Fisheye Camera Calibration and Applications

by

Vinay Kashyap Takmul Purushothama Raju

A Thesis Presented in Partial Fulfillment
of the Requirement for the Degree
Master of Science

ARIZONA STATE UNIVERSITY

December 2014

ABSTRACT

Fisheye cameras are special cameras that have a much larger field of view compared to conventional cameras. The large field of view comes at a price of non-linear distortions introduced near the boundaries of the images captured by such cameras. Despite this drawback, they are being used increasingly in many applications of computer vision, robotics, reconnaissance, astrophotography, surveillance and automotive applications. The images captured from such cameras can be corrected for their distortion if the cameras are calibrated and the distortion function is determined. Calibration also allows fisheye cameras to be used in tasks involving metric scene measurement, metric scene reconstruction and other simultaneous localization and mapping (SLAM) algorithms. The fisheye camera is considered a central omnidirectional cameras and in recent years many different methods to calibrate central omnidirectional cameras have been developed, based on the type of camera model. This thesis presents a calibration toolbox that implements a collection of some of the most widely used techniques for calibration of fisheye cameras under one package. This enables an inexperienced user to calibrate his/her own camera without the need for a theoretical understanding about computer vision and camera calibration. This thesis also explores some of the applications of calibration such as distortion correction and 3D reconstruction.

ACKNOWLEDGMENTS

I would like to acknowledge the strong guidance and support of my thesis advisor Dr. Lina J. Karam in helping me bring this thesis to fruition. I would like to extend my gratitude towards committee members Dr. Pavan Turaga and Dr. Cihan Tepedelenlioglu for agreeing to serve on my committee. I would also like to acknowledge the support offered by Farshad Akhbari of Intel whose group partly supported this thesis.

I would specially like to acknowledge the extremely valuable assistance provided by my fellow lab mates Tejas Borkar, Charan Prakash, Aditee Shrotre, Milind Gide, Jinjin Li, Sam Dodge, Tong Zhu and Bashar Haddad at Image, Video Usability(IVU) lab for their unending support and guidance. Last but not the least, I would like to express my utmost gratitude to my parents, my brother Nithin and my dearest friends Shruti, Nishanth, Naresh, Vidhyacharan, Balaji, Gokul and Varun for their strong love, support, and for being there for me throughout the journey.

TABLE OF CONTENTS

Page

LIST OF TABLES v

LIST OF FIGURES vi

CHAPTER

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Contributions 3

1.3 Thesis Organization 4

2 BACKGROUND 5

2.1 Omnidirectional Camera 5

2.2 Camera Model 6

2.3 Approaches to fisheye camera calibration 9

2.4 Fisheye Distortion Functions 13

3 CALIBRATION 16

3.1 Scaramuzza marker-based calibration 16

3.1.1 Obtaining the external parameters 22

3.1.2 Estimating the intrinsic parameters 24

3.1.3 Non-Linear Refinement 25

3.2 Kannala Marker based calibration 26

3.2.1 Determining principal point and control points 28

3.2.2 Initializing intrinsic parameters 33

3.2.3 Extrinsic parameters 33

3.2.4 Minimization of the projection error 36

3.3 Kannala Auto-Calibration 37

3.3.1 Obtaining Point Correspondences 41

3.3.2 Epipolar geometry and angular error 46
3.3.3 Calibration by minimization of the angular error 50
4 DISTORTION CORRECTION 52
4.1 Inverse distortion functions 53
5 FISHEYE CAMERA CALIBRATION AND DISTORTION CORRECTION (FisheyeCDC) TOOLBOX 60
5.1 Toolbox Layout 60
5.2 Format choices for output of calibration 66
5.3 Calibration Procedure 67
5.4 Distortion Correction procedure 68
5.5 Results 69
6 RECONSTRUCTION ON FISHEYE IMAGES 74
6.1 3D SFM-based Reconstruction for Planar Cameras 74
6.2 Extension of SFM to fisheye cameras 80
6.3 Results 85
7 CONCLUSION 89
7.1 Contributions 89
7.2 Future Work 90
REFERENCES 92

LIST OF TABLES

Table Page

4.1 Parameters of Fisheye Camera obtained by Scaramuzza Calibration[1]. . 57

4.2 Internal Parameters of Fisheye Camera obtained by Kannala marker-based Calibration[2]. 58

4.3 Parameters of Fisheye Camera obtained by Kannala Auto-Calibration[3]. 59

5.1 Performance of Fisheye Camera Calibration in terms of RMS Reprojection Error. 71

5.2 Effect of Number of Calibration Patterns on RMS Error. 71

6.1 Possible Projection Matrices obtained from Decomposition of Essential Matrix. 79

LIST OF FIGURES

Figure Page

2.1 Fisheye Lens and Fisheye Image. 6

2.2 Perspective Camera Model. C is the Camera Center; O is the Principal Point; **M** is the World Point being Imaged; **m** is the Projection of **M** on the Camera Plane; $\mathbf{Z_c}$ is the Optical Axis. 7

2.3 Projections Functions. Plots A to E correspond respectively, to Projections in Equations (2.1) and (2.5)- (2.8). 9

2.4 Fisheye Camera Model. C is the Camera Center; O is the Principal Point; **M** is the World Point being Imaged; m is the Projection of M on the Image Plane; f is the Focal Length; $\mathbf{Z_c}$ is the Optical Axis; θ is the angle of incoming ray with the Optical Axis; r is the Radial Distance of m from O; ψ is the Angle made by the Radial line of m with x Axis of the Image Plane; **m'** is the Projection of **M** if the Camera were following a Perspective Projection Model and r' is the Radial distance of $\mathbf{m}'$ from O. 10

2.5 Examples of 2D planar Calibration Patterns. 11

2.6 Example of 3D Calibration Pattern. 12

3.1 The Mapping of a Scene Point **X** into a Sensor Plane to a Point **u"** [4]. 17

3.2 Geometric Interpretation of Fisheye Lens Projection[4]. Projection of the vector **q"**$->$ **p"**$->$ **u"** into a Sensor Plane π through a Function $g(r)$. 19

3.3 Checkerboard Images required for Calibration. 21

3.4 Checkerboard Images with Extracted Corners. 22

3.5 Fisheye Camera Model. 27

3.6 Example Calibration Patterns. 29

Figure Page

3.7 Ellipse Fitting to Image Boundary. 31

3.8 Control Point Extraction from Calibration Image. 32

3.9 Reprojection onto Unit Sphere. 34

3.10 General Model for Central Camera [5]. 38

3.11 Polar Coordinates of 3D Point X. 40

3.12 Computing Difference-of-Gaussian for different Octaves. 43

3.13 Detecting Extremal Points across Scales. 44

3.14 Epipolar Geometery of Fisheye Cameras. The Fisheye Cameras represented by Spherical Retinas whose Radius is 1. (e_1, e_1') represents the Epipoles of Camera 1 with centre O and (e_2, e_2') represents the Epipoles of camera 2 with centre O'. The Epipolar curves are shown in Orange on the two Retinas and **q,q'** represent the Unit Vectors of **OP,O'P** and intersect the Spherical Retina at **Q,Q'**. 47

3.15 Angular Error. 49

4.1 Forward and Backward Models for Fisheye Cameras. 53

4.2 Relation between a Pixel in Fisheye and Corrected Image. 54

4.3 Distortion Correction using Fisheye Camera Parameters that are obtained using Scaramuzza method[1] with the distortion coefficient Length of four. 56

4.4 Distortion Correction using Fisheye Camera Parameters that are obtained using Kannala marker-based Method[2] with the Distortion Coefficient length of Four. 57

4.5 Distortion Correction using Fisheye Camera Parameters that are obtained using Kannala Auto-Calibration Method[3]. 58

Figure Page

5.1 Layout of FisheyeCDC Toolbox. 61

5.2 Toolbox with Calibration Tab Active. 62

5.3 Toolbox with Distortion Correction Tab Active. 64

5.4 Warning and Error Messages. 65

5.5 Effect of Number of Calibration Images on RMS Reprojection Error. . . 72

5.6 Error Plot of Reprojected Points in x and y Directions. (a) and (b) show the Error Plots in x and y Directions of Kannala marker-based technique[2] for Three and Four Calibration Images respectively;(c) and (d) show the Error Plots in x and y directions of Scaramuzza technique[1] for Three and Four Calibrations Images respectively. 73

5.7 Error Plot of Reprojected points in x and y Directions for Kannala Auto-Calibration[3]. 73

6.1 Epipolar Geometery between Two Cameras viewing the Same Point. O_1 and O_2 are the Camera Centers with Camera Planes C_1 and C_2 respectively; x_1 and x_2 are the Projections of Scene Point X on C_1 and C_2 respectively; e_1 and e_2 denote the Epipoles of the 2 Cameras; l_1 and l_2 are the respective Epipolar lines. 75

6.2 Image Correspondences after the RANSAC Stage. 81

6.3 Flowchart for 3D Reconstruction using Fisheye Cameras. 84

6.4 Original Fisheye Images. 86

6.5 Image Matches after RANSAC Stage. 86

6.6 Regions of Known Depth enclosed by White Rectangles and their Corresponding Feature Points. 87

6.7 Top View of Scene setup showing Distances of Regions from the Camera. 87

Figure Page

6.8 Calculated Depth versus Ground Truth Comparison for the Regions of Interest. 88

Chapter 1

INTRODUCTION

This chapter presents the motivations behind the work in this thesis and summarizes the contributions and organization of this thesis.

1.1 Motivation

In recent decades, with significant progress in semiconductor industry, there has been an increase in the types of camera sensors that have been introduced into the market at very low prices. This has made cameras available for a number of new age applications involving automation, computer vision, robotics, entertainment, and surveillance where they are being used not just for traditional imaging but also for other computational tasks, and for measurement and understanding applications.

Various combinations of sensors and optical elements lead to different types of cameras. The type of lens attached to the imaging sensor defines the geometry of the camera, which in turn determines its field of view (FOV). Based on the FOV, cameras can be differentiated into directional and omnidirectional cameras. Directional cameras typically have a narrow field of view and a perspective projection. Omnidirectional cameras include dioptric (Fisheye) and catadioptric (Para, Hyperbolic and spherical) cameras. Omnidirectional cameras generally have a large field of view and a non-linear projection function which makes the images captured by them appear distorted. In this thesis we are primarily interested in omnidirectional cameras and in particular fisheye cameras as they are being widely used in several recent applications.

There are many approaches to planar camera calibration [6][7][8]. Most of these methods use a calibration target and perform localization of control points of the calibration target captured in the images of the camera. Many toolboxes [9][10] are publicly available for the calibration of planar cameras. Such toolboxes typically determine the focal length *f*, principal point O and small non-linear distortions if any. Certain toolboxes [9] also output information about camera motion in terms of rotation and translation vectors. Most toolboxes designed for planar cameras cannot be used with fisheye cameras because the fisheye non-linear distortion is too large to be modeled by planar camera-specific projection functions. This is where calibration for omnidirectional cameras comes into the picture.

In this thesis fisheye cameras are modeled as central cameras. Central cameras have a single center of projection, so that every pixel in the sensed images measures the irradiance of the light passing through the same viewpoint in one particular direction. In other words, all the rays entering the camera meet at a point known as the camera center. Moreover, this central camera model allows applying the known theory of epipolar geometry [11] directly on unit vectors describing the directions of the incoming rays. The large distortion function of the fisheye camera can be modeled using various functions[12][2][1]. Knowledge of the distortion function also allows us to correct the image for it to appear as if captured by a perspective camera. This is specifically due to the single viewpoint property of the fisheye camera that permits the generation of geometrically correct perspective images from the corresponding captured fisheye images. Fisheye cameras have a single view point as they are central cameras and the incoming rays meet at the camera center.

There are numerous methods that were proposed to calibrate a fisheye camera depending on the prior knowledge of the scene, camera manufacturer specifications and other parameters. Each of these model the fisheye camera using different parameters which are then used to describe the properties of the camera. In this thesis, some of the most popular fisheye calibration techniques are evaluated and incorporated into a user-friendly and automated fisheye camera calibration package.

Since there are such wide uses of fisheye cameras, even in fields that are not particularly related to computer vision or photogrammetry such as the automation industry and exploratory sciences to name a few, some practitioners and users of calibration systems might not have in-depth knowledge about camera models and parameters. Even for users and researchers familiar with camera calibration methods, evaluating and running separately calibration methods is a time consuming task. We have designed an interactive user-friendly Fisheye camera Calibration and Distortion Correction (FisheyeCDC) toolbox that incorporates state-of-the-art approaches with the goal of making the process of calibration easier for practitioners and users without having to worry about programming details and about having to become familiar with different implementations using different platforms. The developed FisheyeCDC toolbox also consists of a distortion-correction mode that allows users to correct for distortions in the fisheye images and convert them to planar images.

1.2 Contributions

In this thesis a user-friendly graphical user interface based toolbox is to perform calibration of fisheye cameras as well as distortion correction of fisheye camera images. The toolbox incorporates some the most widely used calibration techniques in one package. The usage of the toolbox does not require any prior knowledge of camera calibration techniques. The main contribution of this work is incorporating the various existing methods for marker-based calibration and also an implementation of an auto-calibration method that is based on estimating epipolar geometry from computed putative matches obtained from two images observing the same scene. The user has the option to use a marker-based calibration where images of a calibration pattern are provided as the input for calibration[2][13][1][14], or to use an auto-calibration method where the input to the calibration consists of two images of the same scene taken from different viewpoints[3]. In both cases the toolbox outputs the camera calibration parameters including the distortion function parameters, principal point, and the dimensions (width and height) of the images being used for calibration.

Once the calibration is done, the user has the option of saving the distortion parameter in a file to be used at a later stage. Each of these calibration methods typically results in different parameters defining the distortion of the camera. The developed toolbox provides a distortion correction functionality using the camera calibration parameters. The user has the option of distortion correction immediately after the calibration is done or, if the user already calibrated the camera at an earlier time, he/she can just load the saved calibration parameters and perform the distortion correction. Additionally, as a part of this thesis, using internal camera calibration parameters, we implemented and demonstrated three-D structure reconstruction (depth estimation) directly from fisheye images.

1.3 Thesis Organization

This thesis is organized as follows. Chapter 2 gives a background about the various camera imaging models. Chapter 3 gives a detailed description of the calibration procedure for each of the calibration techniques incorporated in the developed Fisheye camera Calibration and Distortion Correction(FisheyeCDC) toolbox. Chapter 4 gives the theory behind distortion correction using the calibration parameters. Chapter 5 describes the implemented FisheyeCDC toolbox and its functionalities. Chapter 6 presents an application of calibration where the fisheye camera calibration parameters are used for implementing structure from motion on fisheye images. Chapter 7 summarizes the contributions of this thesis and proposes possible future enhancements of this work.

Chapter 2

BACKGROUND

This chapter provides some background information about omnidirectional cameras, including types of omnidirectional cameras and how the fisheye camera is related to an omnidirectional camera. It also provides a brief overview of fisheye camera representations, models and their respective distortion functions.

2.1 Omnidirectional Camera

Omnidirectional cameras are a class of cameras that have a very wide field of view. Their projection model differs from the traditional perspective projection model of a planar camera in the sense that projection functions for omnidirectional cameras are usually non-linear. Their major advantage is their wide field of view compared to planar cameras.

Omnidirectional cameras can be broadly classified as central and non-central cameras. A camera can be represented by a subset of the set of lines in $\mathbb{P}^3$ and a central camera in particular is defined as a subset of lines in $\mathbb{P}^3$ passing through a single point called a projection center C. Examples of central cameras include fisheye cameras, central aligned para-catadioptric[4], hyperbolic catadioptric[4] and spherical catadioptric cameras[4]. A non-central camera can be defined as the camera whose incoming rays do not intersect at a point. In other words, non-central cameras do not have a single effective view-point. The rotating camera [15], omnivergent camera[16] and conical-mirror catadiotpric camera[17] are examples of non-central cameras. This thesis is concerned with the fisheye camera. The fisheye camera is typically represented using a spherical camera model as discussed in Section 2.2. Figure 2.1 shows a fisheye

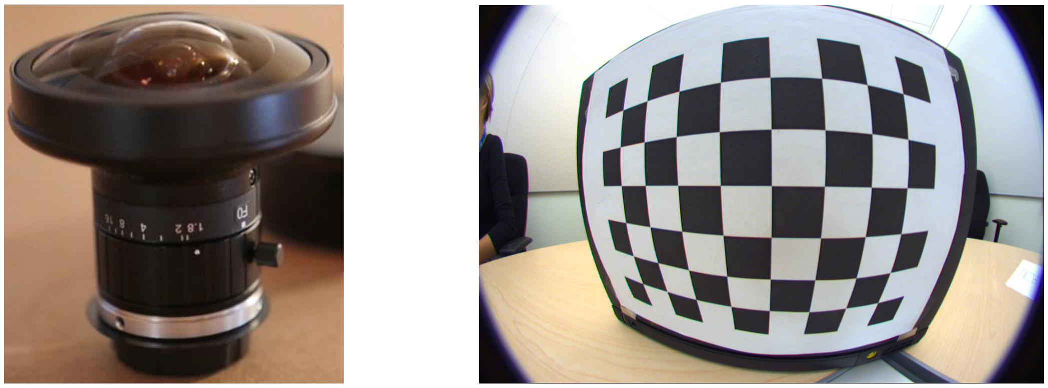

(a) Fisheye Lens(Fujinon FE185C086HA-1). (b) Image from fisheye camera.

Figure 2.1: Fisheye lens and Fisheye image.

lens (Fujinon FE185C086HA-1) which in combination with an image sensor forms the fisheye camera and an image captured by the fisheye camera. From Figure 2.1b it can be seen that the image is highly distorted at the edges due to the non-linear projection model.

2.2 Camera Model

This section describes the perspective(planar) and fisheye camera models. A typical perspective camera can only image points in a narrow field of view in front of the camera imaging plane. The perspective projection of a pinhole camera is described as follows

$$r = f \tan\theta \tag{2.1}$$

where θ is the angle between the principal axis and the incoming ray as shown in Figure 2.2. O is the principal point, which is the point of intersection of the principal axis and the imaging plane. f is the focal length and r is the distance between the image point and the principle point. The standard perspective camera model maps all the scene points $\mathbf{M}$ to one image point C called the optical center of the camera.

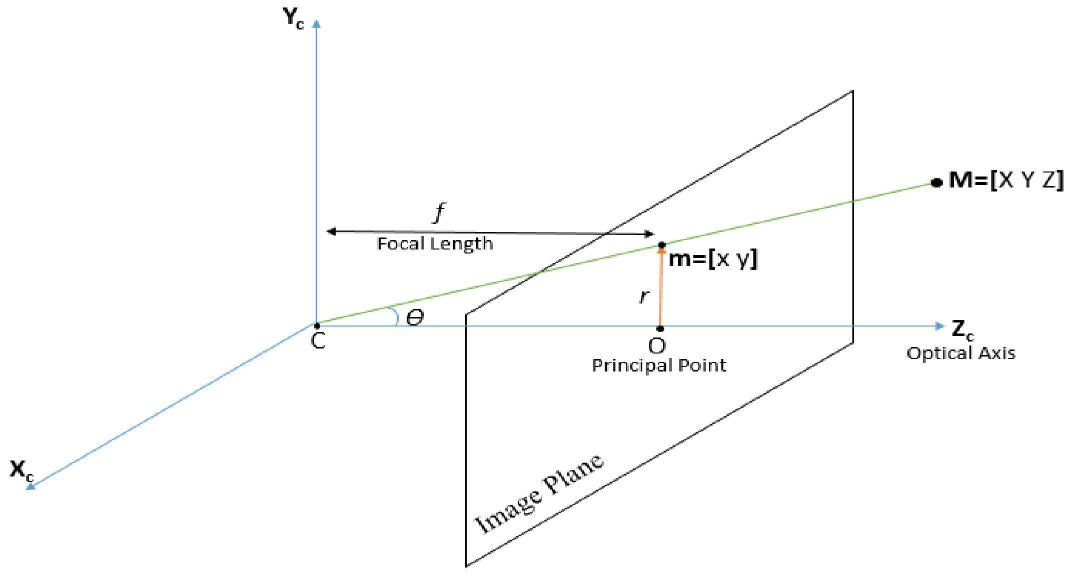


Figure 2.2: Perspective camera model. C is the camera center; O is the principal point; $\mathbf{M}$ is the world point being imaged; $\mathbf{m}$ is the projection of $\mathbf{M}$ on the camera plane; $\mathbf{Z_c}$ is the optical axis.

The line joining the optical center and the scene point intersects the imaging plane at a 2D point **m** which forms the image co-ordinate of the scene point **M**. We can define a matrix **P** such that

$$\alpha \mathbf{m} = \mathbf{PM} \tag{2.2}$$

where P$\in \mathbb{R}^{3X4}$ is a projection matrix, **M**$\in \mathbb{R}^4$ is a homogenized scene point, **m**$\in \mathbb{R}^3$ represents an homogenized image point and α is a scalar number. Homogenized co-ordinates are expressed with a **1** augmented after the co-ordinates. For example **M=[X Y Z 1**] represents a point X, Y, Z in space. Similarly **m=[x y 1**] represents an image co-ordinate [x y] on the imaging plane. Further, the Equation (2.2) can be represented as[6] :

$$\alpha \mathbf{m} = \mathbf{K}[\mathbf{R}\ \mathbf{t}]\mathbf{M} \tag{2.3}$$

where the projection matrix is decomposed into an intrinsic parameter matrix $\mathbf{K}$ called the camera matrix and a rotation $\mathbf{R}$ and translation $\mathbf{t}$ matrices corresponding to the camera extrinsic parameters. The camera intrinsic matrix is given by

$$\mathbf{K} = \begin{bmatrix} f_x & \gamma & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$\mathbf{K}$ is a 3×3 matrix containing the intrinsic parameters namely the focal lengths along the x and y axes of the image (f_x, f_y), skew between the two image axes (γ), and the principal point (u_0, v_0). In this model we can represent only the scene in front of the camera with the edges of the image plane being the border. This restricts the field of view of the perspective camera to about 60°.

In the case of a fisheye camera, a more radially symmetric projection model is assumed. Different camera types can be designed and can be approximated using different projection models, popular examples of which follow below:

$$r = 2\, f \tan(\theta/2) \tag{2.5}$$

$$r = f\, \theta \tag{2.6}$$

$$r = 2\, f \sin(\theta/2) \tag{2.7}$$

$$r = f \sin(\theta) \tag{2.8}$$

where r is the radial distance of the pixel from the principal point and θ is the angle made by the incoming ray with the optical axis as shown in Figure 2.4. Equations (2.5)-(2.8) represent, respectively, a stereograhic projection, an equidistance projection, an equisolid projection, and an orthogonal projection. Figure 2.3 shows the corresponding projection functions (plots B to E) that can be used to model a fisheye lens/camera. For comparison, Figure 2.3 also shows the perspective projection (Plot A).

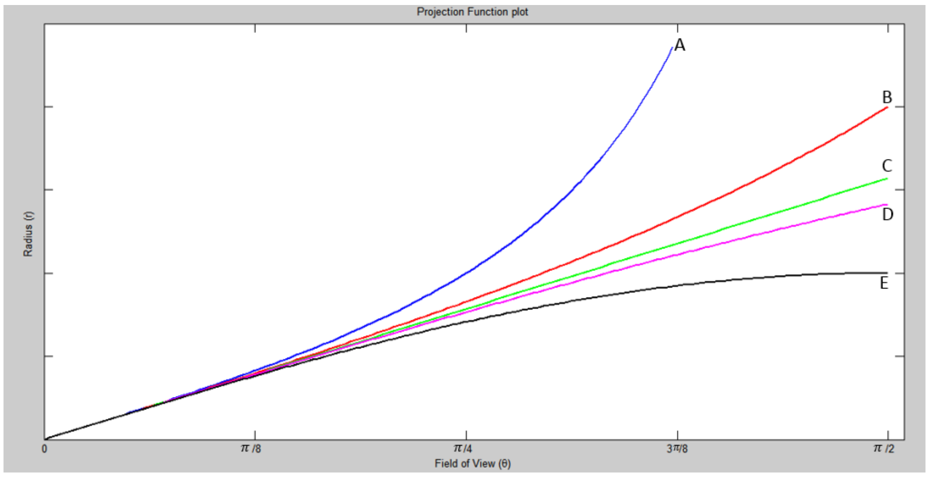


Figure 2.3: Projections functions. Plots A to E correspond respectively, to projections in Equations (2.1) and (2.5)- (2.8).

The general fisheye camera model is shown in Figure 2.4. From Figure 2.4, it can be seen that image of the 3D point in space **M** = [**X, Y, Z**] is **m** = [**x, y**] whereas it would be **m'** in a perspective camera. The point **m** on the image plane is determined by one of the projection functions in Figure 2.3. Similar to the perspective camera, θ defines the angle of incident ray from the point **M** to the camera center C. O refers to the principal point or center of the imaging plane. The distance from the principal point O to the image point **m** is the radius r. For fisheye cameras, this radius r is a function of the incident angle θ. This is a radially symmetric model similar to the model used in [2]. In reality, the manufacturing of a fisheye lens is intricate and often there are deviations from the ideal projection models shown in Figure 2.3. Fisheye calibration is aimed at modeling these imperfect projection models.

