (black pixels were occluded in the original image). (d) The glove normal map before correction exhibits crosstalk between the albedo and normal channels. (e) After correction, crosstalk is reduced. (f) Recovered albedo for comparison.

the right. Figure 8.16 depicts two separate reconstructions of the imaging geometry and the resulting sets of fronto-parallel images. The first is a reconstruction where the object height is not incorporated, so  $z = 0$ . This represents the first photometric stereo pass where a planar homography is used for image rectification. Our known values include the camera poses, the location of the target, the layout of the pixel grid for the fronto-parallel image and the images recorded from the two oblique viewpoints. By fixing all of these and projecting onto the  $z = 0$  plane we produce our two fronto-parallel images (shown in between the two cameras). In each reconstruction the object is shown twice, once for each view, because the views project differently. This is evident in the fronto-parallel images where the red/blue transitions do not line up. Two images of the same object from the same viewpoint should be aligned; the problem is in the object height representation. For the second reconstruction in Figure 8.16 we assign the height  $z = h$  to the object. For the small linear slice the effect is to raise it, but over the entire object we produce an irregular surface. Projecting the two images onto



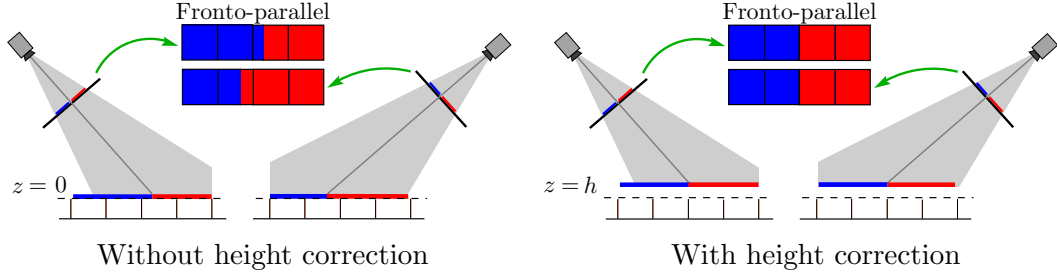


Figure 8.16: Albedo crosstalk can be reduced through parallax correction. *Left* Two views of the same red/blue object where each image has been warped fronto-parallel using an incorrect height value. Notice that the stripes do not line up in the rendered views. *Right* Using the correct height data produces fronto-parallel views where the stripes line up properly. *Green arrows indicate rendering into the fronto-parallel views.*

this height map results in fronto-parallel images that are in alignment.

We present parallax correction as an image warping problem. Given an input image recorded at an oblique viewpoint, a height map for the target object, and the position of the camera relative to the object we aim to compute the warping function that will produce a rerendered fronto-parallel image.

For each input image  $I_i$  we know the camera position  $P_i = K[R_i t_i]$ . From the initial photometric stereo run we have an approximation of the object height  $h_j$  at each of the pixels in the output image. The image location  $(u_j, v_j)$  to sample for pixel  $j$  is given by

$$\begin{pmatrix} u_j \\ v_j \end{pmatrix} = \pi \left( P \begin{bmatrix} x_j \\ y_j \\ h_j \\ 1 \end{bmatrix} \right) \quad (8.13)$$

where  $(x_j, y_j)$  is the grid location of the  $j^{\text{th}}$  pixel in the fronto-parallel image. The image intensity at subpixel location  $(u_j, v_j)$  is sampled through bilinear interpolation. Repeating this for every pixel in the rectified image produces a warping that “pulls” parallax corrected intensity values into the fronto-parallel view.

Occlusions are handled by z-buffering. The distance from each object point  $(x_j, y_j, h_j)$  to the camera is computed and stored in a height ( $z$ ) buffer. As illustrated in Figure 8.17, multiple destination pixels may sample from the same input image pixel. However, that input pixel represents only the intensity at the nearest point. The height buffer is used to determine which destination pixel is closest and should therefore be filled in. All others are marked as invalid so they are not included in the photometric stereo calculations; only visible points supply information for that view.

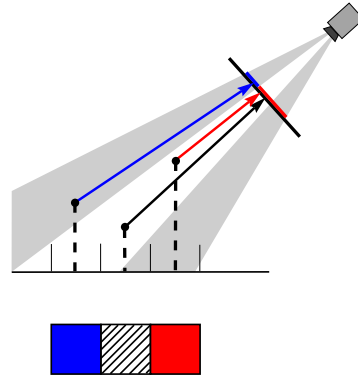


Figure 8.17: Z-buffering is used to avoid rendering of pixels that are obscured in oblique views. In this example, the centre pixel is not rendered into the fronto-parallel view because it is obscured by the rightmost pixel. The z-buffer stores the distance from the height for each pixel to the camera and then only the nearest pixel is rendered.

### 8.3 Results

Figure 8.18 shows the results of the photometric stereo application on a number of different materials. From each set of input images we have computed the surface normals, albedo and integrated the 3D surface shape. The hand demonstrates the ability to accurately recover the normals for an object that is far from planar; the resulting 3D shape is accurate and largely free from the systematic bias that is common when integrating a surface from normal data. The checkered oven glove has been separated into surface normals and albedo — very little crosstalk is evident in the normal map. The polystyrene is a translucent material which does not follow the Lambertian lighting model. Overall this reconstruction is successful, but there is some darkening in the recovered albedo in the vicinity of abrupt changes in depth. These are locations where the material translucence will allow light to shine through and therefore disrupt the photometric stereo assumptions. The asphalt example demonstrates the ability of this system to reconstruct an object photographed outdoors in sunlight. Our flash apparatus causes the background illumination from the sun to be insignificant, so it is not necessary to capture objects in complete darkness. This greatly simplifies the acquisition process, and makes the system highly portable.

Figure 8.19 shows the results of an experiment designed to push the limits of the photometric stereo system. A human face is far from planar, skin is quite specular (non-Lambertian), and we are highly tuned to recognizing facial deformations. The recovered surface model is very true to life. The specular properties of skin only produced visible artifacts on the eyelids; notice the peaks at the centre of each eye. These

