12-2005

Universal Geometric Camera Calibration with Statistical Model Selection

Chistopher Paul Broaddus
University of Tennessee - Knoxville

To the Graduate Council:

I am submitting herewith a thesis written by Chistopher Paul Broaddus entitled "Universal Geometric Camera Calibration with Statistical Model Selection." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Mongi Abidi, Major Professor

We have read this thesis and recommend its acceptance:

Besma Abidi, Seong Kong

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a thesis written by Christopher Broaddus entitled "Universal Geometric Camera Calibration with Statistical Model Selection." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Mongi Abidi
Major Professor

We have read this thesis
and recommend its acceptance:

Besma Abidi

Seong Kong

Accepted for the Council:

Anne Mayhew

Vice Chancellor and
Dean of Graduate Studies

(Original signatures are on file with official student records)

Universal Geometric Camera Calibration with Statistical Model Selection

A Thesis
Presented for
Master of Science
Degree
The University of Tennessee, Knoxville

Christopher Paul Broaddus

December 2005

ACKNOWLEDGEMENTS

I never quite realized how academically immature I was, and still am. This is probably the last thing I should be admitting in the first section of my thesis, before you even read the abstract. It is hard to fathom how I actually made it to where I am today, academically and personally, despite many paths, detours and struggles. I can attribute this to my dad, Craig Broaddus, and my step-mom, Cindy Broaddus, for encouraging me to follow my passion. I cannot forget my beloved brother, Michael Broaddus, for helping me stay on a fairly straight path. I would also like to give special thanks to my best friend, Lori Brewer, for being a friend like no other, and all the red ink spilt on my reports.

I would like to express my gratitude to Dr. Mongi Abidi for giving me the opportunity to study, and work in the IRIS Lab, and to Dr. Besma Abidi for her efforts as my advisor. I give special thanks to Dr. Andrei Gribok for being an inspiration to me and the stimulating discussions. I owe a significant amount of thanks to Dr. David Page, as he always supported me.

Dedicated to Tammy Broaddus, 1992

ABSTRACT

We propose a new universal camera calibration approach that uses statistical information criteria for automatic camera model selection. It requires the camera to observe a planar pattern from different positions, and then closed-form estimates for the intrinsic and extrinsic parameters are computed followed by nonlinear optimization. In lieu of modeling radial distortion, the lens projection of the camera is modeled, and in addition we include decentering distortion. This approach is particularly advantageous for wide angle (fisheye) camera calibration because it often reduces the complexity of the model compared to modeling radial distortion. We then apply statistical information criteria to automatically select the complexity of the camera model for any lens type. The complete algorithm is evaluated on synthetic and real data for several different lens projections, and a comparison between existing methods which use radial distortion is done.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AIC - Akaike information criteria

BIC - Bayesian information criteria

CAIC - Correction Akaike information criteria

DIAC - Dual image of the absolute conic

DLT - Direct linear transformation

IAC - Image of the absolute conic

LPDD - Lens projection and decentering distortion model

MDL - Minimum description length

MSE - Mean-square-error

NDLT - Normalized direct linear transformation

PDF - Probability density function

RDDD - Radial distortion and decentering distortion model

RMS - Root-mean-square error

SSD - Shortest data description

LIST OF NOTATIONS

$\mathbf{M}$ - World point

$\mathbf{m}$ - Image point

(u,v) - Image point coordinates

$\mathbf{C}$ - Camera center

f - Focal length

α/β - Aspect ratio

s - Skew

$\mathbf{p}=(u_0,v_0)$ - Principal point

$\mathbf{P}$ - Projection matrix

$\mathbf{H}$ - Homography

$\mathbf{t}$ - Translation vector

$\mathbf{R}$ - Rotation matrix

p - Number of radial distortion or lens projection coefficients (based on context)

q - Number of coefficients in the decentering distortion model

$\boldsymbol{\kappa}$ - Radial distortion or lens projection coefficients (based on context)

$\boldsymbol{\rho}$ - Decentering distortion coefficients

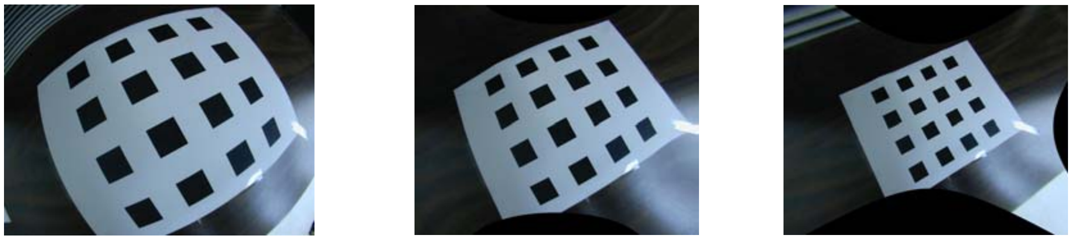
1 INTRODUCTION

The first step and fundamental problem in nearly every precision computer vision application which utilizes cameras is *calibration*. So what is camera calibration? When we are working with cameras, we need to know its characteristics, such as: how it represents color, how a point in space projects onto the camera imager, and what is the physical location of the camera relative to an object in space. In this thesis we are concerned with solving the last two, which is called *geometric camera calibration*. In geometric camera calibration we usually assume little or no prior knowledge of the camera. We are simply given a camera, a black box, and we model the inputs and outputs. In a perfect world we would design an ideal camera according to a mathematical model, and have the ability to manufacture it to exact specifications. Unfortunately, the manufacturing process is not perfect, and the real camera inputs and outputs will never match what the original model predicts exactly. The calibration step models the error, so the relationship between the real and ideal camera is known. This could be used to correct the real camera, so it becomes closer to the ideal one, or it could be used to compute statistics.

Over the years the objective of camera calibration has not changed, but the process has evolved considerably. In the early days, before the computer, calibration was mostly a mechanical procedure which utilized instruments, such as collimators and geodetic theodolites, in the photogrammetry community. These instruments were used to model the lens to identify its center and visible distortion. Instrumentation based calibration was used during World War I when the US government discovered the benefits of aerial surveying and started submitting cameras to the National Bureau of Standards for calibration. Aerial surveying received even more attention during World War II, after which several countries believed there was a need for standardization in calibration techniques, and a meeting was held between camera manufactures, calibration authorities, and photogrammetrists. The onset of computers marked the beginning of modern calibration techniques. These latter methods are largely analytical, which harnessed the speed of computers to compute solutions which would have otherwise been too tedious. Today, it is these calibration techniques which are largely used in computer vision research, attributed to Tsai [Tsai86] who joined the gap between the photogrammetry and computer vision communities. The most recent calibration methods do not use any instruments to make measurements. Instead, modern calibration techniques utilize mathematical models, analytical solutions and computer algorithms.

1.1 Motivation

In camera calibration, the parameters of a mathematical model are recovered, but are we calibrating the correct model? Figure 1.1 shows an original image taken from a full frame fisheye camera, and the corrected versions of this image after calibration of two different models with different complexities.



(a) (b) (c)

Figure 1.1: Set of images illustrating an insufficient model. a) The original image, (b) corrected image using insufficient model, and (c) corrected image using sufficient model. Notice how lines in (c) appear straight, but do not in (b).

We see that lines which should appear straight in Figure 1.1 (b) are actually curved caused by an insufficient model. However, if we increase the complexity of the model slightly we achieve the results shown in Figure 1.1 (c), which is the expected outcome.

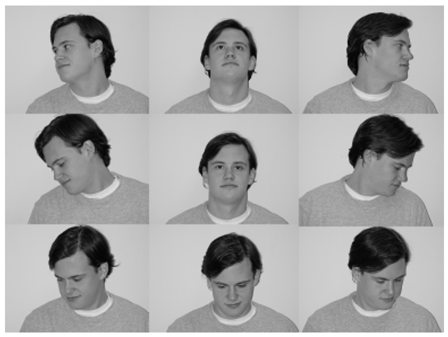
The most popular calibration methods take several images of a known model (most often a precise 2D or 3D grid pattern) from different camera positions. The projection of features from the model onto the image sensor is approximated with a pinhole camera model. The deviation from the pinhole camera is modeled as *radial distortion* and *decentering distortion* [Heikkila97][Heikkila00][Lenz87][Tsai86][Tsai87][Zhang00A]. All of these methods measure some combination of radial and decentering distortion. But the drawback to these methods is they require prior knowledge, namely the focal length, or do not perform well on wide angle cameras. They also do not include a way to automatically select the complexity of the model so that the best model is used regardless if the lens is rectilinear or fisheye.

This brings us to our motivation. We want to develop a calibration method that works equally well on a wide range of cameras, regardless of the quality or lens type, such as rectilinear or fisheye. In addition, we want to calibrate the least complex camera model that sufficiently models the camera. Wide angle cameras, such as fisheye cameras, are perhaps better approximated by modeling the *lens projection*, as opposed to radial distortion. Kannala and Brandt [Kannala04] used this approach to calibrate a fisheye camera for use in 3D reconstruction. However, they assume prior knowledge of the focal length, and the applicability to other lens projections, such as perspective and stereographic, was left unclear.

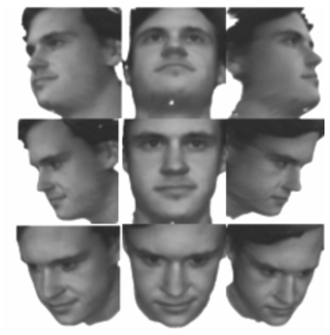
In this paper, we propose a new camera calibration technique which addresses the shortcomings of previous approaches. Namely, how complex should the camera model be to sufficiently model the camera, regardless of lens projection? Thus it should work equally well on a rectilinear or fisheye camera. To solve this, we apply statistical information criteria to automatically select the complexity of the lens and decentering distortion model of the particular camera, and evaluate the results for several sets of synthetic and real data. We provide results to show modeling lens projection performs as well, if not better, with little or no extra complexity, compared to modeling radial distortion for several different lens projections and a variety of real cameras. We also test an alternation technique during optimization mentioned by Weng [Weng92] and Zhang [Zhang00A] on several different lens projections. We found little benefit when calibrating a perspective camera, but significant improvement for other lens projections.

1.2 Application

The applications of geometric camera calibration are far reaching. Ever since the camera was invented, researchers have been developing methods for more precise calibration. Long before the camera, scientists had already written the underlying mathematics for modeling how 3D objects in space are represented on a 2D surface. Once the airplane was invented though, the application of aerial surveying became clear, which was a stimulus for research in developing models and calibration methods. In aerial photography, a plane is equipped with a camera mounted on the underside, which takes several images as the plane is in motion. The process of making scaled maps and measurements from these images is aerial photogrammetry. Prior to computer vision, calibration received much attention from photogrammetrists in the early to mid 1900's. Computer vision spun a new set of applications for calibration. One example is in 3D reconstruction where a laser scanner, or several images from multiple view points, is used to reconstruct a scene in 3D. Figure 1.2 is an example of a 3D model generated from several poses of a face.



(a)



(b)

Figure 1.2: Example of a 3D model generated from several poses. (a) 2D poses and (b) 3D model.

1.3 Contributions

Existing methods in camera calibration use some prior knowledge of the characteristics of the camera. For instance, this knowledge could be knowing if the camera is rectilinear or wide-angle, or low or high quality. Based upon this information the user chooses a sufficient model, and if it fails to produce acceptable results a different model is adopted. We took a different approach by removing the user from the model selection process using statistical model selection. If we were to remove this process from the user, than we need to have a model that suffices for a gamut of catadioptric cameras, i.e. rectilinear to fisheye. Traditional methods using *radial distortion* are difficult to calibrate without prior knowledge of the focal length. Since we assume no prior knowledge of the camera, these methods tend to get stuck in poor local minimums for wide-angle cameras using the closed-form estimate of the focal length. We thus adopted a different technique for modeling the lens based on modeling the *lens projection*, rather than modeling the deviation from the pinhole camera, i.e. radial distortion. In the experimental results chapter evidence is shown which supports this approach by analyzing the calibration errors of several synthetic and real cameras.

1.4 Document layout

The organization of this thesis is as follows. Chapters 2-5 are an overview of related work in camera calibration. They also provide the foundation to understand much of the theory in modern camera calibration. Chapter 2 describes the basic camera model and ways to parameterize a rotation matrix. Even though in this paper we assume the rotation matrix is parameterized as Euler angles, there are other techniques which may be superior depending on the application. Chapter 3 discusses lens modeling, where we specify the two different approaches. Chapter 4 canvasses much of the theory for direct linear transformation (DLT) based camera calibration, which is the basis for most modern calibration methods. We also discuss statistical model selection at the basic level to understand how it is applied to our problem; this is a huge field and can easily be a dissertation topic. Chapter 5 gives an overview of our algorithm and Chapter 6 gives our experimental results. Finally, our conclusions are in Chapter 7.

2 CAMERA MODELS

2.1 Pinhole camera

In this section we describe the pinhole camera that models the ideal perspective projection which is illustrated in Figure 2.1. In the pinhole camera, a point in space $\mathbf{M} = (X, Y, Z)^{\mathrm{T}}$ is projected onto the *image plane* to image point $\mathbf{m} = (u, v)^{\mathrm{T}}$ so the ray from $\mathbf{M}$ to $\mathbf{m}$ passes through the *camera center* (*center of projection*) $\mathbf{C}$. The *focal length* f is the distance from the camera center to the image plane. The *principal point* $\mathbf{p} = (u_0, v_0)$ is the point where the *principal axis* meets the image plane. From the similarity of triangles the point $\mathbf{m}$ on the image plane can be described in terms of the focal length and the coordinates of $\mathbf{M}$:

$$(X, Y, Z)^{\mathrm{T}} \mapsto (f\, X/Z\, ,\, f\, Y/Z)^{\mathrm{T}} = (u, v)^{\mathrm{T}}. \tag{2.1}$$

This can be written in matrix notation using *homogenous coordinates* by augmenting $\mathbf{M}$ and $\mathbf{m}$ with a 1 so that $\mathbf{M} \sim (X, Y, Z, 1)^{\mathrm{T}}$ and $\mathbf{m} \sim (u, v, 1)^{\mathrm{T}}$, where ~ denotes up to a scale factor. Homogenous coordinates and projective spaces are used throughout the field of computer vision and an abundant amount of information exists on the subject [Faugeras93][Hartley00].

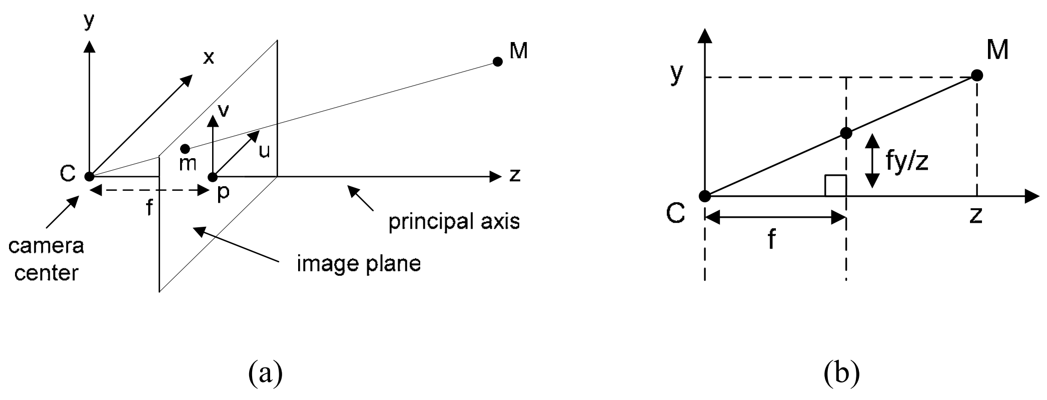


Figure 2.1: Two diagrams describing thee pinhole camera. a) The pinhole camera geometry and (b) a profile description to show the similarity of triangles.

Unless otherwise mentioned, the rest of this paper will assume homogenous representation. The pinhole camera is simply expressed in homogenous coordinates as

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \tag{2.2}$$

The camera calibration matrix in the above equation,

$$\mathbf{K} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{2.3}$$

holds the internal parameters of the camera. The pinhole camera does not include parameters for rectangular pixels, non-orthogonal image axes or principal point offset. Additionally, it assumes the point $\mathbf{M}$ is in camera coordinates. The next section describes the projective camera that is much more flexible and more accurately models real cameras.

2.2 Projective camera

The pinhole camera provides the foundation for the projective camera, absent the restrictions. The *internal* parameters, which are the parameters that model the internal aspects of the camera, include the focal length f, aspect ratio α/β (rectangular pixels), skew s (non-orthogonal image axes) and principal point (u_0, v_0) (location of image center).

The rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t} = (t_x, t_y, t_z)^{\mathrm{T}}$ comprise the *external* parameters which transfer $\mathbf{M}$ to camera coordinates as illustrated in Figure 2.2. Although the rotation matrix may be parameterized different ways, such as axis/angle, quaternions or Cayley-Klein parameters, we will assume the rotation matrix is parameterized as Euler angles (χ, φ, γ).

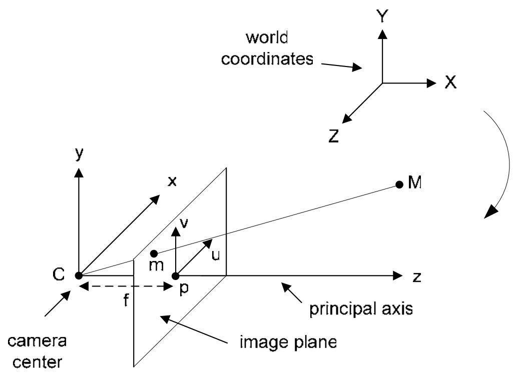


Figure 2.2: Diagram of the projective camera geometry.

Aspect ratio, skew and principal point: Whereas the pinhole camera assumes the two image axes have equal scale in both directions. CCD cameras on the other hand have the possibility of having non-square pixels. The pixels per unit distance in the x and y directions are m_x and m_y respectively. Then the inhomogeneous representation of the mapping $(X,Y,Z)^{\mathrm{T}} \mapsto (u,v)^{\mathrm{T}}$ is

$$(X,Y,Z)^{\mathrm{T}} \mapsto \left(fm_x\, X/Z\,,\, fm_y\, Y/Z\right)^{\mathrm{T}} = (u,v)^{\mathrm{T}}. \tag{2.4}$$

In lower quality cameras, the image axes might not be orthogonal. We can include this non-orthogonality with the skew parameter $s = \tan\theta$ as

$$(X,Y,Z)^{\mathrm{T}} \mapsto \left(fm_x\, X/Z + sY/Z\,,\, fm_y\, Y/Z\right)^{\mathrm{T}} = (u,v)^{\mathrm{T}}. \tag{2.5}$$

The principal point (u_0, v_0) can be included as an offset into the image, which can be written in the inhomogeneous representation as

$$(X,Y,Z)^{\mathrm{T}} \mapsto \left(fm_x\, X/Z + sY/Z + u_0\,,\, f\, m_y Y/Z + v_0\right)^{\mathrm{T}} = (u,v)^{\mathrm{T}}. \tag{2.6}$$

Keeping in mind the origin of the image is usually in the upper left corner. Homogenous representation combines all these as

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim \begin{pmatrix} \alpha & s & u_0 & 0 \\ 0 & \beta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.7}$$

with $\alpha = fm_x$ and $\beta = fm_y$. The camera calibration matrix for the projective camera is

$$\mathbf{K} = \begin{pmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.8}$$

Rotation and translation: The projective camera does not restrict the point in space $\mathbf{M}$ to be in camera coordinates. This other coordinates system, the *world coordinate frame*, is related to camera coordinates via a rotation $\mathbf{R}$ and translation $\mathbf{t}$. The projective camera, represented by $\mathbf{P}$ and called the projection matrix, can be concisely written as

$$\mathbf{P} = \mathbf{K}[\mathbf{R} \mid \mathbf{t}] \tag{2.9}$$

with $\mathbf{R}$ a 3×3 rotation matrix:

$$\mathbf{R} = \begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{12} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{2.10}$$

and a $\mathbf{t}$ a 3×1 translation vector

$$\mathbf{t} = \left(t_x, t_y, t_z\right)^{\mathrm{T}}. \tag{2.11}$$

2.3 Properties and parameterization of R

The rotation matrix has a certain set of properties that all rotation matrices share. If these properties are not satisfied, or are almost satisfied, this will introduce numerical errors in the computations. This section defines a rotation matrix, its properties and then describes common methods of parameterization.