2.3 Approaches to fisheye camera calibration

There are numerous approaches to calibrating the fisheye camera. They can be broadly separated into two categories mainly marker-based calibration and auto-

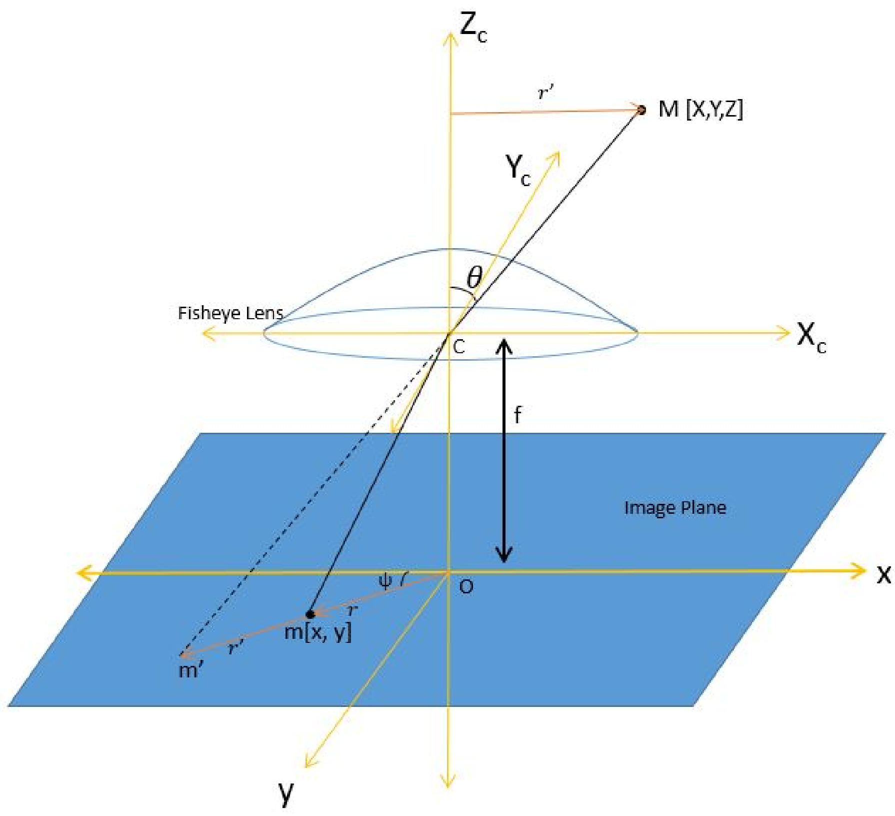
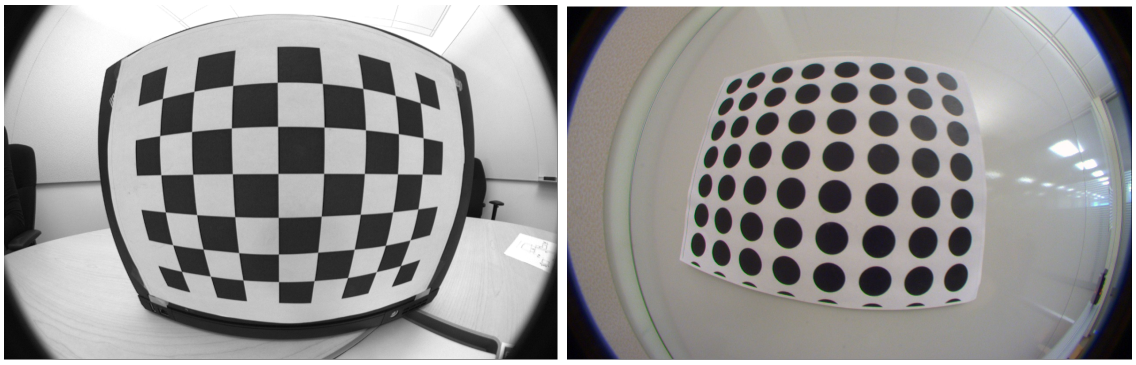


Figure 2.4: Fisheye camera model. C is the camera center; O is the principal point; **M** is the world point being Imaged; m is the projection of M on the image plane; f is the focal length; $\mathbf{Z_c}$ is the optical axis; θ is the angle of incoming ray with the optical axis; r is the radial distance of m from O; ψ is the angle made by the radial line of m with x axis of the image plane; **m'** is the projection of **M** if the camera were following a perspective projection model and r' is the radial distance of $\mathbf{m}'$ from O.

calibration. As part of this work, the marker-based methods of [2],[1], and the auto-calibration method of [3] have been integrated in the developed GUI-based calibration and distortion-correction toolbox.

Marker-based calibration: These methods use a marker or a calibration pattern to obtain a relationship between pattern points in 3D space and their projections



(a) Checkerboard calibration pattern. (b) Circular calibration pattern.

Figure 2.5: Examples of 2D planar calibration patterns.

on the 2D camera image plane. These methods often require a few images of a known pattern or object to be captured by the camera. Knowing this relationship, one can obtain the distortion function and other internal parameters of the fisheye camera. Marker-based calibration itself can be divided into a few more categories based on the type of calibration pattern used. Broadly they can be categorized as 2D pattern calibration[2][1][18][19][20][21] and 3D point based calibration[22]. As the name suggests, 2D pattern calibration uses a planar 2D pattern of known geometry. There are certain points on the calibration pattern, whose position in space is known prior to them being imaged. In [2] the calibration pattern consists of a grid of circular patterns and the control points consist of the centroids of these circular patterns. In [1] the calibration pattern consists of a checkerboard image and the control points are corners of the each block in the checkerboard pattern. An example of the 2D calibration patterns used in [2] and [1] are shown in Figure 2.5. Once the image is captured and the control points determined in the image, the distortion is modeled based on the relationship between positions of the control points in the captured image and known 3D corresponding points. This allows the estimation of the camera intrinsic parameters. Figure 2.6 shows the 3D calibration pattern used in [22]. The

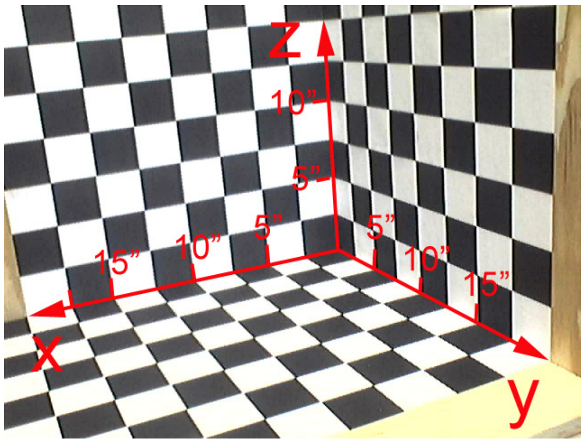


Figure 2.6: Example of 3D calibration pattern.

method [22] discrete linear Transform(DLT) based approach and requires only a single image of the 3D calibration pattern. The control points are again corners of the checkerboard on each of the planes. Here a homography can be estimated between the imaged pattern and the original calibration pattern based on the control points. The resulting homography is used to calibrate the camera and determine the intrinsic parameters.

Auto-Calibration: These methods are also referred to as self-calibration techniques as calibration is done without the support of a marker whose geometry is known. More often than not such methods require more than a single image of the scene or some specific requirements in the structure of the scene itself. Certain self calibration techniques use only point correspondences in multiple views, without needing to know either the 3D location of the points or the camera locations[3][4]. Here, the camera distortion function is estimated along with the external parameters of the camera motion between the images. Most of these methods are based on the fact that central cameras obey epipolar geometry as will be discussed in Sections 3.3.2 and 6.1. Other auto-calibration methods include line-based calibration methods[23] in which calibration is done by detecting straight lines in the scene. The detection of straight is performed by detecting curved lines in a single captured image[23][24]. Distortion parameters are estimated by determining the distortion function parameters that can best transform curved lines into straight lines. Such methods require only a single image of a scene that have a few well defined straight lines. The disadvantage of this method is that one can obtain only the intrinsic parameters of the camera and cannot estimate extrinsic parameters between 2 camera views.

2.4 Fisheye Distortion Functions

This section presents the functions that are typically used to model the distortion introduced due to a fisheye lens. As indicated in Section 2.2, in practice the design of fisheye lenses may not follow one of the projection functions given by Equations (2.5)-(2.8). Hence, other functions need to be used to model the additional introduced distortion. A radial distortion model for planar cameras using odd order polynomials was proposed in [25], and the authors of [8] and [6] adopted the model of [25] for the calibration of off-the-shelf planar cameras. The Equation is of the form

$$D_x = x_d(k_1 r^2 + k_2 r^4 + ..) \tag{2.9}$$

$$D_y = y_d(k_1 r^2 + k_2 r^4 + ..) \tag{2.10}$$

where (x_d, y_d) are the distorted image co-ordinates, $r = \sqrt{x_d^2 + y_d^2}$ is the radial distance from the center to the distorted image pixel, and (D_x, D_y) are the distortions in the X and Y directions. [8] used just one coefficient k_1 in Equations(2.9) and (2.10) to model the small distortions in the planar camera. Though this model was reasonably accurate for planar cameras, it cannot model fisheye cameras particularly well because of the high levels of radial displacement.

Some other distortion functions developed specifically for fisheye cameras include the fisheye polynomial model[1], fisheye transform[26], the FOV model[23], the polynomial fisheye transform[26] and the division model[24][12]. Some distortion models developed for catadioptric cameras[5] were shown to work well for fisheye cameras as described in [3]. The polynomial fisheye model [1] is given by

$$\theta = \sum_{i=1}^{\infty} k_n r^n = k_1 r + k_2 r^2 + k_3 r^3 + ... + k_n r^n + ... \tag{2.11}$$

where r is the radial distance from the principal point(center of the image) and θ is the angle between the incoming ray and the optical axis as shown in Figure 2.4. It has been suggested in [27] that a fourth-order form of the model is adequate to simulate the distortion introduced by fisheye lenses. The field of view (FOV) model proposed in [23] is given by

$$r_d = \frac{1}{\omega} tan^{-1}\left(2 r_u tan\left(\frac{\omega}{2}\right)\right) \tag{2.12}$$

where ω is the field of view of the camera in radians, r_d is the radial distance from the principal point in the distorted image and r_u is the radial distance from the principal point in the planar image. The FOV model is based on an simple optical model of the fisheye camera[23]. The fisheye transform in [26] is described as

$$r_d = s \ln(1 + \lambda r_u) \tag{2.13}$$

where s is a scalar and λ controls the amount of distortion. The division model [12] is given by

$$r_d = \frac{r_u}{1 - \lambda r_u^2} \tag{2.14}$$

where λ controls the amount of distortion present. A variant of this model has been adopted by [4] and [28] to solve the problem of auto-calibration converting the epipolar constraint discussed in Section 6.1 to a Quadriatic Eigenvalue problem[12]. The model used here is

$$\theta = \frac{a}{1 + b r_d^2} \tag{2.15}$$

where θ is the angle between the incident ray and the optical axis of the camera as shown in Figure 2.4, and a and b are parameters that control the amount of distortion. The model used in [3] was developed originally for a catadioptoric case in [5]. It was later shown that the same model can be applied for a for a fisheye camera.

$$r_d = \frac{(l+1)\sin\theta}{l+\cos\theta} \tag{2.16}$$

where r_d and θ are the same as in Equation (2.15) and l is the parameter that models the distortion. The calibration toolbox in [13] uses a polynomial model defined by

$$r_d = k_1\theta + k_2\theta^3 + k_3\theta^5 + ... \tag{2.17}$$

. In this thesis, the distortion models of [1], [2] and [3] given by Equations (2.11), (2.17) and (2.16), respectively are used as part of the developed calibration and distortion-correction toolbox. These models and their implementations are discussed in Chapter 3.

Chapter 3

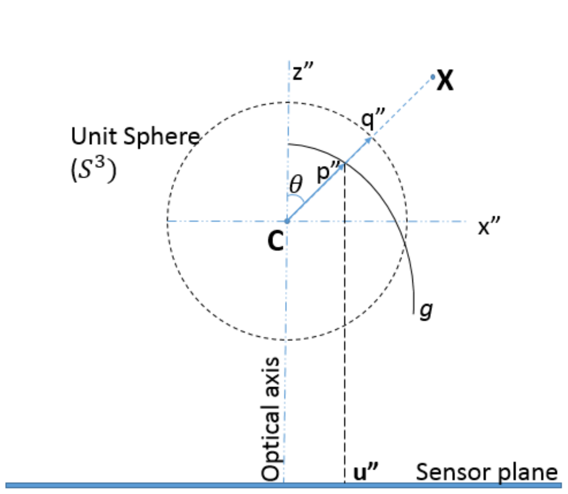
CALIBRATION

This chapter provides a detailed overview of the methods used for calibration of fisheye cameras and that are implemented in our calibration toolbox. As mentioned in Section 2.4, these methods include two marker-based calibration techniques [1][3] and an auto-calibration technique[3].

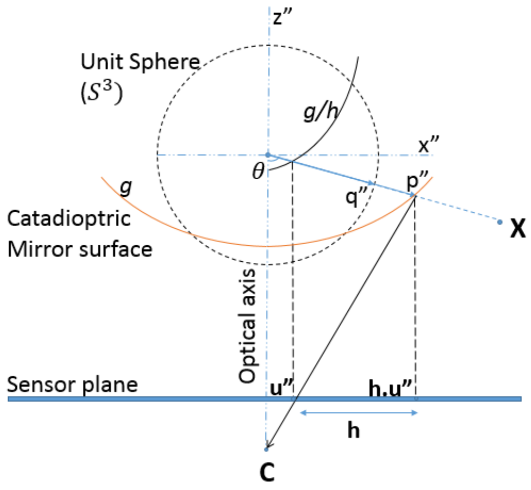
3.1 Scaramuzza marker-based calibration

This method was proposed by Scaramuzza et al. in [1] and was implemented in [14]. It models the distortion function as a Taylor series expansion whose coefficients are estimated by a two-step least-squares minimization method. This method was developed for general omnidirectional cameras but, in this thesis, its application for central fisheye cameras is considered. The omnidirectional camera model used here is a generalization of the spherical model used in [4]. Figure 3.1 shows spherical models for fisheye and catadioptric cameras.

Consider a point $\mathbf{X}$ being observed by the omnidirectional camera with center $\mathbf{C}$. The projection of point $\mathbf{X}$ on the unit sphere centered around the camera center $\mathbf{C}$ is given by the unit vector $\mathbf{q''} \in S^3$. Another vector $\mathbf{p''}$ which is in the same direction as unit vector $\mathbf{q''}$ and which maps to the sensor point u''. The vectors $\mathbf{p''}$ and $\mathbf{q''}$ are related by the functions h and g. The function h describes the mirror shape and mirror properties in case of a catadioptric camera as we can see in Figure 3.1b and function g is the radial distortion function. Hence, the vector $\mathbf{p''}$ can be represented



(a) Spherical Model-Fisheye Lens.



(b) Spherical Model-Catadioptric Camera.

Figure 3.1: The Mapping of a scene point **X** into a sensor plane to a point **u"** [4].

as

$$\mathbf{p"} = \begin{pmatrix} h(r)\mathbf{u"} \\ g(r) \end{pmatrix} \tag{3.1}$$

where $\mathbf{u"} = (u", v")^T$ are the sensor point coordinates, $r = \sqrt{u"^2 + v"^2}$ is the radial distance of **u"** from the optical axis and h and g are functions that depend on the radial distance r [4][28]. The functions g and h are dependent on the internal parameters of the camera as well. For a perspective projection, both g and h are 1; so, Equation (3.1) becomes

$$\mathbf{p"} = \begin{pmatrix} \mathbf{u"} \\ 1 \end{pmatrix} \tag{3.2}$$

For a fisheye camera, the function $h = 1$. The projection Equation (3.1) for a fisheye camera becomes

$$\mathbf{p"} = \begin{pmatrix} \mathbf{u"} \\ g(r) \end{pmatrix} \tag{3.3}$$

Several models were proposed for function $g(r)$. One such model proposed in [4] expresses $g(r)$ in terms of radial distance, r, and the angle of incidence, θ(Figure 3.2) as follows:

$$g(r,a,b) = \frac{r}{\tan\theta} \tag{3.4}$$

Figure 3.2 shows geometric interpretation of the fisheye lens model. C is the camera center and $g(r)$ represents the distortion function. As shown in Figure 3.2, θ is the angle between incident ray and the optical axis. Linear and nonlinear models for θ were defined in [4] as follows:

$$\theta = ar \qquad (Simple\ linear\ model) \tag{3.5}$$

$$\theta = \frac{ar}{1+br^2} \qquad (Sigma\ fisheye\ camera) \tag{3.6}$$

$$\theta = \frac{1}{b}\sin^{-1}\left(\frac{br}{a}\right) \qquad (Nikon\ fisheye\ camera) \tag{3.7}$$

where a and b are the parameters which describe the radial distortion of the fisheye cameras.

This model was extended by the authors of [27] to have a single model for all omnidirectional cameras[1]. Instead of having two separate functions g and h, a single function g/h was proposed to model the cameras. Since $h = 1$ for fisheye cameras [28], the distortion function is described by $g(r)$. Also, the linear and non-linear models described in [4] are replaced by a single Taylor series expansion to have a more generic form to model different cameras as shown below [1][27]:

$$g(r,\mathbf{a}) = a_0 + a_1 r + a_2 r^2 + a_3 r^3 + ... + a_N r^N \tag{3.8}$$

where the coefficients $\mathbf{a} = a_0, a_2, a_3...a_N$ and N are the calibration parameters, also referred to as intrinsic parameters and r again is the radial distance from the optical axis. Hence, we can rewrite the projection function of Equation (3.3) as

$$\mathbf{p"} = \begin{pmatrix} \mathbf{u"} \\ a_0 + a_1 r + a_2 r^2 + ... + a_N r^N \end{pmatrix} \tag{3.9}$$

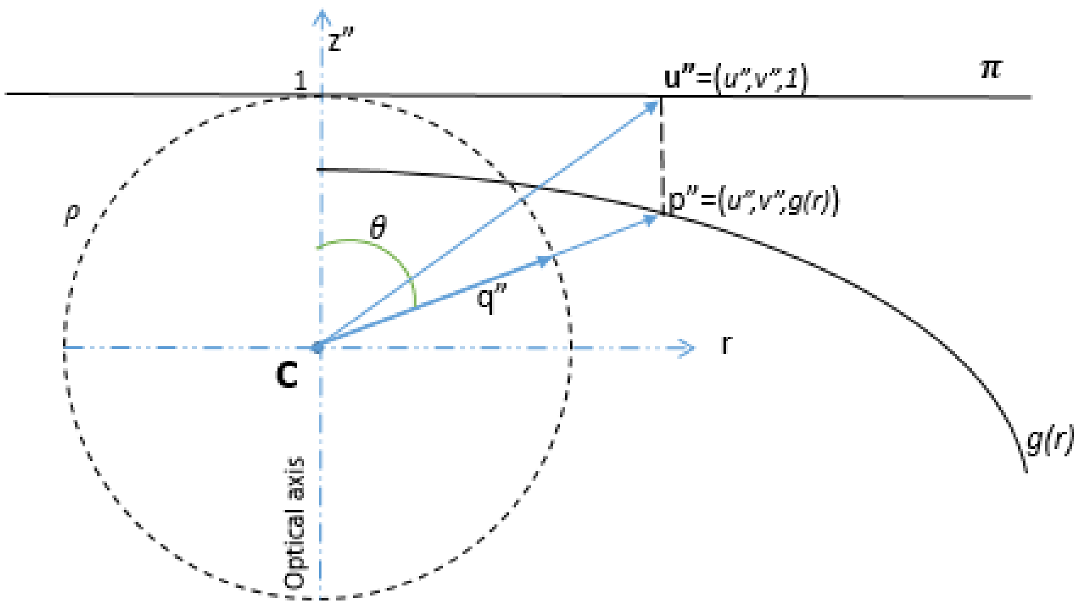


Figure 3.2: Geometric interpretation of fisheye lens projection[4]. Projection of the vector **q”**$-$ > **p”**$-$ > **u”** into a sensor plane π through a function $g(r)$.

It is shown in [29], [28] and [11] that for fisheye and catadioptric lenses:

$$\left.\frac{dg}{dr}\right|_{r=0} = 0$$

This makes the coefficient $a_1 = 0$ and thus Equation (3.8) becomes

$$g(r, \mathbf{a}) = a_0 + a_2 r^2 + a_3 r^3 + ... + a_N r^N \tag{3.10}$$

The sensor plane is expressed in metric coordinates and can be seen as the plane of the CCD sensor on which the light ray is incident. The image plane expressed in pixel coordinates is the image obtained by digitization process. There might exist small misalignments of axes in the digitization process. Hence, an affine transform **A** is defined which describes the relationship between the sensor plane coordinates and image plane coordinates due to digitization process and the axis misalignments. This affine transformation **A** is of the form

$$\mathbf{u''} = A\mathbf{u} + t \tag{3.11}$$

where $\mathbf{u} = (u, v)^T$ are image coordinates, $A \in \mathbb{R}^{2\times2}$ is the affine transform and $t \in \mathbb{R}^{2\times1}$ is a translation if any is present. Now, the relationship between a pixel point **u** and a scene point **X** is

$$\lambda.\mathbf{p''} = \lambda.f(\mathbf{u''}) = \lambda.f(A\mathbf{u} + t) = P\mathbf{X} \tag{3.12}$$

where P is the projection matrix and f is the imaging function and is given by $f(u'') = [u'', g(r)]^T$. Hence, in the final form Equation (3.12) becomes

$$\lambda. \begin{bmatrix} u'' \\ v'' \\ z'' \end{bmatrix} = \lambda.f(A\mathbf{u} + t) = \lambda. \begin{bmatrix} (A\mathbf{u} + t) \\ g(r) \end{bmatrix} = P\mathbf{X} \tag{3.13}$$

where the function g is defined in Equation (3.10), $r = \sqrt{u''^2 + v''^2}$ and P is the projection matrix which relates the world coordinates to the sensor coordinates. The projection matrix can be decomposed into the camera extrinsic parameters including a rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$ and a translation vector $\mathbf{T} \in \mathbb{R}^{3\times1}$. It is important to note that the intrinsic parameters are the same for all the views of a single camera whereas the extrinsic parameters vary for each view captured by the fisheye camera as they describe the poses between the views. Calibration is performed to determine the intrinsic parameters of the camera as well as to determine poses between camera positions which are described by the extrinsic parameters.

Calibration in the Scaramuzza method requires a few images of a 2D checkerboard pattern of known dimensions to be imaged at different unknown positions by the fisheye camera as shown in Figure 3.3. These positions are related to each other by the rotation and translation parameters (R, T) which form the extrinsic parameters. The

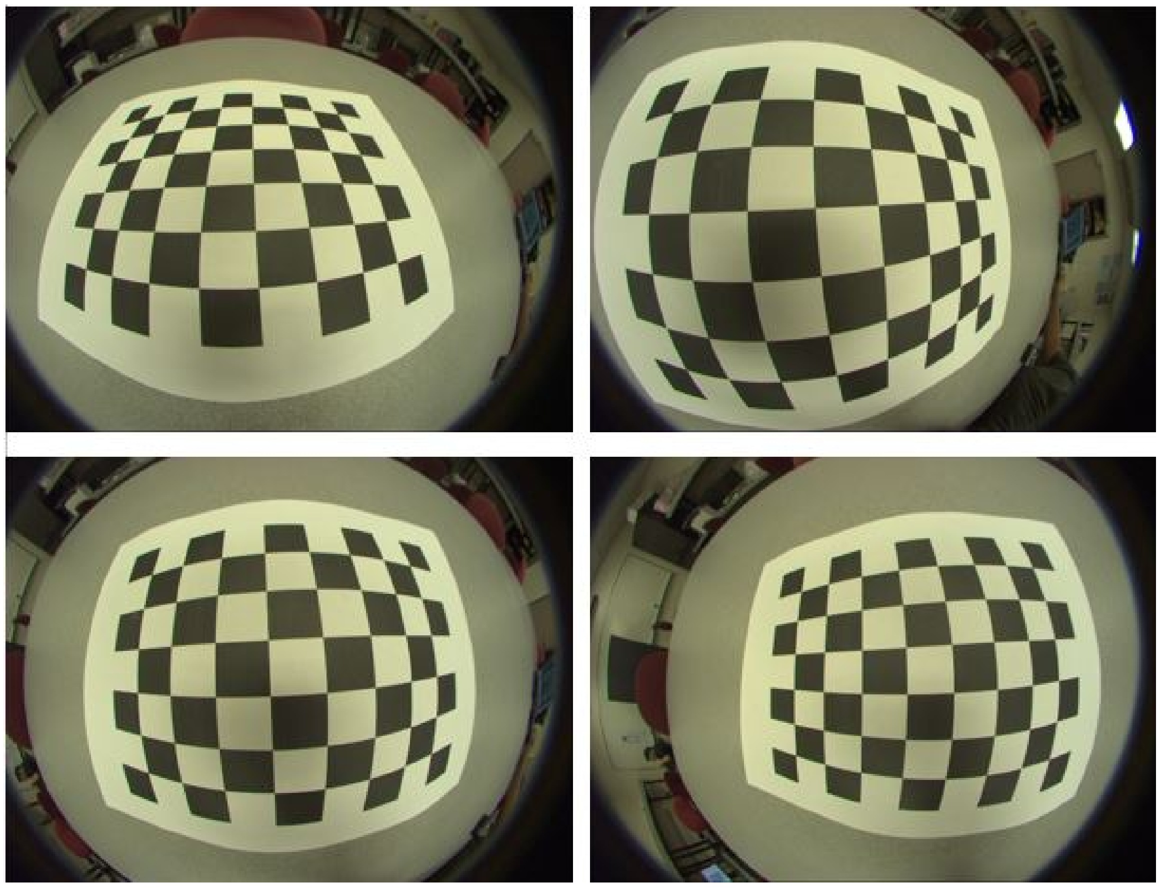


Figure 3.3: Checkerboard images required for calibration.

corners in the checkerboard are detected using the method of [30]. This latter method is based on the checkerboard detection algorithm implemented in OpenCV [31] with a few modifications to take into account highly blurred and distorted images. Once the checkerboard corners are detected for each of the camera views, correspondences between the checkerboard corners in space and in image coordinates is known. An example of the checkerboard pattern with extracted and numbered corners is shown in Figure 3.4. Consider Im^k to be the k^{th} captured image of the calibration pattern, i to be number of checkerboard corners detected on each of the patterns and let $M_{k,i}$=$[X_{k,i}$,$Y_{k,i}$,$Z_{k,i}]^T$ be the 3D coordinates of the checkerboard corners in the pattern coordinate system and $m_{k,i}$=$[u_{k,i}$, $v_{k,i}]^T$ be the image coordinates of the corresponding

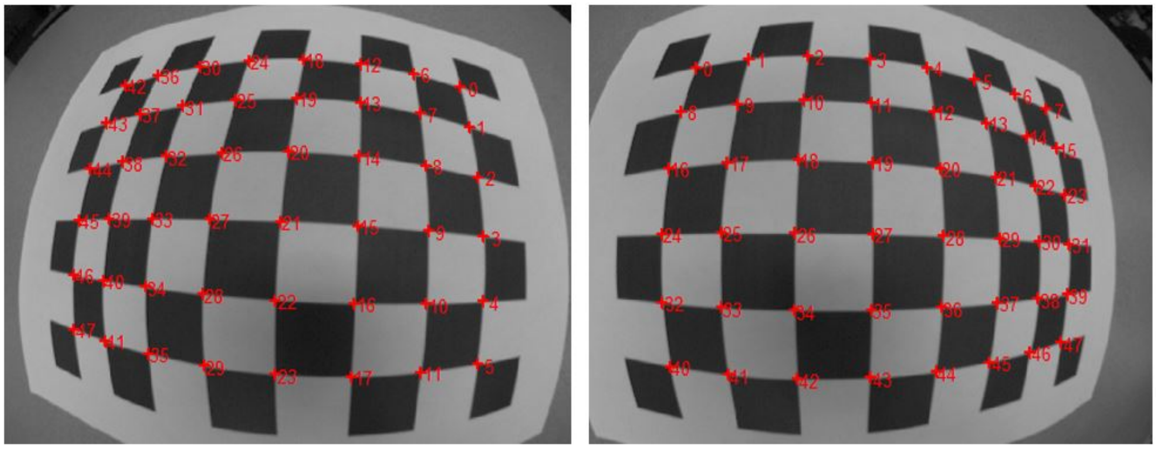


Figure 3.4: Checkerboard images with extracted corners.

checkerboard corners. These correspondences are used to obtain the intrinsic and extrinsic parameters for each view during the calibration process.

Calibration is done in 2 stages. In the first stage the extrinsic parameters are estimated as explained in Section 3.1.1 with the assumption that affine transformation matrix **A** which gives the relationship between the sensor plane and the image plane is **I**, i.e., sensor plane and the image plane are perfectly aligned. Using the estimated extrinsic parameters the intrinsic parameters **a** are estimated as described in Section 3.1.2. Next using these values of **a**, the affine transformation **A** is estimated in an iterative minimization process. Once the intrinsic and extrinsic parameters are obtained, the principal point is obtained on based the reprojection errors and finally a non-linear minimization step described in Section 3.1.3 is used to refine all the parameters[27].