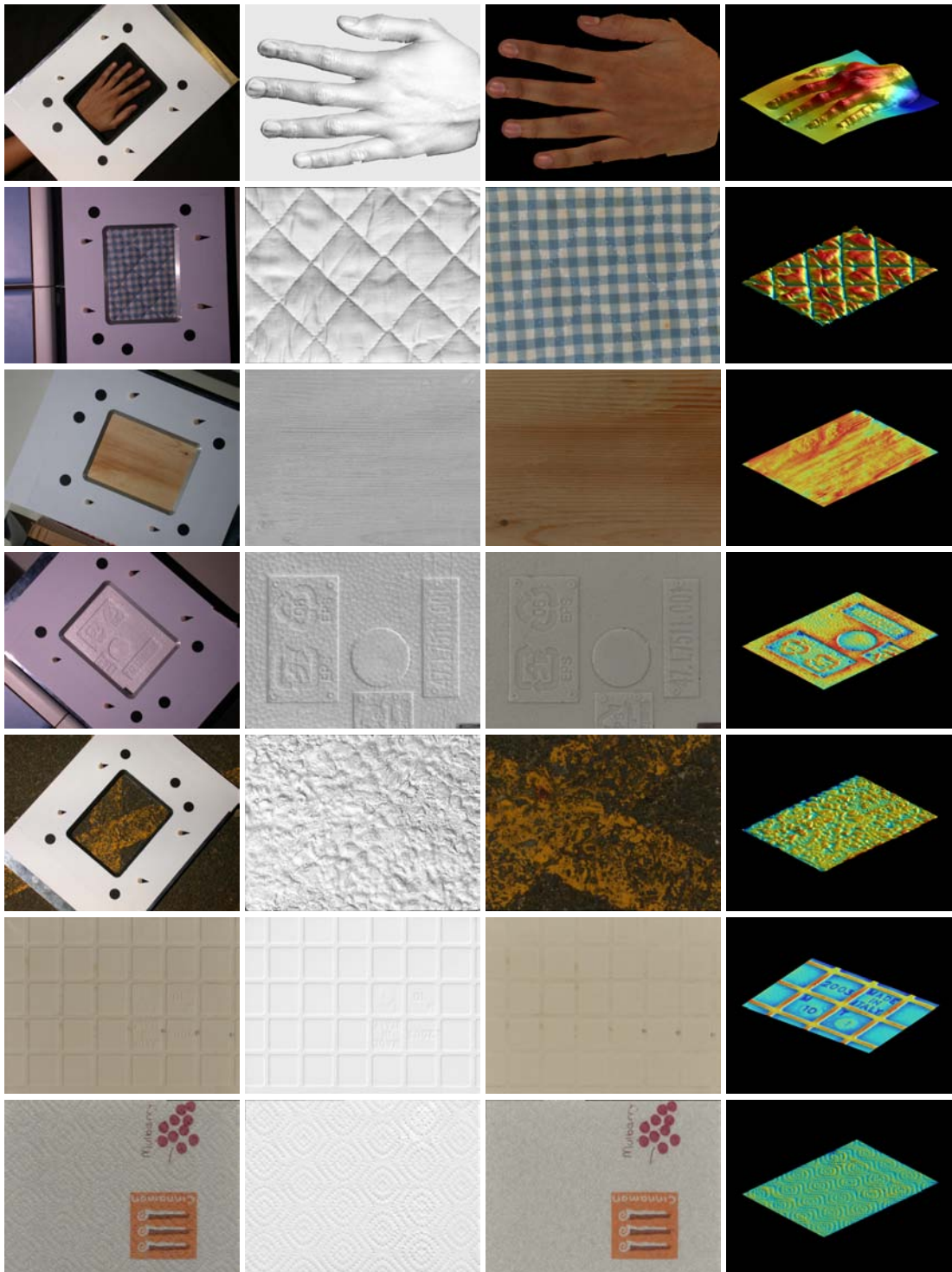


Figure 8.18: Results of the photometric stereo application on a selection of materials. From top: Hand, Padded glove, Wood, Polystyrene (non-Lambertian), Tarmac, Tile, Paper towel. From left to right: Input image, Normals, Albedo, 3D reconstruction.

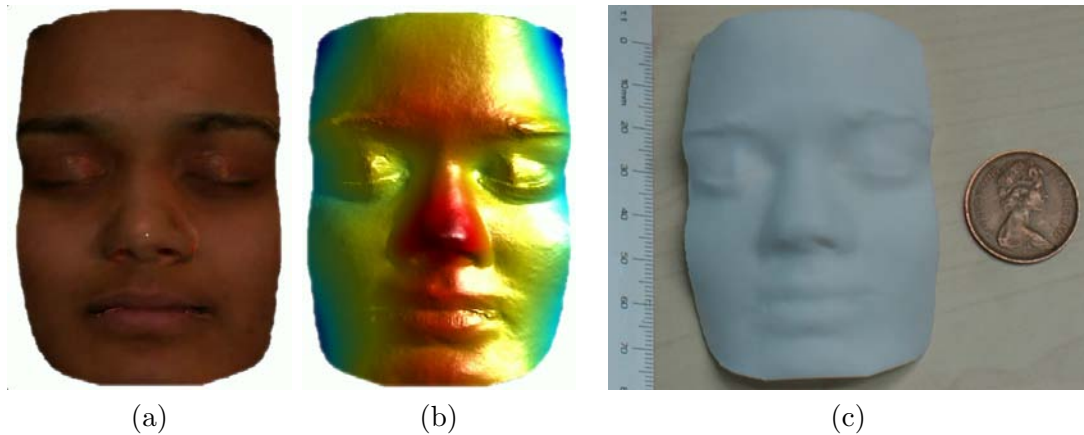


Figure 8.19: Face reconstruction through photometric stereo. (a) Recovered albedo demonstrates accurate skin tones. (b) Depth shaded surface model (c) Mask created by rapid prototyping at one quarter scale.

are the result of reflections in each input view which introduce noise into the surface normal computations. Sharp depth changes and self-shadowing caused problems at the corners of the mouth where there were not enough view with visible pixels to allow the computation of meaningful surface information. Orange pixels in the albedo map indicate the same problem.

## 8.4 Conclusions

We set out to build a photometric stereo system that was inexpensive to construct, simple to operate, highly portable, could operate in ambient lighting, recovered accurate height data, and produced high resolution albedo and normal maps. The results shown above provide qualitative evidence to the quality of the output data. For quantitative results and a more in-depth treatment of the photometric aspects of the project the reader is referred to (Paterson et al. 2005, Paterson 2005). The aim in relation to this thesis was to demonstrate an application of the camera localization and distortion correction methods. Accurate fiducial detection is the key to the system's portability: because the camera position can be computed for each image the operator is free to photograph with a handheld camera. Precise camera pose optimization is also key to making the moving camera feasible. Fiducials, pose and distortion correction are all required to precisely align the input images. Without such accurate alignment the photometric stereo input data would be nonsense.

## Chapter 9

# Conclusion

This thesis has addressed three important components of any system that makes measurements using a camera: 1) correcting lens distortion, 2) calibrating and localizing the camera, and 3) reliably detecting features within the images. A camera based measurement system must address all three. A perfect lens distortion model cannot be calibrated without precisely detected image features. Likewise, detected image locations can only be transformed into metric coordinates up to the accuracy of the camera calibration and distortion correction. My findings in each of these key areas are described in the following sections. The thread that ties them all together is that accurate modelling followed by careful fitting of those models to data yields highly accurate camera calibration and pose data.