According to *Euler's Rotation Theorem*, a rotation in Euclidean 3D-space can be represented with three parameters. The process of parameterization is representing a 3×3 rotation matrix with a reduced set of parameters. For example, a rotation matrix can be parameterized into Euler angles consisting of three parameters. However, other parameterizations such as quaternions have four parameters, but have advantages over Euler angles.

2.3.1 Definition of R

A rotation matrix $\mathbf{R} \in \Re^{n \times n}$ is an orthogonal matrix such that $\mathbf{R}\mathbf{R}^{\mathrm{T}} = \mathbf{R}^{\mathrm{T}}\mathbf{R} = \mathbf{I}$ and $\det(\mathbf{R}) = 1$ for proper rotation. Equivalently, denoting the column vectors as $\mathbf{r}_i$ then the equality $\mathbf{r}_i^{\mathrm{T}}\mathbf{r}_j = \delta_{ij}$ must be satisfied, where δ_{ij} is the Kronecher symbol

$$\delta_{ij} = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ if } i \neq j \end{cases}. \tag{2.12}$$

2.3.2 Euler angles

We know from Euler's Rotation Theorem that any rotation may be described by three parameters. A rotation matrix can be computed by considering the rotation around each axis, called the *pitch (tilt)* χ, *yaw (azimuth)* φ and *roll* γ. The direction of rotation is assumed to be in the clockwise direction around the axis when looking down axis from the origin as in Figure 2.3.

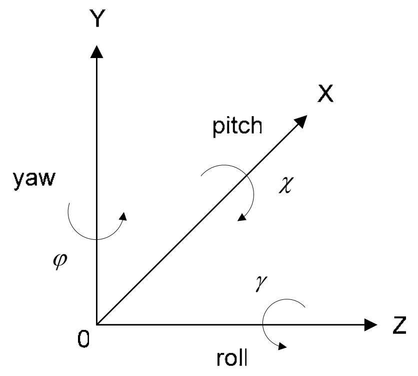


Figure 2.3: Diagram of pitch, yaw and roll.

Converting from Euler angles to rotation matrix

The three rotation matrices corresponding to the three axes are constructed as

$$\begin{aligned}\mathbf{R}_x &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\chi) & \sin(\chi) \\ 0 & -\sin(\chi) & \cos(\chi) \end{pmatrix} \\ \mathbf{R}_y &= \begin{pmatrix} \cos(\varphi) & 0 & -\sin(\varphi) \\ 0 & 1 & 0 \\ \sin(\varphi) & 0 & \cos(\varphi) \end{pmatrix} \\ \mathbf{R}_z &= \begin{pmatrix} \cos(\gamma) & \sin(\gamma) & 0 \\ -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix}\end{aligned} \tag{2.13}$$

and the rotation matrix is the product of the individual rotations

$$\mathbf{R} = \mathbf{R}_z \mathbf{R}_y \mathbf{R}_x . \tag{2.14}$$

Converting from rotation matrix to Euler angles

There are numerous ways to extract the Euler angles from the rotation matrix. A straightforward technique is

$$\begin{aligned}\varphi &= \sin^{-1} r_{31} \\ \chi &= \tan^{-1}\left(-\frac{r_{32}}{\cos\varphi}, \frac{r_{33}}{\cos\varphi} \right) \\ \gamma &= \tan^{-1}\left(-\frac{r_{21}}{\cos\varphi}, \frac{r_{11}}{\cos\varphi} \right)\end{aligned} \tag{2.15}$$

where $\tan^{-1}$ is the two argument inverse tangent.

2.3.3 Axis/angle

An arbitrary rotation matrix $\mathbf{R}$ can be represented as a rotation around an axis $\mathbf{n} = (n_1, n_2, n_3)^{\mathrm{T}}$ by a rotation angle θ. The rotation axis has 2-DOF, since only the direction is important. The rotation angle adds one more DOF making 3-DOF, which is consistent with Euler's Rotation Formula. The four axis/angle parameters can be concisely written using only three parameters as

$$\begin{aligned}\omega &= (\omega_x, \omega_y, \omega_z) \\ &= \frac{\mathbf{n}}{\|\mathbf{n}\|}\theta\end{aligned} \tag{2.16}$$

with the magnitude of ω being the angle of rotation θ and its vector components describe the axis of rotation. However, if using this parameterization in unconstrained nonlinear optimization, keep in mind that $\|\omega\|$ is not guaranteed to be θ during the iteration process when using this minimal representation.

Converting from axis/angle to rotation matrix

A rotation matrix $\mathbf{R}$ can be written as an exponential of the antisymmetric matrix $\mathbf{H}$

$$\mathbf{R} = e^{\mathbf{H}} = \sum_{n=0}^{\infty} \frac{\mathbf{H}^n}{n!} \tag{2.17}$$

with

$$\mathbf{H} = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}. \tag{2.18}$$

The *Rodrigues' Rotation Formula* is a convenient way to compute the rotation matrix $\mathbf{R}$ from the antisymmetric matrix $\mathbf{H}$ and rotation angle θ directly as

$$\mathbf{R} = \mathbf{I} + \sin\theta\,\mathbf{H} + \left(1 - \cos\theta\,\mathbf{H}^2\right). \tag{2.19}$$

Converting from rotation matrix to axis/angle

The axis of rotation and rotation angle are computed from the rotation matrix by eigenvalue decomposition. The three eigenvalues of $\mathbf{R}$ are $\left(1, \cos\theta \pm i\sin\theta\right)$. The axis of rotation is the eigenvector associated with the eigenvalue of 1, and the rotation angle is $\theta = \tan^{-1}\left(\pm\sin\theta, \cos\theta\right)$, computed from the real and imaginary parts of the remaining eigenvalues. The sign ambiguity can be resolved by converting the two possible solutions back to rotation matrices and comparing to the original rotation matrix.

2.3.4 Quaternions

A quaternion is an extension of an imaginary number denoted as

$$q = q_0 + iq_1 + jq_2 + kq_3. \tag{2.20}$$

This is usually written in vector form as

$$\mathbf{q} = \left(q_0, q_1, q_2, q_3\right)^{\mathrm{T}} = \left(q_s, \mathbf{q}_v^{\ \mathrm{T}}\right)^{\mathrm{T}} \tag{2.21}$$

with q_s the scalar component and $\mathbf{q}_v^{\ \mathrm{T}}$ the vector part. The magnitude of a quaternion is $\|\mathbf{q}\| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$ which is normalized to unity for unit quaternions. Quaternions are often used in precision applications because of their numerical stability in nonlinear optimization [Hornegger99][Schmidt01]. They are similar to axis/angle in that they have four elements, even though a rotation has 3-DOF. The unit length constraint again has to be considered during unconstrained nonlinear optimization. Schmidt and Niemann [Schmidt01] proposed a technique to use quaternions in unconstrained nonlinear optimization with results in photogrammetric bundle-adjustment.

Converting from rotation matrix to quaternion

The quaternion $\mathbf{q} = (q_0, q_1, q_2, q_3)^{\mathrm{T}}$ is computed from a rotation matrix as defined by equation 2.10 by solving the following system of equations:

$$\begin{aligned}
q_0 q_1 &= \frac{1}{4}(r_{32} - r_{23}) \\
q_0 q_2 &= \frac{1}{4}(r_{13} - r_{31}) \\
q_0 q_3 &= \frac{1}{4}(r_{21} - r_{12}) \\
q_1 q_2 &= \frac{1}{4}(r_{12} + r_{21}) \\
q_1 q_3 &= \frac{1}{4}(r_{13} + r_{31}) \\
q_2 q_3 &= \frac{1}{4}(r_{23} + r_{32}).
\end{aligned} \tag{2.22}$$

Converting from quaternion to rotation matrix

The rotation matrix corresponding to the quaternion $\mathbf{q} = (q_0, q_1, q_2, q_3)^{\mathrm{T}}$ is

$$\mathbf{R} = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{pmatrix}. \tag{2.23}$$

3 LENS MODELING

Camera systems are built from many different elements: multiple lenses, sensor element, camera assembly, etc. Ideally all these elements would fit together perfectly, to mathematical precision. However, this is never the case. Since the invention of the camera researchers in photogrammetry, and more recently computer vision have tried to model camera systems for accurate metrology, rigid and non-rigid object reconstruction, and countless other applications.

We typically think of a camera as being one that takes a perspective image, but perspective projection is not the only way to map points onto a planar surface. An example of this is the circular fisheye, which has a field-of-view (FOV) of approximately180°. With this camera projections of straight lines in the scene appear curved. This is sometimes seen in hemispherical or spherical maps of the globe. In the study of maps the question is: “how do you project a sphere onto a planar image?”. When in camera calibration the question is: “given the planar image of the object what is the projection?”.

There are two ways of modeling the way a point in space projects on the camera sensor. If we think of a perspective camera as being ideal, then we model the real camera as a deviation from a perspective camera called *radial distortion*. The other approach is not to think of an ideal camera; instead the *lens projection* is modeled directly. The difference between modeling radial distortion and projection is emphasized because both have their advantages and disadvantages. Notably, if we are modeling a quality perspective camera then the projective camera model is sufficient, and there is minimal complexity in the radial distortion model. However, if we are modeling lens projection, then there is increased complexity due to the nonlinearity of perspective projection. Other types of cameras, such as wide-angle and fisheye, are perhaps better modeled using lens projection rather than radial distortion.

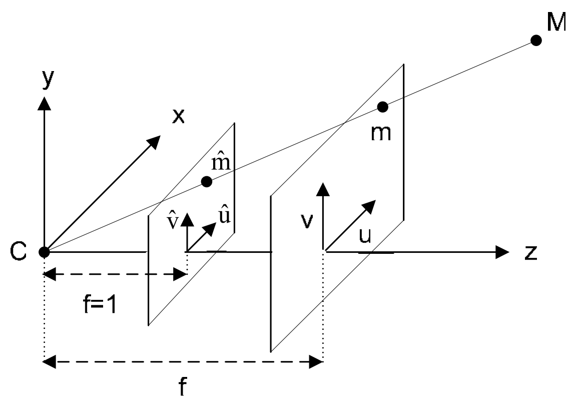


Figure 3.1: Diagram of two image planes: the true image plane and normalized image plane.

We will first setup a few ideals and notation. The image points $\mathbf{m}_i$ are normalized to unit focal length using the inverse of the camera calibration matrix as illustrated in Figure 3.1:

$$\hat{\mathbf{m}}_i = \mathbf{K}^{-1}\mathbf{m}_i . \tag{3.1}$$

The normalized image points $\hat{\mathbf{m}}_i = (\hat{u}_i, \hat{v}_i)^{\mathrm{T}}$ are then converted to polar coordinates (r_i, θ_i)

$$\begin{aligned} r_i &= \sqrt{\hat{u}_i^2 + \hat{v}_i^2} \\ \theta_i &= \tan^{-1}\left(\frac{\hat{v}_i}{\hat{u}_i}\right) \end{aligned} \tag{3.2}$$

and $\phi_i = \tan^{-1}(r_i / f)$ the angle between the principal axis and the incoming ray. Since the points are assumed to be normalized to unit focal length $f = 1$.

Section 3.1 describes some basic lens projections and section 3.2 discusses modeling radial distortion, along with other types of distortion commonly seen in camera calibration.

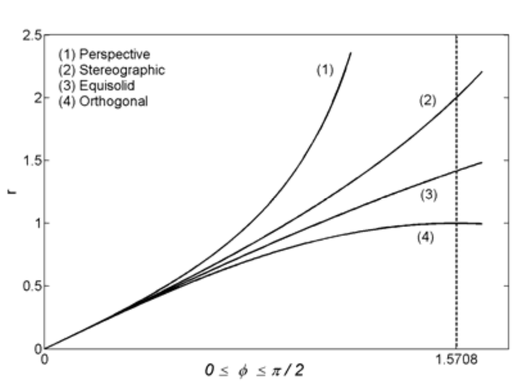
3.1 Modeling lens projection

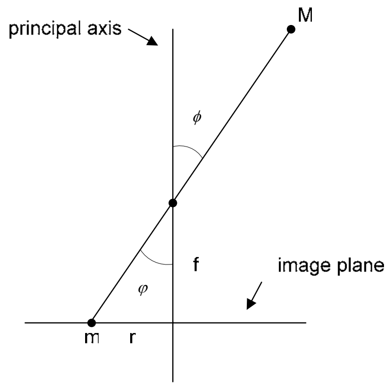
Cameras are typically built to follow a perspective projection model probably because that is the way humans perceive the world. The lenses that are mounted on these cameras are called rectilinear lenses which map lines in the world to lines in the image. Lenses of this type include normal and telephoto lenses. For normal lenses, the field-of-view (FOV) is around 40-50 degrees, and telephoto lenses can have a FOV as small a 1 degree. For normal and telephoto use, rectilinear lenses are desirable. However, the perspective projection has an asymptote at 180° FOV, as illustrated in Figure 3.2a, which causes objects to appear stretched near the edge of the image. This makes it impossible to build a rectilinear lens with 180° FOV, and extremely difficult to build a rectilinear lens above 100° FOV. Other types of projections have been proposed or used which overcome these problems and are listed in Table 3.1.

Figure 3.2a shows the geometry and behavior of these projections. When modeling the lens projection, the radial distance r is a function of the angle between the principal axis and the incident ray from the world point. These projections map lines which do not run through the center of the image to curves. Objects near the edge of the image are no longer stretched, but they are distorted. Lenses of this type, called wide-angle lenses, usually have a FOV greater than 50°.

Table 3.1: Types of lens projections.

	Name	Formula
1	Perspective	$r = f \tan \phi$
2	Stereographic	$r = 2f \tan(\phi/2)$
3	Equidistant	$r = f\phi$
4	Orthogonal	$r = f \sin \phi$





(a) (b)

Figure 3.2: Lens projection diagram. a) Plot of the ideal projections in Table 3.1 and (b) a diagram of the geometry.

Two specific wide-angle lenses are full-frame fisheye, with a FOV of 180° across the diagonal, and circular fisheye, with a FOV of 180° in all directions.

In practice, real cameras do not exactly follow the projections in Table 3.1. We use a polynomial of the following form to approximate the real lens projection:

$$r(\phi) = \kappa_1\phi + \kappa_2\phi^3 + \kappa_3\phi^5 + \dots \tag{3.3}$$

3.2 Modeling distortion

There are several different types of distortion, which commonly occur due to imperfections of the lens design and the manufacturing process. Projection modeling, which was discussed in the previous section, and radial distortion modeling are closely related. One or the other, not both, needs to be performed depending on the application.

3.2.2 Radial distortion

Modeling distortion differs from modeling lens projection in that it is a function of the radial distance r of point $\mathbf{m}$, with $\mathbf{m}$ the perspective projection of point $\mathbf{M}$ as illustrated in Figure 3.3. Modeling distortion is usually done using a polynomial [Slama80] of the form

$$r_d(r) = \kappa_1 r + \kappa_2 r^3 + \kappa_3 r^5 + \dots \tag{3.4}$$

Alternative models have been proposed for different types of cameras. Basu and Licardie [Basu95] used a logarithmic model for the fisheye

$$r_d(r) = s\log(1 + \lambda r). \tag{3.5}$$

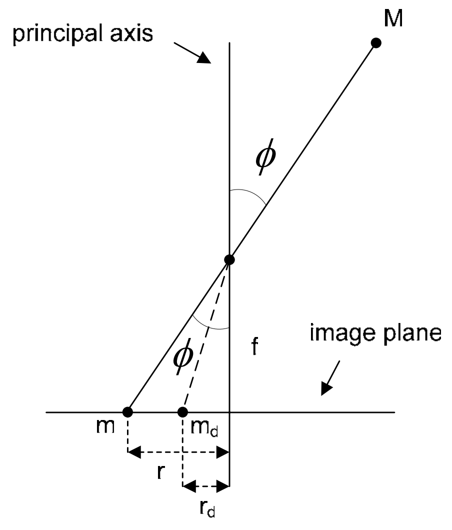


Figure 3.3: Diagram of the geometry for modeling radial distortion.

The advantage of the logarithmic model is in the stability of the nonlinear optimization, partly contributed to having only one parameter, and the asymptotic behavior of a logarithm function. However, it sacrifices flexibility to achieve this.

3.2.2 Decentering distortion

Optical systems are generally a composite of lens elements which are subject to a various amount of decentering distortion [Brown66][Slama80][Weng92]. This occurs when the centers of the lens elements are not strictly collinear. This type of distortion has a radial and tangential component, which just means the distortion acts differently on the image axes. It can be modeled as

$$\mathbf{\Delta d} = \begin{pmatrix} \left(2\rho_1 uv + \rho_2\left(r^2 + u^2\right)\right)\left(1 + \rho_3 r + \rho_4 r^3 + \ldots\right) \\ \left(\rho_1\left(r^2 + v^2\right) + 2\rho_2 uv\right)\left(1 + \rho_3 r + \rho_4 r^3 + \ldots\right) \end{pmatrix} \tag{3.6}$$

with $(\rho_1, \rho_2, \ldots)$ the decentering distortion coefficients. The coefficients ρ_1 and ρ_2 are typically the only ones used in practice, neglecting higher order terms.

3.2.3 Thin prism distortion

Thin prism distortion occurs due to imperfections in the lens design and manufacturing, as well as camera assembly [Weng92]. It too acts in the radial and tangential directions and can be expressed as

$$\mathbf{\Delta s} = \begin{pmatrix} s_1 r^2 \\ s_2 r^2 \end{pmatrix}. \tag{3.7}$$

Higher order terms can be included but rarely are in practice. In actuality, thin prism distortion can be neglected and compensated for by higher order radial and decentering distortion models [Folm-Hansen99].

4 CALIBRATION AND MODEL SELECTION

There are several techniques to estimate the parameters of a camera. Early methods derive explicit solutions to the camera parameters discussed in Chapter 2. The classic method in computer vision was developed by Tsai [Tsai86][Tsai87] which merged computer vision and photogrammetry, and also Abidi and Eason [Abidi85]. More recent methods are based on projective geometry [Zhang00A][Hartley00][Heikkila00]. These methods first estimate the projection matrix and then extract the camera parameters from it. If the projection matrix is estimated from a coplanar model, rather than a 3D model, then difficulties arise in extracting the intrinsic parameters. In this case, multiple images of a coplanar target are needed. However, all parameters can be extracted from a projection matrix computed from a single 3D model. The type of target used is application dependent. If in a laboratory setting, then precision coplanar and 3D targets are readily available. However, in the field coplanar targets are probably easier to handle.