3.1.1 Obtaining the external parameters

First the external parameters for each camera view are extracted. Since we consider a planar pattern, the Z coordinate in the pattern coordinate system will be 0 [1]. Hence, without loss of generality, we assume $Z_{k,i} = 0$. For each point i in each image k, Equation (3.13) becomes

$$\lambda_{k,i} \begin{bmatrix} u_{k,i} \\ v_{k,i} \\ g(r_{k,i}) \end{bmatrix} = \mathbf{P^k X} = [\mathbf{r_1^k} \quad \mathbf{r_2^k} \quad \mathbf{r_3^k} \quad \mathbf{t^k}]. \begin{bmatrix} X_{k,i} \\ Y_{k,i} \\ 0 \\ 1 \end{bmatrix} = [\mathbf{r_1^k} \quad \mathbf{r_2^k} \quad \mathbf{t^k}]. \begin{bmatrix} X_{k,i} \\ Y_{k,i} \\ 1 \end{bmatrix} \tag{3.14}$$

where the $\mathbf{r_j^k} = [r_{j1}^k \quad r_{j2}^k \quad r_{j3}^k]^T, \quad j = 1,2,3$ represents the first 3 columns and $\mathbf{t^k} = [t_1^k \quad t_2^k \quad t_3^k]^T$ represents the last column of the projection matrix $\mathbf{P^k}$ of camera view $\mathbf{Im^k}$. Premultiplying both sides of Equation (3.14) vectorially by $\mathbf{p_{k,i}} = [u_{k,i} \ v_{k,i} \ g(r_{k,i})]^T$ for the i^{th} point of the k^{th} camera view three homogeneous equations are obtained(ignoring the K index for simplicity)

$$v_i \ (r_{31}X_i + r_{32}Y_i + t_3) - g(r_i) \ (r_{21}X_i + r_{22}Y_i + t_2) = 0 \tag{3.15a}$$

$$g(r_i) \ (r_{11}X_i + r_{12}Y_i + t_1) - u_i \ (r_{31}X_i + r_{32}Y_i + t_3) = 0 \tag{3.15b}$$

$$u_i \ (r_{21}X_i + r_{22}Y_i + t_2) - v_i \ (r_{11}X_i + r_{12}Y_i + t_1) = 0 \tag{3.15c}$$

Since $\mathbf{X}$ and $\mathbf{u}$ are known, only Equation (3.15c) can be solved. The unknowns in Equation (3.15c) can be rearranged into a vector $\mathbf{L} = [r_{11}, r_{12}, r_{21}, r_{22}, t_1, t_2]^T$ such that Equation (3.15c) can be represented as[1]:

$$\mathbf{F}.\mathbf{L} = 0 \tag{3.16}$$

A linear estimate of L is obtained by minimizing $\|\mathbf{F}.\mathbf{L}\|^2$ using singular value decomposition (SVD). Hence, using all the checkerboards corners of all the poses a good estimate of $\mathbf{L}$ is obtained and the extrinsic parameters for each camera view are obtained.

3.1.2 Estimating the intrinsic parameters

The extrinsic parameters estimated in Section 3.1.1 are used to estimate the intrinsic parameters $\mathbf{a} = [a_0, a_2, ...a_N]$. The values $\mathbf{L} = [r_{11}, r_{12}, r_{21}, r_{22}, t_1, t_2]^T$ are used in Equations (3.15a) and (3.15b) to estimate $\mathbf{a}$ which describes the shape of the distortion function g [1]. The parameter t_3 was not present in $\mathbf{L}$ as Equation (3.15c) was independent of it. Hence, t_3^k is estimated for each of the views in this step. The parameters to be estimated in Equations (3.15a) and (3.15b) are stacked into a column vector and the equations are rewritten as a system of linear equations[1]; They can be represented as:

$$\begin{bmatrix} A_i^1 & A_i^1 \rho_i^{1^2} & \cdots & \cdots & A_i^1 \rho_i^{1^N} & -v_i^1 & 0 & \cdots & \cdots & 0 \\ C_i^1 & C_i^1 \rho_i^{1^2} & \cdots & \cdots & C_i^1 \rho_i^{1^N} & -u_i^1 & 0 & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ A_i^K & A_i^K \rho_i^{K^2} & \cdots & \cdots & A_i^K \rho_i^{K^N} & 0 & 0 & \cdots & \cdots & -v_i^K \\ C_i^K & C_i^K \rho_i^{K^2} & \cdots & \cdots & C_i^K \rho_i^{K^N} & 0 & 0 & \cdots & \cdots & -u_i^K \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_2 \\ \vdots \\ a_N \\ t_3^1 \\ t_3^2 \\ \vdots \\ t_3^K \end{bmatrix} = \begin{bmatrix} B_i^1 \\ D_i^1 \\ \vdots \\ B_i^K \\ D_i^K \end{bmatrix} \tag{3.17}$$

where

$$A_i^k = r_{21}^k X_i^k + r_{22}^k Y_i^k + t_2^k$$

$$B_i^k = v_i^k \left(r_{31}^k X_i^k + r_{32}^k Y_i^k\right)$$

$$C_i^k = r_{11}^k X_i^k + r_{12}^k Y_i^k + t_1^k$$

$$D_i^k = u_i^k \left(r_{31}^k X_i^k + r_{32}^k Y_i^k\right)$$

The linear least squares solution can be obtained by SVD. Further refinement is done on the estimated parameters in 2 steps. First, the estimated intrinsic parameters

a are used to refine the extrinsic parameters $r_{11}, r_{12}, r_{21}, r_{22}, t_1, t_2, t_3$. then these refined extrinsic parameters are used to refine the intrinsic parameters. Both these refinements are done by linear minimization using SVD.

The estimation of principal point or the center of distortion is done iteratively using the estimated intrinsic and extrinsic parameters and minimizing the sum of squared reprojection errors. This is the point of intersection of the image plane and the optical axis. The radial distortion is symmetric around this point. Ideally the principal point is the center of the image but this can vary due to misalignments of the fisheye lens and the camera sensor plane. The 3D checkerboard corners are reprojected back onto the image. The sum of squared distances between these reprojected points and the original points give the reprojection error. Hence, for each iteration, a set of points are chosen as the potential centers and reprojection error is calculated with of these potential point assumed to be the center. The candidate center point with the minimum reprojection error is selected as the starting point for the next iteration and a set of candidate center points around this point are checked for the reprojection error. This process is repeated until the reprojection error reaches a value less than a given threshold (say 0.5 pixel). This point is taken as the center of distortion or the principal point **O** of the fisheye image. Now that good estimates of all the parameters are determined a non-linear refinement as explained in Section 3.1.3 is applied on the whole system to obtain the affine parameter matrix **A** in Equation (3.11) which encodes the digitization process of the camera and any misalignments between the sensor plane and the image plane.

3.1.3 Non-Linear Refinement

The intrinsic and extrinsic parameters are estimated as described in Sections 3.1.2 and 3.1.1, respectively, through linear estimation. These parameters are refined along with the parameter $\mathbf{A}$ in Equation (3.11) which was assumed to be $\mathbf{I}$ in Section (3.1.1) using a non-linear minimization process. Suppose there are K camera views of the calibration pattern and each pattern has N checkerboard corners. The equation to be minimized is [27]:

$$E = \sum_{i=1}^{K} \sum_{j=1}^{N} \left\| u_j^i - \hat{u}(\mathbf{R^i}, \mathbf{T^i}, \mathbf{A}, \mathbf{O}, a_0, a_2, .., a_N, \mathbf{X^i_j}) \right\| \tag{3.18}$$

where $\hat{u}_j^i$ are the reprojected control points and u_j^i are the detected control points. This non-linear minimization is done using the Levenberg-Marquardt algorithm[32][33]. The initial guesses for the parameters to be refined by this algorithm is given by the linear estimates in Sections 3.1.1 and 3.1.2. The output of non-linear minimization of Equation (3.18) gives the final camera calibration parameters.

3.2 Kannala Marker based calibration

This section describes the camera model proposed in [2] and [34]. The same fisheye model in Figure 2.4 applies here as well. The main difference between this method and the method explained in Section 3.1 is the projection function.

In [34] the projection function is extended from Equations (2.5)-(2.8) to a generic polynomial model is used to incorporate the projections of different types of fisheye lenses. The model used here is:

$$r(\theta) = k_1\theta + k_2\theta^3 + k_3\theta^5 + k_4\theta^7 + k_5\theta^9 + ... \tag{3.19}$$

wheree r is the radial distance of the sensor image point m and θ is the angle of incidence of the incoming ray with the optical axis. It is seen in [2] that the first 5 coefficients are sufficient to give a good estimate of the different projection curves in Figure 2.3. In this thesis, we consider the camera model to consist of the five parameters k_1, k_2, k_3, k_4, k_5. From Figure 3.5 we can represent the image co-ordinate

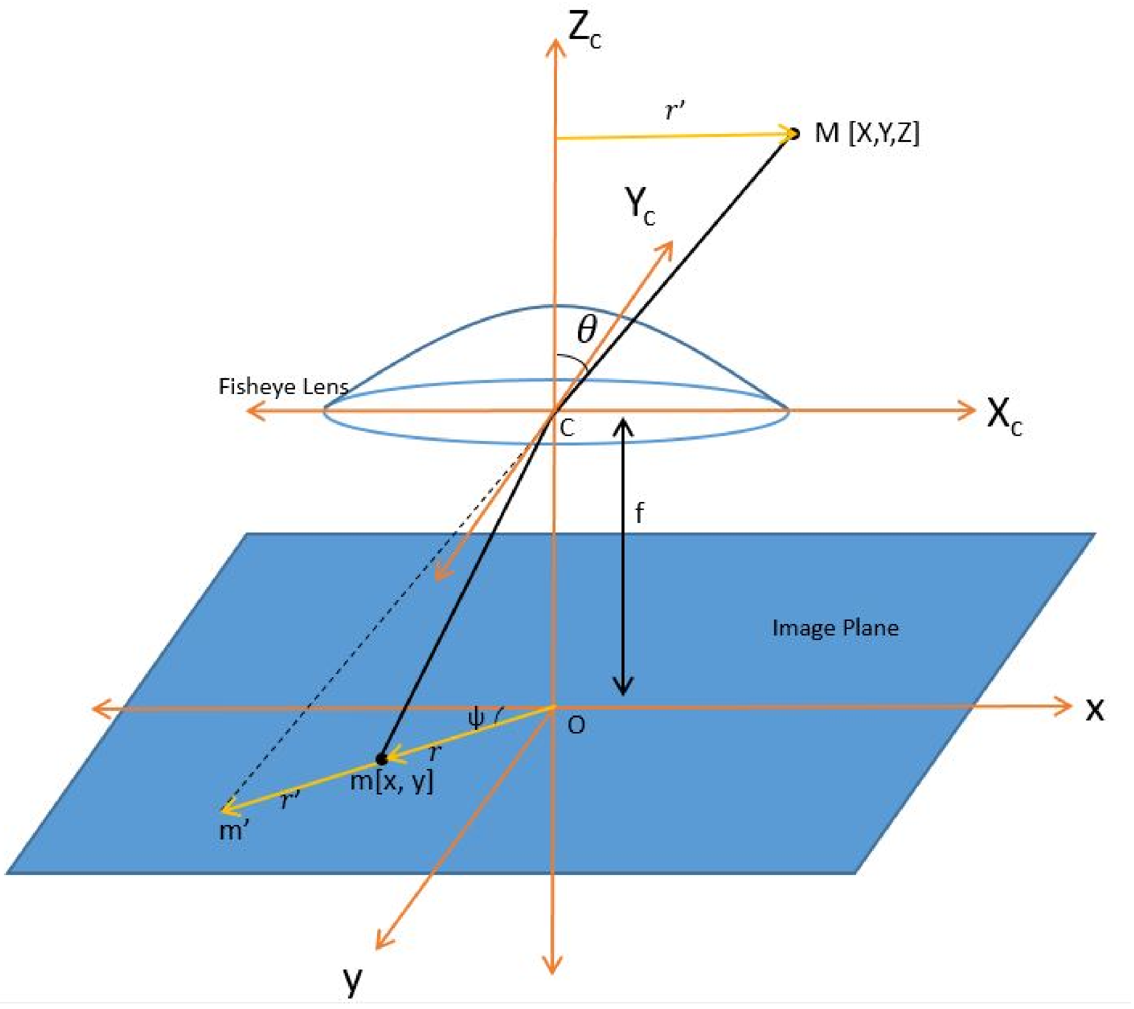


Figure 3.5: Fisheye camera model.

m of the as

$$\mathcal{F} = \begin{bmatrix} x \\ y \end{bmatrix} = r(\theta) \begin{bmatrix} \cos\psi \\ \sin\psi \end{bmatrix} \tag{3.20}$$

where θ and ψ represent the direction of the incoming ray from the 3D point **M**. θ is the angle made by the incoming ray with the optical axis and ψ is the angle made by the radial line with the x axis. A forward projection function $\mathcal{F}$ can be defined for mapping the 3D point **M** to the sensor image point **m**. The forward projection function $\mathcal{F}(\Phi)$ is defined as a function of the incoming ray direction $\Phi = (\theta, \psi)$[34].

The computation of the forward model is important as we can derive the inverse model from it to be used for reprojection and distortion correction.

An affine transform $\mathcal{A}$ is defined [2] to incorporate any misalignments between the sensor plane and the image plane. Given the image center as $[u_o, v_o]^T$ and the number of pixels per unit length in the horizontal and vertical directions as l_u and l_v, respectively, the image pixel coordinates are given by

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} l_u & 0 \\ 0 & l_v \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} u_o \\ v_o \end{bmatrix} = \mathcal{A}(\mathbf{x}) \tag{3.21}$$

. The total forward projection model is a combination of the projection function $\mathcal{F}$ and the affine transform $\mathcal{A}$. Combining Equations (3.20) and (3.21) we get the final forward model[2] as

$$\mathbf{m} = \mathcal{A}.\mathcal{F} = \mathcal{P}_{\mathbf{c}}(\mathbf{\Phi}) \tag{3.22}$$

where $\mathbf{m} = (u, v)^T$ are the image co-ordinates and $\mathcal{P}_c(\Phi)$ is the forward projection model. This model consists of 9 intrinsic parameters. k_1, k_2, k_3, k_4, k_5 from Equation (2.17) make up 5 of the parameters and l_u, l_v, u_0, v_0 from Equation (3.21) make up the other 4 parameters.

Similar to the technique in Section 3.1, this calibration technique uses a planar calibration pattern consisting of circular control points. The input to the calibration system is few camera views of the circular point calibration pattern captured at different unknown positions. A few example calibration patterns are shown in Figure 3.6

3.2.1 Determining principal point and control points

Control points refer to the points in the calibration fisheye image that correspond to known points in the planar 2D calibration pattern. These are necessary to obtain

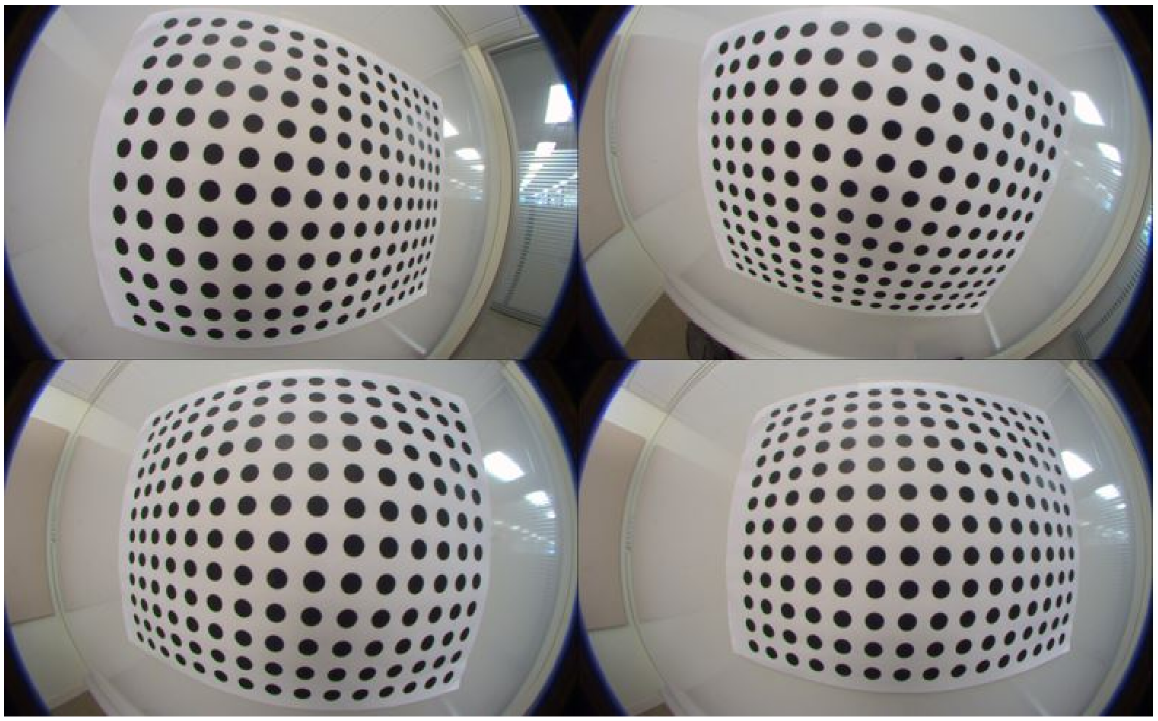


Figure 3.6: Example calibration patterns.

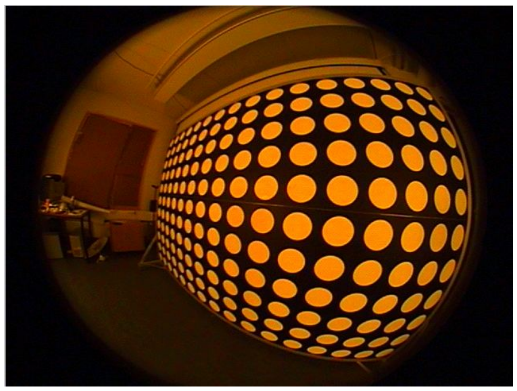
the relationship between a 3D point in space and its projection on the fisheye image. Here, we also include determination of the image boundary and the center of the image or the principal point. These are essential for later stages of calibration as we will see in Section 3.2.2. The principal point of the image is also essential for back-projection during error minimization in Section 3.2.3 and distortion correction in Chapter 4.

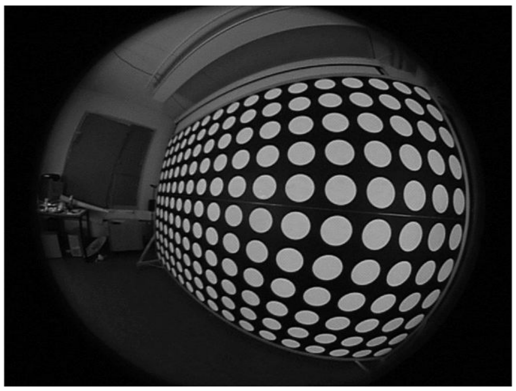
The first step is to determine the ellipse that bounds the image. In the case of where the fisheye image fills an elliptical area on a black background, an ellipse fitting is used to obtain the parameters of the ellipse. The calibration image similar to the one shown in Figure 3.7a is first converted to grayscale (Figure 3.7b). It is then histogram equalized and converted to a binary image using a appropriate threshold. Then a connected component labeling algorithm is applied to the binary image in order to label all the binary 8-connected objects as shown in Figure 3.7c. The component with the maximum number of elements is calculated. This is the part of the image which makes up the black external boundary. A new binary image is created with only this component set to 1 and the rest of the image is set to 0. This is shown in Figure 3.7d where only the boundary of the image is segmented. This boundary is further refined using a finer-connected-components method and we obtain a binary image similar to Figure 3.7e which gives a more well defined boundary. The perimeter to this bounding image is obtained as shown in Figure 3.7f. An ellipse is now fit to this perimeter image to obtain the parameters of the ellipse. The equation of the ellipse to be fit is

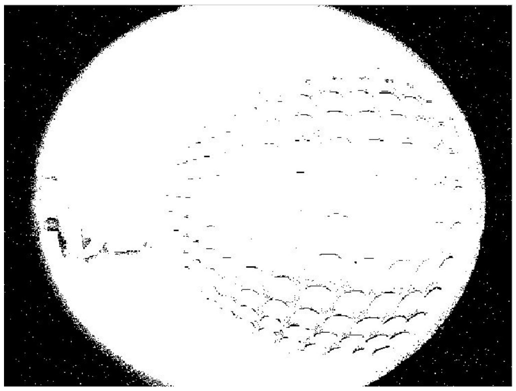
$$\left(\frac{u-u_0}{a}\right)^2 + \left(\frac{v-v_0}{b}\right)^2 = 1 \tag{3.23}$$

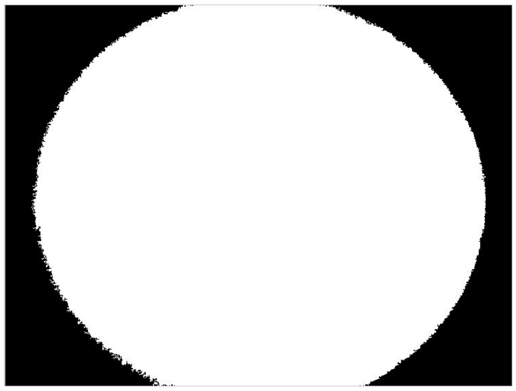
where u_0, v_0, a, b are the obtained parameters from the fitting. (u_0, v_0) represents the center of the ellipse and a, b represent the length of the minor and major axes of the ellipse. The ellipse is fit by a linear minimization using singular value decomposition. Hence, (u_0, v_0) is the center of distortion in the fisheye image which is used in the proceeding stages of calibration. Ideally, for a fisheye image $a = b$. However, these are unequal in the case the sensor plane and image plane are not perfectly perpendicular to the optical axis of the fisheye camera.

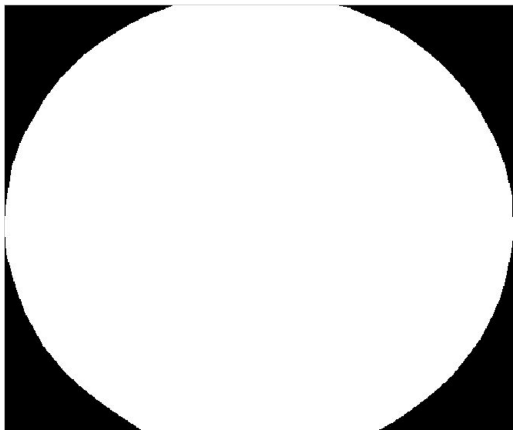
Once the image boundary and the principal point of the camera are determined, the control points are estimated by using a similar connected component approach. A few images are captured as in Figure 3.6. Each of these images are converted first to a grayscale image. Then an optimal value for the threshold for the grayscale image is obtained by fitting two normal distributions to the histogram[13]. This is done first by finding the threshold value which separates the image histogram into 2 distributions. One distribution for the blobs representing the circular points and


(a) Example of calibration image.


(b) Grayscale image.


(c) Binary connected components image.


(d) Segmented bounding box.


(e) Image bounding binary image.

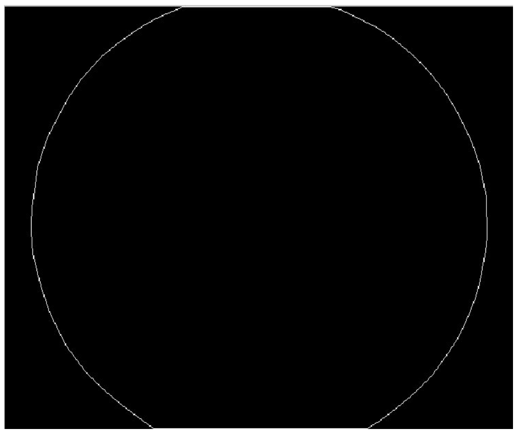
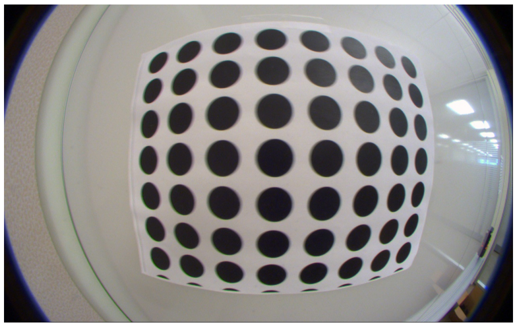
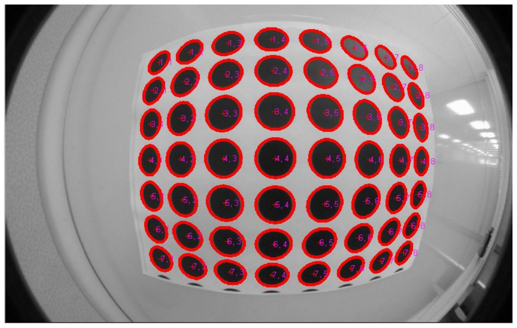
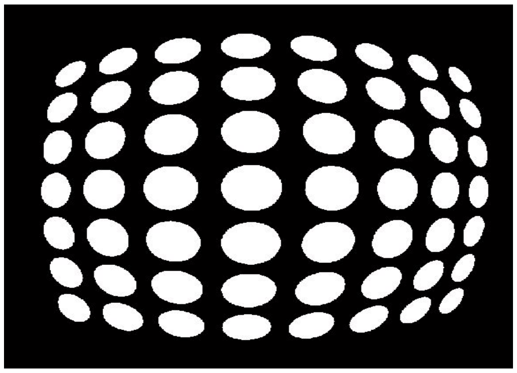

(f) Perimeter of image to which ellipse is fit.

Figure 3.7: Ellipse fitting to image boundary.


(a) Example of calibration image.


(b) Detected blobs and their centroids.


(c) Binary connected components image.

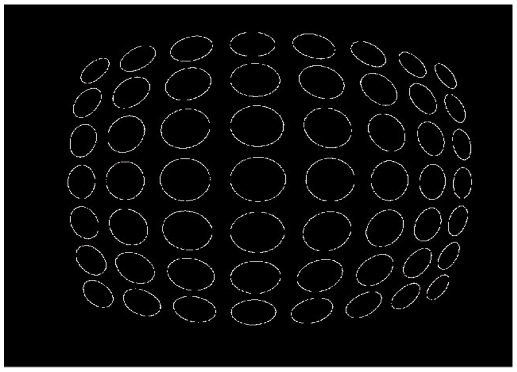

(d) Perimeters of the segmented blobs.

Figure 3.8: Control point extraction from calibration image.

the other for the background pixels. The probabilities, mean and standard deviation for both the distributions are estimated and a Gaussian mixture model is fit to these distributions. From this, an ideal threshold for the grayscale image is obtained which is used to convert the image to a binary image like the one shown in Figure 3.8c. The binary image now consists of only the circular control points.

A connected component algorithm with 4 connectivity is run on this binary image to obtain the labels and centroids for each of the control points as shown in Figure 3.8b. The centroids are ordered based on axis directions entered by the user. These act as the control points for the rest of the calibration procedure, and are shown with the + symbols in the circular points of Figure 3.8b.

Now that control points are determined from the calibration images, we can proceed to the next stage of calibration as discussed in Section 3.2.2.