### 9.1 Modelling lens distortion

The primary contribution of this thesis is a rational function model for lens distortion. A quotient of polynomials permits the accurate modelling of non-linear lens distortion within a convenient mathematical framework. Image coordinates are “lifted” to a higher dimensional (quadratic) space where the distortion can be represented linearly. The formulation is capable of modelling a wide variety of lenses; specific representations were described for pinhole, distorted and catadioptric cameras. These representations use fewer parameters than the full model and are therefore more robust in everyday usage. Finally, epipolar geometry for multiple views was defined within the new framework.

Three calibration techniques for the rational function lens distortion model were presented, each using a different type of image information. Linear calibration from a planar calibration grid is the simplest, and in situations where the camera is available for offline calibration this provides a reliable calibration technique with little or no user intervention required. The plumbline method is based on straightening the curved im-

ages of lines which are truly straight in the real world. An elegant linear factorization solution is presented, though optimization is required to make this a stable calibration technique in the presence of realistic image noise. The sheer volume of data provided by line fitting makes this a precise calibration method in practice. The multiple view calibration method also requires nonlinear optimization to provide stable camera parameters in the presence of image noise. Both the plumline and multiple view calibration methods yield very accurate camera parameters through an optimization process that is simple and robust enough for general application. Reducing the number of model parameters constrains the optimization so that a stable solution can be found reliably.

The Rational Function model accurately represents the distortion present in real world lenses, and provides a simplified and scalable family of calibration procedures. Both fisheye and standard lenses calibrated using the RF model were compared with calibrations by existing methods. The RF model can be fit linearly in a single step, without the need to first estimate the aspect ratio and distortion centre; the residual errors are on the same order as the best lens-specific model available. The performance of our proposed method matched the lens-specific field of view model, yet the proposed method can be calibrated linearly. The plumline calibration method is robust both to changes in initialization values and to image noise. Comparison with the MATLAB Camera Calibration Toolbox revealed that its model does not represent the distortion near the image periphery as accurately as the rational function model does. The RF model is also calibrated from a single image rather than the 14 used by the Toolbox. The plumline method was found to be slightly more precise than the planar grid method calibration. The reduced parametrization incurs a slight loss of accuracy over the full rational function model, but not enough to discourage its use for most applications. The multiple view method for calibrating the rational function model demonstrates that it is possible to perform auto-calibration of fisheye camera parameters from point correspondences in real images. Nonlinear optimization using a reduced parametrization achieves stable convergence and provides an explicit definition of the model parameters. Image rectification results and a scene reconstruction from distorted input images demonstrate that these techniques are applicable to real world scenarios. This work has been cited in (Chen and Ip 2006, Kannala and Brandt 2006, Li et al. 2005, Rosten and Cox 2006, Steele and Jaynes 2006), with significant extensions put forward by Barreto and Daniilidis (2006).

## 9.2 Camera calibration and localization

Given a set of known world positions and their corresponding image locations, camera localization computes the pose of the camera when the image was recorded. This is best considered as a nonlinear optimization problem: by computing the analytic derivatives of the reprojection function it is possible to perform nonlinear pose and focal length optimization in real time. The camera path computed on individual frames is comparable in accuracy to the path obtained through offline bundle adjustment over the entire sequence. Our optical initialization technique for surveying the positions of calibration reference points simplifies this difficult 3D measurement problem. It also demonstrates how a precisely calibrated consumer grade camera can become a powerful yet inexpensive measurement instrument.

Camera localization results were compared with independent absolute measurements of the camera position. A normal lens corrected using the rational function model produced camera locations for each individual frame that are accurate to 0.15 mm RMS and 0.47 mm maximum. The results for a fisheye lens were 0.23 mm RMS and 0.54 mm maximum. This indicates that sub millimeter position accuracy is possible for single frame camera localization using nonlinear optimization with appropriate lens distortion correction. Moreover, these results are comparable to those obtained through bundle adjustment on the entire sequence; single frame methods suitable for realtime applications need not suffer in accuracy for lack of bundle adjustment.

## 9.3 Reliable fiducial detection

We then presented a fiducial detection scheme that uses example based learning to handle lighting variation, scale changes and motion blur. Engineering a fiducial detector by assembling image operations produces a narrowly focused algorithm that is difficult to adopt to new scene challenges. By contrast, our exemplar based method is retrained simply by including relevant image samples. The power of machine learning techniques benefits even the most basic of computer vision tasks, especially when high reliability is included in the design constraints. Evaluation on a varied collection of image video sequences demonstrated that the learnt detector returns fewer false positives than an engineered one, while simultaneously identifying fiducials over a wider operating range. Adding image processing steps such as normalization to the learnt detector actually decreases the performance. This work has been cited by (Gao and Vasconcelos 2004, Fiala 2005, Klein and Drummond 2005, Lepetit and Fua 2005, Yang et al. 2005, Kaján et al. 2006).

## 9.4 Application areas

Although the previous chapter dealt only with a photometric stereo application, the techniques described in this thesis have many potential application areas. One of these is camera localization for indoor virtual reality. This is achieved by placement of known markers within a room. A significant challenge to such marker based optical positioning systems is how to deal with varying lighting and viewpoint conditions, challenges that can be overcome through the use of the fiducial detection techniques from Chapter 7. Another application would be in the area of position measurement in an outdoor context. Combining a camera with a Global Positioning System (GPS) receiver yields a synergy due to the complementary error distributions of the two sensor modalities. Here the optimization techniques from Chapters 6 would be augmented with occasional position fixes from the GPS receiver.

## 9.5 Summary

An inexpensive camera *can* be used to make accurate measurements. All of the images recorded for this thesis were shot using either a consumer video camera or a digital stills camera. We have demonstrated sub-millimeter camera localization even with a fisheye adapter fitted to a video camera. The combination of robust feature detection and precise camera calibration have many potential application areas. We focused on just one: a simple method for capturing texture models by recording real-world surfaces. Our photometric stereo application uses controlled lighting to illuminate an object from several viewpoints and then inverts the illumination equations to recover the object's shape. Careful engineering of all aspects of the processing pipeline yields very high resolution surface geometry. Precise camera localization permits simple, standard equipment to produce high resolution models of 3D geometry and reflectance.

No research topic is ever totally concluded; advancement will continue in camera calibration. This work has shown that with accurate modelling, careful fitting and reliable input data it is possible to obtain very accurate measurements from ordinary cameras.

## 9.6 Further extensions

This thesis has provided some new theory in the area of distortion correction, answered some questions in camera localization precision, and provided a different view of robust fiducial detection. As with most research project, though, it has also prompted additional questions and problems.