Section 4.1 is a review on 2D homography and 3D projection matrix estimation. A 2D homography maps from 2D to 2D space, and a 3D projection matrix maps from 3D to 2D space. Section 4.2 discusses parameters extraction techniques that use multiple images of a coplanar target and also different factorization methods.

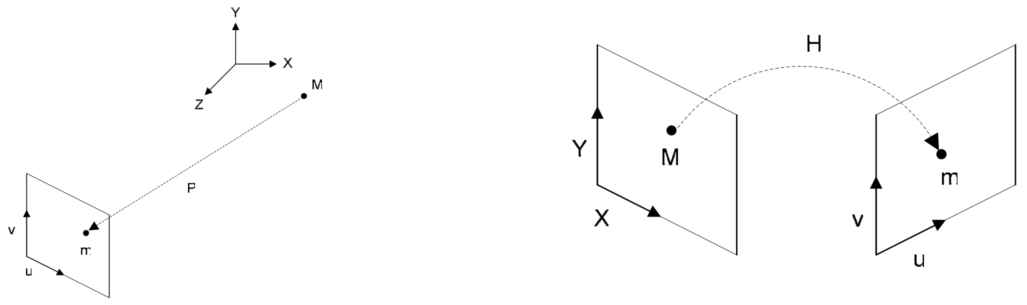
4.1 Homography and projection matrix estimation

The projection matrix $\mathbf{P}$ maps points in world space $\mathbf{M} = (X, Y, Z)$ to points in image space $\mathbf{m} = (u, v)$. When using a 3D model, the world point $\mathbf{M} \in \Re^3$ is mapped to image point $\mathbf{m}$ via the projection matrix $\mathbf{P} : \Re^3 \mapsto \Re^2$ with $\mathbf{P}$ a 3×4 matrix . The transformation is given by the equation $\mathbf{m} = \mathbf{PM}$. Note that the transformation operates in homogenous coordinates, thus $\mathbf{m}$ and $\mathbf{M}$ are augmented with a 1 prior to the operation. In the coplanar case, which uses a 2D model, the point $\mathbf{M} \in \Re^2$ is mapped to point $\mathbf{m}$ via a homography $\mathbf{H} : \Re^2 \mapsto \Re^2$ with $\mathbf{H}$ a 3×3 matrix. A 3×4 projection matrix can be computed from the homography using the orthogonality constraint of the rotation matrix. There are benefits of both methods. Notably, a 3D model will give more accurate calibration, but a precise model is difficult to build. However, calibration techniques which use multiple images of coplanar patterns are highly accurate and the ease of creating the model is desirable. Figure 4.1 illustrate the difference between estimating the projection matrix using 3D and 2D world points.

The next section describes estimating the homography from 2D world points. This is then extended to include 3D world points in section 4.1.2 to compute the projection matrix directly. Section 4.2 describes techniques to compute the projection matrix from a set of homographies and parameterize the projection matrix into physical parameters.

4.1.1 Homography estimation

This section reviews three techniques to estimate 2D homographies. We start with a linear solution (DLT), followed by linear solution with normalization (NDLT), and finally the NDLT with an optimization step. The normalization step adds a significant improvement over the linear solution, especially when noise is present.



(a) (b)

Figure 4.1: Two images illustrating the difference between the projection matrix and homography. a) The projection matrix $\mathbf{P}$ maps the 3D point $\mathbf{M}$ in world coordinates to the 2D point $\mathbf{m}$ in image coordinates. (b) The homography $\mathbf{H}$ maps the 2D point $\mathbf{M}$ to $\mathbf{m}$.

2D direct linear transformation (DLT)

The linear transformation $\mathbf{H} \in \Re^{3\times 3}$ can be computed using the DLT given a set of at least four correspondences $\mathbf{m}_i \leftrightarrow \mathbf{M}_i$ so that $\mathbf{m}_i = \mathbf{H}\mathbf{M}_i$ with $\mathbf{H}$ a 2D projective transformation. The product $\mathbf{H}\mathbf{M}_i$ may be written as

$$\mathbf{H}\mathbf{M}_i = \begin{pmatrix} \mathbf{h}^{1\mathrm{T}}\mathbf{M}_i \\ \mathbf{h}^{2\mathrm{T}}\mathbf{M}_i \\ \mathbf{h}^{3\mathrm{T}}\mathbf{M}_i \end{pmatrix} \tag{4.1}$$

with $\mathbf{h}^{i\mathrm{T}}$ the i^{th} row of $\mathbf{H}$. Taking the cross product of $\mathbf{m}_i$ and $\mathbf{H}\mathbf{M}_i$

$$\mathbf{m}_i \times \mathbf{H}\mathbf{M}_i = \begin{pmatrix} v_i \mathbf{h}^{3\mathrm{T}} \mathbf{M}_i - w_i \mathbf{h}^{2\mathrm{T}} \mathbf{M}_i \\ w_i \mathbf{h}^{1\mathrm{T}} \mathbf{M}_i - u_i \mathbf{h}^{3\mathrm{T}} \mathbf{M}_i \\ u_i \mathbf{h}^{2\mathrm{T}} \mathbf{M}_i - v_i \mathbf{h}^{1\mathrm{T}} \mathbf{M}_i \end{pmatrix} \tag{4.2}$$

which may be written as a homogenous system

$$\begin{pmatrix} w_i \mathbf{M}_i^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & -u_i \mathbf{M}_i^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} & -w_i \mathbf{M}_i^{\mathrm{T}} & v_i \mathbf{M}_i^{\mathrm{T}} \\ -v_i \mathbf{M}_i^{\mathrm{T}} & u_i \mathbf{M}_i^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \tag{4.3}$$

Alternatively, since the rows are linearly dependent only the first two rows are needed

$$\begin{pmatrix} w_i \mathbf{M}_i^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & -u_i \mathbf{M}_i^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} & -w_i \mathbf{M}_i^{\mathrm{T}} & v_i \mathbf{M}_i^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \tag{4.4}$$

This has the form $\mathbf{A}_i \mathbf{h} = \mathbf{0}$, stacking all these equations makes $\mathbf{A}$ a $2n \times 9$ matrix with n the number of correspondences. The solution of the homogenous system $\mathbf{A}\mathbf{h} = \mathbf{0}$ is the right singular vector associated with the smallest singular value, or equivalently the eigenvector of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ associated with the smallest eigenvalue. The 9 element vector $\mathbf{h}$ makes up the components of the homography matrix $\mathbf{H}$

$$\mathbf{H} = \begin{pmatrix} \mathbf{h}^{1\mathrm{T}} \\ \mathbf{h}^{2\mathrm{T}} \\ \mathbf{h}^{3\mathrm{T}} \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}. \tag{4.5}$$

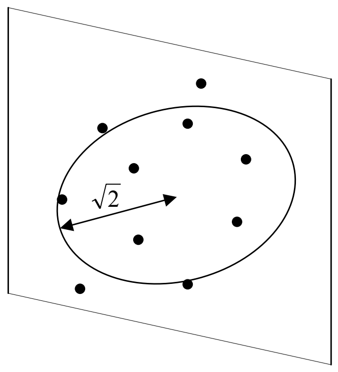


Figure 4.2: Normalized 2D points so the centroid of the points is at the origin and the average distance from the origin is $\sqrt{2}$.

2D normalized direct linear transformation (2D NDLT)

Hartley [Hartley97][Hartley00] used a simple normalizing transformation before applying eight-point algorithm to compute the fundamental matrix which produced results comparable to the best iterative algorithms. The same normalization technique can be used in conjunction with the DLT. The normalizing transformation $\mathbf{T}$ is a similarity transformation that a) translates the centroid of the points to the origin and b) scales the points so the average distance from the origin is $\sqrt{2}$ in the planar case as illustrated in Figure 4.2.

This can be accomplished by the following similarity transformation

$$\mathbf{T} = \begin{pmatrix} \frac{1}{\overline{|\mathbf{X}-\overline{\mathbf{X}}|}} & 0 & -\frac{\overline{\mathbf{X}}}{\overline{|\mathbf{X}-\overline{\mathbf{X}}|}} \\ 0 & \frac{1}{\overline{|\mathbf{Y}-\overline{\mathbf{Y}}|}} & -\frac{\overline{\mathbf{Y}}}{\overline{|\mathbf{Y}-\overline{\mathbf{Y}}|}} \\ 0 & 0 & 1 \end{pmatrix} \tag{4.6}$$

where $\mathbf{X} = (x_1, x_2, \ldots, x_n)$ and $\mathbf{Y} = (y_1, y_2, \ldots, y_n)$ are the coordinates of the 2D points. Two different normalizing transformations are applied, $\mathbf{T}$ and $\mathbf{T}'$, on the model and image points, respectively

$$\begin{aligned} \widetilde{\mathbf{M}} &= \mathbf{TM} \\ \widetilde{\mathbf{m}} &= \mathbf{T}'\mathbf{m}. \end{aligned} \tag{4.7}$$

The DLT algorithm is used on the normalized data yielding a transformation $\widetilde{\mathbf{H}}$ such that $\widetilde{\mathbf{m}}_i = \widetilde{\mathbf{H}}\widetilde{\mathbf{M}}_i$. The matrix $\widetilde{\mathbf{H}}$ can be represented as a 9×1 vector $\mathbf{h} = \left(\widetilde{\mathbf{h}}^{1\mathrm{T}}, \widetilde{\mathbf{h}}^{2\mathrm{T}}, \widetilde{\mathbf{h}}^{3\mathrm{T}}\right)^{\mathrm{T}}$ with $\widetilde{\mathbf{h}}^{i\mathrm{T}}$ the i^{th} row of $\widetilde{\mathbf{H}}$. The DLT on the normalized data is

$$\begin{pmatrix} \widetilde{\mathbf{M}}^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & -\widetilde{u}\widetilde{\mathbf{M}}^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} & -\widetilde{\mathbf{M}}^{\mathrm{T}} & \widetilde{v}\widetilde{\mathbf{M}}^{\mathrm{T}} \end{pmatrix} \widetilde{\mathbf{h}} = 0 \tag{4.7}$$

with $\widetilde{\mathbf{m}} = (\widetilde{u}, \widetilde{v})^{\mathrm{T}}$. Letting $\mathbf{H} = \mathbf{T}'^{-1}\widetilde{\mathbf{H}}\mathbf{T}$ recovers the homography $\mathbf{m} = \mathbf{HM}$ on the actual data.

Normalized 2D DLT

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\mathbf{M}_i \leftrightarrow \mathbf{m}_i$ compute a linear estimate of $\mathbf{H}$.

Algorithm

1. Compute normalized model points $\tilde{\mathbf{M}} = \mathbf{TM}$ and image points $\tilde{\mathbf{m}} = \mathbf{T}'\mathbf{m}$
2. Compute normalized projective transformation matrix $\tilde{\mathbf{H}}$ using DLT
3. Denormalize $\mathbf{H} = \mathbf{T}'^{-1}\tilde{\mathbf{H}}\mathbf{T}$

2D Gold Standard Algorithm

The Gold Standard Algorithm [Hartley00] follows directly from the normalized DLT with an optimization step on the normalized homography $\mathbf{H}$. The linear solution is used as the initial guess for Maximum Likelihood Estimation (MLE)

Gold Standard Algorithm for 2D projective transformation

Objective

Given $n \geq 4$ 2D to 2D point correspondences $\mathbf{M} \leftrightarrow \mathbf{m}$ determine the maximum likelihood estimate of the homography matrix $\mathbf{H}$.

Algorithm

1. Linear solution
    a. Compute normalized model points $\tilde{\mathbf{M}} = \mathbf{TM}$ and image points $\tilde{\mathbf{m}} = \mathbf{T}'\tilde{\mathbf{m}}$
    b. Compute normalized transformation matrix $\tilde{\mathbf{H}}$ using DLT
2. Minimize geometric error with the linear estimate as the initial guess.

$$\min_{\tilde{\mathbf{H}}} \sum_i d\left(\tilde{\mathbf{m}}_i, \tilde{\mathbf{H}}\tilde{\mathbf{M}}_i\right)^2$$

3. Denormalize $\mathbf{H} = \mathbf{T}'^{-1}\tilde{\mathbf{H}}\mathbf{T}$

4.1.2 Projection matrix estimation

This section is similar to the previous section on estimating the homography, so we will simply extend some of the ideas to estimate the projection matrix. The only difference in the 2D DLT and 3D DLT is that $\mathbf{M}$ has an extra component and $\mathbf{P}$ is a 3×4 matrix. The DLT for the projection matrix is

$$\begin{pmatrix} \mathbf{M}_i^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & -u_i\mathbf{M}_i^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} & -\mathbf{M}_i^{\mathrm{T}} & v_i\mathbf{M}_i^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \mathbf{p}^1 \\ \mathbf{p}^2 \\ \mathbf{p}^3 \end{pmatrix} = \mathbf{0} \tag{4.8}$$

with $\mathbf{p}^{i\mathrm{T}}$ the i^{th} row of $\mathbf{P}$. This has the form $\mathbf{A}_i\mathbf{p} = \mathbf{0}$, stacking all the equations makes $\mathbf{Ap} = \mathbf{0}$ with $\mathbf{A}$ a $2n \times 12$ matrix. This homogenous system is solved in exactly the same way as was done with homography estimation. Similarly, the normalizing transformation used in the 3D case a) translates the centroid of the points to the origin and b) scales the points so the average distance from the origin is $\sqrt{3}$. The transformation applied to the 3D model points is

$$\mathbf{T} = \begin{pmatrix} \frac{1}{\overline{|\mathbf{X}-\overline{\mathbf{X}}|}} & 0 & 0 & -\frac{\overline{\mathbf{X}}}{\overline{|\mathbf{X}-\overline{\mathbf{X}}|}} \\ 0 & \frac{1}{\overline{|\mathbf{Y}-\overline{\mathbf{Y}}|}} & 0 & -\frac{\overline{\mathbf{Y}}}{\overline{|\mathbf{Y}-\overline{\mathbf{Y}}|}} \\ 0 & 0 & \frac{1}{\overline{|\mathbf{Z}-\overline{\mathbf{Z}}|}} & -\frac{\overline{\mathbf{Z}}}{\overline{|\mathbf{Z}-\overline{\mathbf{Z}}|}} \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{4.9}$$

We now have the tools to compute the projection matrix using the Gold Standard Algorithm, which is outlined below.

Gold Standard Algorithm for computing the projection matrix

Objective

Given $n \geq 6$ 3D to 2D point correspondences $\mathbf{M} \leftrightarrow \mathbf{m}$ determine the maximum likelihood estimate of the projection matrix $\mathbf{P}$.

Algorithm

4. Linear solution
   c. Compute normalized model points $\widetilde{\mathbf{M}} = \mathbf{TM}$ and image points $\widetilde{\mathbf{m}} = \mathbf{T}'\widetilde{\mathbf{m}}$
   d. Compute normalized projection matrix $\widetilde{\mathbf{P}}$ using DLT
5. Minimize geometric error with the linear estimate as the initial guess.

$$\min_{\widetilde{\mathbf{P}}} \sum_i d\left(\widetilde{\mathbf{m}}_i, \widetilde{\mathbf{P}}\widetilde{\mathbf{M}}_i\right)^2$$

6. Denormalize $\mathbf{P} = \mathbf{T}'^{-1}\widetilde{\mathbf{P}}\mathbf{T}$

4.2 Extracting physical parameters from P

There are a myriad of methods to extract the camera parameters from the projection matrix. This section reviews a few techniques for planar models, which require multiple images, and for 3D models.

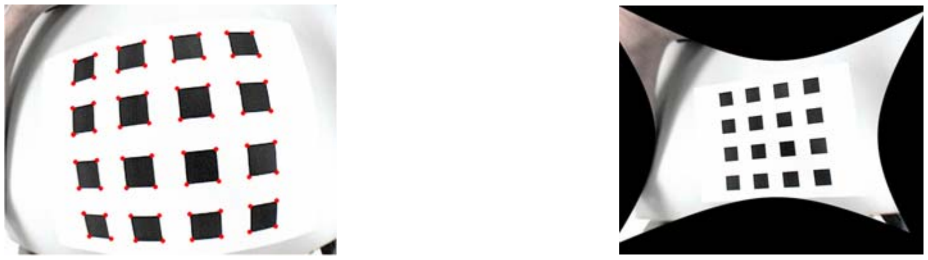


Figure 4.3: An example of a planar calibration model. (a) Original calibration target with the control points shown in red and (b) corresponding image corrected towards a perspective projection.

4.2.1 Coplanar model

There are two main techniques to extract the camera parameters using planar models. Figure 4.3 shows one example of a planar model taken from a wide-angle camera. The first method assumes some prior knowledge, specifically the camera calibration matrix, and only requires a single image. The second method estimates all the parameters, including the camera calibration matrix, using multiple images.

Known K

We use the technique in section 3.1 to compute the homography $\mathbf{H}$ such that $\mathbf{m}_i = \mathbf{HM}_i$. Here, the rotation matrix and translation vector are extracted from the homography using a known camera calibration matrix $\mathbf{K}$. The camera calibration matrix could have been computed u

sing other techniques, or an estimate could be used based on expected values. The 3×3 homography $\mathbf{H}$ can be written as

$$\begin{aligned} \mathbf{H} &= \mathbf{K}[\mathbf{R}_{3\times2} \quad \mathbf{t}] \\ &= [\mathbf{K}\mathbf{R}_{3\times2} \quad \mathbf{K}\mathbf{t}] \end{aligned} \tag{4.10}$$

where $\mathbf{R}_{3\times2}$ is the first 3×2 submatrix of the rotation matrix $\mathbf{R}$. Then $\mathbf{K}$ and the first 3×2 submatrix of $\mathbf{H}$ are used to recover the orientation

$$\mathbf{R}_{3\times2} = \mathbf{K}^{-1}\mathbf{H}_{3\times2}. \tag{4.11}$$

Since a rotation matrix is orthogonal the last column of $\mathbf{R}$ is $\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2$, and the full rotation matrix is $\mathbf{R} = (\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3)$. The translation vector is

$$\mathbf{t} = \mathbf{K}^{-1}\mathbf{h}_3 \tag{4.12}$$

with $\mathbf{h}_3$ the last column of $\mathbf{H}$. The rotation matrix and translation vector are then scaled by dividing through by $\lambda = \sqrt{\sum \mathbf{r}_1}$ where $\sum \mathbf{r}_1$ is the summation of the components of the first column vector of $\mathbf{R}$.