3.2.2 Initializing intrinsic parameters

The intrinsic parameters consist of $k_1, k_2, k_3, k_4, k_5, u_0, v_0, l_u, l_v$ where $k_1, .., k_5$ are the coefficients of the odd powered polynomial equation represented by Equation (3.19). (u_0, v_0) represents the image center or the principal point of the fisheye camera. They are determined using ellipse fitting of the image boundary as discussed in Section 3.2.1. Initially, only k_1, k_2 are estimated by fitting the model $r = k_1\theta + k_2\theta^3$ to one of the models in Equations (2.5)-(2.8) with the manufacturer's value if nominal focal length f and the maximum angle of view θ_{max} if available. $r_{max} = k_1\theta_{max} + k_2\theta^3_{max}$ where r_{max} is the maximum radius of the fisheye image[2]. The parameters $l_u = a/r_{max}$ and $l_v = b/r_{max}$ where a and b are determined as described in Section 3.2.1. Only 2 of the 5 intrinsic parameters k_i are initialized here. The other parameters are included only at the last stage where non-linear minimization is used to minimize the projection errors in Section 3.2.4.

3.2.3 Extrinsic parameters

Assume that M control points are observed in N views. For each view there is a rotation matrix $\mathbf{R_j}$ and translation $\mathbf{t_j}$ describing the camera position with respect to the calibration pattern. Hence,

$$\mathbf{X_c} = \mathbf{R_j}\mathbf{X} + \mathbf{t_j}, \quad j = 1, ..., N. \tag{3.24}$$

where $\mathbf{X_c}$ is the position of the camera and $\mathbf{X}$ is the position of the calibration pattern. The calibration pattern is assumed to be planar and to lie on the XY-plane. The i^{th} control point will have the coordinates $\mathbf{X^i} = (X^i, Y^i, 0)^T$. The corresponding homo-

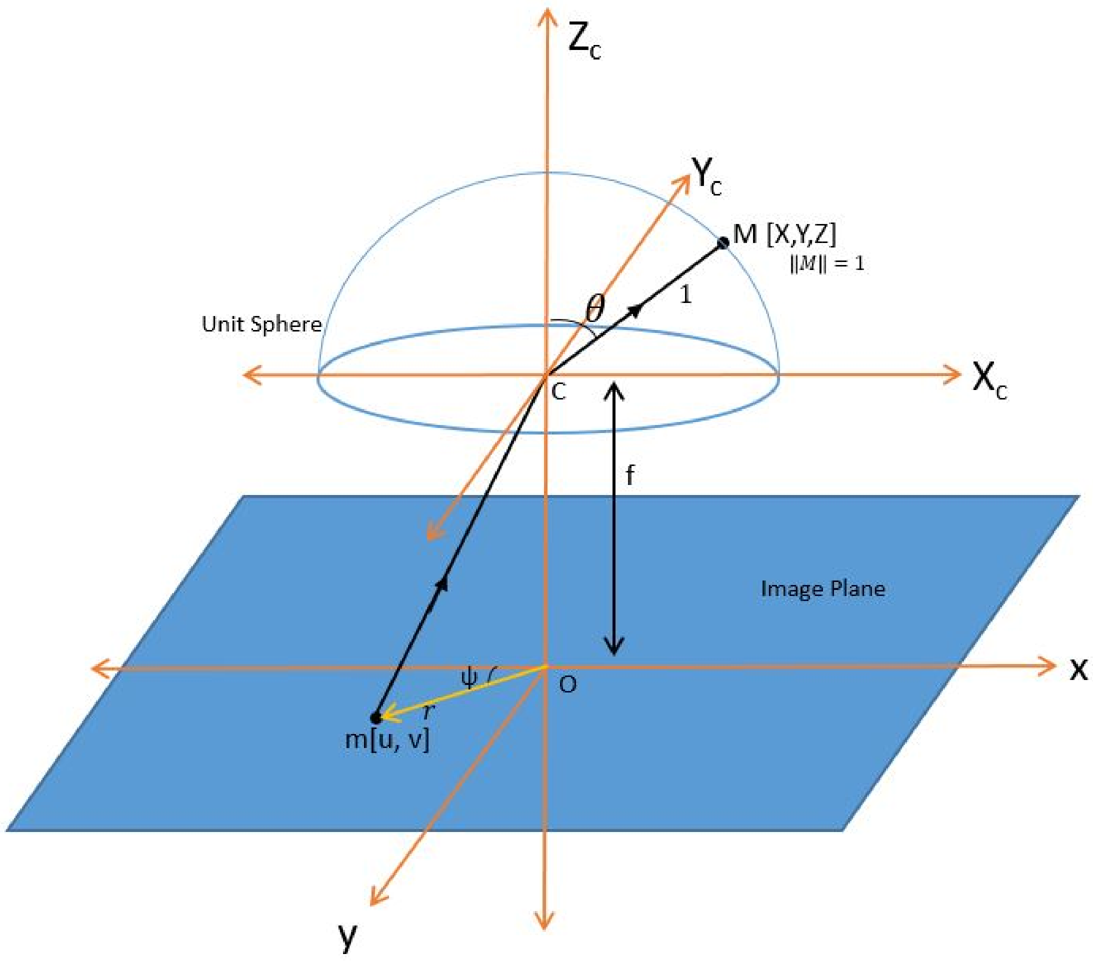


Figure 3.9: Reprojection onto unit sphere.

geneous coordinates in the calibration pattern[34] are denoted by $\mathbf{x_p^i} = (X^i, Y^i, 1)^T$ and the corresponding coordinates in the j^{th} fisheye image is $\mathbf{m_j^i} = (u_j^i, v_j^i)^T$ as shown in Figure 3.5.

Here, for each view j, each of the i image points $\mathbf{m_j^i} = (u_j^i, v_j^i)^T$ is reprojected onto the unit sphere as shown in Figure 3.9. This can be done using a backward model which can be defined as $\mathcal{P}_c^{-1} = \mathcal{A}^{-1}.\mathcal{F}^{-1}$ where $\mathcal{A}$ and $\mathcal{F}$ are defined in Equations (3.21)and (3.20). $\mathcal{A}^{-1}$ can be determined as

$$\begin{bmatrix} x_j^i \\ y_j^i \end{bmatrix} = \begin{bmatrix} 1/l_u & 0 \\ 0 & 1/l_v \end{bmatrix} \begin{bmatrix} u_j^i - u_0 \\ v_j^i - v_0 \end{bmatrix} \tag{3.25}$$

From Equation (3.25) we can calculate the radius as

$$r_j^i = \sqrt{(x_j^i)^2 + (y_j^i)^2}$$
$$\psi_j^i = \tan^{-1}\left(\frac{y_j^i}{x_j^i}\right)$$

To obtain θ_j^i, the cubic Equation (3.26) is solved:

$$k_2(\theta_j^i)^3 + k_1(\theta_j^i) - r_j^i = 0 \tag{3.26}$$

The unit sphere coordinates is then given by $\tilde{x}_j^i = (\sin\psi_j^i \sin\theta_j^i,\ \cos\psi_j^i \sin\theta_j^i,\ \cos\theta_j^i)^T$. The relationship between the points on the calibration pattern and on the unit sphere is a central projection and, hence, there is a planar homography $\mathbf{H_j}$[2] for each view between the camera image and the calibration plane where $s\mathbf{\tilde{x}_j^i} = \mathbf{H_j x_p^i}$. Initial estimate of $\mathbf{H_j}$ is obtained by correspondences between the planar calibration pattern coordinates($\mathbf{x_p^i}$) and the coordinates of the reprojected points on unit sphere ($\mathbf{\tilde{x}_j^i}$) using the discrete linear transform[35]. Normalized coordinates of the calibration pattern with respect to the camera coordinate system is $\mathbf{\hat{x}_j^i} = \mathbf{H_j x_p^i} / \|\mathbf{H_j x_p^i}\|$, where $\mathbf{H_j}$ is the estimated homography as discussed. The homography for the j^{th} view is further refined by minimizing $\sum_{i=1}^{M} \sin^2 \alpha_j^i$, where α_j^i is the angle between the unit vectors $\mathbf{\tilde{x}_j^i}$ and $\mathbf{\hat{x}_j^i}$.

The external parameters for the j^{th} view are $\mathbf{R_j}$ and $\mathbf{t_j}$ and are related to its homography $\mathbf{H_j}$ as follows:

$$s\mathbf{\hat{x}_j^i} = \mathbf{H_j x_p^i} = [\mathbf{R_j} \quad \mathbf{t_j}] \begin{bmatrix} X^i \\ Y^i \\ Z^i \\ 1 \end{bmatrix} = [\mathbf{r_j^1}\ \mathbf{r_j^2}\ \mathbf{r_j^3}\ \mathbf{t_j}] \begin{bmatrix} X^i \\ Y^i \\ 0 \\ 1 \end{bmatrix} = [\mathbf{r_j^1}\ \mathbf{r_j^2}\ \mathbf{t_j}] \begin{bmatrix} X^i \\ Y^i \\ 1 \end{bmatrix} \tag{3.27}$$

where $[X^i, Y^i, Z^i]$ are the coordinates of the i^{th} control points on the calibration pattern in space in the pattern coordinate system. Since a planar pattern is used, the Z^i component is 0. Hence, from Equation (3.27) the homography can be represented by $\mathbf{H_j} = [\mathbf{r_j^1}\ \mathbf{r_j^2}\ \mathbf{t_j}]$. These form the extrinsic parameters for each camera view. Now that both intrinsic and extrinsic parameters are estimated and initialized, a non-linear minimization step can be used to refine the parameters as discussed in Section 3.2.4

3.2.4 Minimization of the projection error

In Section 3.2.2, only the first 2 k_1, k_2 parameters were estimated. Here, the rest of the parameters $k_3, ..., k_5$ are included into the model in Equation (3.26) with them initially set to 0. Hence, Equation (3.26) becomes

$$k_5(\theta_j^i)^9 + k_4(\theta_j^i)^7 + k_3(\theta_j^i)^5 + k_2(\theta_j^i)^3 + k_1(\theta_j^i) - r_j^i = 0 \tag{3.28}$$

Both the intrinsic and extrinsic parameters are refined by a non-linear minimization of the reprojection error. Combining the estimated parameters and using them in Equation (3.22) we can get a projection function $\mathcal{P}_j$ for each camera view. Using this estimated projection function, the control points from the calibration pattern are projected onto the image. These reprojected points can be represented as

$$\mathbf{\hat{m}_j^i} = \mathcal{P}_\mathbf{j}(\mathbf{X^i}) \tag{3.29}$$

where $\mathbf{\hat{m}_j^i}$ represents the reprojection of the i^{th} control point in the j^{th} view and X^i is the coordinate of the i^{th} control point in the pattern coordinate system.

From Section 3.2.1 we already obtained the measured projection coordinates for each of the control points in each of the views and denoted this by $\mathbf{m_j^i}$. If the estimated camera projection function $\mathcal{P}_j$ correctly models the actual projection function of the camera, the coordinates of the estimated control points $\mathbf{\hat{m}_j^i}$ and the measured control points $\mathbf{m_j^i}$ will perfectly coincide for each of the j views. In practice however, since approximated values have been used for the intrinsic and extrinsic parameters these points will not coincide and the distance between the measured and estimated control point is called the reprojection error. A non-linear minimization step can be applied to this stage to minimize the sum of the reprojection error over all the control points and all the views. This uses the estimated intrinsic and extrinsic parameters as a starting point and refines them till a global minimum of the cost function (which is the reporjection error in this case) is obtained. This non-linear minimization can be represented as

$$\sum_{j=1}^{N}\sum_{i=1}^{M} d(\mathbf{m_j^i}, \mathbf{\hat{m}_j^i})^2 \tag{3.30}$$

This non-linear minimization step is carried out by using the Levenberg-Marquardt algorithm[32][33]. The refined parameters after the minimization step can be used for distortion correction and other applications which require the camera calibration parameters.

3.3 Kannala Auto-Calibration

Sections 3.1 and 3.2 described marker-based methods where a calibration pattern was used to help calibrate the fisheye camera. This section discusses the theory and implementation of an auto-calibration technique developed in [3]. This method is based on using point correspondences between 2 images captured by a fisheye camera and estimating the camera parameters by minimizing the angular error[36]. A generic model for the central catadioptric camera developed in [5] is extended to be used for the fisheye cameras. The model is shown in Figure 3.10. The **Z** axis is the optical axis and **Z=1** is a virtual image plane.

The point **X** in space is mapped to **x** on the virtual image plane. [5] defines a generic model where a combination of functions maps the 3D point **X** in space to an

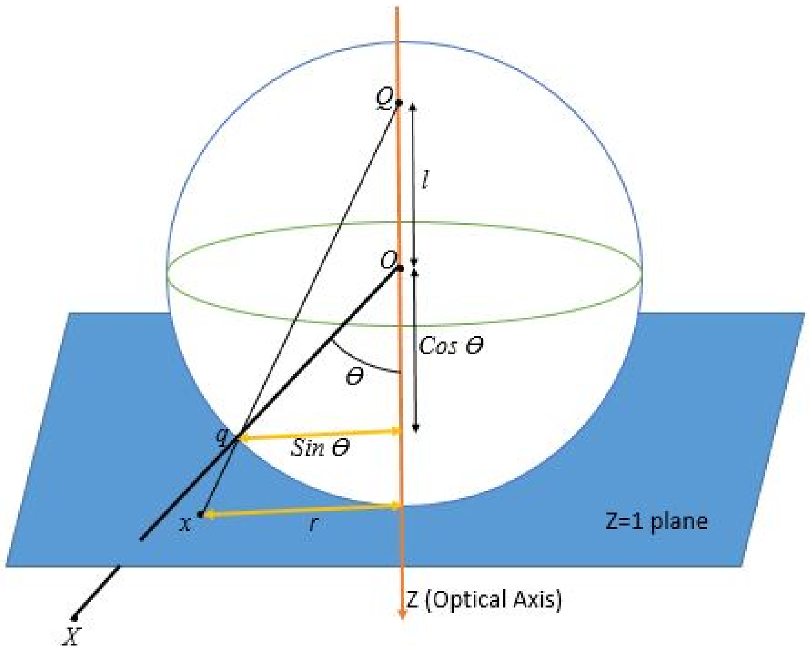


Figure 3.10: General model for central camera [5].

observed image point **m** as shown in Equations (3.31) to (3.33)

$$\mathbf{q} = \mathcal{G}(\mathbf{X}) \tag{3.31}$$

$$\mathbf{x} = \mathcal{H}(\mathbf{q}) \tag{3.32}$$

$$\mathbf{m} = \mathcal{A}(\mathbf{x}) \tag{3.33}$$

where $\mathbf{X} = (X, Y, Z)^T$ is a 3D point, $\mathbf{q}$ is the projection of the 3D point $\mathbf{X}$ onto the unit sphere, $\mathbf{x} = (x, y, 1)^T$ is a projection of point $\mathbf{q}$ on the virtual image plane when viewed from a position $\mathbf{Q}$ on the optical axis as shown in Figure 3.10, and finally $\mathbf{m} = (u, v, 1)^T$ is an image point related to the point $\mathbf{x}$ on the virtual image by an affine transformation $\mathcal{A}$. An interesting point about the design of this model is that it is assumed that the point $\mathbf{q}$ is perspectively projected onto the virtual image plane from the point $\mathbf{Q}$ which lies on the optical axis. The distance $l = \|OQ\|$ is the parameter of the camera which defines the amount of distortion.

The model defined above and proposed in [5] has the form,

$$r = \frac{(l+1)\sin\theta}{l+\cos\theta} \tag{3.34}$$

This can be seen in Figure 3.10 where the corresponding sides of similar triangles have the same ratio, i.e., $\dfrac{r}{\sin\theta} = \dfrac{l+1}{l+\cos\theta}$. It is seen in [3] that the model in Equation (3.34) can be fit to all the models in Equations (2.5)-(2.8). The function $\mathcal{G}$ is used to project the point $\mathbf{X}$ onto the unit sphere. This basically means obtaining the unit direction vector in the direction of $\mathbf{X}$ from the camera center $\mathbf{O}$ (Figure 3.10). This can be represented by $\mathcal{G}(\mathbf{X}) = \mathbf{X}/\|\mathbf{X}\| = (\cos\psi\sin\theta,\ \sin\psi\sin\theta, \cos\theta)^T$, where ψ and θ are the polar angle coordinates of $\mathbf{X}$ as shown in Figure 3.11. The function $\mathcal{H}$ which maps the point $\mathbf{q}$ on the unit sphere to the point $\mathbf{x}$ on the virtual image plane $\mathbf{Z}$=1 can be represented by $\mathbf{x} = \mathcal{H}(\mathbf{q}) = (r\cos\psi, r\sin\psi, 1)^T$ where r is the radial projection function in Equation (3.34) which is a function of θ. The affine transformation $\mathcal{A}(.)$ can be defined as $\mathbf{m} = \mathcal{A}(x) = \mathbf{K}\mathbf{x}$ where $\mathbf{K}$ is the affine transformation matrix given by

$$\mathbf{K} = \begin{bmatrix} f & sf & u_0 \\ 0 & \gamma f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.35}$$

which contains the conventional parameters of the pinhole camera as in [6]. f is the effective focal length, (u_0, v_0) describe the optical center, s, γ are the skew and ratio of size of pixels. In our implementation, we assume $s = 0$ and $\gamma = 1$ as modern day image sensors have negligible skew. The initial value of f is set to 500 and (u_0, v_0) are initially set to the image center. These are refined to their optimal values in during non-linear minimization stage. Hence, the forward camera model $\mathcal{P}$ described by Equations (3.31)-(3.33) and which describes the mapping of a 3D point to an image point, can be represented as $\mathbf{m} = \mathcal{P}(\mathbf{\Phi})$ where the directions of the incoming ray is represented by $\Phi = (\theta, \psi)$ as in Figure 3.11. The backward model can be computed

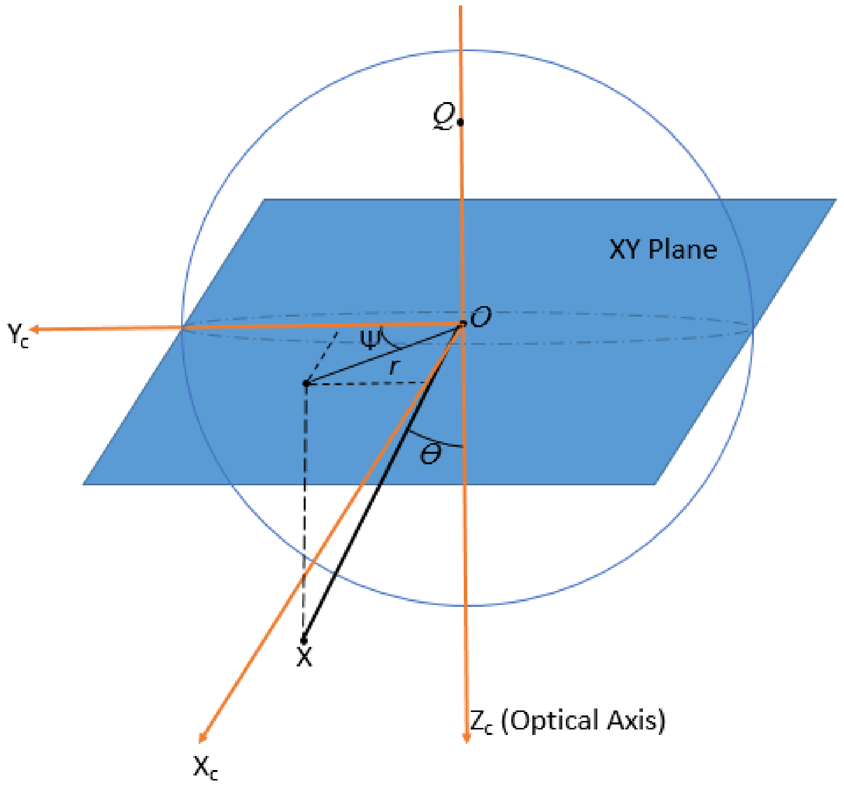


Figure 3.11: Polar coordinates of 3D point X.

easily by inverting the forward model and can be viewed as $\mathbf{\Phi} = \mathcal{P}^{-1}(\mathbf{m})$. It is computed in two steps by first inverting $\mathcal{A}$ in Equation (3.33) and then inversion of r in Equation (3.34). The affine transformation $\mathcal{A}$ can be inverted just by finding the inverse of the affine transformation matrix $\mathbf{K}$. The inverse of r can be obtained by taking squares of both sides of Equation (3.34) which gives

$$l^2 r^2 + 2lr^2 \cos\theta + r^2 \cos^2\theta = (l+1)^2 \sin^2\theta \tag{3.36}$$

Solving the quadriatic equation and solving for $\cos\theta$ we obtain the angle θ as:

$$\theta = \cos^{-1}\left(\frac{-lr^2 + \sqrt{l^2r^4 - (r^2+(l+1)^2)(l^2r^2-(l+1)^2)}}{(r^2+(l+1)^2)}\right) \tag{3.37}$$

From these values of θ and $\mathbf{K^{-1}}$, the unit vectors in the direction of the 3D point $\mathbf{X}$ can be calculated given its corresponding image point $\mathbf{m}$. This is essential for the non-linear optimization stage discussed further in Section 3.3.3. The correspondences of image points **m,m'** in two images are obtained using a scale invariant feature detector[37] as explained in Section 3.3.1.

3.3.1 Obtaining Point Correspondences

This section describes how image point correspondences are obtained when we have two images of the same scene taken at different poses. A scale invariant feature transform developed in [37] is used to obtain key features in the image along with their descriptors. These descriptors are later used for matching the feature points between images to get the correspondences. This method with a shortened acronym SIFT is one of the most popular feature detector/descriptor used for feature detection and matching of images as it is invariant to scale, rotation and illumination. SIFT mainly has five steps, scale-space extrema detection, keypoint localization, orientation assignment, keypoint descriptor and keypoint matching.

Detection of scale space extrema The first stage of keypoint detection is to identify locations and scales that can be repeatably assigned under different viewing conditions of the same object[37]. This is done by searching for features that are invariant in all scales using a continuous function of scale known as scale space. The Gaussian function represents such a scale space as shown in [38]. If $I(x, y)$ is the image, then scale space can be defined as a function $L(x, y, \sigma)$

$$L(x, y, \sigma) = G(x, y, \sigma) \star I(x, y) \tag{3.38}$$

where $\star$ is convolution operation in x, y and G is a Gaussian represented by

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp^{-\dfrac{(x^2 + y^2)}{2\sigma^2}}$$

To detect stable keypoint locations, a difference-of-Gaussian(DoG) is convolved with the image as shown in Equation (3.38)[37]. The DoG can be represented as

$$D(x, y.\sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) \star I(x, y) \tag{3.39}$$

For easy computation of this DoG function, the initial image is convolved with incremental values of σ to produce images separated by a constant factor k in scale space. The adjacent Gaussian images are subtracted for to obtain the DoG result. An octave of the scale space is where the final value of the σ is double the initial value. This way the scale space s is divided into different octaves. Each octave is further divided into s intervals, where s is an integer. Hence, the constant factor k can be defined in terms of s as $k = 2^{1/s}$. In each octave the scale σ goes from a value σ_0 to $2\sigma_0$, where σ_0 is the beginning scale for each octave. Once a complete octave has been processed, the Gaussian image is downsampled by a factor of two and the above process is repeated. This has the same effect of using the σ values from $2\sigma_0$ to $4\sigma_0$ on the 2^{nd} octave but is much faster because of fewer computations [37]. Once the DoG images are obtained, the local extrema detection is performed where each pixel is searched across a 3×3 window in three adjacent scale spaces. Hence, each pixel in the DoG image is compared with eight pixels in its own scale, apart from nine pixels in the next scale and nine pixels in the previous scale making a grand total of twenty-six pixels being checked as shown in Figure 3.13. The pixel is selected only if it is the maximum or minimum value of all these neighbors. This maximum or minimum pixel is called a keypoint. Each octave must have at least $s+3$ Gaussian blurred images so that the extremal point detection covers the complete octave. An example of DoG pyramid for different octaves is shown in Figure 3.12.

Keypoint Localization As the value of σ increases, especially when going from a lower octave to a higher octave, the individual discrete points of the DoG represent

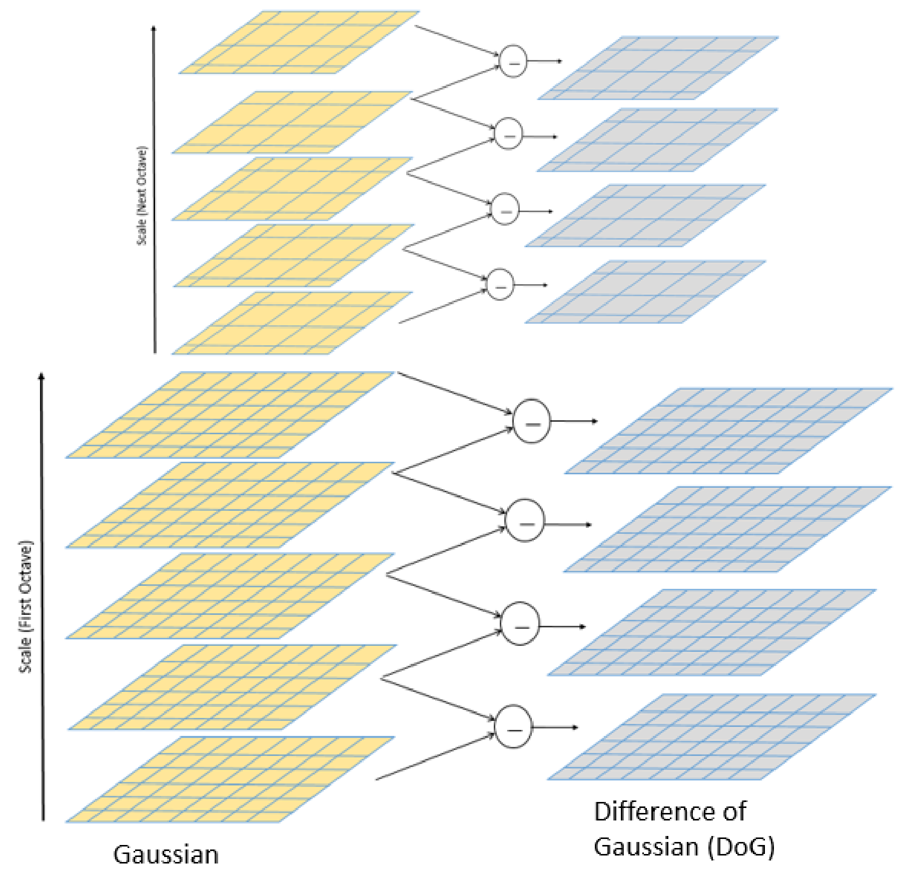


Figure 3.12: Computing Difference-of-Gaussians for different Octaves.

increasingly coarse samples of the original image. An extremum in the higher octaves may not point directly to a pixel that should represent the keypoint. Greater precision is achieved in the localization of the extremum with respect to the original image by estimating a second order derivative of $D(x, y, \sigma)$ at the sampling points in the DoG pyramid. This allows the localization of the extrema with sub-pixel accuracy. A Taylor series expansion of $D(x, y, \sigma)$ in the vicinity of $\vec{X_0} = (x_0, y_0, \sigma_0)$, which represents an extremum found in the extrema detection stage can be represented as

$$D(\vec{X}) \backsimeq D(\vec{X_0}) + J^T(\vec{X_0})\vec{X} + \frac{1}{2}\vec{X}^T H(\vec{X_0})\vec{X} \tag{3.40}$$

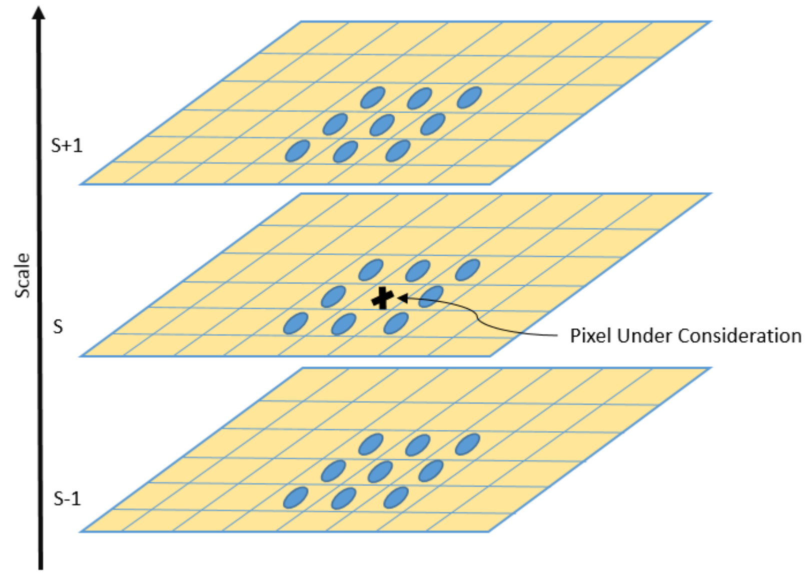


Figure 3.13: Detecting extremal points across scales.

where J is the gradient vector estimated as $\vec{X_0} = \left(\frac{\partial D}{\partial x}, \frac{\partial D}{\partial y}, \frac{\partial D}{\partial \sigma}\right)$, and H is the hessian at $\vec{X_0}$. At the true extremum, the derivative in Equation (3.40) will be zero. Taking the derivative on both sides of Equation (3.40) with respect to variable $\vec{X}$, the true extremum is given by $J\vec{X} = -H^{-1}(\vec{X_0}).J(X_0)$. On the edges, there are some unstable keypoints which have large curvatures across the edge but small curvatures in the perpendicular direction. The parameters of a 2×2 Hessian matrix are used to calculate the ratio of the curvature across the edge and the curvature in the perpendicular direction. If the ratio is larger than a threshold, the keypoint is discarded.