Perhaps the most interesting avenue for future work on these topics is the factorization algorithm for linear distortion model calibration. Noise in measured image point positions causes the methods described in this thesis to break down. The root of this problem may lie in the imposition of the rank constraint on the solution matrix; the truncation method discards too much useful information. A method for imposing the rank constraint that does not involve such a drastic truncation may resolve this difficulty.

Chapter 7 reported fiducial localization accuracy of  $\frac{1}{20^{th}}$  of a pixel. This is a measurement for a single camera under indoor lighting conditions. Further study is required to explore the relationship between feature localization accuracy and feature detection invariance. I expect that an increase in invariance to differing scene conditions would be accompanied by a decrease in localization precision.

The fiducial detection methods could be improved by incorporating faster classification algorithms. The nearest neighbour classifier used in this work includes dataset reduction to remove redundant training data. The speed and generality of the classifier is strongly influenced by the type of reduction that is imposed. A technique such as Locality Sensitive Hashing may provide comparable (or faster) classification speed, without the sensitivity observed with condensed nearest neighbour. Further study may also be warranted on the effects of using training data transformations to expand the range of the available data. I found in §7.3.3 that varying the training data provided greater detection rate increases than performing the same transformations on the test video.

Given the practical focus of this work, another logical extension would be to prepare a general purpose camera calibration and distortion package similar to the MATLAB Camera Calibration Toolbox. Although that suite works well, the need for multiple calibration images and a rather limited distortion algorithm no longer represents the state of the art. The techniques that I have presented in this thesis represent the theory required to implement a robust calibration package that could also solve for distortion from a single image.

## Appendix A

# Pinhole Camera Calibration Methods

This appendix outlines two well-known techniques for recovering the extrinsic parameters of a camera from point correspondences. They are included here as a reference for calibrating pinhole cameras.

### A.1 Extrinsic from Coplanar Data

The method of Simon et al. (2000) can be used to compute the rotation and translation for a camera if point correspondences to planar calibration data are available.

The relationship between the world coordinates and the image points is given by

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \simeq \mathbf{P} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}. \quad (\text{A.1})$$

Here we are using  $\simeq$  to denote equivalence up to a scale factor. By aligning the world coordinate system with the target plane the  $z$  coordinate becomes zero. The projection matrix  $\mathbf{P}$  can be factored into the internal calibration  $\mathbf{K}$  and a rotation and translation between the camera coordinates and the world coordinate system

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \simeq \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \begin{pmatrix} x \\ y \\ 0 \\ 1 \end{pmatrix}. \quad (\text{A.2})$$

The rotation matrix consists of three columns, but since the  $z$  value in the world coordinates is always zero this can be reduced to

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \simeq \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad (\text{A.3})$$

where  $\mathbf{r}_i$  denotes a column of the rotation matrix. The homography equation that relates the world coordinates to the image coordinates (A.1) is of the same form as (A.3). It is now just a problem of decomposing the homography into the intrinsic calibration matrix  $\mathbf{K}$ , rotation  $\mathbf{R}$  and translation  $\mathbf{t}$ . Since  $\mathbf{K}$  is considered to be known, it can be factored out of the homography and the remaining columns equated with the columns of the modified homography  $\mathbf{H}'$ .

$$\mathbf{H}' = \mathbf{H}\mathbf{K}^{-1} \quad (\text{A.4})$$

$$[\mathbf{h}'_1 \ \mathbf{h}'_2 \ \mathbf{h}'_3] = [\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{t}] \quad (\text{A.5})$$

The first two columns should be of unit length, but may not be due to noise in the image acquisition process. To correct this, the average length is computed as  $\sqrt{\|\mathbf{h}'_1\| \cdot \|\mathbf{h}'_2\|}$  and then the entire matrix is divided through by this amount. The columns of the rotation matrix must also be orthogonal, so they are rotated according to

$$\mathbf{r}'_1 = \frac{\mathbf{c}}{\|\mathbf{c}\|} + \frac{\mathbf{d}}{\|\mathbf{d}\|} \quad (\text{A.6})$$

$$\mathbf{r}'_2 = \frac{\mathbf{c}}{\|\mathbf{c}\|} - \frac{\mathbf{d}}{\|\mathbf{d}\|} \quad (\text{A.7})$$

where  $\mathbf{c} = \mathbf{r}_1 + \mathbf{r}_2$ ,  $\mathbf{d} = \mathbf{r}_1 \times \mathbf{r}_2$  and  $\times$  denotes the vector cross-product. The third column of the rotation matrix is given by

$$\mathbf{r}'_3 = \frac{\mathbf{r}'_1 \times \mathbf{r}'_2}{\|\mathbf{r}'_1 \times \mathbf{r}'_2\|}, \quad (\text{A.8})$$

so the full rotation and translation are

$$[\mathbf{R} \mid \mathbf{t}] = [\mathbf{r}'_1 \ \mathbf{r}'_2 \ \mathbf{r}'_3 \mid \mathbf{t}]. \quad (\text{A.9})$$

## A.2 Extrinsic from General Point Data

Recall from §2.4.3 that the DLT computes the transformation  $\mathbf{H}$  that relates two sets of points:

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \quad (\text{A.10})$$

The projection of world points  $\mathbf{X}$  into image points  $\mathbf{x}$  can be expressed as

$$\mathbf{x} = \pi(\mathbf{P}\mathbf{X}) \quad (\text{A.11})$$

where the  $3 \times 4$  matrix  $\mathbf{P}$  can be computed via the DLT. It is then necessary to decompose  $\mathbf{P}$  to separate the internal and external camera parameters

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \mid \mathbf{t}]. \quad (\text{A.12})$$

The camera extrinsics are shown in Figure 6.1. The calibration and rotation matrices are obtained as described in §4.1 of (Hartley and Zisserman 2003)§4.1. This ensures that  $\mathbf{K}$  is upper triangular and that  $\mathbf{R}$  is a valid rotation matrix. The camera centre  $\mathbf{C}$  (expressed in the world coordinate system) is related to the coordinate origin  $\mathbf{t}$  (i.e. the origin of the world coordinate frame expressed in the camera coordinate system) by

$$\mathbf{t} = -\mathbf{R}\mathbf{C} \quad (\text{A.13})$$

where

$$\mathbf{C} = \pi(\text{nullspace}(\mathbf{P})). \quad (\text{A.14})$$

We have now decomposed the camera projection matrix obtained by the DLT into its component intrinsic and extrinsic camera parameters. The reader is referred to (Hartley and Zisserman 2003) for a more detailed description of this technique.