Closed-form solution from IAC

Zhang [Zhang00A] used the Image of the Absolute Conic (IAC) to parameterize a set of homographies computed from multiple images of a 2D model, since typical methods based on RQ factorization and Cholesky decomposition do not work for a projection matrix computed from a 2D model. The IAC $\omega = \mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}$ and the homography $\mathbf{H} = \mathbf{K}(\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t})$ relating a model plane in the world coordinate system to its image are used to place two constraints on the intrinsic parameters. Given a homography $\mathbf{H}$ we may write

$$\mathbf{H}^{\mathrm{T}}\omega\,\mathbf{H} = \mathbf{H}^{\mathrm{T}}\mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}\mathbf{H} \tag{4.13}$$

and since $\mathbf{R}$ is orthonormal: $\mathbf{h}_1^{\mathrm{T}}\omega\,\mathbf{h}_1 = 1$, $\mathbf{h}_2^{\mathrm{T}}\omega\,\mathbf{h}_2 = 1$ and $\mathbf{h}_1^{\mathrm{T}}\omega\,\mathbf{h}_2 = 0$. Hence the two constraints are

$$\begin{aligned}\mathbf{h}_1^{\mathrm{T}}\omega\,\mathbf{h}_1 - \mathbf{h}_2^{\mathrm{T}}\omega\,\mathbf{h}_2 &= 0\\ \mathbf{h}_1^{\mathrm{T}}\omega\,\mathbf{h}_2 &= 0.\end{aligned} \tag{4.14}$$

Rewriting ω in terms of (α,β,s,u_0,v_0) gives

$$\omega \sim \begin{pmatrix}\omega_{11} & \omega_{12} & \omega_{13}\\ \omega_{21} & \omega_{22} & \omega_{23}\\ \omega_{31} & \omega_{32} & \omega_{33}\end{pmatrix} = \begin{pmatrix} \dfrac{1}{\alpha^2} & -\dfrac{s}{\alpha^2\beta} & \dfrac{sv_0 - u_0\beta}{\alpha^2\beta}\\ -\dfrac{s}{\alpha^2\beta} & \dfrac{s^2}{\alpha^2\beta^2}+\dfrac{1}{\beta^2} & \dfrac{s(sv_0-u_0\beta)}{\alpha^2\beta^2}+\dfrac{v_0}{\beta^2}\\ \dfrac{sv_0-u_0\beta}{\alpha^2\beta} & \dfrac{s(sv_0-u_0\beta)}{\alpha^2\beta^2}+\dfrac{v_0}{\beta^2} & \dfrac{(sv_0-u_0\beta)^2}{\alpha^2\beta^2}+\dfrac{v_0}{\beta^2}+1\end{pmatrix}$$

which is a symmetric matrix that may be defined by a 6-tuple vector $\hat{\omega} = (\omega_{11},\omega_{12},\omega_{22},\omega_{13},\omega_{23},\omega_{33})^{\mathrm{T}}$. Letting the i^{th} column of $\mathbf{H}$ be $\mathbf{h}_i = (h_{1i},h_{2i},h_{3i})^{\mathrm{T}}$ we may write

$$\mathbf{h}_i^{\mathrm{T}}\hat{\omega}\,\mathbf{h}_j = v_{ij}^{\mathrm{T}}\hat{\omega} \tag{4.15}$$

with

$$v_{ij} = \left(h_{1i}h_{1j},\, h_{1i}h_{2j}+h_{2i}h_{1j},\, h_{2i}h_{2j},\, h_{3i}h_{1j}+h_{1i}h_{3j},\, h_{3i}h_{2j}+h_{2i}h_{3j},\, h_{3i}h_{3j}\right)^{\mathrm{T}}. \tag{4.16}$$

Combining the two constraints in equation 4.14 as a homogenous system gives

$$\mathbf{V}\hat{\omega} = \begin{bmatrix} v_{12}^{\mathrm{T}}\\ (v_{11}-v_{22})\end{bmatrix}\hat{\omega} = \mathbf{0}. \tag{4.17}$$

If there are n images of the model plane, then stacking (4.17) makes $\mathbf{V}$ a $2n \times 6$ matrix with a unique solution when $n \geq 3$. Once we have ω we can solve for $(\alpha, \beta, s, u_0, v_0, \lambda)$ with λ a scale factor yielding

$$\begin{aligned}
v_0 &= (\omega_{12}\omega_{13} - \omega_{11}\omega_{23}) \big/ (\omega_{11}\omega_{22} - \omega_{12}^2) \\
\lambda &= \omega_{33} - \left[\omega_{13}^2 + v_0(\omega_{12}\omega_{13} - \omega_{11}\omega_{13})\right] \big/ \omega_{11} \\
\alpha &= \sqrt{\lambda / \omega_{11}} \\
\beta &= \sqrt{\lambda\omega_{11} \big/ (\omega_{11}\omega_{22} - \omega_{12}^2)} \\
s &= \omega_{12}\alpha^2\beta / \lambda \\
u_0 &= sv_0 / \alpha - \omega_{13}\alpha^2 / \lambda .
\end{aligned} \tag{4.18}$$

Once the intrinsic parameters have been solved the extrinsic parameters are computed as

$$\begin{aligned}
\mathbf{r}_1 &= \lambda\mathbf{K}^{-1}\mathbf{h}_1 \\
\mathbf{r}_2 &= \lambda\mathbf{K}^{-1}\mathbf{h}_2 \\
\mathbf{r}_3 &= \mathbf{r}_1 \times \mathbf{r}_2 \\
\mathbf{t} &= \lambda\mathbf{K}^{-1}\mathbf{h}_3
\end{aligned} \tag{4.19}$$

with $\lambda = 1 \big/ \left\|\mathbf{K}^{-1}\mathbf{h}_1\right\| = 1 \big/ \left\|\mathbf{K}^{-1}\mathbf{h}_2\right\|$.

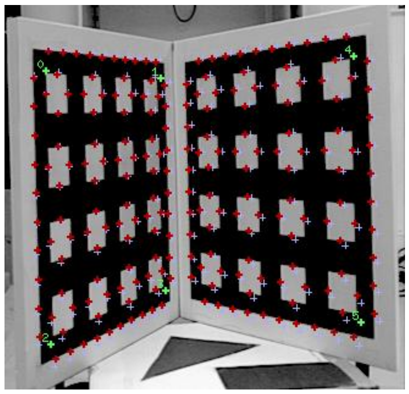


Figure 4.4: An example of a 3D calibration model.

4.2.1 Non-coplanar model

There are several techniques to extract the camera parameters when using a 3D calibration model as illustrated in Figure 4.4. In this section, we describe a few techniques that have been proposed. The decision to use a non-coplanar model depends on the application, specifically the required precision. Interestingly enough, RQ factorization, and Cholesky factorization of the IAC produce exactly the same results in our analysis on synthetic 3D data; and Faugeras' method produces nearly the same results.

Known K

Similar to recovering the orientation knowing the camera calibration matrix in the coplanar case, we can do the same in the 3D case. The projection matrix $\mathbf{P}$ can be factored as

$$\begin{aligned}\mathbf{P} &= \mathbf{K}[\mathbf{R} \quad \mathbf{t}] \\ &= [\mathbf{KR} \quad \mathbf{Kt}]\end{aligned} \tag{4.20}$$

where $\mathbf{R}$ is a rotation matrix and $\mathbf{t}$ the translation vector. Then the known $\mathbf{K}$ and the first 3×3 submatrix of $\mathbf{P}$ are used to recover the orientation

$$\mathbf{R} = \mathbf{K}^{-1}\mathbf{P}_{3\times 3}. \tag{4.21}$$

The translation vector is

$$\mathbf{t} = \mathbf{K}^{-1}\mathbf{P}_4 \tag{4.22}$$

with $\mathbf{P}_4$ the last column of $\mathbf{P}$. The rotation matrix and translation vector are then scaled by dividing through by $\lambda = \sqrt{\sum_i \mathbf{r}_{1i}^2}$ where $\sum_i \mathbf{r}_{1i}^2$ is the squared summation of the components of the first column vector of $\mathbf{R}$.

RQ factorization

The projection matrix $\mathbf{P} = \mathbf{K}[\mathbf{R} \mid \mathbf{t}]$ can be factored as $\mathbf{P} = [\mathbf{KR} \mid \mathbf{Kt}] = [\mathbf{U} \mid \mathbf{V}]$. The first 3×3 submatrix $\mathbf{U} = \mathbf{KR}$ is the product of an upper triangular and rotation matrix. RQ factorization is used to compute the camera calibration matrix $\mathbf{K}$ and rotation matrix $\mathbf{R}$. The translation vector is $\mathbf{t} = \mathbf{K}^{-1}\mathbf{V}$ with $\mathbf{V}$ the right most 3×1 vector of $\mathbf{P}$. The camera calibration matrix computed using this method will be of the form

$$\mathbf{K} = \begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.23}$$

with $\theta = \tan^{-1} s$ the angle between image axes and α/β the aspect ratio. The rotation matrix and translation vector are then scaled by dividing through by $\lambda = \sqrt{\sum \mathbf{r}_1}$ where $\sum \mathbf{r}_1$ is the summation of the components of the first column vector of $\mathbf{R}$.

Factor DIAC using Cholesky factorization

Seedahmed and Habib [Seedahmed02] used the orthogonality of $\mathbf{R}$ and Cholesky factorization to recover the camera calibration matrix. Letting $\mathbf{U} = \mathbf{KR}$, then

$$\begin{aligned} \mathbf{UU}^{\mathrm{T}} &= (\mathbf{KR})(\mathbf{KR})^{\mathrm{T}} \\ &= \mathbf{KRR}^{\mathrm{T}}\mathbf{K}^{\mathrm{T}} \\ &= \mathbf{KIK}^{\mathrm{T}} \\ &= \mathbf{KK}^{\mathrm{T}} \end{aligned} \tag{4.24}$$

since the rotation matrix is orthogonal. The product $\omega^* = \mathbf{UU}^{\mathrm{T}} = \mathbf{KK}^{\mathrm{T}}$ is the dual image of the absolute conic (DIAC). Using Cholesky decomposition the known DIAC ω^* can be factored into $\mathbf{KK}^{\mathrm{T}}$ where $\mathbf{K}$ is an upper triangular matrix. The normalized camera calibration matrix is computed by dividing through by the element in the last row and column K_{33}. The Cholesky factorization will not reveal a correct decomposition due to the missing structure in terms of lower-upper ordering. An iterative step is needed to correctly decompose ω^* using Cholesky decomposition.

Iteratively factor the DIAC to recover the camera calibration matrix

Objective

Compute the camera calibration matrix from the DIAC via Cholesky decomposition by iteratively updating the principal point.

Algorithm

1. Compute projection matrix
2. Form the DIAC ω^*
3. Apply Cholesky factorization to recover $\mathbf{K}$
4. Normalize $\mathbf{K}$ by dividing through by K_{33}
5. Extract principal point from $\mathbf{K}$
6. Displace the observed image coordinates using the principal point
7. Repeat steps 1-7 until convergence

The rotation matrix and translation vector are recovered using the same technique found in the RQ factorization method. Again, proper scaling must be done.

Factor IAC using Cholesky factorization

Seedahmed and Habib [Seedahmed02] also proposed a non-iterative algorithm that produces the correct ordering in terms of the lower-upper matrix by factoring the matrix $\omega = \left(\mathbf{U}\mathbf{U}^{\mathrm{T}}\right)^{-1}$, which is the image of the absolute conic (IAC).

Factor the IAC to recover the camera calibration matrix

Objective

Compute the camera calibration matrix from the IAC via Cholesky decomposition.

Algorithm

1. Compute projection matrix
2. Form the IAC ω
3. Apply Cholesky factorization to ω
4. Invert factorized matrix to recover $\mathbf{K}$
8. Normalize $\mathbf{K}$ by dividing through by K_{33}

Both of the methods described in [Seedahmed02] require the submatrix $\mathbf{U}$ to be positive definite. This should be the case when working with a 3D model, but not necessarily with a 2D model. Similarly, the rotation matrix and translation vector are recovered using the previously mentioned technique.

Faugeras method

Faugeras and Toscani [Faugeras87] recover the camera parameters based on the fact that $\mathbf{R}$ is orthogonal and $\mathbf{P}$ is defined up to a scale factor. All the camera parameters can be recovered as long as the scale factor k is known, which corresponds to knowing whether the world coordinates system is in front or behind the camera ($t_z < 0$ or $t_z > 0$). In accordance with the original notation found in [Faugeras87], the projection matrix can be denoted as

$$\mathbf{P} = \begin{pmatrix} \mathbf{I}_1 & I_{14} \\ \mathbf{I}_2 & I_{24} \\ \mathbf{I}_3 & I_{34} \end{pmatrix} \tag{4.25}$$

where $\mathbf{I}_1$, $\mathbf{I}_2$ and $\mathbf{I}_3$ are the 1×3 row vectors of $\mathbf{P}$, and I_{14}, I_{24} and I_{34} are the last components of each row. Then the closed-form solution to recover the camera parameters is

$$\begin{aligned}
& k = \sqrt{\mathbf{I}_3 \mathbf{I}_3^{\mathrm{T}}} \\
& t_z = \mathbf{I}_{34} / k \\
& \text{if } t_z < 0 \text{ then} \\
& \quad t_z = -t_z \\
& \quad k = -k \\
& \text{end} \\
& r^{3\mathrm{T}} = \mathbf{I}_3 / k \\
& u_0 = \mathbf{I}_1 \mathbf{I}_3^{\mathrm{T}} / k \\
& v_0 = \mathbf{I}_2 \mathbf{I}_3^{\mathrm{T}} / k \\
& \alpha = \sqrt{\mathbf{I}_1 \mathbf{I}_1^{\mathrm{T}} / \left(k^2 - u_0^2\right)} \\
& \beta = \sqrt{\mathbf{I}_2 \mathbf{I}_2^{\mathrm{T}} / \left(k^2 - v_0^2\right)} \\
& r^{1\mathrm{T}} = \left(\mathbf{I}_1 - u_0 \mathbf{I}_3\right) / (k\alpha) \\
& r^{2\mathrm{T}} = \left(\mathbf{I}_2 - v_0 \mathbf{I}_3\right) / (k\beta) \\
& t_x = \left(\mathbf{I}_{14} - u_0 \mathbf{I}_{34}\right) / (k\alpha) \\
& t_y = \left(\mathbf{I}_{24} - v_0 \mathbf{I}_{34}\right) / (k\beta) \\
& s = (1/\beta)\left(\mathbf{I}_1 \mathbf{I}_2^{\mathrm{T}} / k^2 - u_0 v_0\right)
\end{aligned} \tag{4.26}$$

where $\mathbf{r}^{i\mathrm{T}}$ is the i^{th} row of $\mathbf{R}$. The derivation of the solution is not derived here due to its length. However, the original paper does derive the solution.

4.3 Model selection

Model selection picks the best model when several competing models can be used to explain an observation. Akaike [Akaike74] laid the foundation for statistical model selection, for use in time series analysis, using what is called Information Theoretic Criterion (AIC). In AIC, the model selected is the one that minimizes the error of a new observation. It has the form

$$\mathrm{AIC} = -2\log L(\boldsymbol{\theta};\mathbf{m}_i) + 2k \tag{4.27}$$

where k is the number of parameters in the model and $L(\boldsymbol{\theta};\mathbf{m}_i)$ is the likelihood of the model parameters $\boldsymbol{\theta} = (\mathbf{K},\mathbf{R},\mathbf{t},\boldsymbol{\kappa},\boldsymbol{\rho})$ given the observations $\mathbf{m}_i$. The model with the lowest AIC score is selected according to this criterion. The first term in equation 4.27 is a measure of the goodness of fit of the model, and the second term penalizes higher complex models.

We will denote the estimated projection of point $\mathbf{M}_j$ as $\breve{\mathbf{m}}_i$ according to the model parameters $\boldsymbol{\theta}$. The sum-square-error (SSE) is computed as $\mathrm{SSE} = \sum_i r_i^2$ with $r_i = \|\mathbf{m}_i - \breve{\mathbf{m}}_i\|$ the difference between the measured and estimated image points.

Assuming the noise in the data is Gaussian distributed, the probability of $\mathbf{m}_i$ given the model $\boldsymbol{\theta}$ is the product of the individual probability density functions (PDF's) of each point, assuming the errors on all points are independent [Harltey00]. The PDF of the noise perturbed data is given by

$$\Pr(\mathbf{m}_i \mid \boldsymbol{\theta}) = \prod_i \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) e^{-r_i^2/(2\sigma^2)} \tag{4.28}$$

where σ^2 is the variance of the noise. Then the log-likelihood of the model parameters $\boldsymbol{\theta}$ given the observations $\mathbf{m}_i$ is

$$\begin{aligned}\log L(\boldsymbol{\theta};\mathbf{m}_i) &= \arg_{\boldsymbol{\theta}}[\log \Pr(\mathbf{m}_i \mid \boldsymbol{\theta})] \\ &= -\frac{1}{2\sigma^2}\sum_i r_i^2 + \text{constant.}\end{aligned} \tag{4.29}$$

The maximum log-likelihood estimate (MLE) is the set of parameters $\boldsymbol{\theta}$ that maximizes $\log L(\boldsymbol{\theta};\mathbf{m}_i)$. What we observe is that minimizing the SSE is equivalent to maximizing the log-likelihood, which is in-turn equivalent to maximizing the likelihood of the model parameters $\boldsymbol{\theta}$. Therefore, by substituting equation 4.29 into equation 4.27 and simplifying, we can write AIC in the following form:

$$\text{AIC} = \frac{1}{\sigma^2}\sum_i r_i^2 + 2k \tag{4.30}$$

Similarly, we can do the same with all the criterions in Table 4.1.

Table 4.1: List of model selection criterions.

Name	Formula
AIC [Akaike74]	$-2\log L(\boldsymbol{\theta};\mathbf{m}_i) + 2k$
MDL [Rissanen78]	$-2\log L(\boldsymbol{\theta};\mathbf{m}_i) + 1/2\, k \log N$
BIC [Schwarz78]	$-2\log L(\boldsymbol{\theta};\mathbf{m}_i) + 2k \log N$
SSD [Rissanen78]	$-2\log L(\boldsymbol{\theta};\mathbf{m}_i) + k \log[(N+2)/24] + 2\log(k+1)$
CAIC [Bozdogan87]	$-2\log L(\boldsymbol{\theta};\mathbf{m}_i) + k(\log N + 1)$

5 ALGORITHM OVERVIEW

In this chapter, we give an overview of the calibration algorithm based on the theory from Chapter 4. We first discuss the setup, and then move onto the linear solution for the camera parameters, nonlinear optimization and finally model selection.

5.1 Homography estimation

A point in the world coordinates $\mathbf{M}$ is projected to its image $\mathbf{m}$ by the projection matrix $\mathbf{P}$ which maps from $\mathrm{P}^3 \rightarrow \mathrm{P}^2$ (projective mapping):

$$\begin{aligned} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} &\sim \mathbf{K}\begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \end{pmatrix}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \\ &= \mathbf{P}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \end{aligned} \tag{5.1}$$

Letting all the points in world coordinates lie on a plane, i.e. $Z = 0$, the projection matrix reduces to a mapping from $\mathrm{P}^2 \rightarrow \mathrm{P}^2$:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim \mathbf{K}\begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{t} \end{pmatrix}\begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} = \mathbf{K}\begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{pmatrix}\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}. \tag{5.2}$$

Then the model point and image point are related by a 3×3 homography $\mathbf{H}$:

$$\mathbf{m} \sim \mathbf{HM} \text{ with } \mathbf{H} = \mathbf{K}\begin{pmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{pmatrix}. \tag{5.3}$$

We use the Gold Standard Algorithm from Chapter 3 to compute the homographies from the model plane to each of the images taken from unknown vantage points. Figure 5.1 shows two of eight images in one of the calibration sets with the model points mapped to the image via the computed homographies appearing as red dots. The red dots should correspond to the corners of the black squares. Since the images were taken with a wide-angle camera the nonlinearity of the lens projection will be considered in a later stage.