Orientation Assignment Each detected extremum is assigned a dominant local orientation. This gives a notion of description with the retained extremum. To do this, the gradient vector of the Gaussian smoothened image L at the scale σ closest to the scale of the detected extremum is computed. At each point in a $K \times K$ neighborhood around the extremum, the gradient magnitude $m(x, y)$ and the gradient orientation $\theta(x, y)$ are calculated as

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \qquad (3.41)$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1)))/(L(x+1,y) - L(x-1,y)) \qquad (3.42)$$

$\theta(x, y)$ is weighted with $m(x, y)$[37], a histogram is constructed for $\theta(x, y)$ using 36 bins spanning the full 360 degree range. The bin in which the histogram peak occurs, gives the dominant local orientation.

Keypoint descriptor Here a descriptor is assigned to the retained extremum. The same gradient information as was calculated in the previous step is used, except for the fact that the gradient directions are measured relative to the dominant local orientation. For an extremum, the 16×16 surrounding area of the keypoint is divided into 4×4 sub-regions and is used to calculate the descriptor. After Gaussian smoothing, the gradient magnitudes and orientations of the samples in the 4×4 sub-regions are calculated and an 8 bin histogram for each 4×4 sub-region is generated. Stringing together the sixteen 8 bin histograms yields a 128 element descriptor at each retained extremum. Considering the 128 element descriptor as a vector in a 128 dimensional space, its length is normalized to unity to make it invariant to changes in illumination.

Once the keypoints have been detected in both images, a modified a K-D tree algorithm called Best-Bin-First method [39] is used for matching keypoints by finding the descriptor of the keypoint in one view with minimum Euclidean distance from the descriptor in the other view. We used the MATLAB implementation by [40] which is available online.

3.3.2 Epipolar geometry and angular error

This auto-calibration method is based on minimizing angular two-image reprojection error over the camera internal parameters[3]. In this section, details of the epipolar constraints used, their meaning and the contribution of angular error are discussed. Two central cameras with camera centers O and O' observing a point P as shown in Figure 3.14 are assumed. $\mathbf{OP}, \mathbf{O'P}$ represent the vectors of this point in the first and second camera, respectively. **q** and **q'** are the unit directional vectors in the directions of **OP** and **O'P**. Then the epipolar constraint yields

$$\mathbf{q'^T E q} = 0 \tag{3.43}$$

Here **E** is the essential matrix [35]. This essential matrix encodes the relative position of the cameras. We can represent the essential matrix **E** as $\mathbf{E} = [\mathbf{t}]_{\mathbf{x}}\mathbf{R}$, where $[t]_x$ represents a skew symmetric version of the translation matrix **t** and **R** is the rotation matrix. Hence, $\mathbf{R}, \mathbf{t}$ gives the position of camera 2 with respect to camera 1, which defines the extrinsic parameters of camera 2. It is assumed that center **O** of camera 1 is placed at the origin of the world coordinate system. Since **q** and **q'** are unit vectors they can be represented on a sphere and hence can be related to the image coordinates **m** as described in Equations (3.31)-(3.33). It is important to note that since the fisheye cameras are represented with their retinas as unit spheres, they have two epipoles as shown in Figure 3.14 compared to planar cameras which have just one epipole for each camera. This is further discussed in Chapter 6 where we look at structure from motion applications using fisheye cameras.

In general the values of **q** and **q'** are obtained by back projecting noisy image correspondences obtained by SIFT (Section 3.3.1) and by using the inverse projection model as described in Equation (3.37). These do not satisfy Equation (3.43) exactly which leads to corresponding rays not intersecting at a point **P**. The internal

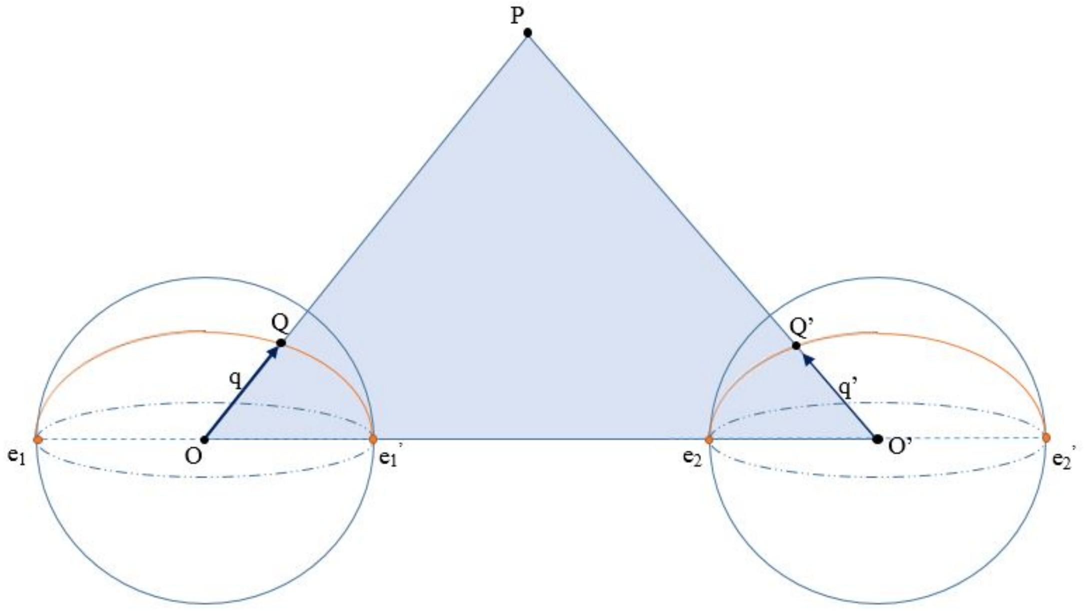


Figure 3.14: Epipolar geometery of fisheye cameras. The fisheye cameras represented by Spherical Retinas whose Radius is 1. (e_1, e_1') represents the epipoles of camera 1 with centre O and (e_2, e_2') represents the epipoles of camera 2 with centre O'. The epipolar curves are shown in orange on the two retinas and **q,q'** represent the unit vectors of **OP,O'P** and intersect the spherical retina at **Q,Q'**.

parameters need to be optimized for these rays to intersect. This means that we have to find the ideal unit direction vectors $\hat{\mathbf{q}}$ and $\hat{\mathbf{q}}'$ which are close to **q** and **q'**. The error criterion used as an error measure between the ideal unit direction vector $\hat{\mathbf{q}}$ and reprojected unit direction vector **q** is the angular error. This angular error[36] is defined as the sum of squared sines of angles between **q** and $\hat{\mathbf{q}}$ and between **q'** and $\hat{\mathbf{q}}'$ and is defined as

$$\mathcal{E}(\mathbf{q}, \mathbf{q}', \mathbf{E}) = \min_{\hat{\mathbf{q}}, \hat{\mathbf{q}}'}(\|\mathbf{q} \times \hat{\mathbf{q}}\|^2 + \|\mathbf{q}' \times \hat{\mathbf{q}}'\|^2) \tag{3.44}$$

where **E** is the essential matrix. The geometric meaning of this error is shown in Figure 3.15. In Figure 3.15a **P** denotes a 3D scene point, **n** is the unit normal vector to the epipolar plane Π, **O,O'** are camera centers, $\mathbf{q}, \mathbf{q}'$ are rays corresponding to image points, $\hat{\mathbf{q}}, \hat{\mathbf{q}}'$ are their optimal estimates lying in the epipolar plane, angles ϕ_1, ϕ_2 are angles between image points and their optimal estimates. Figure 3.15b shows the same image as seen from the side. We can see that the rays reprojected from images $\mathbf{q}, \mathbf{q}'$ are not on the same plane Π and hence do not intersect. This can be improved by minimizing the angles ϕ_1 and ϕ_2 . Hence by definition of angular error and from Figure 3.15, the equation for angular error can also be expressed as

$$\mathcal{E}(\mathbf{q}, \mathbf{q}', \mathbf{E}) = \min_{\mathbf{n}}(\sin^2 \phi_1 + \sin^2 \phi_2) \tag{3.45}$$

$$\mathcal{E}(\mathbf{q}, \mathbf{q}', \mathbf{E}) = \min_{\mathbf{n}}(|\mathbf{n}.\mathbf{q}|^2 + |\mathbf{n}.\mathbf{q}'|^2) \tag{3.46}$$

where $\mathbf{n}$ is the normal to the epipolar plane π and **n.q** is the dot product between $\mathbf{n}$ and $\mathbf{q}$.

This error has an exact closed form solution given in [36] and is given by

$$\mathcal{E}(\mathbf{q}, \mathbf{q}', \mathbf{E}) = \frac{A}{2} - \sqrt{\frac{A^2}{4} - B} \tag{3.47}$$

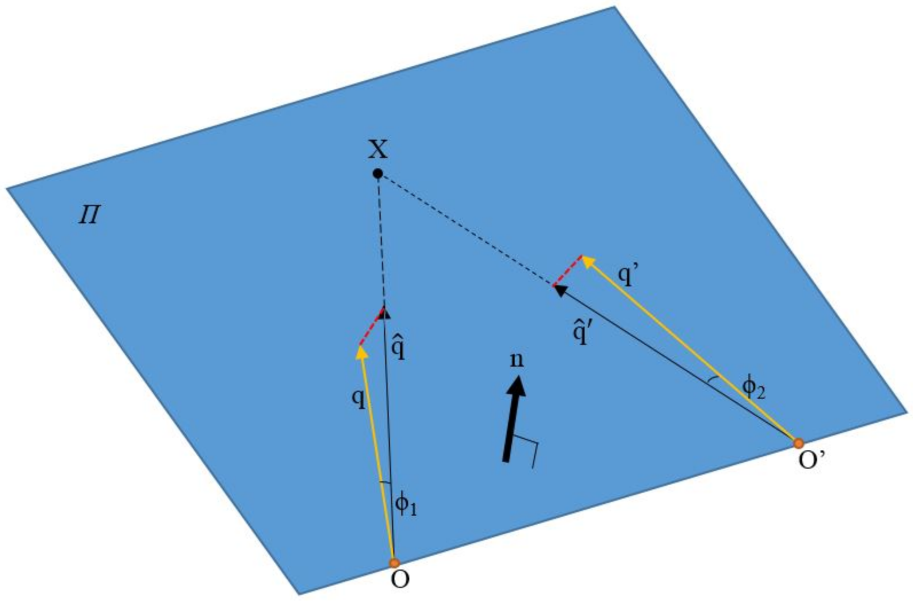
where

$$A = \mathbf{q}^{\mathbf{T}}\mathbf{E}^{\mathbf{T}}\mathbf{E}\mathbf{q} + \mathbf{q}'^{\mathbf{T}}\mathbf{E}\mathbf{E}^{\mathbf{T}}\mathbf{q}'$$

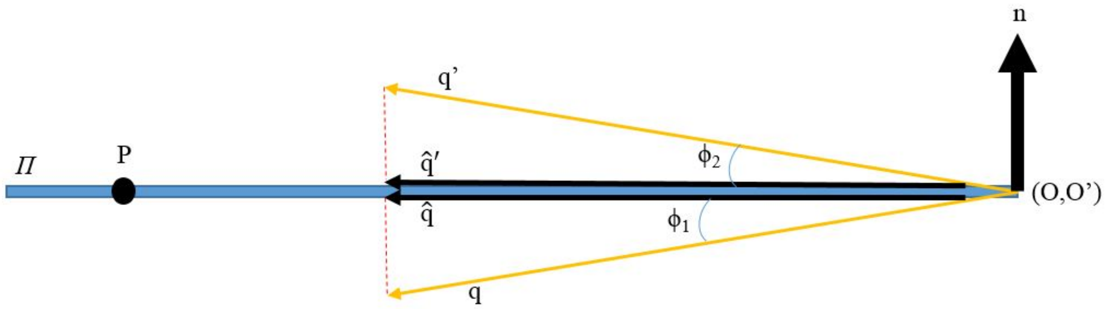
and

$$B = (\mathbf{q}'^{\mathbf{T}}\mathbf{E}\mathbf{q}^2)$$

The main idea in this calibration technique is to minimize the sum of angular errors of point correspondences from two images obtained using SIFT. This sum of angular error is then used as a cost function to be minimized over the camera parameters. This is further explained in Section 3.3.3.



(a) Angular error shown on one epipolar plane Π.



(b) Side view of angular error shown in (a).

Figure 3.15: Angular Error.

3.3.3 Calibration by minimization of the angular error

Calibration using point correspondences and minimization of the angular error is divided into two non-linear minimization steps. This is done by first representing the essential matrix $\mathbf{E}$ in Equation (3.43) as a function of external parameters as discussed in Section 3.3.2[35]. Using the inverse model $\mathcal{P}^{-1}$, the unit vectors **q,q'** can be defined as a function of internal camera parameters ($\mathbf{K}$,l) as described by Equations (3.31)-(3.33). Given n point correspondences $(\mathbf{m_i}, \mathbf{m'_i})$ the cost function to be minimized can be expressed as:

$$C(\mathbf{a}) = \sum_{i=1}^{n} \mathcal{E}(\mathbf{q_i}, \mathbf{q'_i}, \mathbf{E}) = \sum_{i=1}^{n} \mathcal{E}(\mathbf{q}(\mathcal{P}^{-1}(\mathbf{m_i}, \mathbf{a_i})), \mathbf{q}(\mathcal{P}^{-1}(\mathbf{m'_i}, \mathbf{a_i})), \mathbf{E}(\mathbf{a_e})) \tag{3.48}$$

where $\mathbf{a_e}$ are the external camera parameters and $\mathbf{a_i}$ are the internal camera parameters. This cost function is minimized using a non-linear optimization technique. A good estimate of initial values of the parameters are required for performing this optimization. [3] proposes a two-phase optimization approach where the optimization is first performed over the internal parameters using the eight point algorithm to compute a rough essential matrix [35]. The estimated essential matrix is then used as a starting point to minimize the cost function over all the parameters.

Given more than eight correspondences $(\mathbf{m_i}, \mathbf{m'_i})$ and a rough estimate of the internal parameters the unit directional vectors $(\mathbf{q}, \mathbf{q}')$ are calculated by using Equation (3.37). The unit directional vectors corresponding to the image points cab be used in the eight-point algorithm described in [35] to obtain an estimate of the essential matrix $\mathbf{E}$. The obtained essential matrix together with the calculated **(q,q')** can be used to compute the cost function (Equation (3.48)). This cost function is minimized over the internal parameters using non-linear optimization. Then an estimate of rotation and translation parameters are obtained by decomposing the essential matrix as described in Section 6.1. Decomposition of the essential matrix gives four possible values of rotation and translation matrices as shown in Table 6.1. The correct solution is obtained by taking into account the orientation of the unit vectors $(\mathbf{q}, \mathbf{q}')$. The correct solution is the one where the estimated unit direction vectors$(\hat{\mathbf{q}}_\mathbf{i}, \hat{\mathbf{q}}'_\mathbf{i})$ are in the same direction. This implies that if the 3D point is triangulated, the depth will be positive. Once the correct rotation and translation parameters are obtained, they are used as an estimate for a final optimization stage. All the estimated parameters are then used in the minimization of the cost function given by Equation (3.48). The robustness to outliers of image correspondence mismatches due to SIFT is obtained by using the RANSAC algorithm[35]. A subset of point correspondences $n = 8$ is chosen at random and the parameters are estimated by the above algorithm. Using these parameters, the number of inliers according to the angular error (Equation (3.44)) are calculated. The model with the highest number of inliers is selected and the parameters are recomputed using all the inliers. This is further optimized using a final non-linear optimization step which optimizes all the parameters. The output of the optimization is the estimated intrinsic and extrinsic parameters. We have used Levenberg-Marquardt[32][33] optimization for the optimization steps in our implementation.

An important point to be mentioned with regard to this auto-calibration method is the presence of local minima being found in the non-linear optimization step. This is especially the case when the intial guesses for the camera parameters are not good enough for the minimization to find the global minimum. In order to avoid this an iterative approach is used where estimation is done several times using different initial conditions which are determined by a combination of reprojection errors of reconstructed points and minimal feedback from the user. It is still advised to use auto-calibration for high precision only if there are no other alternatives, due to the limitations of the model parameters and problems in optimization[3].

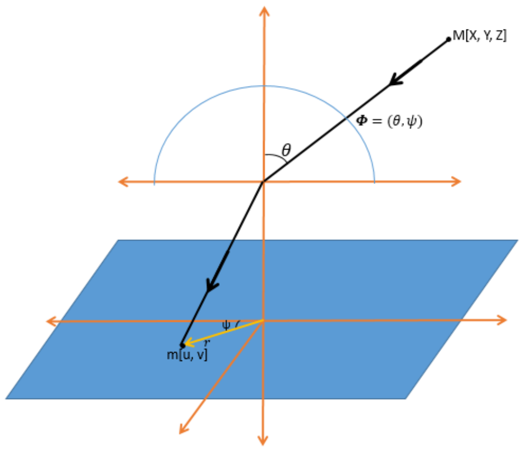
Chapter 4

DISTORTION CORRECTION

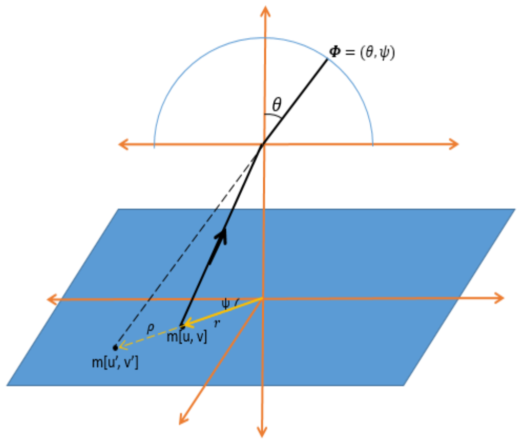
Distortion correction is one of the most direct applications of fisheye camera calibration. Once the camera is calibrated, we obtain the calibration parameters. These calibration parameters describe the distortion function which forms an important part of the forward model $\mathcal{F}$ of the camera which maps the real world co-ordinates to the fisheye camera plane as shown in Figure 4.1a.

The forward camera model describes the mapping of an incoming ray with direction $\mathbf{\Phi} = (\theta, \psi)$ to a pixel $\mathbf{m}(\mathbf{u}, \mathbf{v})$ on to the camera plane, where θ is the angle made by the incoming ray with the optical axis and ψ is the angle made by the line joining the principle point the pixel $m(u, v)$ to the x axis on the image plane. To perform distortion correction the inverse model is to be calculated to determine the direction of ray mapped to each pixel in the fisheye image. In other words, we need to obtain $\mathcal{F}^{-1}$ which maps $\mathbf{m}(\mathbf{u}, \mathbf{v})$ to a unit vector in the direction $\mathbf{\Phi} = (\theta, \psi)$ as illustrated in Figure 4.1b.

Once we get the backward model by inverting the intrinsic parameters of the camera (as discussed in Section 4.1), the distortion corrected image can be obtained by first finding the direction of the ray mapped to a pixel of the fisheye image and then reprojecting this ray using a planar camera (perspective) projection ($\rho = f\,tan(\theta)$, where f is the focal length of the pinhole camera corresponding to the corrected image) onto the distortion-corrected image plane. This is illustrated in Figure 4.1b where ρ is the radial distance from the optical center in the corrected image plane. Hence, distortion correction can be seen as a mapping of each pixel with radius $r = \sqrt{u^2 + v^2}$ in the fisheye image to a pixel with radius $\rho = \sqrt{u'^2 + v'^2}$ in the corrected image where



(a) Forward model. $\mathcal{F}$



(b) Backward model. $\mathcal{F}^{-1}$

Figure 4.1: Forward and backward models for fisheye cameras.

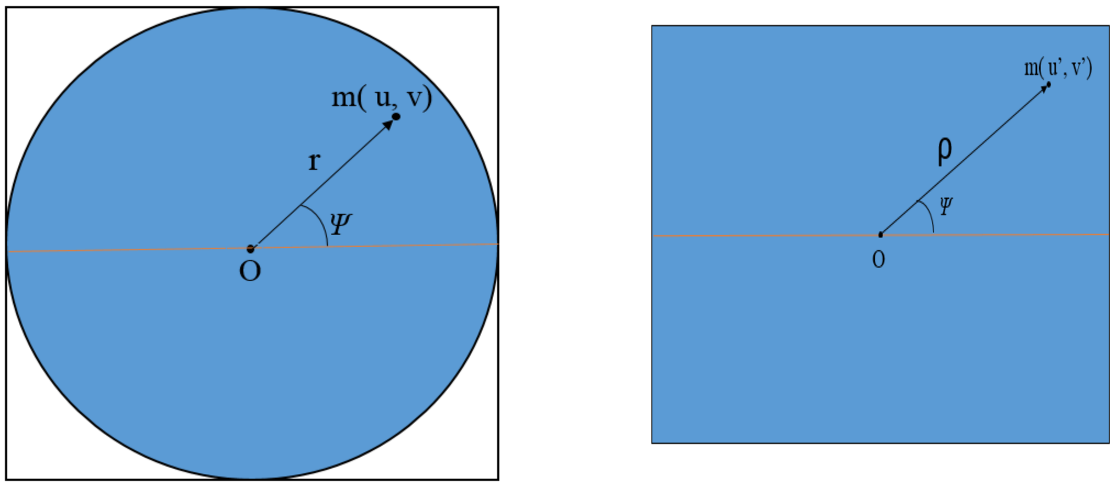
the pixel $m(u, v)$ in the fisheye image is mapped to $m(u', v')$ in the corrected image as shown in Figure 4.2. Hence, a look-up table can be created which maps the position of each pixel in the input fisheye image to a pixel in the output distortion-corrected image. More details about how the mapping is performed are presented in Section 4.1.

4.1 Inverse distortion functions

This section describes how the distortion functions can be inverted to be used in the backward projection model that is employed in the correction of fisheye images. The inverse functions are also used during calculation of reprojection error in the non-linear optimization step of the calibration processes as described in Chapter 3.

As described in Section 3.1 and as given by Equation (3.10), for the Scaramuzza marker-based method[1], the distortion function of a fisheye camera can be expressed as follows:

$$g(r, \mathbf{a}) = a_0 + a_2 r^2 + a_3 r^3 + ... + a_N r^N$$



(a) Fisheye image.   (b) Corrected image.

Figure 4.2: Relation between a pixel in fisheye and corrected image.

where $a_0, a_2, ..., a_N$ are the calibration parameters and r is the radial distance from the optical center of the fisheye camera plane. The general form of the distortion function in terms of θ was defined in Equation (3.4) and is expressed as follows:

$$g(r, a, b) = \frac{r}{\tan(\theta)}$$

where r is the radial distance from optical center and θ is the angle made by incoming ray to the optical axis. The inverse distortion function can be obtained by solving for the roots of Equation (3.10) after equating it with Equation (3.4). Hence, for each pixel in the fisheye image, its radial distance r from the optical center is calculated. This calculated r is used in Equations (3.10) and (3.4) to obtain a function in terms of θ. The roots obtained by solving for the Equation (3.10) after equating it with Equation (3.4) give us the value of θ, i.e, the angle of the incoming ray to the optical axis as shown in Figure 4.1. The value of ψ can be calculated from the x and y co-ordinates of the pixel in consideration as

$$\psi = \tan^{-1}\left(\frac{v}{u}\right) \tag{4.1}$$

So we can obtain $\mathbf{\Phi} = (\theta, \psi)$ the incoming ray direction corresponding to each of the pixels in the fisheye image. This can now be used with the equation of perspective projection $\rho = f \tan(\theta)$ to obtain the radial distance of the pixel in the distortion-corrected image. The value f in this case is the focal length of the pinhole camera corresponding to the corrected image. This can be set to a scalar value to ensure the required FOV in the corrected image is visible. Smaller the value of the focal length used, lesser of the corrected image is visible. The location of the corresponding pixel in the distortion-corrected image can be obtained as

$$u' = \rho \, \cos \, \psi \tag{4.2}$$

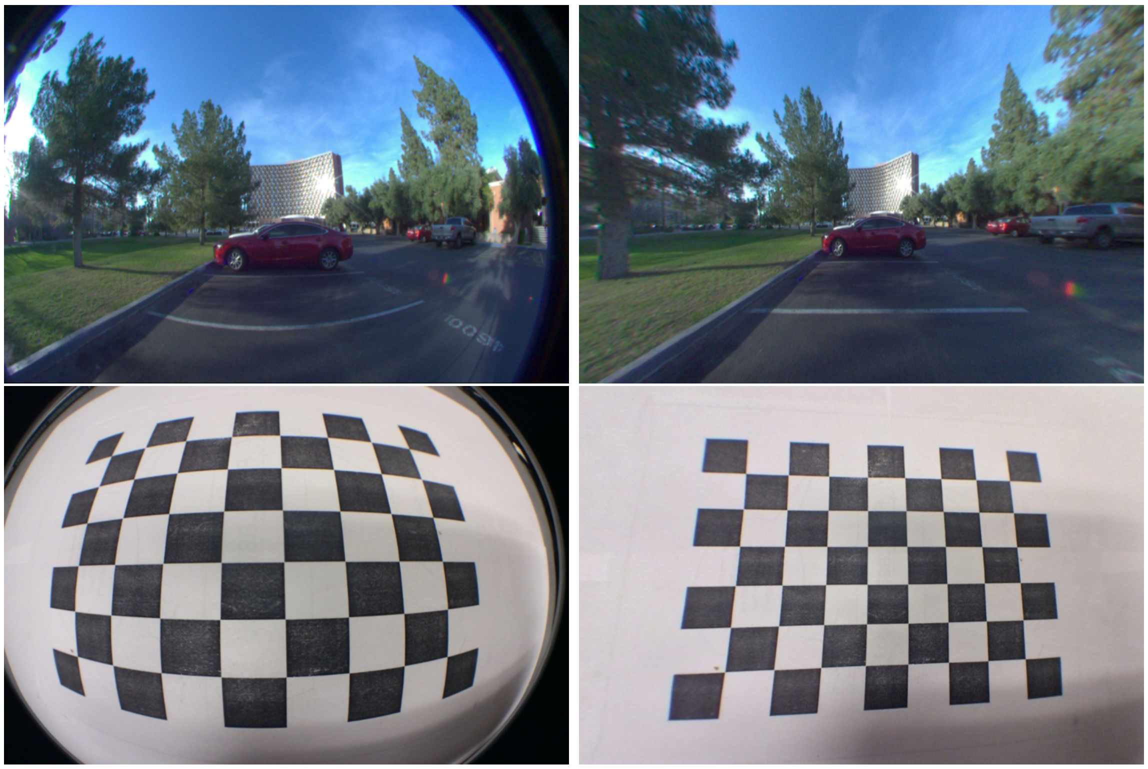
$$v' = \rho \, \sin \, \psi \tag{4.3}$$

This way we can obtain a mapping between the fisheye image and the distorted corrected image. It is important to note that some of the pixels in the corrected image may not be assigned to any pixel in the fisheye image. Hence, we need to perform an interpolation on the distortion corrected image. In our implementation, bilinear interpolation is used. Figure 4.3 shows distortion-correction results that are obtained from a natural image (top) and synthetic image (bottom). The original fisheye images were captured using a Fujinon fe185c086ha-1 fisheye lens attached to a Basler ace acA1300-30uc sensor. Table 4.1 shows the corresponding distortion parameters that were obtained using Scaramuzza calibration and used for the distortion correction.