## Appendix B

# Rotations

*How about sending me a fourth gimbal for Christmas? – Command Module  
Pilot Mike Collins, Apollo 11 Lunar Mission*

A rotation is a linear transformation that preserves the length of vectors and the orientation of space. A rotation group is the set of all rotations within Euclidean space,  $\mathbb{R}^3$ . This is referred to as the *special orthogonal group*  $\text{SO}(3)$ . These can be expressed as a  $3 \times 3$  orthogonal matrix  $\mathbf{R}$  for which  $\det(\mathbf{R}) = 1$ .

Throughout this thesis rotations are represented by such matrices, however it is often beneficial to represent a rotation by fewer parameters than are required for a rotation matrix. This is particularly true when a rotation is one of the variables included in a nonlinear optimization. The optimization must enforce the orthogonality constraints as it varies the rotation matrix; this is most easily accomplished by selecting a minimal parametrization so that the constraint is automatic. The downside of the three-parameter minimal representations is that they are not free from singularities. The key is to select a representation that places the singularities in a region that we can be sure the optimization will avoid.

Euler angles (yaw, pitch and roll) are one common 3 element rotation representation where each element represents an angle about one of the coordinate axes. Euler angles suffer from “gimbal lock” where two of the three gimbals (angular rotations) align the coordinates to cancel out one of the rotation references. In practice, this occurs when the vector being rotated approaches one of the coordinate axes; the rotation about that axis becomes undefined. This singularity at the origin is problematic for optimization.

### B.1 Quaternions

Both Rodrigues (1840) and Hamilton (1844) took the alternative approach of treating a rotation as an angle  $\phi$  about some axis  $\mathbf{n}$  in 3D. This representation is easily parame-

terized using *quaternions*, four dimensional vectors originally developed by Hamilton. A quaternion can be written as

$$\mathbf{q} = q1 + q_x\hat{\mathbf{i}} + q_y\hat{\mathbf{j}} + q_z\hat{\mathbf{k}} \quad (\text{B.1})$$

or as a couple of a real number and a vector of components

$$\mathbf{q} = [q, \mathbf{Q}], \quad \mathbf{Q} = (q_x, q_y, q_z). \quad (\text{B.2})$$

Rotations are best parameterized using the half-angle formula proposed by Rodrigues

$$R(\phi, \mathbf{n}) = [\cos \frac{\phi}{2}, \sin \frac{\phi}{2} \mathbf{n}] \quad (\text{B.3})$$

as it is more general and produces less confusion in the interpretation (Altmann 1989) than Hamilton's full-angle parametrization. A vector  $\mathbf{v}$  can be rotated through the angle  $\phi$  about the axis  $\mathbf{n}$  by

$$\mathbf{v}' = \mathbf{n}^\top \mathbf{v} \mathbf{n} + \cos \phi (\mathbf{v} - \mathbf{n}^\top \mathbf{v} \mathbf{n}) + \sin \phi (\mathbf{n} \times \mathbf{v}). \quad (\text{B.4})$$

The Rodrigues parametrization behaves correctly near the origin, unlike the Euler angles which are undefined for  $\beta = 0, \pi$ . This can cause problems when the Euler construction is used for optimization, as the movement towards a solution becomes extremely slow as the current estimate passes near the origin. Two other advantages of the Rodrigues parametrization are that it uniquely determines the rotation pole (Euler angles do not), and can keep track of  $2\pi$  rotations introduced while multiplying rotations. The main challenge in working with the Rodrigues parametrization is that there is a singularity at  $\phi = \pi$ . To avoid this while working with nonlinear optimizations, treat the initial estimate of the rotation as a static component, and optimize only the change in rotation from this starting point. Provided that the initial estimate is reasonable, the optimization should not be hindered by the singularity.

The multiplication rule for combining rotations is defined in closed form:

$$R(\alpha; \mathbf{l})R(\gamma; \mathbf{m}) = R(\phi; \mathbf{n}) \quad (\text{B.5})$$

where  $\phi = \alpha\gamma - \mathbf{l} \cdot \mathbf{m}$  and  $\mathbf{n} = \alpha\mathbf{m} + \gamma\mathbf{l} + \mathbf{l} \times \mathbf{m}$ . Equations for converting between quaternion rotations and rotation matrices are readily available in geometry texts.

The reader interested in rotation groups and the various parametrizations is referred to Altmann (1986) (which also contains the aforementioned equations).

## Appendix C

### Computing $\mathbf{A}$ from $\mathbf{G}$

The strategy for computing  $\mathbf{A}$  is to extract its (3D) orthogonal complement from the 4D nullspace of  $\mathbf{G}$ . Note that as  $\mathbf{F}$  is rank 2, it has a right nullvector, which we shall call  $\mathbf{e}$ . The nullspace of  $\mathbf{G}$  is the set of  $X$  for which

$$\mathbf{A}'^\top \mathbf{F} \mathbf{A} X = 0 \quad (\text{C.1})$$

and because  $\mathbf{A}'$  has full rank, this is also the set of  $X$  for which

$$\mathbf{F} \mathbf{A} X = 0 \quad (\text{C.2})$$

writing  $\mathbf{N} = \text{null}(\mathbf{G})$ , a  $6 \times 4$  matrix, we have that all such  $X$  are of the form  $\mathbf{N} \mathbf{u}$  for  $\mathbf{u} \in \mathbb{R}^4$ , so

$$\mathbf{F} \mathbf{A} \mathbf{N} \mathbf{u} = 0 \quad \forall \mathbf{u} \in \mathbb{R}^4 \quad (\text{C.3})$$

so  $\mathbf{A} \mathbf{N} \mathbf{u} \propto \mathbf{e} \forall \mathbf{u}$ , i.e. all columns of  $\mathbf{A} \mathbf{N}$  are multiples of  $\mathbf{e}$ , so

$$\mathbf{A} \mathbf{N} = \mathbf{e} \mathbf{v}^\top \quad (\text{C.4})$$

for some  $\mathbf{v} \in \mathbb{R}^4$ , and so

$$\mathbf{A} = \mathbf{e} \mathbf{v}^\top \mathbf{N}^\top + \mathbf{M} \mathbf{N}^\perp \quad (\text{C.5})$$

where  $\mathbf{N}^\perp = \text{null}(\mathbf{N}^\top)^\top$  and  $\mathbf{M}$  is an arbitrary  $6 \times 2$  matrix. Choosing  $\mathbf{e} = -\text{null}((\mathbf{N}^\perp)_{[1:6,4:6]})$  and  $\mathbf{M} = \text{null}(\mathbf{e}^\top)$  means  $\mathbf{A}$  is now a function only of  $\mathbf{v}$ . The  $4 \times 1$  vector  $\mathbf{v}$  is found by non-linear optimization of the following residuals on the original points (where  $\mathbf{A}^* = \mathbf{A}(\mathbf{v})$ ):

$$\sum \left[ (\mathbf{A}^* \boldsymbol{\chi}'_k)^\top \mathbf{A}^* \mathbf{G} \mathbf{A}^{*\top} \mathbf{A}^* \boldsymbol{\chi}_k \right]^2. \quad (\text{C.6})$$