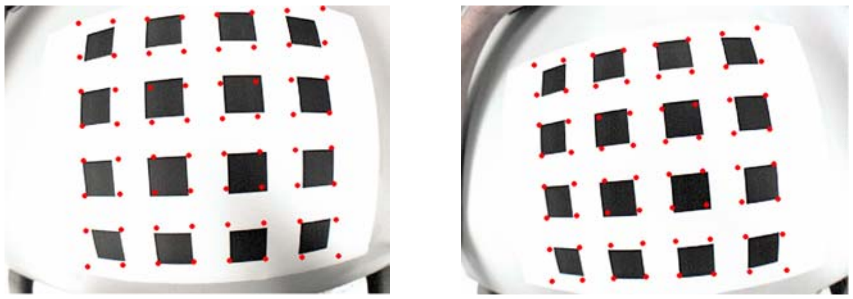


Figure 5.1: Two of eight images taken of a planar grid pattern and the mapping of the model points by the estimated homographies overlaid on the image without considering distortion.

5.2 Solving for intrinsic parameters

With multiple images of the model plane and all the homographies computed which map the model points to the image points; the intrinsic parameters are extracted from the homographies using the technique described by Zhang [Zhang00A] discussed in Chapter 3 via the IAC. A minimum number of three images are needed, with five generally producing stable results. Based on our research, this method will usually overestimate the focal length as the FOV of the camera increases. In addition, the solution for the other parameters will be off target. We apply an alternation technique during optimization discussed in section 5.7, significantly improving the final results.

5.3 Solving for extrinsic parameters

Solving for the extrinsic parameters is straight forward once the camera calibration matrix is known. We use the formulation in equation 4.19 to extract the extrinsic parameters. Once the rotation matrix has been extracted, it is parameterized using Euler angles as described in section 2.3.2. Other parameterizations could be used, which were also discussed in Chapter 2, such as axis/angle and quaternions. Keeping in mind though that axis/angle and quaternions have four parameters, but 3-DOF. Even though axis/angle can be represented with only three parameters, it adds a constraint. So in the bundle adjustment stage when unconstrained nonlinear optimization is performed, all four parameters must be used or a technique which takes the constraint into account must be applied. Hornegger and Tomasi developed a technique to use quaternions in unconstrained nonlinear optimization which only used three parameters [Hornegger99].

5.4 Solving for the lens projection

In practice, the ideal lens projection never extends to the real camera. Thus we use a polynomial to approximate the real lens projection of the following form:

$$r(\phi) = f\sum_{i=1}^{p} \kappa_i \phi^{2i-1} = f\left(\kappa_1\phi + \kappa_2\phi^3 + \ldots + \kappa_p\phi^{(2p-1)}\right) \tag{5.4}$$

We will denote the number of coefficients in equation 5.4 as p.

Once a solution has been computed for the calibration matrix, rotation matrix and translation vector, a least-squares solution to the lens projection coefficients $\boldsymbol{\kappa} = (\kappa_1, \kappa_2, \ldots)^{\mathrm{T}}$ is calculated. Prior to computing the coefficients, we assume the estimated and measured image points, denoted as $(\breve{x}, \breve{y})$ and (x, y) respectively, are normalized to unit focal length by multiplying them by the inverse of the camera calibration matrix. The estimated image points are those computed using the closed-form solution set described in section 5.2, and the measured image points are those that were detected in the image. Then let $\breve{r} = \sqrt{\breve{x}^2 + \breve{y}^2}$ be the radial distance $(\breve{x}, \breve{y})$ is from the principal point, and similarly $r = \sqrt{x^2 + y^2}$ for the measured point (x, y). Then equation 5.4 can be written in matrix notation as

$$\begin{pmatrix} \phi & \phi^3 & \ldots & \phi^{(2p-1)} \end{pmatrix} \begin{pmatrix} \kappa_1 \\ \kappa_2 \\ \vdots \\ \kappa_p \end{pmatrix} = (r) \tag{5.5}$$

with $\phi = \tan^{-1}(\breve{r}/f)$ the angle between the principal axis and the incoming ray. Since the points are normalized to unit focal length: $f = 1$. Stacking equation (5.5) for m points we can write $\mathbf{A}\boldsymbol{\kappa} = \mathbf{b}$ where $\mathbf{A}$ is a $m \times p$ matrix with p the number of coefficients in the lens projection model. The least squares solution is simply

$$\boldsymbol{\kappa} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}\,. \tag{5.6}$$

5.5 Decentering distortion

Decentering distortion occurs when lens elements are misaligned, which was discussed in Chapter 3. Even though decentering distortion may not be needed to model a particular camera, the model selection stage will automatically determine this. The components are modeled as

$$\mathbf{\Delta d} = \begin{pmatrix} \left(2\rho_1 uv + \rho_2\left(r^2 + u^2\right)\right)\left(1 + \rho_3 r + \rho_4 r^3 + \ldots\right) \\ \left(\rho_1\left(r^2 + v^2\right) + 2\rho_2 uv\right)\left(1 + \rho_3 r + \rho_4 r^3 + \ldots\right) \end{pmatrix} \tag{5.7}$$

with $(\rho_1, \rho_2, \ldots)$ the decentering distortion coefficients. The coefficients ρ_1 and ρ_2 are typically the only ones used in practice, neglecting the higher order terms. We denote the number of coefficients used for decentering distortion by q, and initially set all coefficients to zero prior to nonlinear optimization.

5.6 Complete model

The complete camera model includes everything that has been described in the previous sections. The final estimated image point $\breve{\mathbf{m}} = (\breve{x}, \breve{y})^{\mathrm{T}}$ is

$$\begin{pmatrix} \breve{x} \\ \breve{y} \end{pmatrix} = r(\phi)\begin{pmatrix} \cos\vartheta \\ \sin\vartheta \end{pmatrix} + \mathbf{\Delta d} \tag{5.8}$$

where $r(\phi)$ is the lens projection and $\mathbf{\Delta d}$ the decentering distortion. The angle ϑ is the angle the image point is from the x-axis calculated as $\vartheta = \tan^{-1}(y/x)$.

5.7 Bundle adjustment

Once the close-from solutions to the camera parameters are computed, including the lens projection coefficients, the results are refined using Maximum Likelihood Estimation (MLE).

From our experiments, as the lens projection deviates from the perspective projection, alternating between refining $(\mathbf{K}, \mathbf{R}, \mathbf{t})$ and $(\boldsymbol{\kappa}, \boldsymbol{\rho})$ produces significantly better results. Figure 5.2 shows a plot of the mean-square-error (MSE) for different lens projections computed both with and without alternation using our algorithm. Perspective and orthogonal projection had little or no benefit with alternation, but stereographic and equisolid projection had significant improvements.

MLE is performed by minimizing the following functional

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\left\|\mathbf{m}_{ij} - \breve{\mathbf{m}}\left(\mathbf{K}, \mathbf{R}_i, \mathbf{t}_i, \boldsymbol{\kappa}, \boldsymbol{\rho}, \mathbf{M}_j\right)\right\|^2 \tag{5.9}$$

where $\breve{\mathbf{m}}\left(\mathbf{K}, \mathbf{R}_i, \mathbf{t}_i, \boldsymbol{\kappa}, \boldsymbol{\rho}, \mathbf{M}_j\right)$ is the projection of point $\mathbf{M}_j$ in image i computed from equation 5.8. The parameters $(\mathbf{K}, \mathbf{R}, \mathbf{t})$ and $(\boldsymbol{\kappa}, \boldsymbol{\rho})$ are optimized in alternation until convergence. The initial estimates for the decentering coefficients are set to zero, which is satisfactory because decentering distortion is usually small in practice. The Levenberg-Marquardt algorithm is used to perform the MLE. Figure 5.3 gives a flow chart of the complete algorithm.

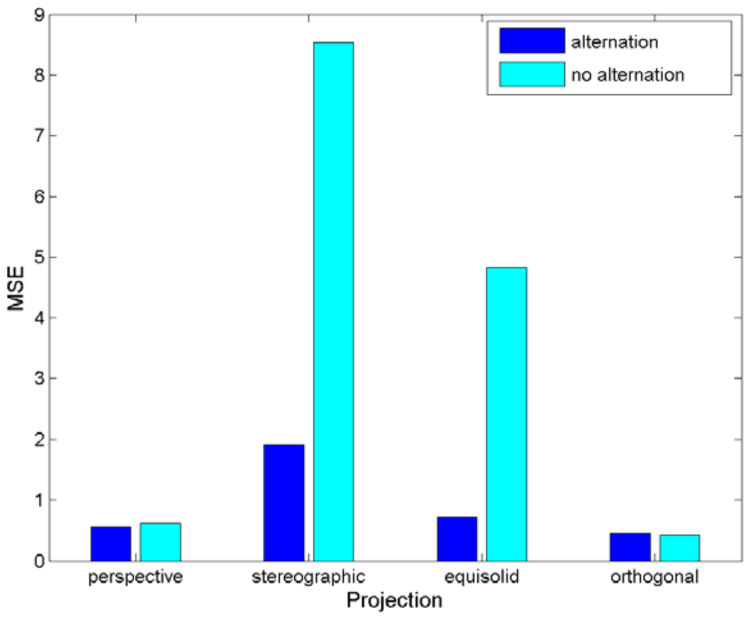


Figure 5.2: Chart of the MSE for different projections with and without alternation. The data was synthetically generated with the same camera parameters found in the results section for synthetic data. To simulate real data we added Gaussian noise with zero mean and unit variance.

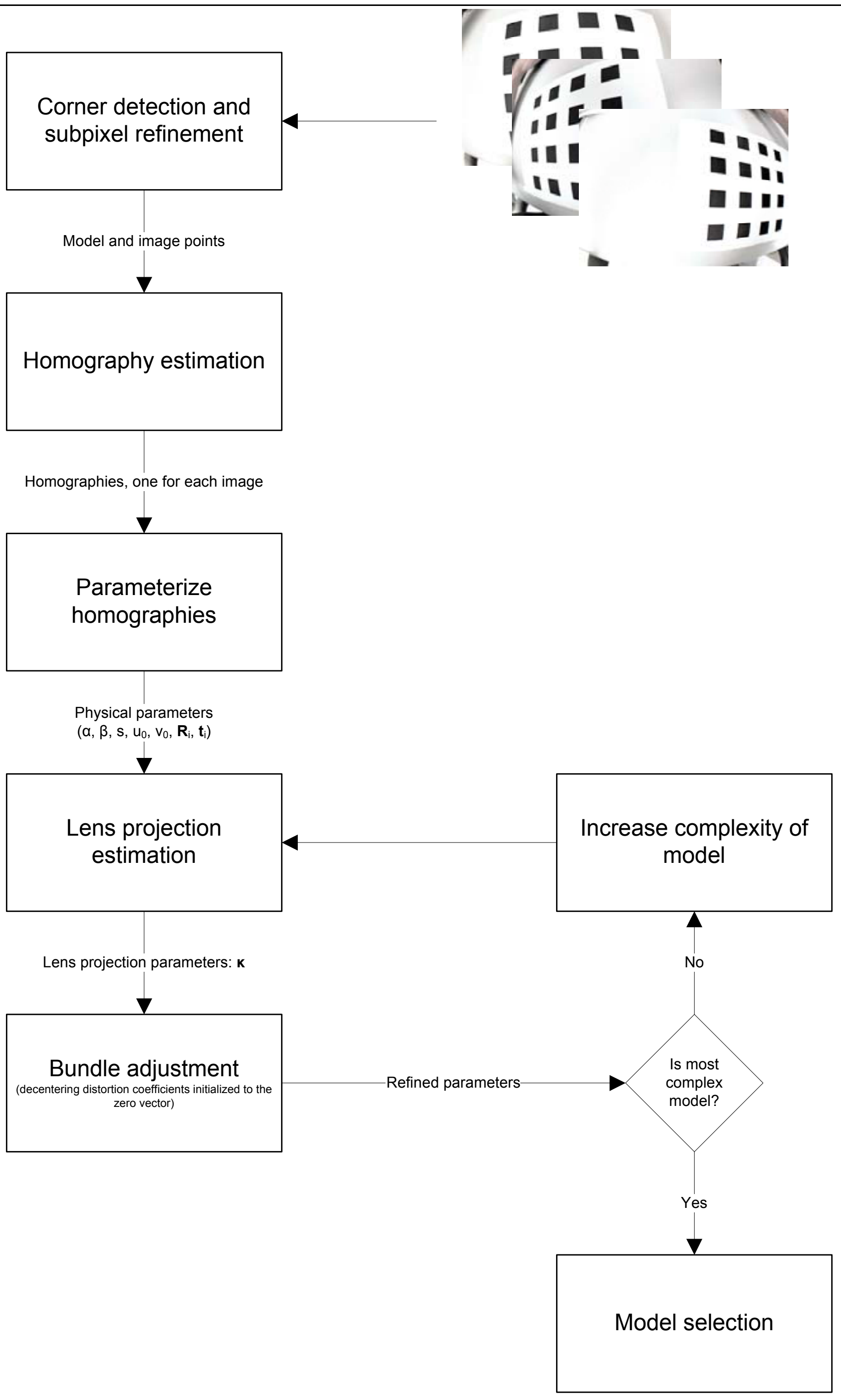

Corner detection and subpixel refinement
Model and image points
Homography estimation
Homographies, one for each image
Parameterize homographies
Physical parameters
(α, β, s, u0, v0, Ri, ti)
Lens projection estimation
Lens projection parameters: κ
Bundle adjustment
(decentering distortion coefficients initialized to the zero vector)
Refined parameters
Is most complex model?
No
Yes
Increase complexity of model
Model selection

Figure 5.3: An overview of the algorithm.

5.8 Model selection

One of the model selection criteria from Chapter 5 is used to find the best model, once several models have been computed. So far we have assumed the variance of the noise is known. We use the formulation in [Gheissari03] to calculate the variance σ^2 of unknown Gaussian noise:

$$\sigma^2 = \sum_i r_i^2 \Big/ \left(N - \hat{k}\right) \tag{5.10}$$

where N is the number of samples and $\hat{k}$ is the number of coefficients of the most complex model in the library. The performance of equation 5.10 is shown in Figure 5.4 for different lens projections. Notice that the noise of a stereographic projection tends to be severely overestimated.

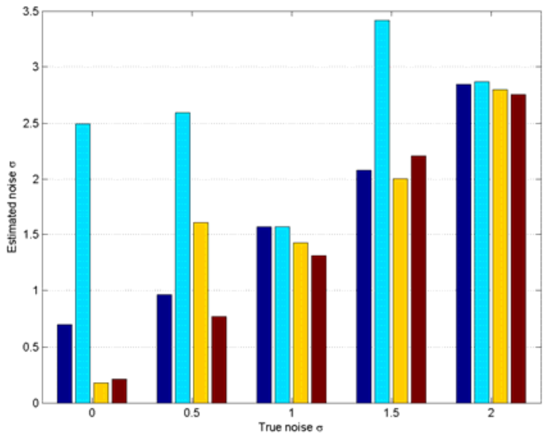


Figure 5.4: Graph of the true and estimated Gaussian distributed noise (standard deviation) for the different lens projections listed in Table 4.1.

6 EXPERIMENTAL RESULTS

We begin this section with the implementation decisions of the calibration procedure and model selection algorithm. We then illustrate the results of our algorithm on multiple sets of synthetic and real data which include several different lens projections.

6.1 Implementation

The algorithm takes several sets of data as input: the 2D model plane points, and several sets of 2D image points. Since our algorithm is a DLT based algorithm, the first component of the software is to compute several sets of 2D homographies relating the model and image points. The homographies are computed using the normalized DLT and optimized using Levenberg-Marquardt algorithm described as the Gold Standard Algorithm in section 4.1.1. The second component decomposes the homographies into the intrinsic and extrinsic parameters of the camera using the closed-form solution from the IAC discussed in section 4.2.1. The third component estimates the lens projection of the camera in a least-squares sense. The fourth component refines all the parameters in bundle-adjustment. The last component takes several optimized models of increasing complexity and selects the best model using statistical information criteria. The majority of the software has been developed by the author in Matlab. However, several existing functions have been utilized from multiple sources.

6.2 Synthetic data

Four sets of data were generated using the lens projections listed in Table 3.1: (1) perspective, (2) stereographic, (3) equisolid and (4) orthogonal. We chose these lens projections because they represent wide range of cameras on the market, varying in order (1-4) from rectilinear to fisheye. The purpose of this experiment was to compare the robustness of our model, which comprises of *lens projection* and *decentering distortions (LPDD),* vs. models that use *radial* and *decentering distortions (RDDD)* [Heikkila97][Heikkila00][Zhang00A]. We apply both methods to several lens projections, and compare the complexity of the model and camera parameters with respect to the types of lens projection. The information criterions listed in Table 4.1 were used to choose the complexity of the models. We let p be the number of coefficients in the lens projection model for LPDD, or the radial distortion model for RDDD, and q the number of coefficients in the decentering distortion model.

The intrinsic parameters of the synthetic *perspective* camera had the following values: $\alpha = 800$, $\beta = 800$, $s = 0$, $u_0 = 320$ and $v_0 = 240$. The other three synthetic cameras, namely stereographic, equisolid, and orthogonal, were set to smaller focal lengths: $\alpha = 160$ and $\beta = 160$. The image resolution was 640×480, and the model plane consisted of $8 \times 8 = 64$ corner points. Five different images of the model were generated for each lens projection with the following Euler angles:

$$\begin{aligned}
&\mathbf{r}_1 = (0.2244,0,0)^{\mathrm{T}}, \\
&\mathbf{r}_2 = (0,0.3491,0)^{\mathrm{T}}, \\
&\mathbf{r}_3 = (0,0,0.2618)^{\mathrm{T}}, \\
&\mathbf{r}_4 = (0.3491,0.3491,0)^{\mathrm{T}} \text{ and} \\
&\mathbf{r}_5 = (0,0.3491,0.3491)^{\mathrm{T}},
\end{aligned} \tag{6.1}$$

and the translation vectors for perspective projection were:

$$\begin{aligned}
\mathbf{t}_1 &= (-120,-120,500)^{\mathrm{T}}, \\
\mathbf{t}_2 &= (-145,-120,450)^{\mathrm{T}}, \\
\mathbf{t}_3 &= (-145,-145,600)^{\mathrm{T}}, \\
\mathbf{t}_4 &= (-95,-120,425)^{\mathrm{T}}, \\
\mathbf{t}_5 &= (-170,-95,500)^{\mathrm{T}}
\end{aligned} \tag{6.2}$$

and for the other three projections:

$$\begin{aligned}
\mathbf{t}_1 &= (-120,-120,90)^{\mathrm{T}}, \\
\mathbf{t}_2 &= (-145,-120,40)^{\mathrm{T}}, \\
\mathbf{t}_3 &= (-145,-145,80)^{\mathrm{T}}, \\
\mathbf{t}_4 &= (-95,-120,120)^{\mathrm{T}} \text{ and} \\
\mathbf{t}_5 &= (-170,-95,40)^{\mathrm{T}}.
\end{aligned} \tag{6.2}$$

We chose the values of these parameters so the collection of images spanned a large portion of the image plane. We then added Gaussian noise to the image points which had unit variance and zero mean.