For the Kannala plane-based calibration model in Section 3.2 the forward model is given by Equation (3.19) i.e,

$$r(\theta) = k_1\theta + k_2\theta^3 + k_3\theta^5 + k_4\theta^7 + k_9\theta^9$$

where r is the radius of the considered pixel in the fisheye camera plane. Similar to the previous case, we can obtain the inverse model (i.e, θ and ψ) by solving for θ in



(a) Fisheye Image. (b) Distortion corrected Image.

Figure 4.3: Distortion correction using fisheye camera parameters that are obtained using Scaramuzza method[1] with the distortion coefficient Length of four.

the Equation (3.19) for the value of r corresponding to each pixel and choosing only real, unique roots whose value lies within the field of view of the camera (usually between $-\pi/2$ and $\pi/2$). ψ can be obtained as above using Equation (4.1) from the pixels in the fisheye image. Hence, we can use the calculated $\Phi = (\theta, \psi)$ to form a distortion-corrected image as previously described. Figure 4.4 shows the original fisheye image on the left and the distortion corrected image using Kannala marker-based calibration[2] on the right. The fisheye image was captured using a Fujinon fe185c086ha-1 fisheye lens attached to a Basler ace acA1300-30uc sensor. Table 4.2 shows the corresponding distortion parameters that were obtained using Kannala

Table 4.1: Parameters of fisheye camera obtained by Scaramuzza calibration[1].

Parameters	Value
Distortion Coefficients	-3.87, 9.30e-04, -1.467e-07, 4.30e-10
Center of distortion (col,row)	395.904, 609.870
Image Size (width,height)	1250,800

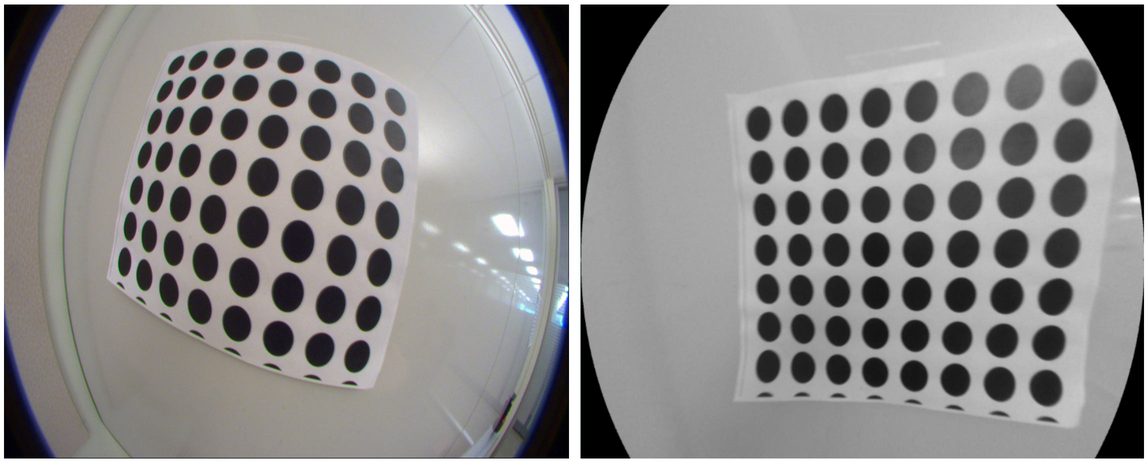


Figure 4.4: Distortion correction using fisheye camera parameters that are obtained using Kannala marker-based method[2] with the distortion coefficient length of four.

marker-based calibration and used for the distortion correction.

In the case of the Kannala auto-calibration technique discussed in Section 3.3 the distortion function is given by Equation (3.34) as follows:

$$r = \frac{(l+1)\sin\theta}{l+\cos\theta}$$

where l is the parameter used to describe the non-linear distortion. The inverse to this model is given in Equation (3.37), i.e.,

$$\theta = \cos^{-1}\left(\frac{-lr^2 + \sqrt{l^2r^4 - (r^2+(l+1)^2)(l^2r^2-(l+1)^2)}}{(r^2+(l+1)^2)}\right)$$

As before, ψ is calculated according to Equation (4.1). As in the previous cases, a mapping between the fisheye and distortion-corrected image in pixels can be obtained

Table 4.2: Internal parameters of fisheye camera obtained by Kannala marker-based calibration[2].

Parameters	Value
Distortion Coefficients	2.294, 5.754e-02, 3.045e-01, -3.484e-01
Center of distortion (col,row)	413.758, 636.544
Image Size (width,height)	1280,800

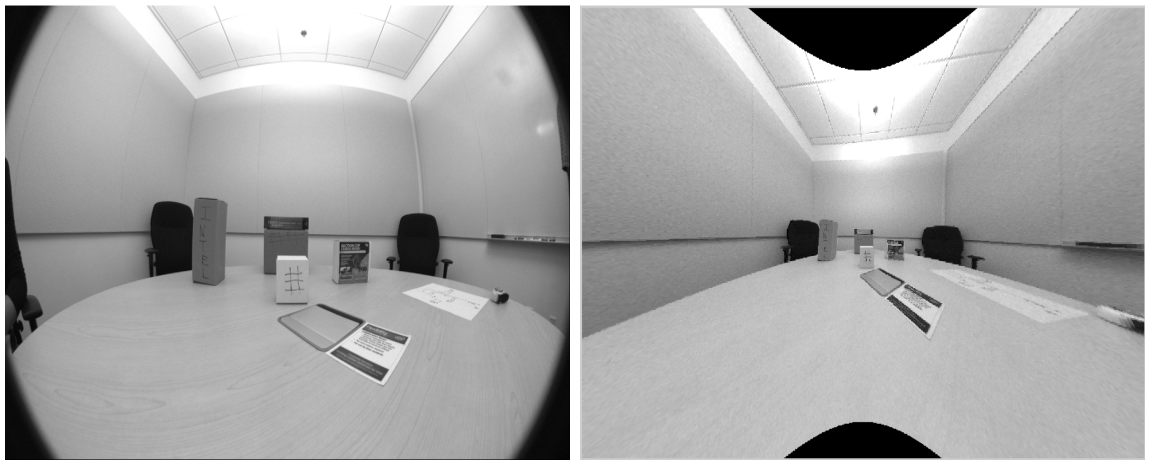


Figure 4.5: Distortion correction using fisheye camera parameters that are obtained using Kannala Auto-Calibration method[3].

and stored in a lookup table that is used for distortion correction. Figure 4.5 shows the original fisheye image on the left and the distortion corrected image using Kannala auto-calibration[3] on the right. The fisheye image was captured using a Fujinon fe185c086ha-1 fisheye lens attached to a Basler ace acA1300-30uc sensor. Table 4.3 shows the corresponding distortion parameters that were obtained using Kannala auto-calibration and used for the distortion correction.

It is important to note that distortion correction can only be done for an angle less then 180 deg. This is due to the fact we cannot represent a point behind the

Table 4.3: Parameters of fisheye camera obtained by Kannala Auto-Calibration[3].

Parameters	Value
Distortion Parameter (l)	3.15
Focal length of fisheye camera	461.77
Center of distortion (col,row)	394.838, 618.007
Image Size (width,height)	1250,800

camera (possible because of FOV $> 180°$) on a plane in front of the camera. Hence when we perform distortion correction we consider only the part of the image where $-70° < \theta < 70°$.

Chapter 5

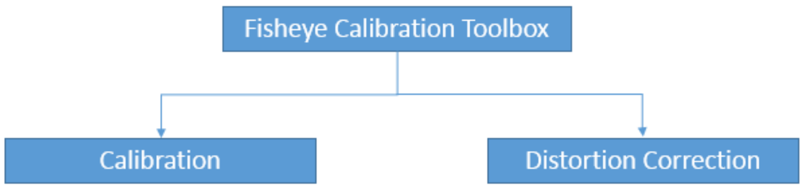
FISHEYE CAMERA CALIBRATION AND DISTORTION CORRECTION (FISHEYECDC) TOOLBOX

This chapter introduces the developed Fisheye Camera Calibration and Distortion-Correction (FisheyeCDC) toolbox's structure and usage. As mentioned earlier, the toolbox was developed for the purpose of simplifying user interaction for the calibration of fisheye cameras. Hence, the design has a simple layout and interface to make the process of calibration easier. This is done by keeping interaction between user and algorithms to a minimum. The toolbox provides an easy, structured and clear interface to help the user as he/she does not have to worry about the implementations of the various algorithms in the toolbox.

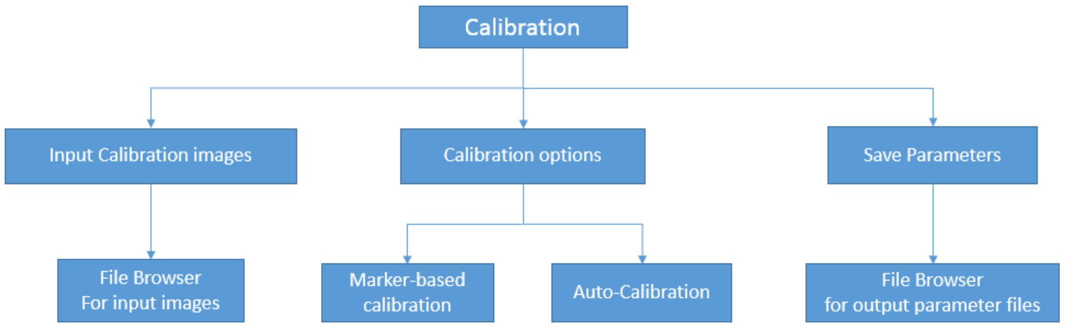
The toolbox was developed in MATLAB due to ease of programming and designing a graphical interface. It also advantageous to use MATLAB as it can run on a number of platforms with little or no customization required. Also, some of the reference calibration methods we have incorporated in our system [13][14] have been developed in MATLAB and hence it is easier to streamline all these methods under one application. Routines that are implemented in the C language can also be called from MATLAB.

5.1 Toolbox Layout

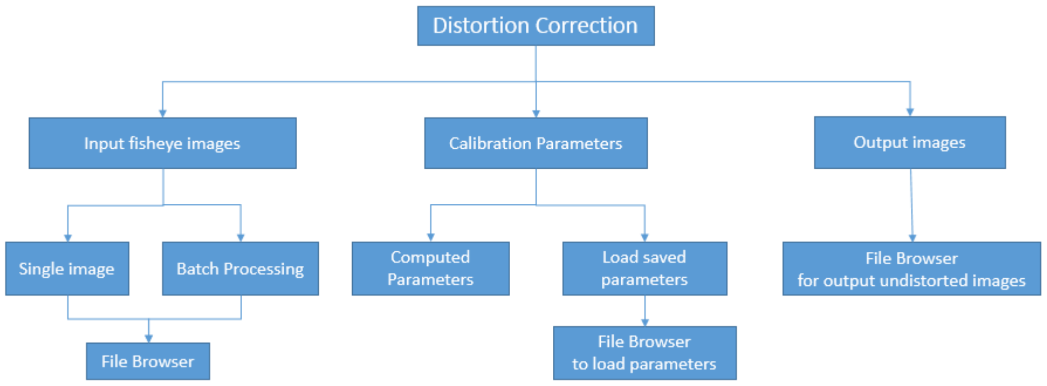
The designed toolbox follows the layout shown in Figure 5.1. The toolbox performs two functions, calibration and distortion correction. Either one of the two functions can be selected by clicking on one of the tabs as shown in Figure 5.2. By default the calibration tab is active. The functionalities of each of the tabs are described below.



(a) Basic layout.

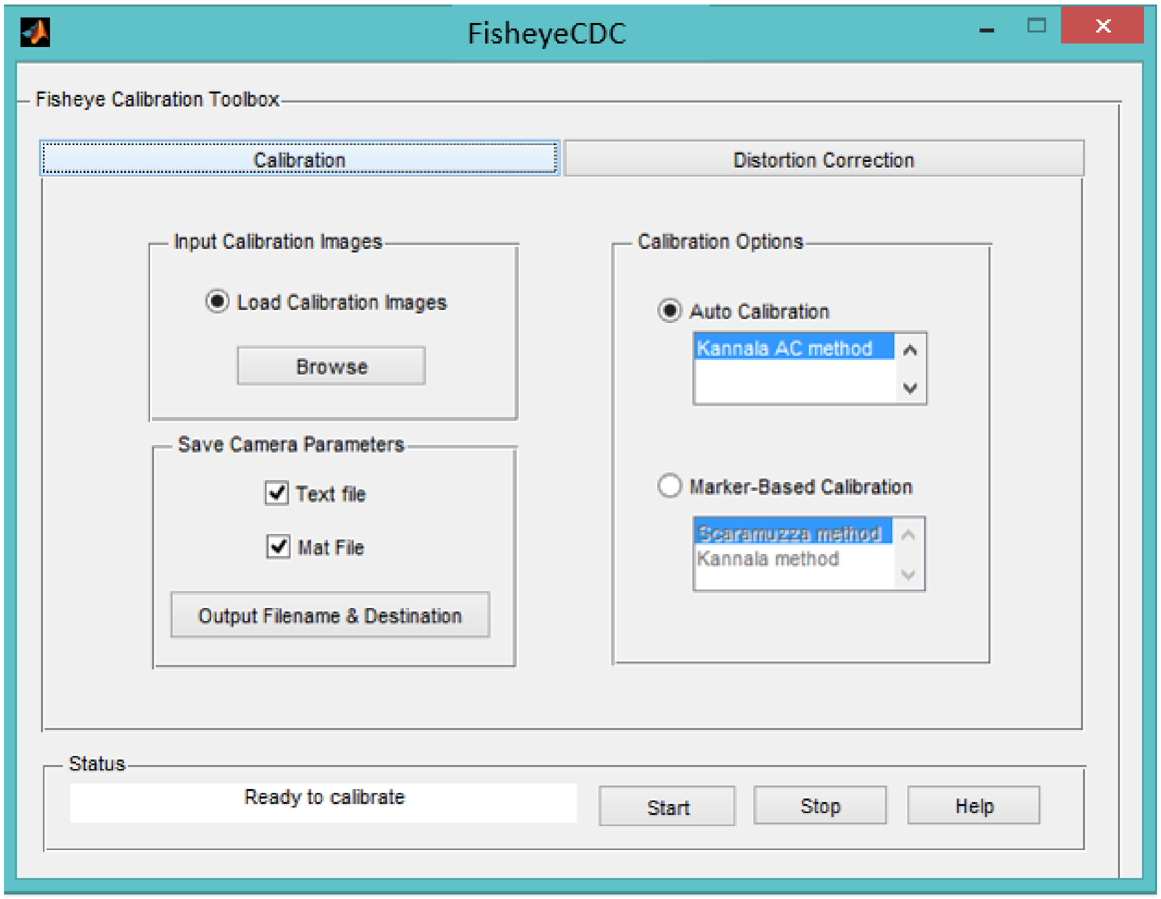


(b) Calibration Tab layout.



(c) Distortion correction Tab layout.

Figure 5.1: Layout of FisheyeCDC Toolbox.



(● Load Calibration Images)

☑ Text file

☑ Mat File

(● Auto Calibration) Kannala AC method

○ Marker-Based Calibration: Scaramuzza method, Kannala method

Status: Ready to calibrate

Figure 5.2: Toolbox with Calibration Tab active.

Calibration Tab: The **Calibration** tab in the GUI must be selected when the user has to perform calibration of the fisheye camera. When the application is initially launched or the **Calibration** tab is selected by the user, the GUI has the appearance shown in Figure 5.2. The layout of the **Calibration** tab is shown in Figure 5.1b. For calibration, the user has to load the calibration images into the toolbox using the **Load Calibration Images** file browser, select the calibration method to be used in the **Calibration Options** panel and specify the formats and the filenames of the output parameter files to be stored. The input images for this stage are the images of the calibration patterns as shown in Figure 2.5, if the marker-based calibration techniques are selected, or two images of the same scene if the auto-calibration technique is used. If less than two images are selected at this stage a warning message prompting the user to select more than two images is displayed (Figure 5.4a). The user then needs to select the type of calibration (Auto or Marker-based calibration) he/she wants to use by clicking on one of the radial buttons in the **Calibration Options** panel and then selecting the method from the list provided under each type. The user also has to select the format he/she would like the output parameter files to be stored in by clicking the corresponding checkboxes and needs to provide the filepath and filename for the output files using the **Output Filename and Destination** browser button in the **Save Camera Parameters** panel. Once the above steps are done, the user can begin calibration by clicking on the start button in the **Status** panel. The process of calibration will start only if all the inputs and output names are set up in the calibration tab. Clicking on **Start** with any of the options not set will result in an error message prompting the user to select or set up the required options as shown in Figure 5.4b.

Distortion Correction Tab: The user must select this tab to perform distortion correction. Once the tab is selected the toolbox appears as shown in Figure 5.3. The user can input either a single image or can process a batch of images depending on the selection in the **Input Images** panel. If single input is selected, the user can then use the **Browse** button and navigate to the fisheye image that he/she wants to undistort. If **Batch Processing** is selected, the **Browse** button now can be used to select the folder which contains the images that need to be corrected for distortion. It is assumed that all the images in this folder are from the same fisheye camera. If distortion correction is being done immediately after the calibration then the user can select the **Use Computed Parameters** button in the **Calibration Parameters** panel. This assumes the images being undistorted are from the same fisheye camera

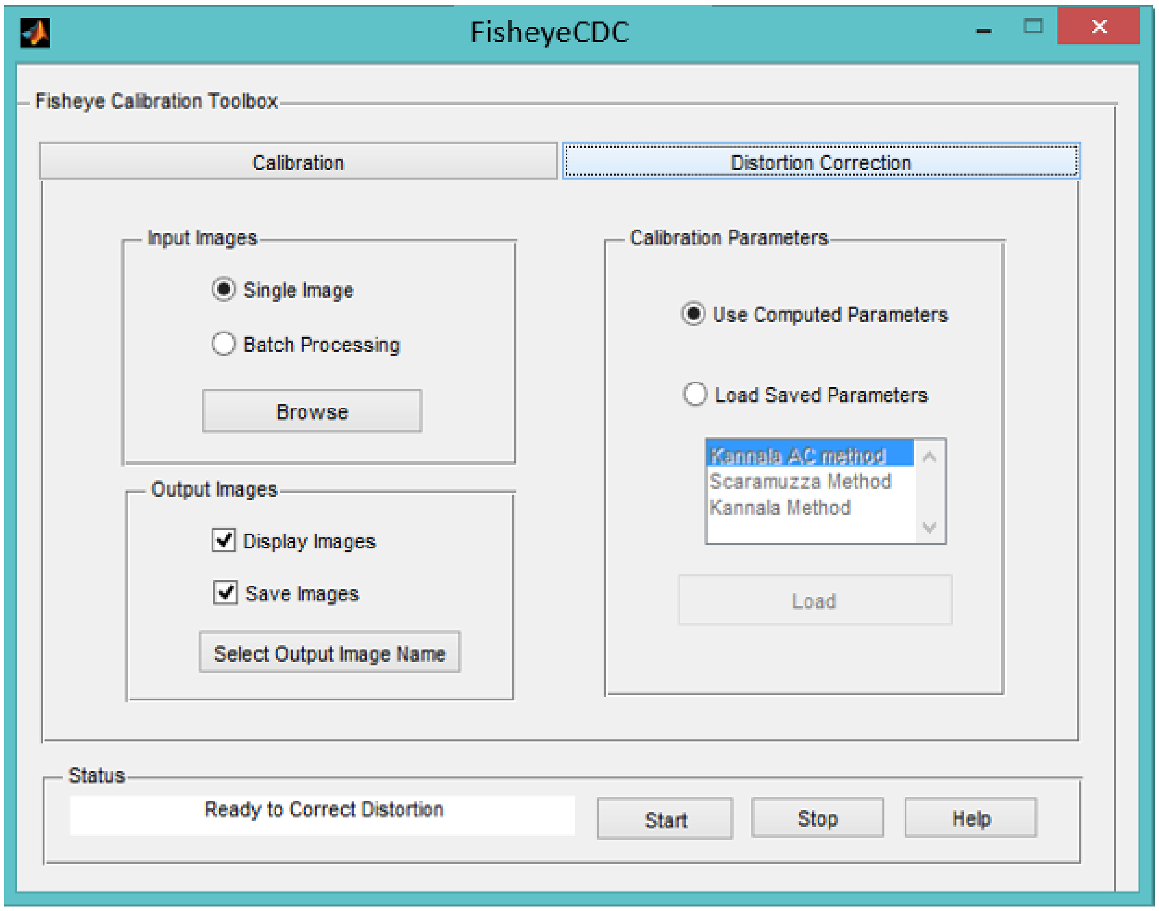
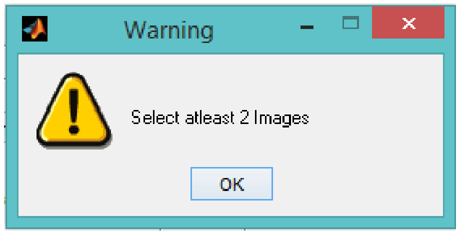


Figure 5.3: Toolbox with Distortion Correction Tab active.

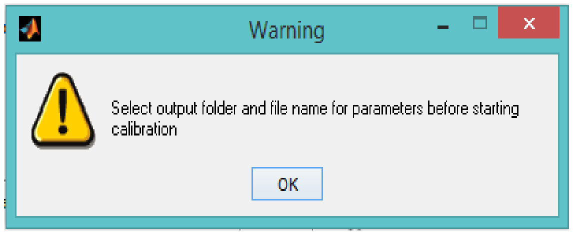
that was just calibrated. If the undistortion has to be performed at a later stage, the user can select the **Load Saved Parameters** option and use the **Load** button to navigate to and load the parameter file of the fisheye camera which was previously calibrated. In this case, the user has to specify which method was used to calibrate the camera from the list in the **Calibration Parameters** panel for the parameter file to have the correct parameters. If the type of parameter file that was loaded was different from the type selected in the list, the user receives a warning as in Figure 5.4c. For example, if the loaded parameter file was for the auto-calibration but the method selected in the list was one of the marker-based calibration methods


Warning

Select atleast 2 Images

OK

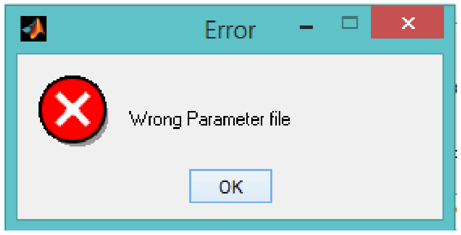
(a) Insufficient images for calibration.


Warning

Select output folder and file name for parameters before starting calibration

OK

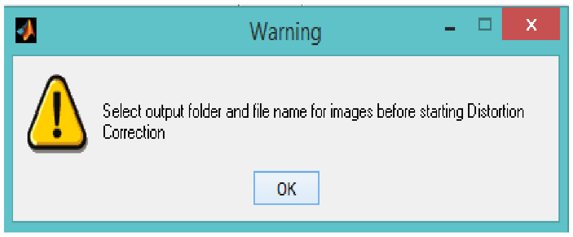
(b) Options not selected for calibration.


Error

Wrong Parameter file

OK

(c) Incorrect parameter file loaded.


Warning

Select output folder and file name for images before starting Distortion Correction

OK

(d) Options not selected for Undistortion.

Figure 5.4: Warning and error messages.

the warning in Figure 5.4c is displayed to the user. The user also has to set up the path and the filenames for the output images. If a single image is being processed(i.e, the **Single Image** button in **Input Images** panel is selected), then the user has to select the output folder and enter the filename for the output image. If undistortion is being performed in batch processing mode, the user only has to select the folder for the output undistorted images to be saved. The filenames of the images will be similar to the input filenames. The **Display Images** check box in the **Output Images** panel can be checked if the user wants to display the images as they are being undistorted. Once all the options have been set, the user has to click on the start button in the algorithm control panel to begin the process of undistortion using the calibrated parameters. Similar to the calibration tab, if all the options are not set before clicking on start, the user receives a warning message as shown in Figure 5.4d.

The **Status** panel is common to both the **Calibration** and **Distortion Correc-**

tion tabs. It notifies the user that the process of calibration or distortion correction can be started once the correct options in all the panels have been selected. The process can be halted at any time by hitting the **Stop** button. The **Help** button leads to a detailed help document for the currently active tab.The **Status** panel also contains the progress and status bar which displays the progress of the calibration or undistortion process. It also displays status of intermediate tasks such as completion of loading of images for calibration or undistortion and completion of corner detection during calibration etc.