# Bibliography

- 2d3 Ltd.: 2003, Boujou: Automated camera tracking. <http://www.2d3.com>.
- Altmann, S. L.: 1986, *Rotations, Quaternions and Double Groups*, Oxford University Press.
- Altmann, S. L.: 1989, Hamilton, Rodrigues, and the quaternion scandal, *Mathematics Magazine* **62**(5), 291–308.
- Arun, K. S., Huang, T. S. and Blostein, S. D.: 1987, Least-squares fitting of two 3-D point sets, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **9**(5), 698–700.
- Barreto, J. and Daniilidis, K.: 2006, Epipolar geometry of central projection systems using Veronese maps, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New York*, Vol. 1, IEEE Computer Society, Los Alamitos, CA, USA, pp. 1258–1265.
- Berchtold, S., Ertl, B., Keim, D. A., Kriegel, H.-P. and Seidl, T.: 1998, Fast nearest neighbor search in high-dimensional spaces, *Proc. ICDE*, pp. 209–218.
- Beyer, H.: 1992, Accurate calibration of CCD cameras, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 96–101.
- Bookstein, F.: 1979, Fitting conic sections to scattered data, *Computer Vision, Graphics and Image Processing* **9**, 56–71.
- Bouguet, J.-Y.: 2003, Camera calibration toolbox for MATLAB, *Technical report*, MRL-INTEL <http://www.vision.caltech.edu/bouguetj/calib.doc/>.
- Brown, D. C.: 1966, Decentering distortion of lenses, *Photogrammetric Engineering* **32**(3), 444–462.
- Brown, D. C.: 1971, Close-range camera calibration, *Photogrammetric Engineering* **37**(8), 855–866.
- Canny, J. F.: 1986, A computational approach to edge detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **8**(6), 679–698.
- Chen, Y. and Ip, H.: 2006, Single view metrology of wide-angle lens images, *The Visual Computer* **22**(7), 445–455.
- Chojnacki, W. and Brooks, M.: 2003, Revisiting Hartley’s normalized eight-point algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**(9), 1172–1177.
- Clapham, C.: 1996, *The Concise Oxford Dictionary of Mathematics*, second edn, Oxford University Press.



- Claus, D.: 2004, Video-based surveying for large outdoor environments, *Transfer of status report*, University of Oxford, Robotics Research Group.
- Claus, D. and Fitzgibbon, A. W.: 2004, Reliable fiducial detection in natural scenes, *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, Springer-Verlag, pp. 469–480.
- Claus, D. and Fitzgibbon, A. W.: 2005a, A plumbline constraint for the rational function lens distortion model, *Proceedings of the 16th British Machine Vision Conference, Oxford*, pp. 99–108.
- Claus, D. and Fitzgibbon, A. W.: 2005b, A rational function lens distortion model for general cameras, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, pp. 213–219.
- Claus, D. and Fitzgibbon, A. W.: 2005c, Reliable automatic calibration of a marker-based position tracking system, *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pp. 300–305.
- Coleman, E. N. and Jain, R.: 1982, Obtaining 3-dimensional shape of textured and specular surface using four-source photometry, *Computer Graphics and Image Processing* **18**, 309–328.
- Cost, S. and Salzberg, S.: 1993, A weighted nearest neighbor algorithm for learning with symbolic features, *Machine Learning* **10**, 57–78.
- Courtney, P. and Thacker, N.: 2001, *Imaging and Vision Systems: Theory, Assessment and Applications*, Vol. 9 of *Advances in Computation : Theory and Practice*, NOVA Science Books, chapter Performance Characterisation in Computer Vision: The role of statistics in testing and design.
- Cover, T. M. and Hart, P. E.: 1967, Nearest neighbor pattern classification, *IEEE Trans. Information Theory* **13**, 57–67.
- Dementhon, D. and Davis, L.: 1995, Model based pose in 25 lines of code, *International Journal of Computer Vision* **15**(1/2), 123–141.
- Deriche, R. and Giraudon, G.: 1993, A computational approach for corner and vertex detection, *International Journal of Computer Vision* **10**(2), 101–124.
- Devernay, F. and Faugeras, O.: 2001, Straight lines have to be straight, *Machine Vision and Applications* **13**, 14–24.
- Duda, R. O., Hart, P. E. and Stork, D. G.: 2001, *Pattern Classification*, 2nd edn, John Wiley and Sons.
- Fiala, M.: 2005, ARTag, a fiducial marker system using digital techniques, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego*, Vol. 2, pp. 590–596.
- Fitzgibbon, A. W.: 2001, Simultaneous linear estimation of multiple view geometry and lens distortion, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.
- Forsyth, D. and Ponce, J.: 2003, *Computer Vision: A Modern Approach*, Prentice Hall, chapter 25: Application: Finding in digital libraries, pp. 599–619.

- Foxlin, E. and Naimark, L.: 2003, Miniaturization, calibration & accuracy evaluation of a hybrid self-tracker, *IEEE International Symposium on Mixed and Augmented Reality*.
- Frankot, R. and Chellappa, R.: 1988, A method for enforcing integrability in shape from shading algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (10), 439–451.
- Gander, W., Golub, G. and Strebel, R.: 1994, Least-square fitting of circles and ellipses, *BIT* (43), 558–578.
- Gao, D. and Vasconcelos, N.: 2004, Discriminant saliency for visual recognition from cluttered scenes, *Advances in Neural Information Processing Systems*, Vol. 17, pp. 481–488.
- Geyer, C. and Daniilidis, K.: 2000, A unifying theory for central panoramic systems and practical implications, *Proceedings of the 5th European Conference on Computer Vision, Freiburg, Germany*, Vol. 2, Springer-Verlag, pp. 445–461.
- Grossberg, M. and Nayar, S.: 2001, A general imaging model and a method for finding its parameters, *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, Vol. 2, pp. 108–115.
- Hamilton, W. R.: 1844, On quaternions: or a new system of imaginaries in algebra, *Philosophy Magazine* pp. 489–495. 3rd ser. 25.
- Harker, M. and O’Leary, P.: 2005, Computation of homographies, *Proceedings of the 16th British Machine Vision Conference, Oxford*, pp. 310–319.
- Harris, C. G. and Stephens, M.: 1988, A combined corner and edge detector, *Proceedings of the 4th Alvey Vision Conference, Manchester*, pp. 147–151.
- Hart, P. E.: 1968, The condensed nearest neighbor rule, *IEEE Transactions on Information Theory* **14**, 515–516.
- Hartley, R. I.: 1997, In defense of the eight-point algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(6), 580–593.
- Hartley, R. I. and Saxena, T.: 1997, The cubic rational polynomial camera model, *Proceedings of the DARPA Image Understanding Workshop*, pp. 649–653.
- Hartley, R. I. and Zisserman, A.: 2003, *Multiple View Geometry in Computer Vision (Second Edition)*, Cambridge University Press, ISBN: 0521540518.
- Hecht, E.: 1998, *Optics*, 3rd edn, Addison–Wesley, Reading, Massachusetts.
- Heikkilä, J.: 2000, Geometric camera calibration using circular control points, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(10), 1066–1077.
- Heikkilä, J. and Silvén, O.: 1997, A four-step camera calibration procedure with implicit image correction, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Puerto Rico*, pp. 1106–1112.
- Horn, B. and Brooks, M.: 1986, The variational approach to shape from shading, *CVGIP* **33**(2), 174–208.
- Kadir, T., Zisserman, A. and Brady, M.: 2004, An affine invariant salient region detector, *Proceedings of the 8th European Conference on Computer Vision, Prague, Czech Republic*, Springer-Verlag.