We measure the error of the overall model using mean-square-error (MSE), measured in pixels, between the actual measured and estimated image points. Table 6.1 and Table 6.2 list the calibration results for the four tested lens projections. From the tables, the results for perspective projection are close to the ground truth, and the MSE is relatively small. For the other lens projections, α and β tended to be considerably smaller in magnitude compared to the ground truth in both models, except for orthogonal projection using the RDDD model, this is expanded on this later in this section when analyzing the lens projection. However, the MSE in the RDDD model is significantly larger compared to the LPDD model for stereographic, equisolid and orthogonal projection. This is illustrated in Figure 6.1, which is a plot of the MSE for all the tested lens projections.

Table 6.1: Calibration results for synthetic cameras using LPDD model and MDL.

Projection	α	β	s	u_0	v_0	MSE
(1) Perspective	809.04	808.55	-0.02	315.40	228.06	0.3500
(2) Stereographic	97.87	97.84	0.13	319.67	240.91	0.4571
(3) Equisolid	102.92	102.96	-0.01	320.86	240.14	0.4041
(4) Orthogonal	105.60	105.76	0.05	320.25	239.89	0.5035

Table 6.2: Calibration results for synthetic cameras using RDDD model and MDL.

Projection	α	β	s	u_0	v_0	MSE
(1) Perspective	792.76	790.94	-0.11	315.84	239.00	0.3990
(2) Stereographic	84.15	85.10	-1.52	307.52	264.44	3.2148
(3) Equisolid	118.93	119.25	0.20	317.17	248.00	1.2878
(4) Orthogonal	163.42	162.54	0.40	320.64	244.88	1.3435

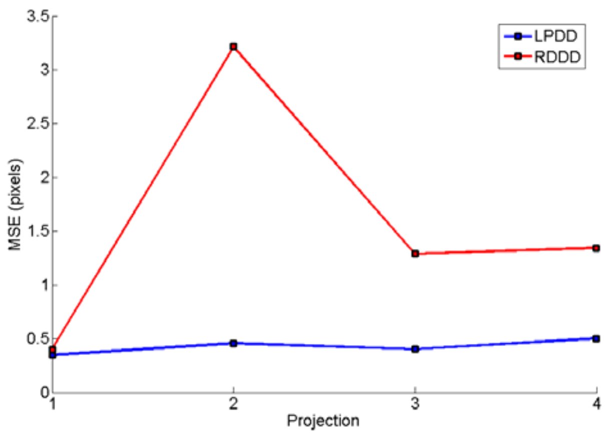


Figure 6.1: Graph of the mean-square-error (MSE) for several different lens projections in pixels: (1) perspective, (2) stereographic, (3) equisolid and (4) orthogonal.

Initially, we may think the complexity of the LPDD model is higher than that of the RDDD model because the errors are small regardless of the lens projection. Table 6.3 and Table 6.4 list the complexity of the LPDD and RDDD models chosen by the different information criterions, respectively. We plotted the results from MDL in Figure 6.2 to show how the complexity changes as a function of the lens projection. MDL was chosen over the other criterions to generate these plots because it *always* selected a complexity less than or equal to that of the other criterions, without sacrificing a significantly lower error. With the RDDD model the complexity increased as the FOV increased. However, the LPDD stayed level for all the lens projections.

The results of the calibration procedure for the intrinsic parameters are listed in Table 6.1 and Table 6.2 for each lens projection for both models. The MSE corresponds to the model selected by the corresponding MDL criterion. The values for perspective projection for both methods are close to the ground truth, but vary widely for the other projections. Even though α and β are not close to the ground truth for the other projections, the MSE stays relatively small in the LPDD model, which is not the case in the RDDD model. In the RDDD model, the MSE is over four times that of LPDD model for stereographic projection, and over two times that for equisolid and orthogonal projections.

We also plotted the estimated lens projections with the true lens projections in Figure 6.3. The estimated lens projection in the perspective case is nearly identical to the theoretic. In the other lens projections, the estimated lens projections are similar in curvature, but vary in amplitude. This is presumably due to the error in the estimated focal length since the focal length and the lens projection are highly correlated. The lens projection or radial distortion model parameters will compensate when the focal length is off target. This is to be expected when estimating the parameters of wide angle camera with a linear solution under the pinhole model.

Table 6.3: Complexity of the LPDD model for several lens projections.

Projection	AIC		MDL		BIC		SSD		CAIC	
	p	q	p	q	p	q	p	q	p	q
(1) Perspective	5	2	2	0	5	0	5	0	5	0
(2) Stereographic	2	0	2	0	2	0	2	0	2	0
(3) Equisolid	2	0	2	0	2	0	2	0	2	0
(4) Orthogonal	2	0	2	0	2	0	2	0	2	0

Table 6.4: Complexity of the RDDD model for several lens projections.

Projection	AIC		MDL		BIC		SSD		CAIC	
	p	q	p	q	p	q	p	q	p	q
(1) Perspective	1	0	1	0	1	0	1	0	1	0
(2) Stereographic	3	2	3	2	3	0	3	3	3	0
(3) Equisolid	4	2	4	2	4	2	4	2	4	2
(4) Orthogonal	4	2	4	2	4	2	4	2	4	2

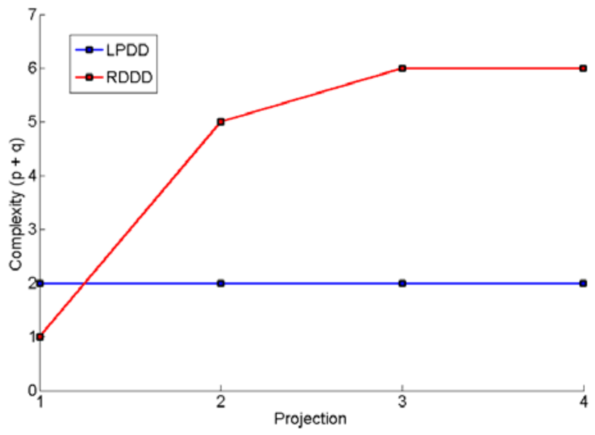
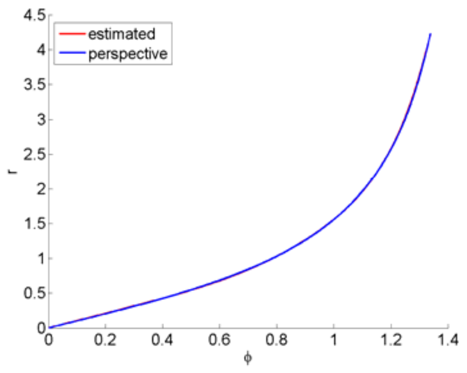
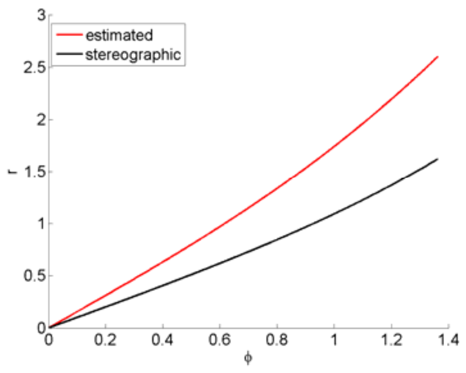


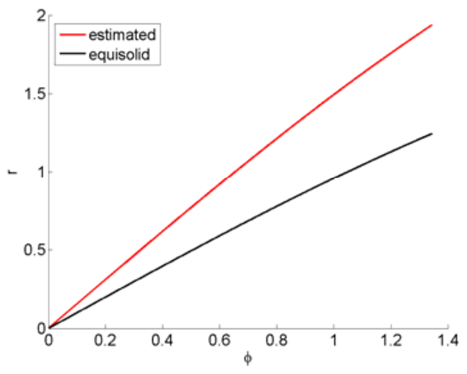
Figure 6.2: Graph of the complexity of the LPDD and RDDD models for several different lens projections: (1) perspective, (2) stereographic, (3) equisolid and (4) orthogonal. The complexity for the LPDD model is calculated as the sum of the number of lens projection and decentering distortion model coefficients in Table 6.3 corresponding to the MDL criterion. Similarly, the complexity of the RDDD model is calculated as the sum of the radial distortion and decentering distortion model in Table 6.4 corresponding to the MDL criterion.



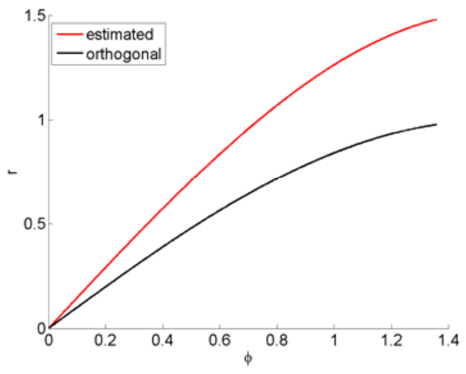
(a)



(b)



(c)



(d)

Figure 6.3: Plots of the estimated lens projections for synthetic data for (a) perspective, (b) stereographic, (c) equisolid, and (d) orthogonal projections using MDL. The red curve is the estimated lens projection and the black curve is the true lens projection. The unit of Φ is in radians, and the unit of r is the same as in Figure 3.2 (a), i.e normalized to unit focal length.

6.3 Real data

We applied our calibration algorithm to four different real cameras: (1) PULNiX CCD camera with 6 mm lens (data taken from [ZhangData]), (2) IQEye3 with a verifocal FUJINON 1.4-3.1 mm lens set to wide angle, and a Nikon with a fisheye FC-E8 lens set to two different zoom settings to produce a (3) full frame fisheye (180˚ across the diagonal) and (4) circular fisheye (180˚ in all directions). Each set contained eight images with 64 corners on each image for a total of 512 corners (except for Zhang's data which contains 5 images each with 256 corners [Zhang00B]).

The layout of the results in this section is similar to that in the previous section. Figure 6.4 shows how the MSE increases exponentially as the FOV increases, denoted by the numbering: (1) being rectilinear camera and (4) a circular fisheye. Zhang [Zhang00A] achieved a root-mean-square (RMS) error on his publicly available dataset [Zhang00B] of 0.335, where only radial distortion was modeled. This corresponds to an MSE of approximately 0.1122. Our LPDD method achieved an MSE of 0.0298 using the same number of coefficients which is a 73% improvement. These values are listed in Table 6.5 and Table 6.6 for comparison. The RDDD method also achieved a lower MSE than Zhang when adding two extra coefficients for decentering distortion, selected by MDL. The complexity of the models for each camera can be seen in Figure 6.5, and the numerical values are listed in Table 6.7 and Table 6.8 for the LPDD and RDDD models, respectively. In each case, except for the circular fisheye, the complexity selected by MDL was less when modeling lens project. Even in the case of the circular fisheye, the complexity was the same, however the MSE was considerably less.

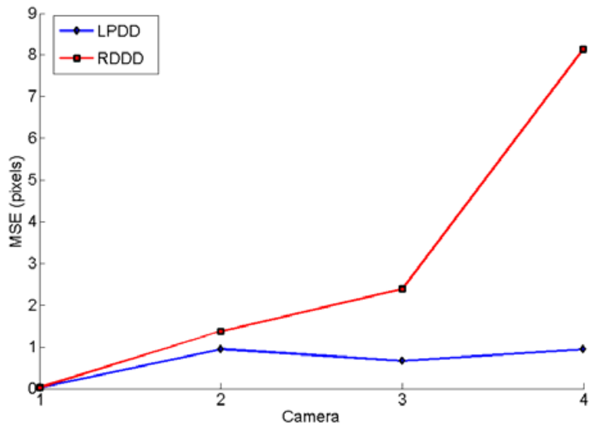


Figure 6.4: Graph of the mean-square-error (MSE) for the different cameras: (1) Rectilinear, (2) wide angle, (3) full frame and (4) circular fisheye, modeled using LPDD (blue) and RDDD (red) models.

Table 6.5: Calibration results for real cameras using LPDD model and MDL.

Projection	α	β	s	u_0	v_0	MSE
(1) Zhang (rectilinear)	821.08	821.12	0.23	303.90	207.55	0.0298
(2) Wide angle	229.03	229.43	0.56	333.93	257.75	0.9520
(3) Full frame	219.19	218.67	0.05	414.94	324.58	0.6639
(4) Circular fisheye	149.73	149.57	0.04	411.73	315.61	0.9405

Table 6.6: Calibration results for real cameras using RDDD model and MDL.

Projection	α	β	s	u_0	v_0	MSE
(1) Zhang	832.05	831.98	0.25	303.76	212.25	0.0287
(2) Wide angle	121.57	122.45	0.71	326.74	269.81	1.3790
(3) Full frame	132.13	131.29	0.17	408.79	307.45	2.3857
(4) Fisheye	170.56	170.33	0.11	413.40	324.25	8.1422

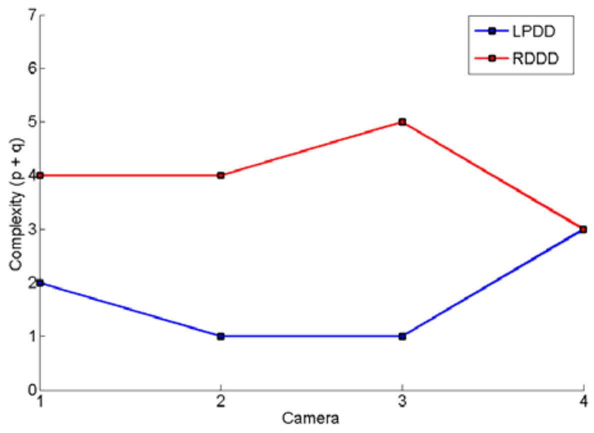


Figure 6.5: Graph of the complexity for the different cameras: (1) Rectilinear, (2) wide angle, (3) full frame and (4) circular fisheye, modeled using LPDD (blue) and RDDD (red) models. The complexity for the LPDD model is calculated as the sum of the number of lens projection and decentering distortion model coefficients in Table 6.7 corresponding to the MDL criterion. Similarly, the complexity of the RDDD model is calculated as the sum of the number of radial distortion and decentering distortion model coefficients in Table 6.8 corresponding to the MDL criterion.

Table 6.7: Complexity of model selection for LPDD model for real cameras.

Projection	AIC		MDL		BIC		SSD		CAIC	
	p	q	p	q	p	q	p	q	p	q
(1) Zhang	4	0	2	0	2	0	2	0	2	0
(2) Wide angle	1	2	1	0	1	2	1	2	1	0
(3) Full frame	1	2	1	0	1	0	1	0	1	0
(4) Fisheye	3	0	3	0	3	0	3	0	3	0

Table 6.8: Complexity of model selection for RDDD model for real cameras.

Projection	AIC		MDL		BIC		SSD		CAIC	
	p	q	p	q	p	q	p	q	p	q
(1) Zhang	2	2	2	2	2	0	2	2	2	2
(2) Wide angle	2	2	2	2	2	0	2	0	2	0
(3) Full frame	3	2	3	2	3	2	3	2	3	2
(4) Fisheye	3	0	3	0	3	0	3	0	3	0

To get a better understanding of the estimated lens projection for comparison to the ideal ones, we plotted the estimated lens projection with perspective and orthogonal projections (the two extremes) in Figure 6.6. In these plots, the focal length is normalized to unity, thus they have the same scale as in Figure 3.2. What is interesting about these plots is the estimated lens projections for cameras appear to be linear. This is also true for the fisheye camera in Figure 6.6 (d), even though MDL selected three coefficients for the lens projections (Table 6.7).

In all the experiments, the LPDD method outperformed the RDDD method except for Zhang rectilinear camera, but was higher only by a small margin (3.8%). We can clearly see in Figure 6.4 the exponential increase of the MSE for the RDDD model as the camera approaches a circular fisheye, denoted by the numbering (1-4), whereas the MSE for LPDD is small and stable for all cameras. Also, the complexity of the model is less than or equal to that of RDDD as shown in Figure 6.5 for all cameras.

This calibration technique can be used in wide area surveillance and video tracking. Figure 6.7 shows three original images taken from the Nikon circular fisheye camera and the corresponding corrected versions after calibration. We use the large FOV of these wide angle cameras to monitor large areas, and relay information to PTZ cameras that zoom in to acquire a close-up view of suspicious activity. In correcting these images towards a perspective projection we have traded one distortion for another as clearly seen in Figure 6.7 (e). The perspective distortions become visible as the FOV approaches 180°'s. Also, the resolution of the fisheye images is lower near the perimeter of the image. Hence, corrected versions appear slightly blurred near the perimeter. The original images which were used to calibrate the different cameras and the corrected versions are shown in Figure 6.8 through 6.11. The first column contains the original images, and the second shows the corrected version towards an ideal perspective one. The corners were detected using the Harris corner detector with sub-pixel refinement, except for Zhang's data. We corrected the images by applying the complete model in equation 5.8.

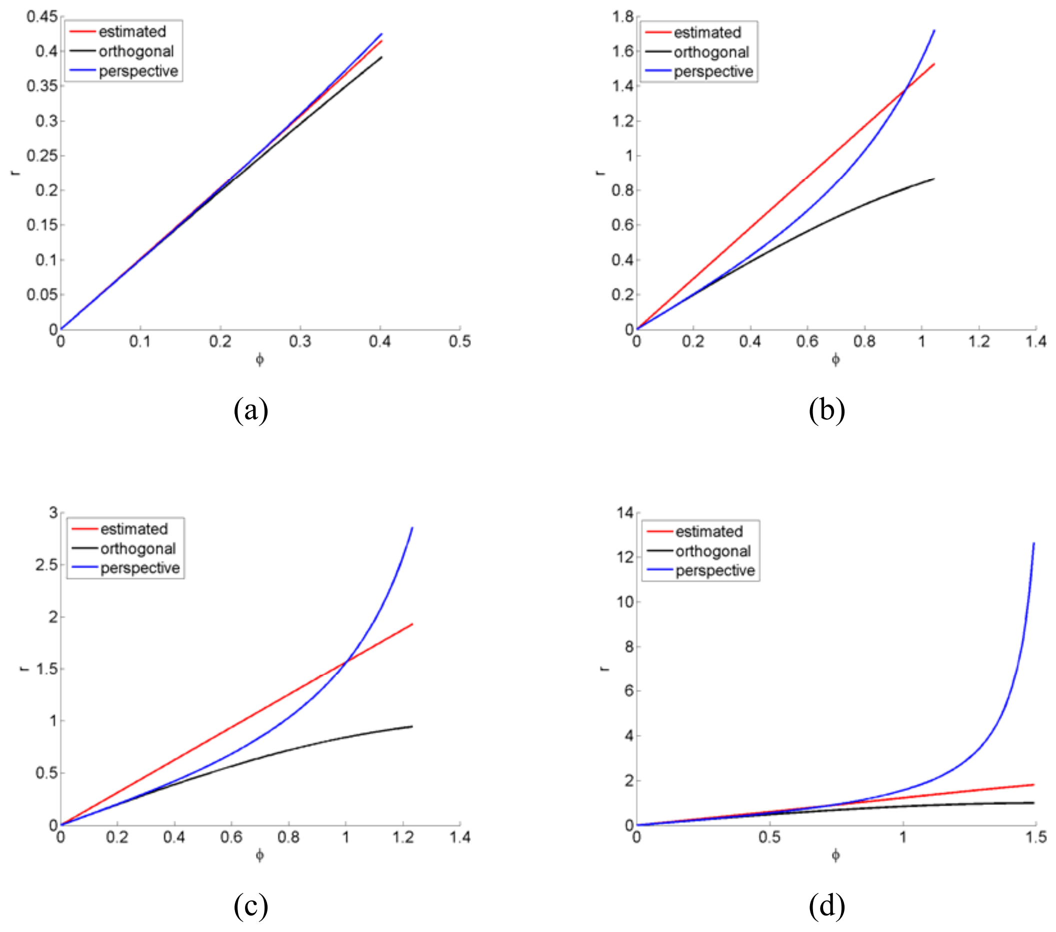


Figure 6.6: The estimated projection for (a) PUNiX camera [Zhang00B], (b) IQEye3 wide angle, (c) full frame fisheye, and (d) circular fisheye using MDL model selection. The unit of Φ is in radians, and the unit of r is the same as in Figure 3.2 (a), i.e normalized to unit focal length.