5.2 Format choices for output of calibration

The output from the calibration step are the parameters for the camera. These parameters depend on the type of method being used for calibration. In general the parameters output from each of the method as the coefficients or parameters for distortion function, the principal point of the image or the center of the distortion, the focal length and the dimensions of the images used for calibration. In some cases a few extra parameters may be present such as affine transformation coefficients. We have a choice of two supported formats for storing these calibration files They can be stored as a MATLAB matrix file(.mat) and/or a text file(.txt) file. In the matrix file the system parameters are stored as a data structure with the appropriate fields for each of the parameters. This makes it easier to load and read the file in MATLAB especially for our undistortion application. The data structure is checked when the file is loaded during undistortion to make sure that the right parameter file is being used. The text file contains a header that describes the calibration method used for calibrating the camera along with the calibration parameters. The text file version is human readable. The text file is also useful if the user wants to use the camera with applications that are developed in different programming languages such as C,C++

and Java to name a few.

5.3 Calibration Procedure

This section describes how to perform calibration using the developed FisheyeCDC toolbox. First the toolbox has to be set up for calibration, as described in Section 5.1, with the appropriate options and parameters such as output file formats, file names and destination folders. To calibrate the camera the following steps can be followed:

1. Click on the **Calibration** tab at the top of the toolbox.

2. Click on **Browse** in the **Input Calibration** panel to open the file browser and select the required calibration images.The user will be notified once the images have been loaded.

3. In the **Calibration Options** panel, select the type of calibration method (Auto or Marker-based calibration). Based on the selection, the list containing the appropriate methods will be displayed. Select the desired calibration method from the list.

4. In **Save Camera Parameters** panel, select the desired format of the output parameter file required. Click on the **Output Filename and Destination** button to select the output folder and enter the name of the output file.

5. Click on **Start** to begin the process of calibration.

Some aspects of the calibration depend on the calibration algorithm being used and in some cases may require some user interaction to provide some of the initial parameters to perform calibration. In the auto-calibration technique, once the ellipse fitting is completed and the center is calculated, the user is prompted to check if the ellipse and camera centers look visually correct as these values will be used in the algorithms as initial conditions for the non-linear minimization steps. The user is also prompted after calibration is done to check if the undistorted image is visually correct. If the user is not satisfied with the result, the calibration is repeated with different initial parameters. In the case of using the Scaramuzza technique, the user is asked to enter some details about the calibration pattern (number of checkerboards, approximate centers, size of checkerboards etc). Here the user also has the option of selecting the checkerboard corners manually if he/she does not prefer using automatic corner detection. In the Kannala plane/marker-based technique, the user is prompted to enter some basic information about the camera if it is available (e.g., manufacturer focal length, type of projection etc) and also some details about the calibration pattern (e.g., number of blobs in the x and y directions). The user also has to select the bounding box of the calibration pattern inside which the blobs or circular control points are detected. This is done by an interactive procedure where the user is prompted to select the polygon that encloses the imaged calibration pattern.

The user is informed once the calibration procedure is complete and the results are saved in the appropriate locations and in the required formats. These parameters are also retained in the memory in case the user wants to perform distortion correction.

5.4 Distortion Correction procedure

This section describes how to perform distortion correction of fisheye images using the FisheyeCDC toolbox. the procedure consists of the following steps.

1. Click on the **Distortion Correction** tab at the top of the toolbox.

2. Select the desired processing mode in the **Input Images** panel

   - If the single image mode is selected, click on the **Browse** button and navigate to the image to undistort and load the image.

- If the batch processing mode is selected, click on **Browse** and select the folder containing the fisheye images to be corrected. It is important to note that all the images in this folder must be from the same fisheye camera.

3. Calibration parameters

   - If performing distortion correction immediately after calibration, select the use **Computed Parameters** option in the **Calibration Parameters** panel.
   - If performing the operation at a different time, then select **Load Saved Parameters**, choose the type of calibration method that was used to calibrate the camera and click on **Load** and navigate to the parameter file.

4. Select the **Output Filename and Destination** folder from the **Output Images** panel in case of undistorting in single image mode. In batch processing mode, select the output folder where the undistorted images need to be stored.

5. Click on the **Start** button in the **Status** panel to start the distortion correction process. The user is notified once the process is complete.

The process of undistortion requires no interaction from the user after the start button is pressed. The status of the process is displayed on the status bar in the **Status** panel.

5.5 Results

This section presents the results of calibration using the different techniques that are available in the FisheyeCDC toolbox. A good method to check the accuracy of calibration is to use the root mean square (RMS) error also known as the reprojection error which can be defined as the distance between modeled and measured control points. We know the positions of modeled control points as the pixel coordinates of either checkerboard corners in the case of Scaramuzza technique or the centroids of the circular points in the Kannala marker-based technique. These are obtained in the keypoint localization stages of each of the algorithms and can be represented as $\mathbf{m_{ij}}$. We also know 3D coordinates $\mathbf{M_{ij}}$ of the control points (from the calibration patterns). These are reprojected onto the image using the estimated camera internal and external parameters to form the measured control point coordinates which can be represented by $\mathbf{m'_{ij}}$. The reprojection error can now be defined as follows:

$$Err = \frac{\sqrt{\sum_{i=1}^{N}\sum_{j=1}^{K}(\mathbf{m_{i,j}} - \mathbf{m'_{i,j}})^2}}{NK} \tag{5.1}$$

where N is the number of calibration patterns and K is the number of control points in each pattern. In case of auto-calibration, two images of checkerboards were used and point correspondences were manually picked. These correspondences act as the modeled points. These corresponding points were also reconstructed to obtain their 3D coordinates which were then reprojected onto the image using calibration parameters to the form the measured control points. The lower the value of the reprojection error, the better is the calibration result.

A coefficient length of 4 was used during calibration for the Scaramuzza and Kannala marker-based calibration methods. The reprojection errors for each method used is shown in Table 5.1. As expected it can be observed that the marker-based methods give comparable results while the auto-calibration technique performs worse than the marker-based methods.

We also test the effect of the number of calibration images on the RMS reprojection errors for the marker-based calibration techniques. The results in Table 5.2 show that

Table 5.1: Performance of fisheye camera calibration in terms of RMS reprojection error.

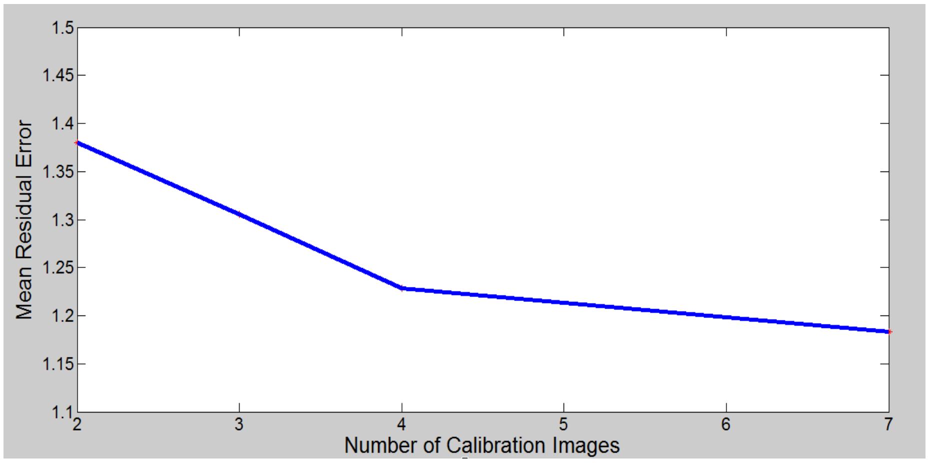
Method	RMS reprojection error (Pixels)
Scaramuzza marker-based technique[1]	1.1662
Kannala marker-based technique[2]	1.1831
Kannala auto-calibration technique[3]	2.682

Table 5.2: Effect of number of calibration patterns on RMS error.

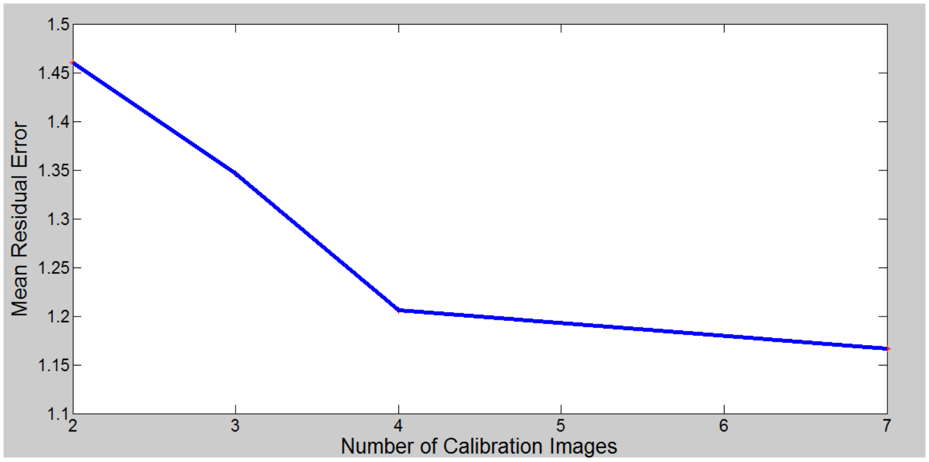
No. of calibration images	Error(Kannala method)	Error(Scaramuzza method)
2	1.3798	1.4599
3	1.3056	1.3466
4	1.2287	1.2059
7	1.1831	1.1662

the error exhibits a decreasing tendency with the increase in number of calibration pattern images. The plot of RMS reprojection error in function of the calibration images is shown in Figure 5.5 for the marker-based calibration techniques. The error in the x and y direction was computed for each control point as the difference between the x and y coordinates of each of the measured and modeled control point. They show the error for each of the reprojected points compared to the modeled point. Figure 5.6 shows these error plots for the marker-based techniques when using three and four calibration pattern images.

For auto-calibration two images of checkerboards were used and point correspondences between them were picked manually. Using the computed calibration parameters the 3D points were reconstructed and then reprojected back onto the images. The resulting reprojection error is shown in Table 5.1 and Figure 5.7 shows the error plot



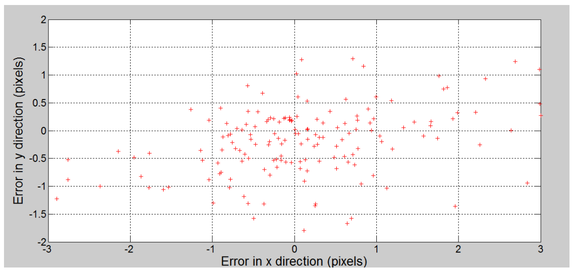
(a) Kannala marker-based method



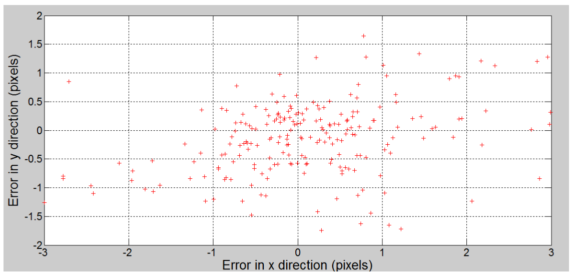
(b) Scaramuzza marker-based method.

Figure 5.5: Effect of number of calibration images on RMS reprojection error.

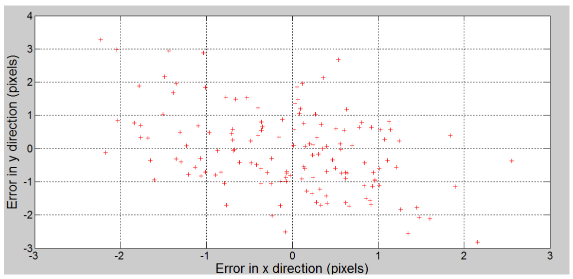
in the x and y directions as the difference of measured and modeled control points. Compared to marker-based techniques (Figure 5.6), the auto-calibration method results in more scattered plots (Figure 5.7), which indicates lower accuracy. This is because we are estimating the distortion of the fisheye camera using just one parameter (l in Equation (3.34) compared to the marker-based techniques in [2] and [1] which use more parameters to model the distortion.



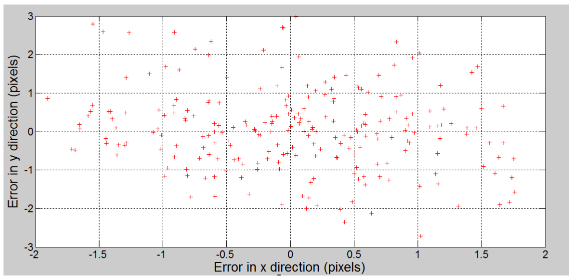
(a)



(b)



(c)



(d)

Figure 5.6: Error plot of reprojected points in x and y directions. (a) and (b) show the error plots in x and y directions of Kannala marker-based technique[2] for three and four calibration images respectively;(c) and (d) show the error plots in x and y directions of Scaramuzza technique[1] for three and four calibrations images respectively.

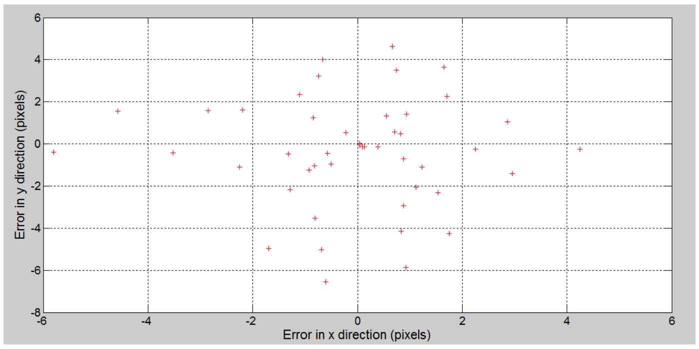


Figure 5.7: Error plot of reprojected points in x and y directions for Kannala Auto-Calibration[3].

Chapter 6

RECONSTRUCTION ON FISHEYE IMAGES

A major application of computer vision is 3D reconstruction of a scene using captured images. Reconstruction mainly consists of obtaining a sparse or dense depth map of a scene using images. It is important for many applications such as robotics for vision based SLAM (simultaneous localization and mapping), visual odometry, rendering in graphics, augmented reality, geographical photo-tagging, defect detection in manufacturing, driver assist systems in automobiles, to name a few. There are many methods to perform 3D reconstruction using different camera types. Stereo cameras, RGB-depth cameras, structured light cameras are some examples of cameras that are used for this purpose. Determining the camera parameters is an important component of 3D reconstruction. This can be achieved through off-line or on-the-fly camera calibration. One method for 3D reconstruction is structure from motion(SFM) which has been a widely studied topic over the last few decades especially for planar cameras[41] [42]. This chapter discusses how we can perform 3D reconstruction using images of a scene captured by the same fisheye camera from unknown positions using the concept of structure from motion.

6.1 3D SFM-based Reconstruction for Planar Cameras

Structure from motion (SFM) is based on the concept of epipolar geometry of multiview images of the same scene[35]. The geometry between two views of the same scene can be represented by epipolar geometry. Consider two cameras with centers O_1, O_2 viewing the same scene. Let C_1 and C_2 be their corresponding camera planes. In what follows, these two cameras will be referred to as C_1 and C_2 for short.

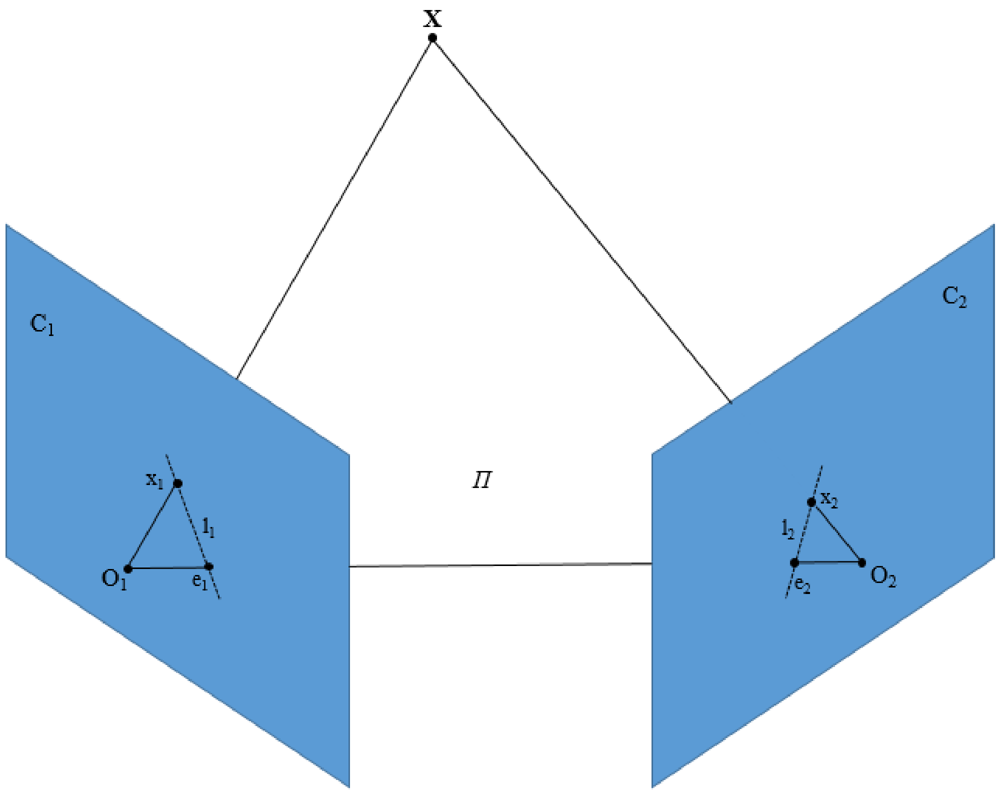


Figure 6.1: Epipolar geometery between two cameras viewing the same point. O_1 and O_2 are the camera centers with camera planes C_1 and C_2 respectively; x_1 and x_2 are the Projections of scene point X on C_1 and C_2 respectively; e_1 and e_2 denote the epipoles of the 2 cameras; l_1 and l_2 are the respective epipolar lines.

A scene point X is projected onto camera image planes C_1 and C_2 as x_1 and x_2 respectively(Figure 6.1). The plane π formed by the lines joining X, O_1 and X, O_2 is called the epipolar plane. The points e_1 and e_2 at which the line joining the camera centers O_1, O_2 meets the camera planes are called the epipoles. The lines l_1, l_2 that are formed, respectively, by joining e_1 to x_1 and e_2 to x_2 are called the epipolar lines. From Figure 6.1, it can be seen that the projection of ray O_1X on the camera image plane C_2 is the epipolar line l_2 and similarly projection of ray O_2X on camera image plane C_1 is the epipolar line l_1. Any point x_1 on camera image plane C_1 that matches a point x_2 in camera image plane C_2 must lie on the epipolar line l_1. Hence, we see that there is mapping between a point on the first camera image plane to an epipolar line on the second plane and vice versa. This mapping is given by fundamental matrix[35] and can be represented as

$$l_2 = F_{12}x_1 \tag{6.1}$$

$$l_1 = F_{21}x_2 \tag{6.2}$$

where F_{12} represents the fundamental matrix that relates the point x_1 to the line l_2 and F_{21} relates x_2 to l_1. From basic geometry, the equation of a point x lying on a line l is $l^T x = 0$ or $x^T l = 0$. Since the point x_2 lies on line l_2 and using Equation (6.1) we have

$$x_2^T l_2 = 0$$

or

$$x_2^T F_{12} x_1 = 0 \tag{6.3}$$

Equation (6.3) is known as the epipolar constraint equation [35]. The fundamental matrix F_{12} is represented in this thesis as $\mathbf{F}$ and is a matrix of rank 2 which encodes the information about the camera locations with respect to a scene. The fundamental matrix between two cameras can be estimated using the eight point algorithm [35]. This requires at least eight points of correspondences between the images captured by the two cameras or two different views of the same camera. In practice, the RANSAC[43] method is used to obtain good inliers. In this method, eight point correspondences are selected at random from the set of correspondences and the fundamental matrix is estimated using the discrete linear transform (DLT)[35]. This estimated fundamental matrix is used in Equation (6.3) with all the point correspondences. The points which are correctly matched will have very low value of error whereas mismatches will have an higher value of error. The matches whose error is less than a certain threshold are considered inliers and the others are considered outliers. The fundamental matrix which has the most inliers in the set is selected and it is re-estimated using all the inliers in the set using singular value decomposition which is a linear least-squares minimization.

The calibration matrix K of a camera gives the relationship between the image co-ordinates and the image sensor co-ordinates for a camera[35]. If the cameras C_1 and C_2 are calibrated and their camera intrinsic matrices are known to be $\mathbf{K_1}$ and $\mathbf{K_2}$, respectively, these matrices give us the relation between the camera image plane and the camera CCD sensor. Hence, we can rewrite the fundamental matrix as:

$$F = K_2^{-T}\mathbf{E}K_1^{-1} \tag{6.4}$$

where E is known as the essential matrix and the epipolar constraint equation (Equation (6.3)) can be rewritten as

$$x_2'^T\mathbf{E}x_1' = 0 \tag{6.5}$$

where x_1' and x_2' represent the normalized coordinates corresponding to points x_1 and x_2 respectively. Normalized co-ordinates are obtained as $x_1' = K_1^{-1}x_1$ and $x_2' = K_2^{-1}x_2$. Normalized co-ordinate x' can be thought of as the image of 3D point X on a camera having the identity matrix I as calibration matrix[35].

The projection matrix $\mathbf{P}$ in Equation (2.2) can be written as a combination of $\mathbf{R}$ and $\mathbf{t}$ as

$$P = [R|t]$$

where $\mathbf{R}$ is a 3×3 rotation matrix and $\mathbf{t}$ is a 3×1 translation vector:

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{12} & r_{22} & r_{23} \\ r_{13} & r_{23} & r_{33} \end{bmatrix} \tag{6.6}$$

and

$$\mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{6.7}$$

Camera 1 is assumed to be at the origin and hence its projection matrix is $P_1 = [I_{3\times 3}|0]$. Consequently, if camera 2 is rotated and translated by $\mathbf{R}$ and $\mathbf{t}$, its projection matrix is $P_2 = [\mathbf{R}|\mathbf{t}]$. The essential matrix can also be represented as $\mathbf{E} = [\mathbf{t}]_{\mathbf{x}}\mathbf{R}$ where $[\mathbf{t}]_{\mathbf{x}}$ represents a skew symmetric form of the translation matrix $\mathbf{t}$ and $\mathbf{R}$ represents the rotation matrix between the two poses of the camera[35]. The essential matrix can be decomposed into rotation and translation matrices which relate the positions of the cameras with respect to each other. Decomposition of the essential matrix can be performed using singular value decomposition. Suppose SVD of E is $Udiag(1,1,0)V^T$. The possible rotation and translation matrices are $\mathbf{R} = UWV^T$ or $\mathbf{R} = UW^TV^T$ where

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.8}$$

and $\mathbf{t} = \mathbf{u}_3$ or $\mathbf{t} = -\mathbf{u}_3$ where $\mathbf{u}_3 = U(0,0,1)^T$. This gives us 4 different possible projection matrices for camera 2 with respect to camera 1 as shown in Table 6.1.The correct projection matrix is chosen after triangulating all the points correspondences to obtain their respective 3D coordinates and choosing the projection matrix which gives all the z components (depths) of the 3D points to be positive[35]. It is important to note that the fundamental matrix has 5 degrees of freedom and hence the term t_z in Equation (6.7) is estimated only up to a scale.

The triangulation is based on the linear triangulation method for noiseless correspondences as described in [35]. For every point m_i in the image Equation (2.2) can

Table 6.1: Possible projection matrices obtained from decomposition of essential matrix.

Number	Possible Projection Matrix (P_2)
1	$[UWV^T \vert -\mathbf{u}_3\ ]$
2	$[UWV^T \vert +\mathbf{u}_3\ ]$
3	$[UW^TV^T \vert -\mathbf{u}_3\ ]$
4	$[UW^TV^T \vert +\mathbf{u}_3\ ]$

be written as

$$m_i \times PM_i = 0 \tag{6.9}$$

where $M_i = [X_i, Y_i, Z_i, 1]^T$ is the homogeneous 3D point corresponding to the 2D point $m_i = [x_i, y_i, 1]^T$. Hence, each image point gives three equations of which two are linearly independent. This can be represented as $AM = 0$ where M is a column vector obtained by stacking all the 3D points M_i, and the matrix A is $m \times P$ for all the point correspondences m_i that are stacked to form a column vector m. A can be represented as

$$A = \begin{bmatrix} x_1P_1^3 - P_1^1 \\ y_1P_1^3 - P_1^2 \\ x_2P_2^3 - P_2^1 \\ y_2P_2^3 - P_2^2 \end{bmatrix} \tag{6.10}$$

where P_1^j is the j^{th} row of the projection matrix P_1 of camera C_1 and P_2^j is the j^{th} row of the projection matrix P_2 of camera C_2. The 3D points $\mathbf{M}$ corresponding to the 2D points $\mathbf{m}$ are estimated by performing a singular value decomposition on the matrix A which solves the system of equations in a least squares sense. In this way a sparse depth map can be obtained using structure from motion.

6.2 Extension of SFM to fisheye cameras

In this section the concept of structure from motion for planar cameras is extended to be applied to fisheye cameras. Fisheye cameras are central cameras as discussed in Section 2.1. The epipolar geometry is applicable to all central cameras and hence is applicable to fisheye cameras too albeit with certain modifications. We consider the camera to be calibrated and its distortion function and intrinsic parameters known. Hence, we know the direction vector related to each of the pixels of the fisheye camera. Since the direction vectors are known, the unit direction vectors for each of them can be calculated.

The fisheye camera's retina is assumed to be similar to a unit sphere (Figure 3.14) as compared to a planar retina of a perspective camera (Figure 6.1). One major difference between the planar and fisheye camera is that instead of epipolar lines we have the presence of epipolar curves as shown in Figure 3.14. Hence, a point Q' on the unit sphere retina corresponding to a unit directional vector q' in the camera with center O' that matches a point Q on the unit sphere retina corresponding to a unit directional vector q in the camera with center O lies on the epipolar curve e_2e_2'(Figure 3.14). q and q' are unit directional vectors corresponding to a point correspondence in the fisheye image. These unit directional vectors are obtained using the camera calibration parameters. Once the camera is calibrated, the backward model for the camera is used to reproject points on the image plane onto a unit circle as shown in Figure 3.9. We use two images of a scene captured by the camera at unknown locations with unknown rotation and translation. The point correspondences between these two images can be obtained using the scale invariant feature transform[37] as explained in Section 3.3.1. These point correspondences are further refined using the RANSAC method which is extended to use unit directional vectors for matches instead of just

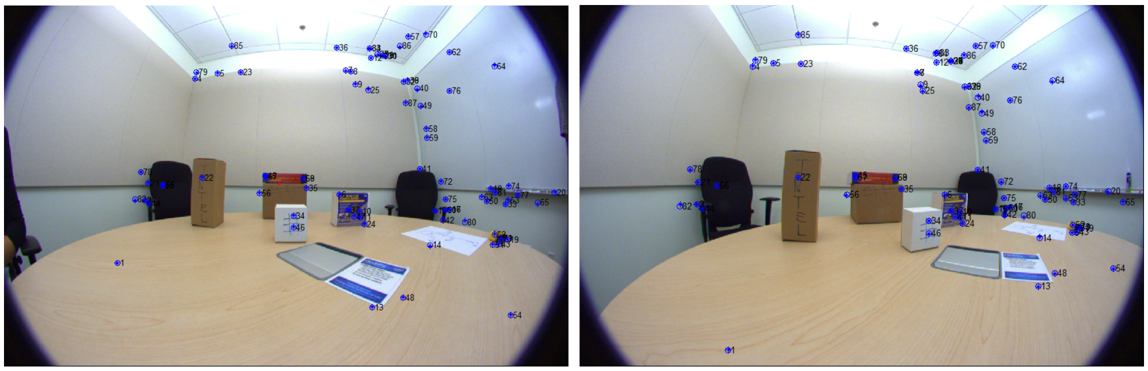


Figure 6.2: Image correspondences after the RANSAC Stage.

image coordinates[4]. The error criteria for obtaining inliers is the angular error as described in Section 3.3.2. The inliers obtained after the RANSAC stage are shown in Figure 6.2. Once the vectors corresponding to image matches are obtained they are used in the epipolar constraint given by Equation (6.3) to obtain the fundamental matrix using the calibrated eight point algorithm [35]. These act like the normalized coordinates described in Section 6.1 and the estimated fundamental matrix turns out to be the essential matrix[28]. The best estimate for the essential matrix is obtained using all inliers of the point correspondences resulting from RANSAC and which lie within an angular error criteria. Once the essential matrix is obtained, it is decomposed into rotation and translation matrices as discussed in Section 6.1. The rotation and translation matrices can be combined to form the four possible projection matrices as shown in Table 6.1, and the best one is chosen using the direction vectors. For the projection matrix to be valid, the direction of the reprojected unit directional vector and the ray joining the center of the camera to the triangulated 3D point should be the same and hence their dot product must be zero. This ensures that the triangulated 3D point has a positive depth. Triangulation is done similar to the linear triangulation method as described in Section 6.1.