- Kaján, L., Kertész-Farkas, A., Franklin, D., Ivanova, N., Kocsor, A. and Pongor, S.: 2006, Application of a simple likelihood ratio approximant to protein sequence classification, *Bioinformatics* **22**(23), 2865–2869.
- Kannala, J. and Brandt, S.: 2006, A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **28**(8), 1335–1340.
- Kato, H. and Billinghurst, M.: 1999, Marker tracking and HMD calibration for a video-based augmented reality conferencing system, *Int'l Workshop on AR*, pp. 85–94.
- Kilpelä, E.: 1980, Compensation of systematic errors of image and model coordinates, *International Archives of Photogrammetry* **XXIII**(B9), 407–427.
- Kim, J. S., Kim, H. W. and Kweon, I. S.: 2005, Geometric and algebraic constraints of projected concentric circles and their applications to camera calibration, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**(5), 637–642.
- Klein, G. and Drummond, T.: 2002, Tightly integrated sensor fusion for robust visual tracking, *Proceedings of the British Machine Vision Conference*, Vol. 2, pp. 787–796.
- Klein, G. and Drummond, T.: 2005, A single-frame visual gyroscope, *Proceedings of the 16th British Machine Vision Conference, Oxford*, Vol. 2, pp. 529–538.
- Klette, R. and Schlüns, K.: 1996, Height data from gradient maps, in S. Solomon, B. Batchelor and F. Waltz (eds), *Proc. SPIE, Machine Vision Applications, Architectures, and Systems Integration V*, Vol. 2908, pp. 204–215.
- Lenz, R. K. and Tsai, R. Y.: 1988, Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **10**(5), 713–720.
- Lepetit, V. and Fua, P.: 2005, *Monocular Model-based 3D Tracking of Rigid Objects*, now Publishers Inc.
- Levenberg, K.: 1944, A method for the solution of certain problems in least squares, *Quarterly of Applied Mathematics* **2**, 164–168.
- Li, H., Hartley, R. and Wang, L.: 2005, Auto-calibration of a compound-type omnidirectional camera, *Digital Image Computing: Techniques and Applications* p. 26.
- Longuet-Higgins, H. C.: 1981, A computer algorithm for reconstructing a scene from two projections, *Nature* **293**, 133–135.
- Lowe, D.: 1999, Object recognition from local scale-invariant features, *Proceedings of the 7th International Conference on Computer Vision, Kerkyra, Greece*, pp. 1150–1157.
- Malbezin, P., Piekarski, W. and Thomas, B.: 2002, Measuring ARToolkit accuracy in long distance tracking experiments, *Int'l AR Toolkit Workshop*.
- Mallon, J. and Whelan, P. F.: 2004, Precise radial un-distortion of images, *International Conference on Pattern Recognition*, pp. 18–21.
- Marquardt, D.: 1963, An algorithm for least-squares estimation of nonlinear parameters, *SIAM Journal of Applied Mathematics* **11**, 431–441.

- Marschner, S., Westin, S., Lafortune, E., Torrance, K. and Greenberg, D.: 1999, Image-based BRDF measurement including human skin, *Proceedings, Eurographics Workshop on Rendering*, pp. 131–144.
- Matas, J., Chum, O., Urban, M. and Pajdla, T.: 2002, Robust wide baseline stereo from maximally stable extremal regions, *Proceedings of the British Machine Vision Conference*, pp. 384–393.
- Mičušík, B.: 2004, *Two-View Geometry of Omnidirectional Cameras*, PhD thesis, Center for Machine Perception, Czech Technical University in Prague.
- Moravec, H.: 1979, Visual mapping by a robot rover, *Proceedings of the 6th International Joint Conference on Artificial Intelligence*, pp. 599–601.
- Moré, J. J., Garbow, B. S. and Hillstom, K. E.: 1980, User guide for MINPACK-1, *Technical Report ANL-80-74*, Argonne National Laboratory, Argonne, IL, USA [www.netlib.org/minpack/](http://www.netlib.org/minpack/).
- Nistér, D., Stewénius, H. and Grossmann, E.: 2005, Non-parametric self-calibration, *IEEE International Conference on Computer Vision (ICCV)*, Vol. 1, Beijing China, pp. 120–127.
- Noakes, L. and Kozera, R.: 2003, Nonlinearities and noise reduction in 3-source photometric stereo, *Journal of Mathematical Imaging and Vision* **18**(2), 119–127.
- Pajdla, T.: 2001, Epipolar geometry of some non-classical cameras, *Computer Vision Winter Workshop*, Slovenian Pattern Recognition Society, Ljubljana, Slovenia, pp. 223–233.
- Pajdla, T.: 2002, Stereo with oblique cameras, *International Journal of Computer Vision* **47**(1), 161–170.
- Paterson, J.: 2005, *Acquisition of Geometry and Reflectance of Objects, Including the Human Face, for Real-Time Systems*, PhD thesis, University of Oxford.
- Paterson, J. A., Claus, D. and Fitzgibbon, A. W.: 2005, BRDF and geometry capture from extended inhomogeneous samples using flash photography, *Computer Graphics Forum (Special Eurographics Issue)* **24**(3), 383–391.
- Peleg, S., Ben-Ezra, M. and Pritch, Y.: 2001, Omnistereo: Panoramic stereo imaging, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **23**(3), 279–290.
- Pless, R.: 2003, Using many cameras as one, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp. 587–593.
- Rademacher, P. and Bishop, G.: 1998, Multiple-center-of-projection images, *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics* **32**, 199–206.
- Ramalingam, S. and Sturm, P.: 2004, A generic structure-from-motion algorithm for cross-camera scenarios, *Proceedings of the IEEE Workshop on Omnidirectional Vision*, pp. 175–186.
- Ramalingam, S., Sturm, P. and Lodha, S. K.: 2005, Towards complete generic camera calibration, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. 1093–1098.
- Ripley, B. D.: 1997, Why do nearest-neighbour algorithms do so well? SIMCAT (Similarity and Categorization), Edinburgh.