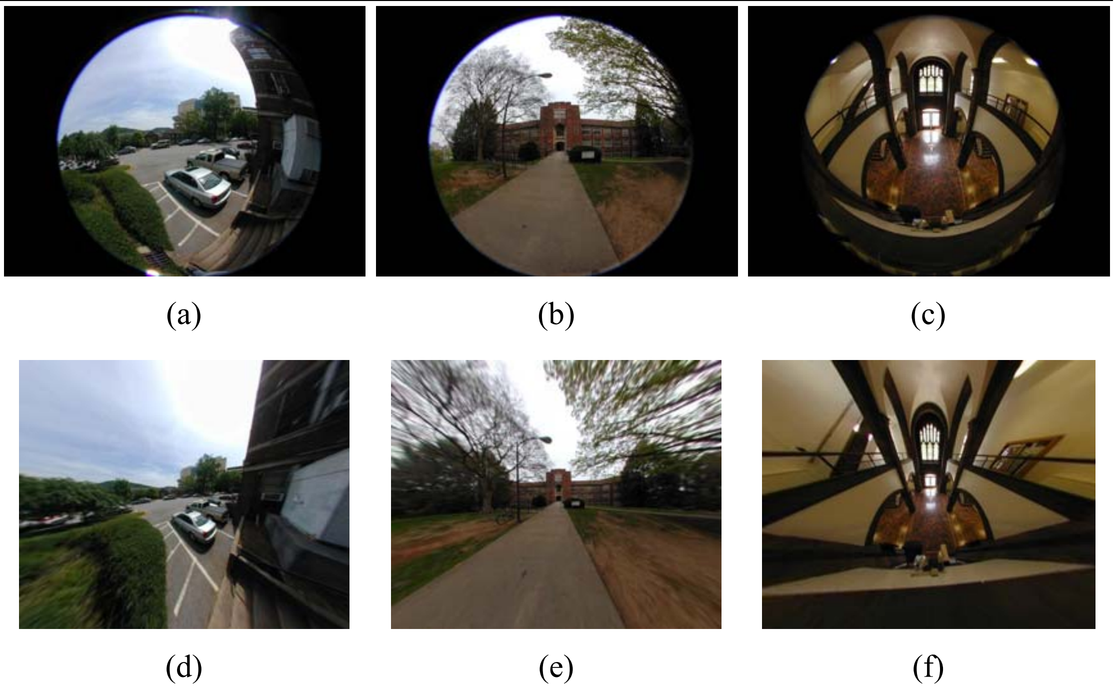


Figure 6.7: Three images and perceptively corrected versions. Image (a), (b) and (c) are three original images taken from the Nikon circular fisheye and (d), (e) and (f) are the corresponding corrected version after calibration, respectively.

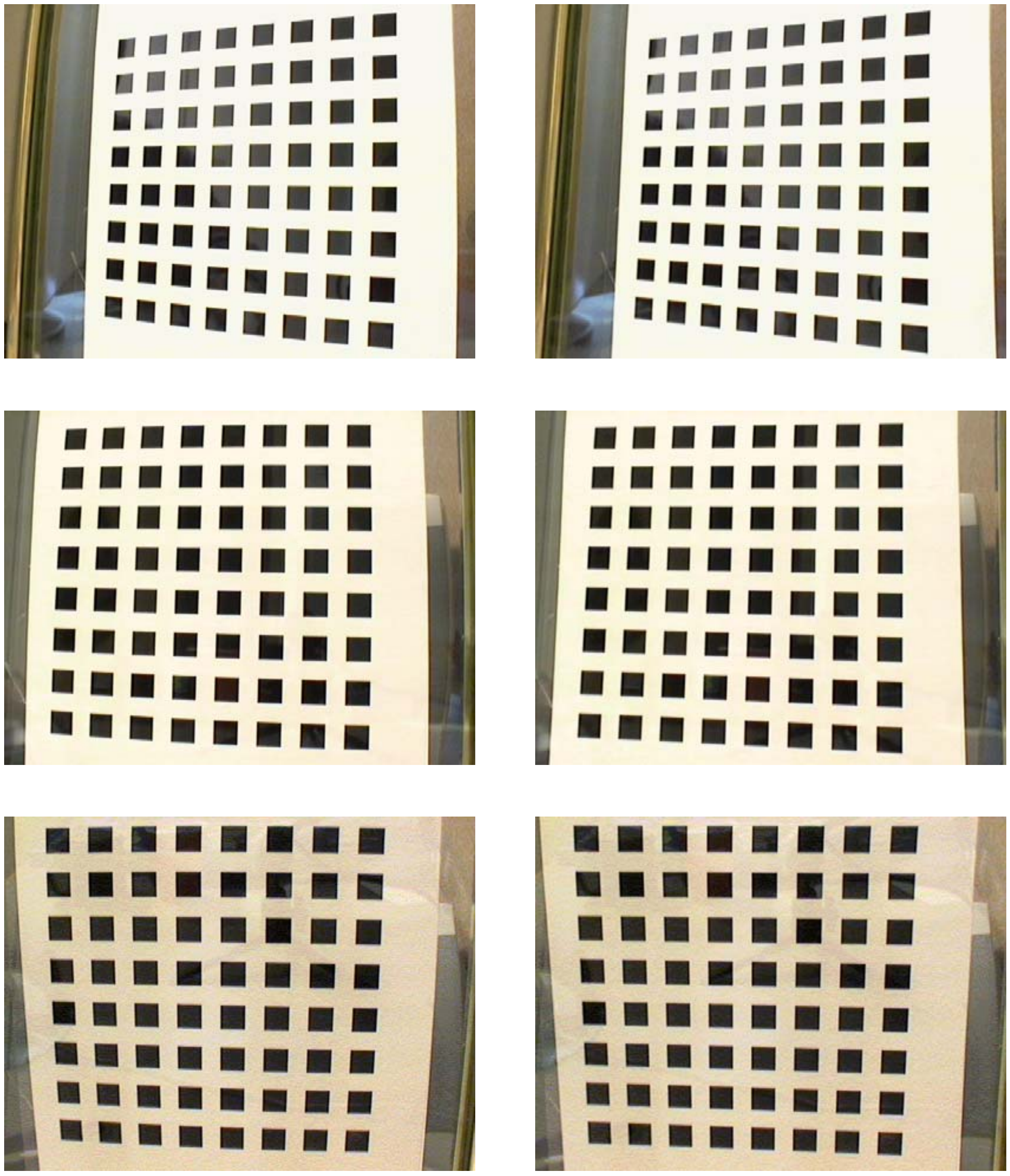


Figure 6.8: (left) Three out of the the original set of five images taken from the PUNix camera [Zhang00B] and (right) the corresponding corrected versions after calibration.

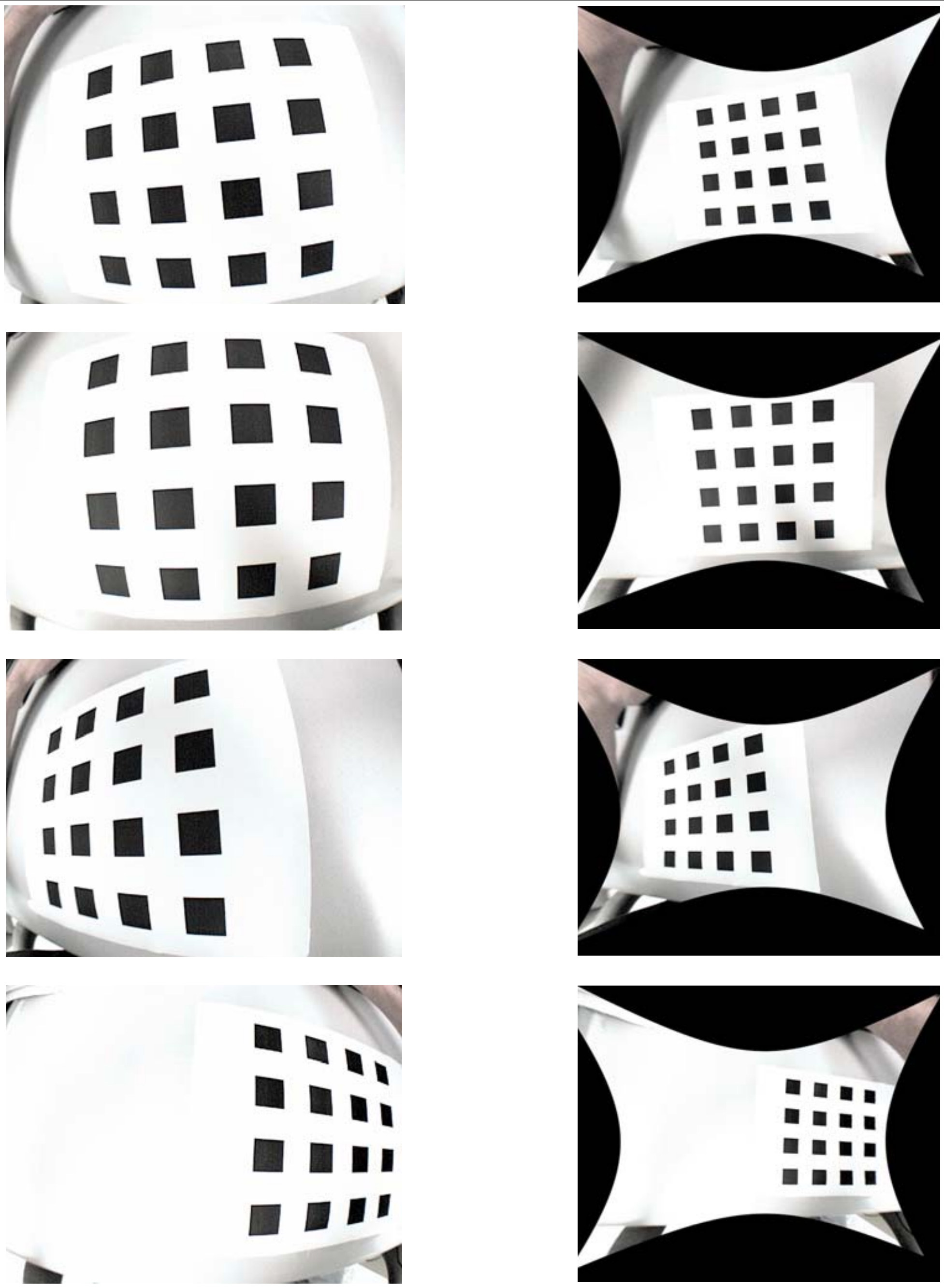


Figure 6.9: (left) The original set of images taken from the IQEye3 wide-angle camera and (right) the corresponding corrected versions after calibration.

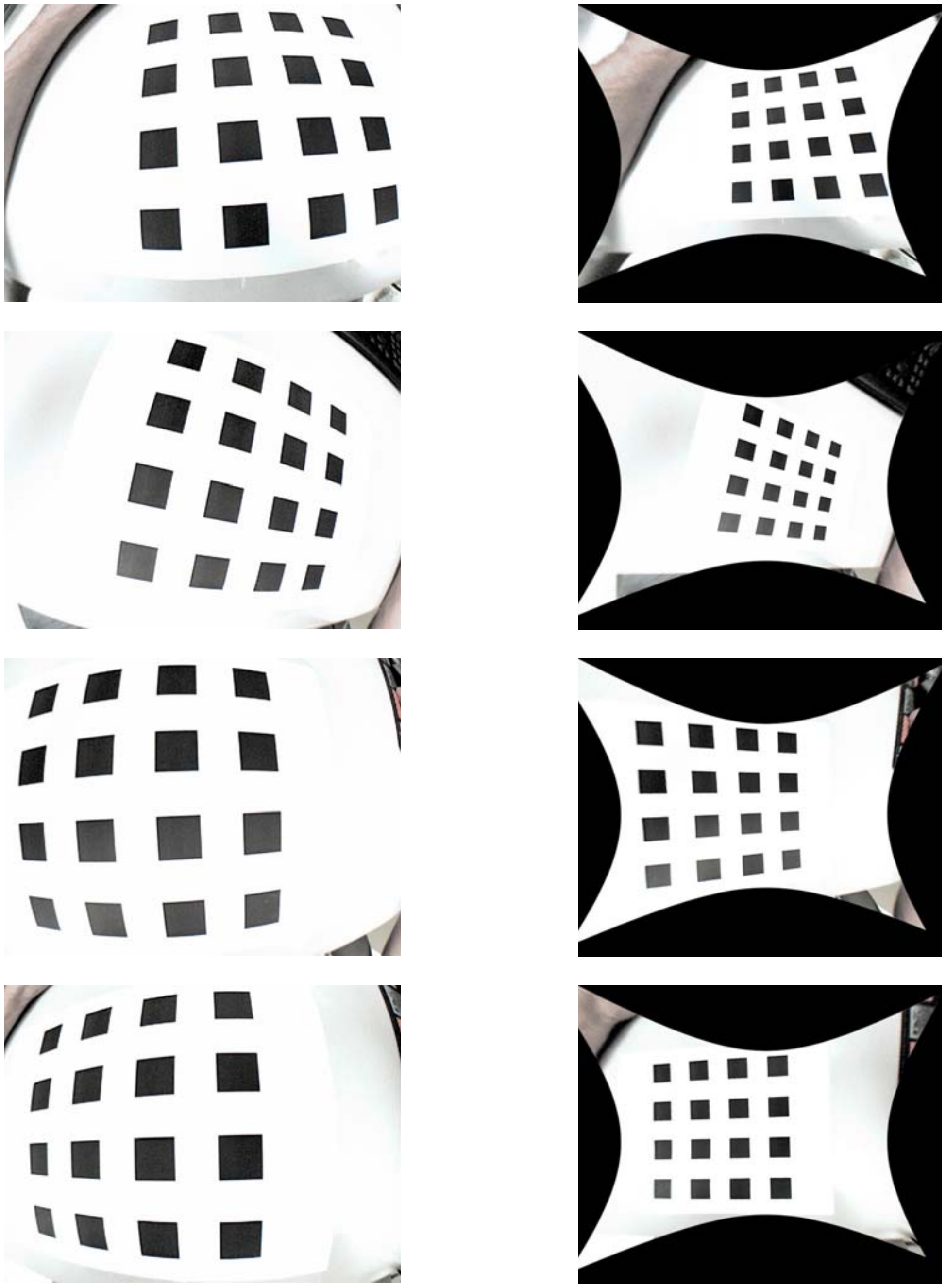


Figure 6.9: Continued

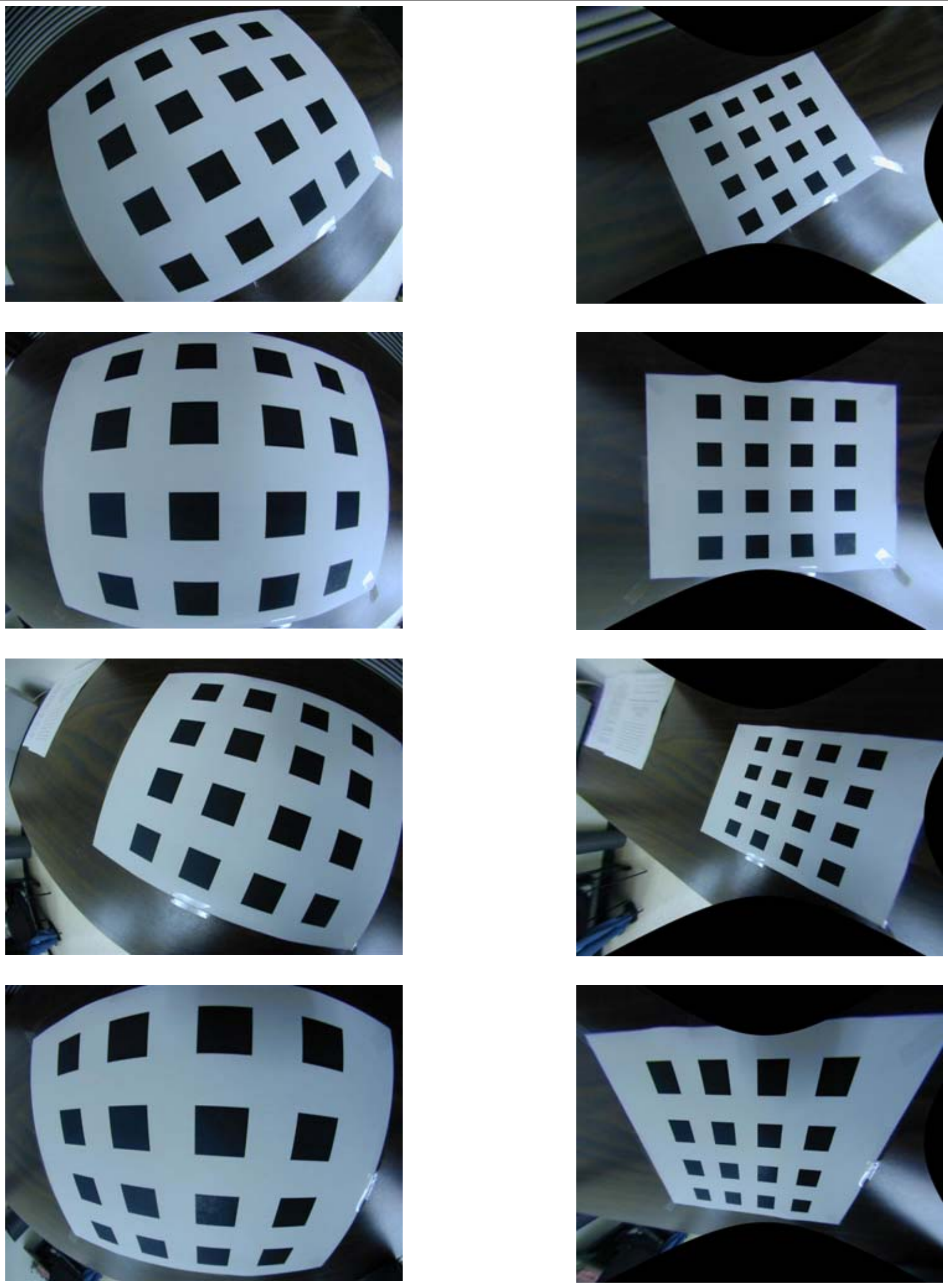


Figure 6.10: (left) The original set of images taken from the full frame fisheye camera and (right) the corresponding corrected versions after calibration.