Suppose we have a calibrated fisheye camera, we can obtain the unit directional vectors p_1 and p_2 corresponding to image points u_1 and u_2 as described in Section 4.1 In our application, each view is considered to correspond to a camera even if these were taken by the same camera. As described in Section 6.1, we can also obtain the projection matrices P_1 and P_2 for view 1(camera 1) and view 2(camera 2). We assume camera 1 to be at world origin and hence $P_1 = [I_{3\times3}|0]$ and $P_2 = [R|T]$ is obtained by the decomposition of the essential matrix.

$$\alpha p_1 = P_1 X \tag{6.11}$$

$$\alpha p_2 = P_2 X \tag{6.12}$$

Consequently, from Equation (6.11)

$$\alpha x_1 = P_1^1 X$$
$$\alpha y_1 = P_1^2 X$$
$$\alpha z_1 = P_1^3 X$$

where P_i^j is the j^{th} row of the i^{th} projection matrix. Similarly, from Equation 6.12, we obtain for second view:

$$\alpha x_2 = P_2^1 X$$
$$\alpha y_2 = P_2^2 X$$
$$\alpha z_2 = P_2^3 X$$

Using Equation (6.9) and extending Equation (6.10) to include $p_i = (x_i, y_i, z_i)$ as we are using the unit vector to triangulate instead of the normalized coordinates

$m_i = (x_i, y_i, 1)$ in the planar camera case we get

$$A = \begin{bmatrix} x_1 P_1^3 - z_1 P_1^1 \\ x_1 P_1^2 - y_1 P_1^1 \\ y_1 P_1^3 - z_1 P_1^2 \\ x_2 P_2^3 - z_2 P_2^1 \\ x_2 P_2^2 - y_2 P_2^1 \\ y_2 P_2^3 - z_2 P_2^2 \end{bmatrix} \tag{6.13}$$

and solving for X in $AX = 0$ in the least squares sense using SVD we can obtain an estimate of the 3D point up to a scale. Once the 3D points for all the point correspondences are estimated they are reprojected back onto the fisheye image and the reprojection error is calculated as sum of squares of the errors between the reprojected points and the original fisheye points. We accept the estimate of the reconstructed points if the reprojection error is less than 1 pixel.

The flowchart for the process of 3D reconstruction is shown in Figure 6.3

The algorithm for 3D reconstruction is as follows[4]:

1. Calibrate the fisheye camera to be used for the reconstruction using one of the methods described in chapters 3 and 5 and obtain the camera parameters. Here we use the Scaramuzza method for calibration.

2. Capture two images of the scene to be reconstructed using the calibrated fisheye camera.

3. Obtain feature correspondences between the 2 image frames using a feature point detector and matching procedure(e.g., SIFT [37]).

4. Use the calibration parameters to obtain the unit direction vectors for each of the point correspondences in both the images as described in Section 4.1. In

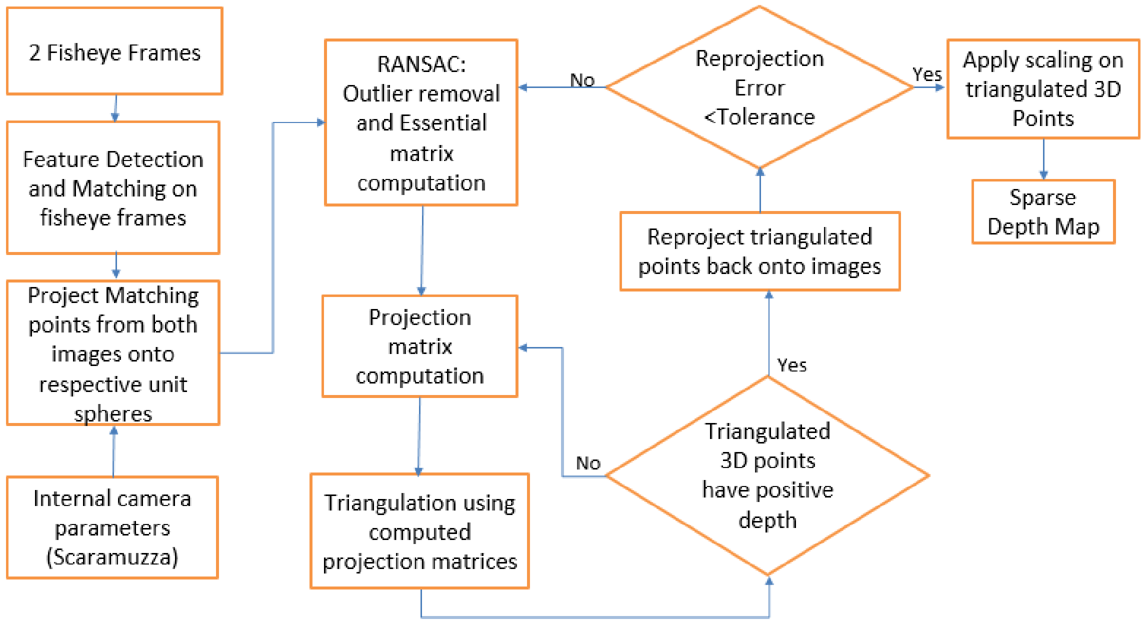

2 Fisheye Frames
Feature Detection and Matching on fisheye frames
Project Matching points from both images onto respective unit spheres
Internal camera parameters (Scaramuzza)
RANSAC: Outlier removal and Essential matrix computation
Projection matrix computation
Triangulation using computed projection matrices
Reprojection Error <Tolerance
No
Yes
Apply scaling on triangulated 3D Points
Sparse Depth Map
Reproject triangulated points back onto images
Yes
Triangulated 3D points have positive depth
No

Figure 6.3: Flowchart for 3D Reconstruction using fisheye cameras.

other words, project the points from both images onto their respective unit spheres.

5. Remove outliers using the RANSAC framework and obtain the best estimate of the essential matrix that minimizes the angular error over all the inliers.

6. Compute the possible projection matrices P as shown in Table 6.1.

7. Triangulate all the inliers using the projection matrix and linear triangulation.

8. Choose the projection matrix which gives a positive depth to all the triangulated points. If none of the projection matrices give this result, the estimated essential matrix is wrong and hence, the process is repeated from step 4.

9. Reproject all the triangulated points back onto the images and calcualte the reprojection error between the original points and the reprojected points.

10. If reprojection error is less then the given threshold then we choose the set of 3D points used as the final results else we repeat the process from step 4.

11. If the true depth to any one of the points is known then scale the points using the true depth to obtain a depth map that reflects true depth of the reconstructed points. If not we obtain the depth map up to a multiplicative scale.

As part of this work, reconstruction from views taken using a fisheye camera was imlpemented and tested. We did not include in our implementation any non-linear optimization stages that can further fine tune the results to give more stable 3D points. A bundle adjustment stage could be added at the end of the process. Bundle adjustment is a non-linear optimization step which jointly optimizes the reconstructed 3D points along with the camera poses, i.e, the camera projection matrices.

6.3 Results

This section describes some results of reconstruction carried out using calibrated fisheye cameras. We captured some images of a scene with known distances from the camera and implemented the algorithm described in Section 6.2. The original fisheye images were captured using a Fujinon FE185C086HA-1 fisheye lens attached to a Basler ACE acA1920-25uc image sensor. Some captured images are shown in Figure 6.4. The cameras were calibrated using the Scaramuzza calibration technique; the resulting camera parameters as shown in Table 4.1. Point correspondences between 2 images were obtained using SIFT as described in Section 3.3.1 and refined using RANSAC. A subset of correspondences that were used for essential matrix estimation are shown in Figure 6.5. These are the final inliers on whom the epipolar constraint is applied. Figure 6.6 shows the regions whose distance from the camera is known for the purpose of performance evaluation. The distances in inches for each of the

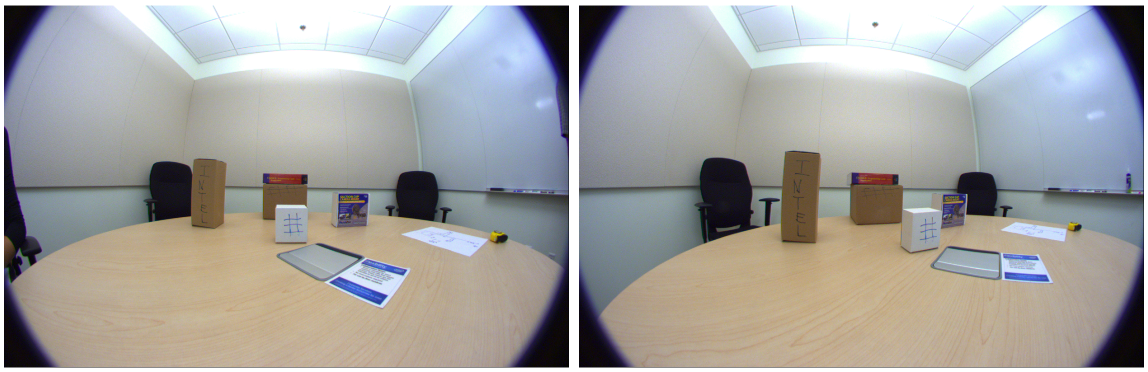


Figure 6.4: Original fisheye images.

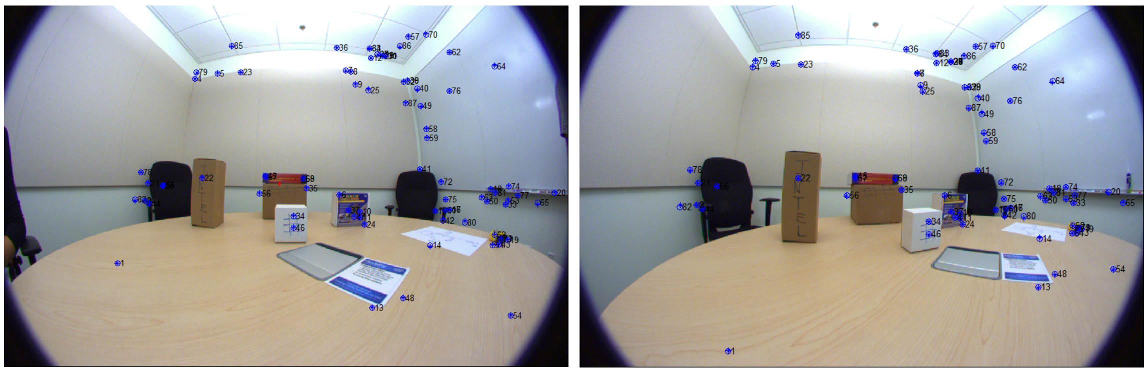


Figure 6.5: Image matches after RANSAC Stage.

regions from the camera are shown in Figure 6.7. Figure 6.8 shows the depths of the scaled reconstructed points plotted against ground truths for the corresponding regions. The scaling was performed using the known depth of region 3. The calculated relative depth for region 3 is divided by the known ground truth depth of region 3 to obtain a scale factor. The relative depths of all the other regions are multiplied by this scale factor to obtain the true depth. We can see that there are few errors in the depth estimation compared to ground the truth especially in region 2 and region 4. This is due to the lack of a sufficient number of feature points in those regions. Also, as discussed previously, the current implementation does not include any non-linear optimization bundle adjustment stage which could reduce the depth estimation error.

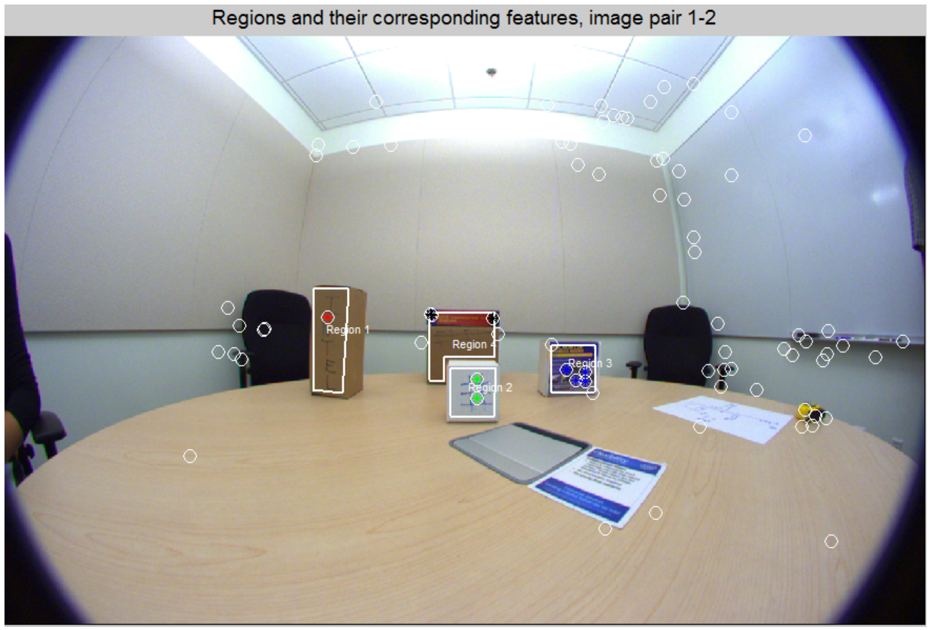


Figure 6.6: Regions of known depth enclosed by white rectangles and their corresponding feature points.

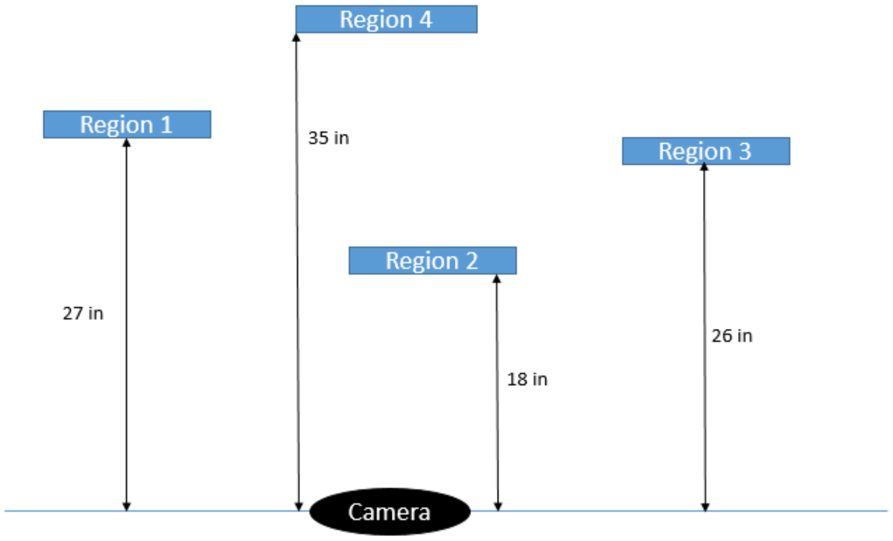


Figure 6.7: Top view of scene setup showing distances of regions from the camera.

The resulting average mean-squared depth estimation error is 2.86 inches.

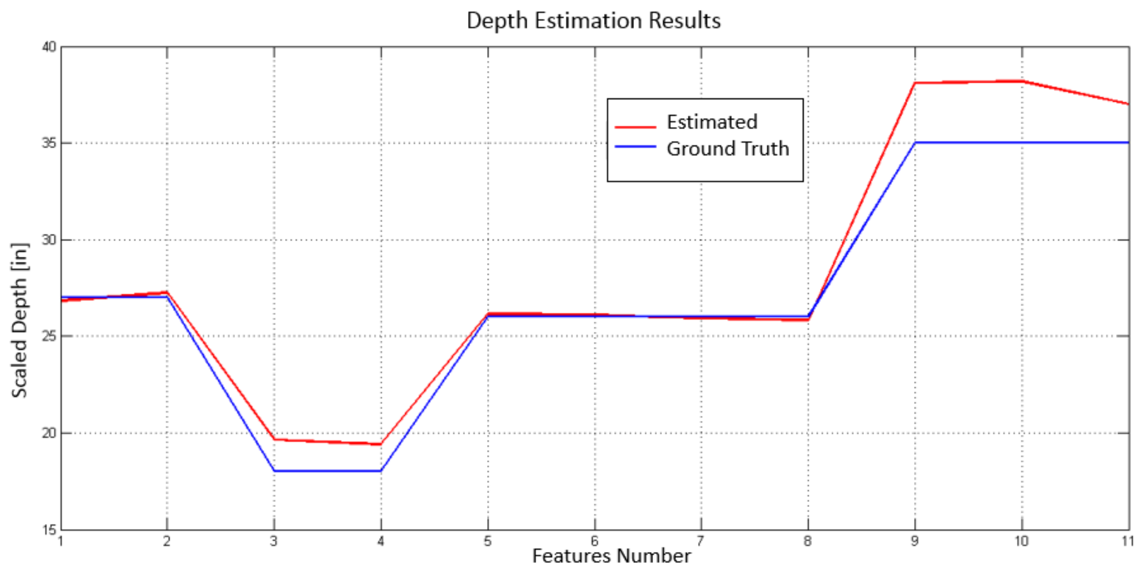


Figure 6.8: Calculated depth versus ground truth comparison for the regions of interest.

Chapter 7

CONCLUSION

This thesis implements a toolbox to perform the calibration of fisheye cameras and distortion correction. It contributes to the field of camera calibration with widespread applications in different domains. This chapter summarizes the contributions of this thesis and discusses the future scope.

7.1 Contributions

The contributions of this thesis can be summarized as follows:

- A user-friendly toolbox with a graphical user interface to perform fisheye camera calibration is implemented for users and practitioners who maybe unfamiliar with calibration procedures.The toolbox can also be used by researchers who are familiar with camera calibration as a tool to compare various methods.
- The toolbox incorporates some of the most widely used algorithms for fisheye camera calibration under one easy-to-use package.
- The implemented toolbox has options of auto-calibration and marker-based calibration techniques for the user. One can use either of the methods depending on required accuracy, flexibility and prior knowledge about the fisheye camera.
- The camera parameters can be stored in different formats giving the user the freedom to use the calibration parameters in other applications.
- Distortion correction of images for fisheye images using the internal calibration parameters is implemented as part of the toolbox which gives the user a

perspectively corrected image similar to the one captured by a planar camera.

- The distortion correction maybe done immediately after the calibration procedure or at a later time by loading the saved parameters into the toolbox.
- The distortion correction can be done on single images or in batches in the batch processing mode.
- The calibration accuracy for the different calibration methods is compared.
- Depth reconstruction using structure-from-motion directly on fisheye images was implemented as an application of fisheye camera calibration.

7.2 Future Work

This section lists out the possible future improvements that can be carried out for the toolbox :

- Incorporate more fisheye calibration methods into the toolbox to have a one stop tool for all calibrations.
- Better feature matching and robust estimation techniques specific to fisheye and omni-directional cameras, especially when considering auto-calibration.
- More flexibility for a user to incorporate his/her own method into the toolbox.
- Further extend to provide the robust external parameters for camera motion which could help with visual simultaneous localization and mapping and other robotic vision applications.
- Incorporate 3D reconstruction as part of the toolbox to provide sparse/dense depth map given a set of multi-view images.

- Incorporate bundle adjustment and other non-linear global optimizations to further refine the 3D reconstructed results.

- Make the fisheye 3D reconstruction real time for using in mobile and other real-time applications.

- Speed up implementation by developing more algorithms in C/C++ and interfacing with current toolbox.

- Further automate calibration procedures for user to have minimum interaction with the system.

- Develop web-based version of the toolbox for easier use and wider reach of the toolbox software.

- Extend to make a toolbox for calibration of any type of camera under one application.

REFERENCES

[1] D. Scaramuzza, A. Martinelli, and R. Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Proc. of The IEEE International Conference on Computer Vision Systems (ICVS)*, 2006.

[2] J. Kannala and S. S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1335–1340, 2006.

[3] J. Kannala, S. S. Brandt, and J. Heikkilä. Self-calibration of central cameras from point correspondences by minimizing angular error. In *Computer Vision and Computer Graphics. Theory and Applications - International Conference, VISIGRAPP 2008, Funchal-Madeira, Portugal, January 22-25, 2008. Revised Selected Papers*, pages 109–122, 2008.

[4] B. Micusík and T. Pajdla. Structure from motion with wide circular field of view cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28: 2006, 2006.

[5] X. Ying and Z. Hu. Catadioptric camera calibration using geometric invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:1260–1271, 2004.

[6] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:1330–1334, 1998.

[7] B. Triggs. Autocalibration from planar scenes. In *European Conference on Computer Vision*, 1998.

[8] R. Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, 3:323–344, 1987.

[9] J. Yves-Bouguet. Complete camera calibration toolbox for matlab, 2000 (Accessed December 2nd, 2013). URL `http://www.vision.caltech.edu/bouguetj/index.html`.

[10] J. Heikkilä. Geometric camera calibration using circular control points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:1066–1077, 2000.

[11] T. Svoboda and T. Pajdla. Epipolar geometry for central catadioptric cameras. *International Journal of Computer Vision*, 49:23–37, 2002.

[12] A. W. Fitzgibbon. Simultaneous linear estimation of multiple view geometry and lens distortion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 125–132, 2001.

[13] J. Kannala. Camera calibration toolbox for generic lenses, 2008 (Accessed March 3rd, 2013). URL `http://www.ee.oulu.fi/~jkannala/calibration/calibration.html`.

[14] D. Scaramuzza, A. Martinelli, and R. Siegwart. A toolbox for easy calibrating omnidirectional cameras. In *Proc. of The IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2006.

[15] H. Shum and R. Szeliski. Stereo reconstruction from multiperspective panoramas. In *Seventh International Conference on Computer Vision (ICCV'99)*. Institute of Electrical and Electronics Engineers, Inc., September 1999.

[16] S. M. Seitz, A. Kalai, and H. Shum. Omnivergent stereo. *International Journal of Computer Vision*, 48(3):159–172, 2002.

[17] E. Brassart, L. Delahoche, C. Cauchois, C. Drocourt, C. Pgard, and E. Mouaddib. Experimental results got with the omnidirectional vision sensor: Syclop. volume 00, Los Alamitos, CA, USA, 2000. IEEE Computer Society.

[18] C. Mei and P. Rives. Single view point omnidirectional camera calibration from planar grids. In *IEEE International Conference on Robotics and Automation*, Roma, Italy, April 2007.

[19] D. Xiao-Ming, W. Fu-Chao, and W. Yi-Hong. An easy calibration method for central catadioptric cameras. *ACTA AUTOMATICA SINICA*, 33:801–808, 2007.

[20] S. Gasparini, P. F. Sturm, and J. P. Barreto. Plane-based calibration of central catadioptric cameras. In *IEEE International Conference on Computer Vision*, pages 1195–1202, 2009.

[21] S. Shah and J. K. Aggarwal. A simple calibration procedure for fish-eye (high-distortion) lens camera. In *Proceedings of the 1994 International Conference on Robotics and Automation, San Diego, CA, USA, May 1994*, pages 3422–3427, 1994.

[22] L. Puig, Y. Bastanlar, P. Sturm, J. J. Guerrero, and J. Barreto. Calibration of central catadioptric cameras using a dlt-like approach. *International Journal of Computer Vision*, 93(1):101–114, March 2011.

[23] F. Devernay and O. Faugeras. Straight lines have to be straight: Automatic calibration and removal of distortion from scenes of structured enviroments. *Machine Vision Applications*, 13(1):14–24, August 2001. ISSN 0932-8092.

[24] C. Bräuer-Burchardt and K. Voss. A new algorithm to correct fish-eye- and strong wide-angle-lens-distortion from single images. In *IEEE International Conference on Image Processing*, pages 225–228, 2001.

[25] D. C. Brown. Decentering Distortion of Lenses. *Photometric Engineering*, 32(3): 444–462, 1966.

[26] A. Basu and S. Licardie. Alternative models for fish-eye lenses. *Pattern Recognition Letters*, 16(4):433–441, April 1995. ISSN 0167-8655.

[27] D. Scaramuzza. *Omnidirectional Vision: from Calibration to Robot Motion Estimation.* PhD thesis, Swiss Federal Institute of Technology Zurich (ETHZ), February 2008.

[28] B. Micusík and T. Pajdla. Estimation of omnidirectional camera model from epipolar geometry. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2003), 16-22 June 2003, Madison, WI, USA*, pages 485–490, 2003.

[29] J. J. Kumler and M. L. Bauer. Fish-eye lens designs and their relative performance. *Proceedings of SPIE Conference*, 4093:360–369, 2000.

[30] M. Rufli and D. Scaramuzza. Automatic detection of checkerboards on blurred and distorted images. In *2008 IEEE International conference on Robots and systems*, 2008.

[31] V. vezhnevets. Opencv calibration object detection, 2003 (last accessed 2005). URL `http://graphicon.ru/oldgr/en/research/calibration/opencv.html`.

[32] K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Applied Math.*, 2:164–168, 1944.

[33] D. W. Marquardt. An Algorithm for Least-Squares Estimation of Nonlinear Parameters. *SIAM Journal on Applied Mathematics*, 11:431–441, 1963.

[34] J. Kannala and S. S. Brandt. A generic camera calibration method for fish-eye lenses. In *17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23-26, 2004.*, pages 10–13, 2004.

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

[36] J. Oliensis. Exact two-image structure from motion. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 24(12):1618–1633, 2002.

[37] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004. ISSN 0920-5691.

[38] A. P. Witkin. Scale-space filtering. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'83, pages 1019–1022, San Francisco, CA, USA, 1983. Morgan Kaufmann Publishers Inc.

[39] J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, CVPR '97, pages 1000–, Washington, DC, USA, 1997. IEEE Computer Society. ISBN 0-8186-7822-4.

[40] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008(Last accessed Sep 2014). URL `http://www.vlfeat.org/`.

[41] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Trans. Graph.*, 25(3):835–846, July 2006. ISSN 0730-0301.

[42] Microsoft photosynth. URL `https://photosynth.net/`.

[43] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981. ISSN 0001-0782.