- Rodrigues, O.: 1840, Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et la variation des coordonnées provenant de ses déplacements considérés indépendamment des causes qui peuvent les produire, *Journal de Mathématiques Pures et Appliquées* **5**, 380–440.
- Rosten, E. and Cox, S.: 2006, Accurate extraction of reciprocal space information from transmission electron microscopy images, *Advances in Visual Computing. LNCS 4292*, Vol. 1, pp. 373–382.
- Sample, J. G. and Kneebone, G. T. (eds): 1998, *Algebraic Projective Geometry*, Clarendon Press.
- Sampson, P. D.: 1982, Fitting conic sections to 'very scattered' data: An iterative refinement of the Bookstein algorithm, *Computer Vision, Graphics, and Image Processing* **18**, 97–108.
- Shum, H. and He, L.: 1999, Rendering with concentric mosaics, *Computer Graphics* **33**(Annual Conference Series), 299–306.
- Simon, G., Fitzgibbon, A. and Zisserman, A.: 2000, Markerless tracking using planar structures in the scene, *Proc. International Symposium on Augmented Reality*, pp. 120–128.
- Simpson, J. A. and Weiner, E. S. C. (eds): 1989, *The Oxford English Dictionary*, Oxford University Press.
- Slama, C.: 1980, *Manual of Photogrammetry*, 4th edn, American Society of Photogrammetry, Falls Church, VA, USA.
- Smith, S. M. and Brady, J. M.: 1997, SUSAN—a new approach to low level image processing, *International Journal of Computer Vision* **23**(1), 45–78.
- Sproull, R. F.: 1991, Refinements to Nearest-Neighbor searching in  $k$ -dimensional trees, *Algorithmica* **6**(4), 579–589.
- Steele, R. and Jaynes, C.: 2006, Overconstrained linear estimation of radial distortion and multi-view geometry, *Proceedings of the 8th European Conference on Computer Vision, Graz, Austria*, Springer-Verlag, pp. 253–264.
- Stein, G. P.: 1997, Lens distortion calibration using point correspondences, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Puerto Rico*.
- Stewénus, H., Oskarsson, M. and Åström, K.: 2005, Reconstruction from planar motion image sequences with applications for autonomous vehicles, *Scandinavian Conference on Image Analysis (SCIA)*, Joensuu, Finland.
- Strand, R. and Hayman, E.: 2005, Correcting radial distortion by circle fitting, *Proceedings of the 16th British Machine Vision Conference, Oxford*, pp. 89–98.
- Sturm, P.: 2002, Mixing catadioptric and perspective cameras, *Proceedings of the IEEE Workshop on Omnidirectional Vision*, pp. 37–44.
- Sturm, P.: 2005, Multi-view geometry for general camera models, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. 206–212.
- Sturm, P. and Ramalingam, S.: 2003, A generic calibration concept: Theory and algorithms, *Technical Report Research Report 5058*, INRIA, France.

- Sturm, P. and Ramalingam, S.: 2004, A generic concept for camera calibration, *Proceedings of the European Conference on Computer Vision*, Vol. 2, Springer-Verlag, pp. 1–13.
- Sutherland, I. E.: 1963, Sketchpad: A man-machine graphical communications system, *Technical Report 296*, MIT Lincoln Laboratories. Also published by Garland Publishing Inc, New York, 1980.
- Swaminathan, R. and Nayar, S.: 2000, Nonmetric calibration of wide-angle lenses and polycameras, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(10), 1172–1178.
- Tao, C. and Hu, Y.: 2001, A comprehensive study of the rational function model for photogrammetric processing, *Photogrammetric Engineering and Remote Sensing* **67**(12), 1347–1357.
- Taubin, G.: 1991, Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**(11), 1115–1138.
- The TargetJr Consortium: 2000, VXL C++ libraries for computer vision research and implementation, <http://vxl.sourceforge.net/>.
- Trefethen, L. N.: 1992, The definition of numerical analysis, *SIAM News*.
- Trefethen, L. N. and Bau, D.: 1997, *Numerical Linear Algebra*, SIAM.
- Triggs, W., McLauchlan, P., Hartley, R. and Fitzgibbon, A.: 2000, Bundle adjustment: A modern synthesis, in W. Triggs, A. Zisserman and R. Szeliski (eds), *Vision Algorithms: Theory and Practice*, LNCS, Springer Verlag, pp. 298–375.
- Tsai, Y. R.: 1987, A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses, *IEEE Journal of Robotics and Automation* **RA-3**(4), 323–344.
- Tuytelaars, T. and Van Gool, L.: 1999, Content-based image retrieval based on local affinity invariant regions, *International Conference on Visual Information Systems*, pp. 493–500.
- Van Gool, L., Moons, T. and Ungureanu, D.: 1996, Affine / photometric invariants for planar intensity patterns, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, UK*, Springer-Verlag, pp. 642–651.
- Viola, P. and Jones, M.: 2001, Rapid object detection using a boosted cascade of simple features, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 511–518.
- Wei, G. Q. and Ma, S. D.: 1993, A complete two-plane camera calibration method and experimental comparisons, *Proceedings of the 4th International Conference on Computer Vision, Berlin*, pp. 439–446.
- Wei, G. Q. and Ma, S. D.: 1994, Implicit and explicit camera calibration: Theory and experiments, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **16**(5), 469–480.
- Wellner, P.: 1993, Adaptive thresholding for the digital desk, *Technical Report EPC-1993-110*, Xerox.

- Weng, J., Cohen, P. and Herniou, M.: 1992, Camera calibration with distortion models and accuracy evaluation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**(10), 965–980.
- Willson, R. G. and Shafer, S. A.: 1994, What is the center of the image?, *Journal of the Optical Society of America* **11**(11), 2946–2955.
- Wilson, D. R. and Martinez, T. R.: 2000, Reduction techniques for instance-based learning algorithms, *Machine Learning* **38**(3), 257–286.
- Woodham, R. J.: 1980, Photometric method for determining surface orientation from multiple images, *Optical Engineering* **19**(1), 139–144.
- Yang, Q., Steele, R., Nister, D. and Jayne, C.: 2005, Learning the probability of correspondences without ground truth, *Proceedings of the 10th International Conference on Computer Vision, Beijing, China*, Vol. 2, pp. 1140–1147.
- Zalewski, E. F.: 1995, Radiometry and photometry, in M. Bass (ed.), *Handbook of Optics*, 2 edn, Vol. 2, McGraw-Hill, chapter 24.