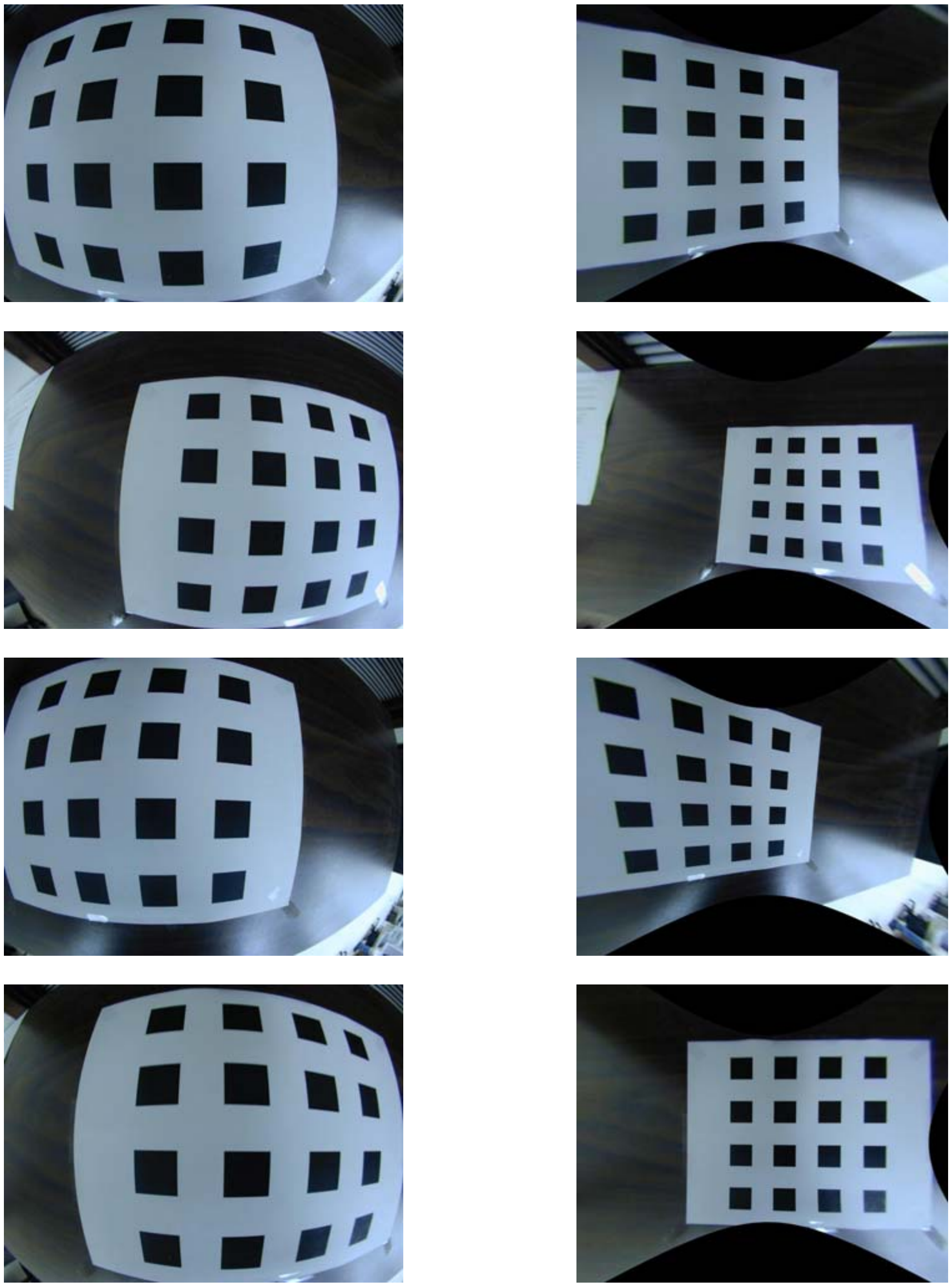


Figure 6.10: Continued

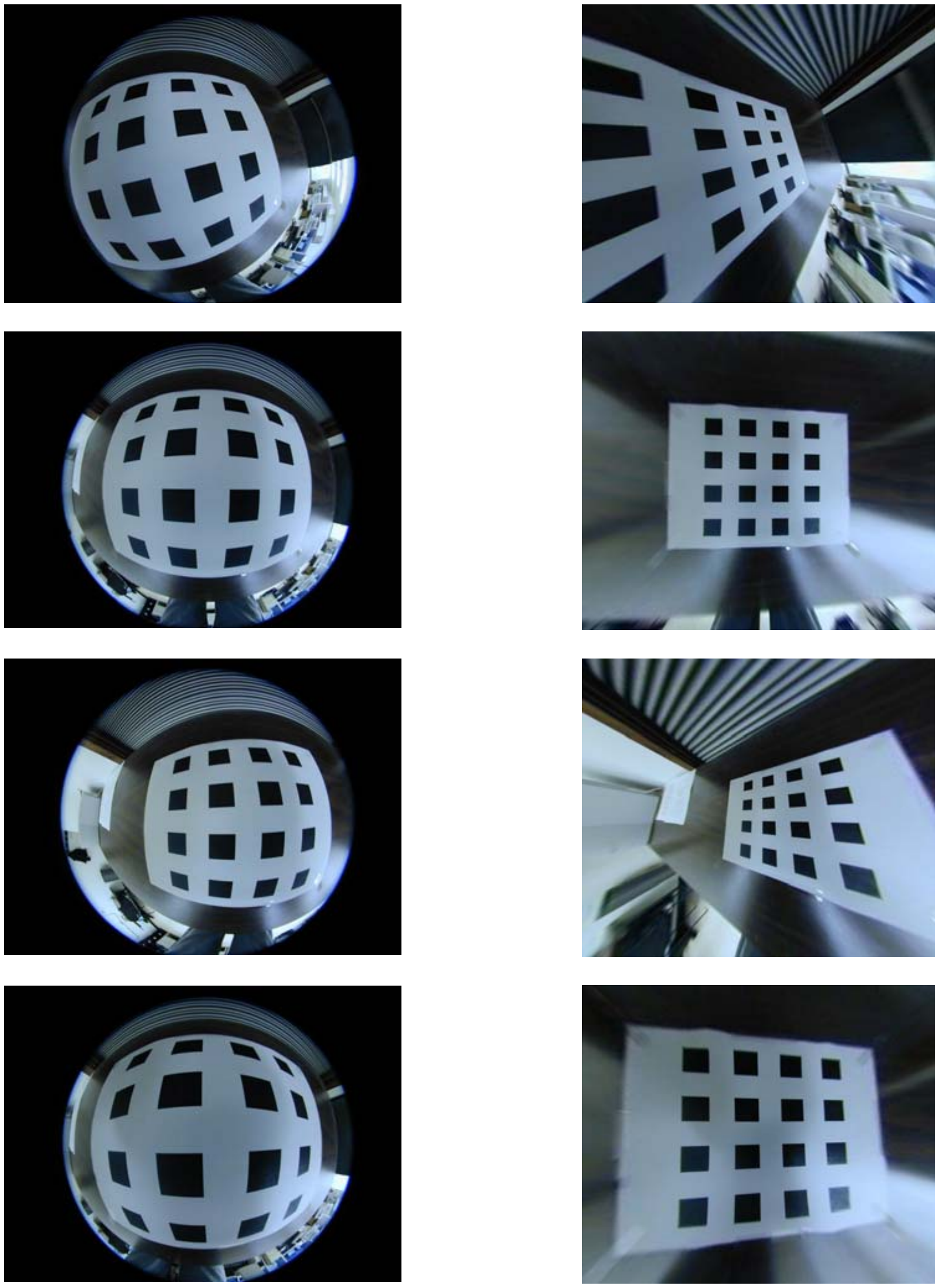


Figure 6.11: (left) The original set of images taken from the fisheye camera and (right) the corresponding corrected versions after calibration.

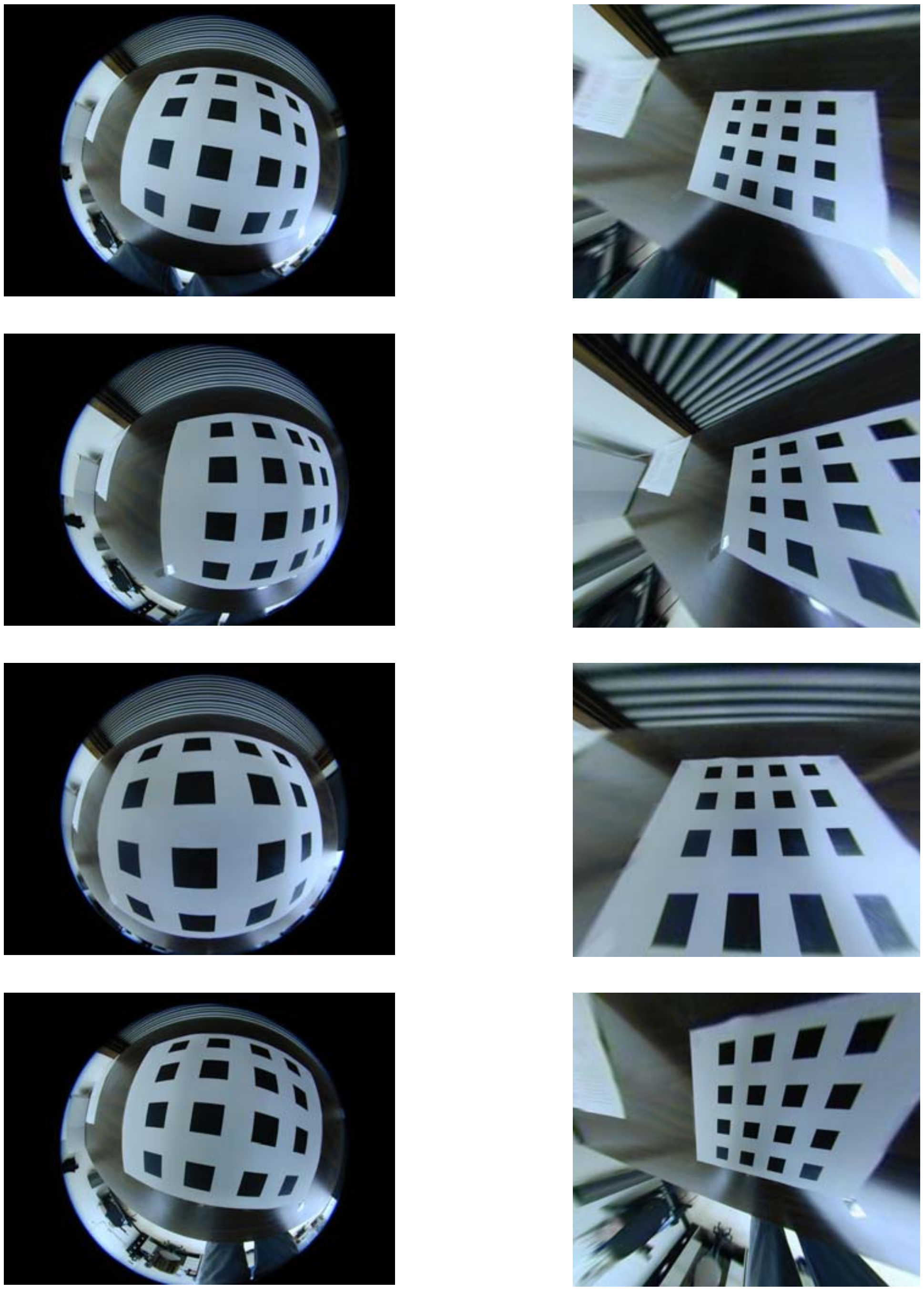


Figure 6.11: Continued

7 CONCLUSIONS AND FUTURE WORK

In this thesis, we have described a general camera calibration technique which performs equally well on a wide range of cameras, regardless of lens projection or quality. To our knowledge, we do not know of an existing calibration technique that performs well across the spectrum of cameras within a unified framework. The technique uses several images of a planar pattern taken at different positions of the camera. Since our method models lens projection, we compared it with that of modeling radial distortion, and in all experiments our method outperformed, or worked as well, as Zhang's methods [Zhang00A] based on modeling radial distortion. We used statistical information criteria to automatically select the complexity (number of coefficients) of the lens projection and decentering distortion model, which allowed us to use the least number of coefficients which sufficiently model the camera.

The contribution of this work lies in universally, fully automatic camera calibration, the application of statistical information criteria to select the complexity of the model, and our experimental results which show this method works better than traditional methods which model radial distortion especially on wide angle cameras. One of the main troubles was convergence to an acceptable local minimum during optimization. We achieved good results with a perspective projection, or rectilinear camera. However, stereographic projection and fisheye cameras were more difficult to calibrate. The alternation technique during bundle adjustment used in the optimization step helped considerably, allowing us to achieve good results across a wide range of lens projections.

We believe the techniques described in this thesis can be fruitful, giving an alternative approach to traditional methods of camera calibration. Generally, when we implement a theory, we want to automate as much of the process as possible. The introduction of model selection in camera calibration brings us one step closer to automation.

We feel that camera calibration has significant room for improvement, especially in the calibration of wide angle and fisheye cameras. There are several problems when calibrating these types of cameras. The extreme distortion exhibited in fisheye images make it more difficult to get accurate detection of the corners of the grid. Since fisheye images also have a 180° FOV in all directions, it is more difficult to accurately represent the FOV by taking images of a planar target. We also used a pinhole model to calibrate cameras which do not obey the pinhole model, such as a wide angle or fisheye, which is quite typical in camera calibration. Thus we expect better results if we use radial distortion or lens projection to extend the pinhole model to accommodate these cameras, which we did in this thesis. This causes problems when computing the initial estimates of the intrinsic and extrinsic parameters from the projection matrix. As we saw in the synthetic test results, the focal length was off target from the ground truth for stereographic, equisolid and orthogonal projections. Perhaps there is a better way to extract the parameters from the projection matrix which considers the lens projection of the camera. If the initial estimates are off target, we cannot depend upon nonlinear optimization to achieve a good local minimum. We also did not analyze the use of 3D targets in calibration, or compare our method to methods which use 3D targets, although we did describe these techniques. In any case, we would expect these methods to reduce the calibration error, but by how much we are not sure. In theory though, if we have three planar targets, it should perform equally well as having a single 3D target. We also did not investigate different parameterization of the rotation matrix, such as Axis/Angle or Quaternions. It is unclear if the use of other parameterizations would yield a significant improvement.

BIBLIOGRAPHY

[Abdel-Aziz71] Y. Abdel-Aziz and H. Karara, "Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry," *In Proceedings of the Symposium on Close-Range Photogrammetry*, American Society of Photogrammetry, Falls Church, VA, pp. 1-18, 1971.

[Abidi85] M. A. Abidi and R. O. Eason, "Camera Calibration in Robotic Vision," In Proceedings of the 4th Scandinavian Conference on Image Analysis, pp. 471-478, 1985.

[Akaike74] H. Akaike, "A New Look at the Statistical Model Identification," *In IEEE Transactions on Automatic Control*, vol. 19, no. 6, 1974.

[Basu95] A. Basu and S. Licardie, "Alternative models for the fish-eye lenses," *Pattern Recognition Letters*, vol. 16, pp. 433-431, 1995.

[Bozdogan87] H. Bozdogan, "Model Selection and Akaike's Information Criterion," *Psychmetrika*, vol. 53, no. 3, pp. 345-370, 1987.

[Brown66] D. Brown, "Decentering distortion of Lenses," *Photogrammetric Engineering*, pp. 444-462, vol. 32, no. 3, 1996.

[Faugeras87] O.D. Faugeras and G. Toscani, "Camera Calibration for 3D Computer Vision," *In Proceedings from the International*

Workshop on Machine Vision and Machine Intelligence, Tokyo, Japan, February, 1987.

[Faugeras93] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*, The MIT Press, Cambridge, Massachusetts, 1993.

[Folm-Hansen99] J. Folm-Hansen, "On Chromatic and Geometrical Calibration,"PhD Dissertation, Technical University of Denmark, 1999.

[Gheissari03] N. Gheissari and A. Bab-Hadiashar, "Model Selection Criteria in Computer Vision: Are They Different?" *In Proceedings of VII Digital Image Computing: Techniques and Applications*, pp.185-194, December 2003.

[Golub96] G. Golub and C. Loan, *Matrix Computations*, 3rd edition, John Hopkins University Press, 1996.

[Hartley97] R. Hartley, "In Defense of the Eight-Point Algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 6, pp. 580-593, June 1997.

[Hartley00] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.

[Heikkila97] J. Heikkila and O. Silven, "A Four-Step Camera Calibration Procedure with Implicit Image Correction," *In Proceedings of*

IEEE Conference on Computer Vision and Pattern Recognition, pp.1106-1112, 1997.

[Heikkila00] J. Heikkila, "Geometric Camera Calibration Using Circular Control Points," *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1066-1077, October 2000.

[Hornegger99] J. Hornegger and C. Tomasi, "Representation Issues in the ML Estimation of Camera Motion," *In Proceedings of The Seventh IEEE International Conference on Computer Vision*, Kerkyra, Greece, pp. 640-647, 1999.

[Kannala04] J. Kannala, "Measuring the Shape of Sewer Pipes from Video," Master's Thesis, Helsinki University of Technology, August 2004.

[Lay96] D. Lay, *Linear Algebra and its Applications*, 2nd edition, Addison-Wesley, 2000.

[Lenz87] R. Lenz and R. Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3D Machine Vision Metrology," *In Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 68-75, Raleigh, NC, 1987.

[Rissanen78] J. Rissanen, "Modeling by Shortest Data Description," *Automatica*, vol. 14, pp. 465-471, 1978.

[Schmidt01] J. Schmidt and H. Niemann, "Using Quaternions for Parametrizing 3-D Rotations in Unconstrained Nonlinear Optimization," *In Proceedings on Vision, Modeling and Visualization*, pp. 399-406, November 2001.

[Schwarz78] G. Schwarz, "Estimating the Dimension of a Model," *Annals of Statistics*, vol. 6, pp. 461-464, 1978.

[Seedahmed02] G. Seedahmed and A. Habib, "Two New Algorithms to Retrieve the Calibration Matrix from the 3-D Projective Camera Model," *ISPRS Commission III Symposium, Photogrammetric Computer Vision*, Graz, Austria, September 9-13, 2002.

[Slama80] C. Slama (ed.), *The Manual of Photogrammetry*, 4th edition, American Society of Photogrammetry, 1980.

[Swaminatha00] R. Swaminathan and S. K. Nayer, "Nonmetric Calibration of Wide-Angle Lenses and Polycameras," *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1172-1178, October 2000.

[Tsai86] R. Tsai, "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision," *In IEEE Conference on Computer Vision and Pattern Recognition*, pp. 364-374, 1986.

[Tsai87] R. Tsai, "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf

TV Cameras," *IEEE Journal of Robotics and Automation*, vol. 3, no. 4, pp. 323-344, August 1987.

[Weng92] J. Weng, P. Cohen and M. Herniou, "Camera Calibration with Distortion Models and Accuracy Evaluation," *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 10, October 1992.

[Zhang00A] Z. Zhang, "A flexible new technique for camera calibration," *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no.11, pp. 1330-1334, November 2000.

[Zhang00B] Z. Zhang, Experimental Data and Results for Camera Calibration, Microsoft Research Technical Report, Available: http://research.microsoft.com/~zhang/calib/, 1998.

Vita

Christopher Paul Broaddus was born in Houston, TX on March 8, 1980. He attended the University of Houston and graduated from Sam Houston State University at the top of his class with a Bachelor of Science in Computer Science and a minor in Mathematics. This led him to pursue a Masters degree in Electrical Engineering at the Imaging, Robotics and Intelligent Systems Laboratory at The University of Tennessee in Knoxville, TN. The education and research experience gained in image processing and computer vision then led him to Sarnoff Research in Princeton, NJ where he is employed full-time in the Vision and Visualization group as technical staff.