Camera Calibration using Adaptive Segmentation and Ellipse Fitting for Localizing Control Points

by

Charan Dudda Prakash

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

Approved August 2012 by the
Graduate Supervisory Committee:

Lina J. Karam, Chair
David Frakes
Antonia Papandreou - Suppappola

ARIZONA STATE UNIVERSITY

August 2012

ABSTRACT

There is a growing interest for improved high-accuracy camera calibration methods due to the increasing demand for 3D visual media in commercial markets. Camera calibration is used widely in the fields of computer vision, robotics and 3D reconstruction. Camera calibration is the first step for extracting 3D data from a 2D image. It plays a crucial role in computer vision and 3D reconstruction due to the fact that the accuracy of the reconstruction and 3D coordinate determination relies on the accuracy of the camera calibration to a great extent.

This thesis presents a novel camera calibration method using a circular calibration pattern. The disadvantages and issues with existing state-of-the-art methods are discussed and are overcome in this work. The implemented system consists of techniques of local adaptive segmentation, ellipse fitting, projection and optimization. Simulation results are presented to illustrate the performance of the proposed scheme. These results show that the proposed method reduces the error as compared to the state-of-the-art for high-resolution images, and that the proposed scheme is more robust to blur in the imaged calibration pattern.

ACKNOWLEDGEMENTS

I would like to acknowledge the precious advice and strong support of my thesis advisor Dr. Lina J. Karam without whom this thesis would not have been possible. I would like to extend my gratitude towards committee members Dr. David Frakes and Dr. Antonia Papandreou-Suppappola for agreeing to be on my committee. I would like to acknowledge the support offered by the labs of Intel as this thesis has been funded in part by Intel. I would especially like to mention Bonnie Bennett, Jeff Pettinato and Nital Patel from Intel for their time and support.

I would also like to thank the support provided by the members of the Image, Video and Usability (IVU) lab. I would like to thank my fellow students and special friends, Lavanya, Vikram, Parag, Jidnyasa and Ragav for all assistance and cooperation when it was most needed. Last but not the least, I would like to express my utmost gratitude to my parents and my dearest friends, especially Abhishek, Adarsh, Aditi, Apurva, Deepak, Komal, Pavan, Rama, Sangeetha, Rujuta and Vimala for their strong love, support, and for being there for me throughout the journey.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

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 image and video processing algorithms, there has been an increasing demand for 3D content in the form of computer graphics, virtual reality and 3D reconstruction. These interests are employed for the purposes of entertainment, computer vision, 3D simulations and communication.

Camera calibration plays a significant role in all these applications. It is the process in which the optical parameters of the camera (intrinsic) and/or the position of the camera (extrinsic), either absolute or with respect to the object of interest are extracted. 3D information is extracted by multiple images of the object of interest from different views using the parameters of the camera and the rotation and translation parameters between multiple cameras. All these parameters can be calculated using camera calibration. Hence, the extracted 3D information depends on camera calibration to a great extent.

Camera calibration determines the intrinsic and/or extrinsic parameters of a camera. The intrinsic parameters of a camera consist of the focal length in the x and y directions, principal point, skews parameters and lens distortion parameters. The extrinsic parameters of a camera include the rotation and translation of the object in world coordinate with respect to camera coordinates. The translation

parameters consist of the displacement of the world coordinate center from the camera coordinate center. The rotation parameters consist of three parameters that define the rotation of the world coordinate system with respect to the camera coordinate system with the camera center as the origin. Camera calibration can be performed by localizing the control points on the calibration pattern in its image(s).

Camera calibration using a calibration pattern is an algorithm which is employed initially once the setup has been installed. It is not run in a repetitive manner. Hence, speed is not of much concern. Accuracy of the camera calibration is verified by calculating the reprojection error of the control points. The known or the manufacturer-specified coordinates of certain feature points in the calibration object (actual coordinates) and the coordinates of the corresponding automatically detected feature points of the calibration object (estimated coordinates) in the image are individually projected to 2D pixel coordinates in the camera image plane using the estimated camera parameters. Then, the error, also known as reprojection error, between the 2D coordinates of the projected estimated coordinates and the 2D coordinates of the projected actual coordinates, is calculated. The smaller the reprojection error, the higher will be the accuracy of camera calibration. There are several existing camera calibration techniques [1] [2] [3] that produce fairly accurate results in terms of accuracy. Nevertheless, there is

a need for higher accuracies in camera calibration in the age of high resolution cameras especially for applications that require high precision.

1.2 Contributions

In this thesis, a novel camera calibration scheme consisting of an iterative refinement of camera parameters with an improved localization of control points based on local adaptive segmentation and ellipse fitting is presented. In the proposed method, the camera calibration parameters are refined in an iterative fashion starting from initial parameters that are estimated using the existing camera calibration methods. The captured images of the circular pattern henceforth known as original images and their planes of orientation referred to as camera planes are undistorted and projected onto a fronto parallel plane. The control points are localized in this fronto parallel plane using a novel approach for more accurately localizing the control points in the images based on adaptive segmentation and ellipse fitting. The coordinates of the boundary of the circles in the imaged calibration pattern are extracted using adaptive segmentation followed by edge detection. The adaptive segmentation is a histogram-based approach and produces improved segmentation under conditions of low contrast and in the presence of blur in the imaged calibration pattern due to defocusing. These coordinates are then input to an optimized ellipse fitting algorithm. The calculated centers of the fitted ellipses serve as control points. The proposed ellipse fitting algorithm combines two state-of-the-art algorithms and picks the ellipse with the

least error in terms of algebraic distance. The localized control points are then projected to their corresponding camera image planes using estimated camera parameters that are refined by minimizing the reprojection error.

1.3 Thesis Organization

This thesis is organized as follows: Chapter 2 provides the background required for camera calibration. Chapter 3 describes the previous work that is related to this thesis. Chapter 4 describes the proposed camera calibration method. Chapter 5 presents experimental results using images captured under different conditions of varying blur and pattern sizes. Comparisons with an existing state-of-the-art technique are also provided in this chapter. Chapter 6 summarizes the contributions of this thesis and proposes future directions of research.

2. BACKGROUND

This chapter gives some background knowledge in relation to the techniques and equations required for camera calibration. Section 2.1 illustrates the basic pinhole camera model. The different types of initialization of intrinsic parameters of the camera are described in Section 2.2. Section 2.3 explains the nonlinear lens distortions in a practical camera. Section 2.4 describes nonlinear optimization techniques for camera calibration.

2.1 Pinhole model of a camera

Fig. 1. illustrates the pinhole camera model. Using homogeneous coordinates, let a 2D point in an image plane be denoted by $\mathbf{x}=[x\ y\ 1]^T$ and the corresponding 3D point be denoted by $\mathbf{X}=[X\ Y\ Z\ 1]^T$. Using the pinhole camera model, the two are related as follows:

$$s\mathbf{x} = \mathbf{A}[\mathbf{R} \quad \mathbf{t}]\mathbf{X} \qquad \text{with } \mathbf{A} = \begin{bmatrix} f_x & \gamma & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where s is an arbitrary scaling factor, **R, t** are extrinsic parameters and **A** is called the camera intrinsic matrix. **A** 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). Modern day cameras are well manufactured such that skew can be assumed to be zero. **t** is a 3×1 translation matrix containing the displacement in the x, y and z directions, respectively. **R** is a 3×3 rotation matrix which is defined by three angles ω, φ and ψ along the x, y, and z axes respectively using Rodrigues transform [4].

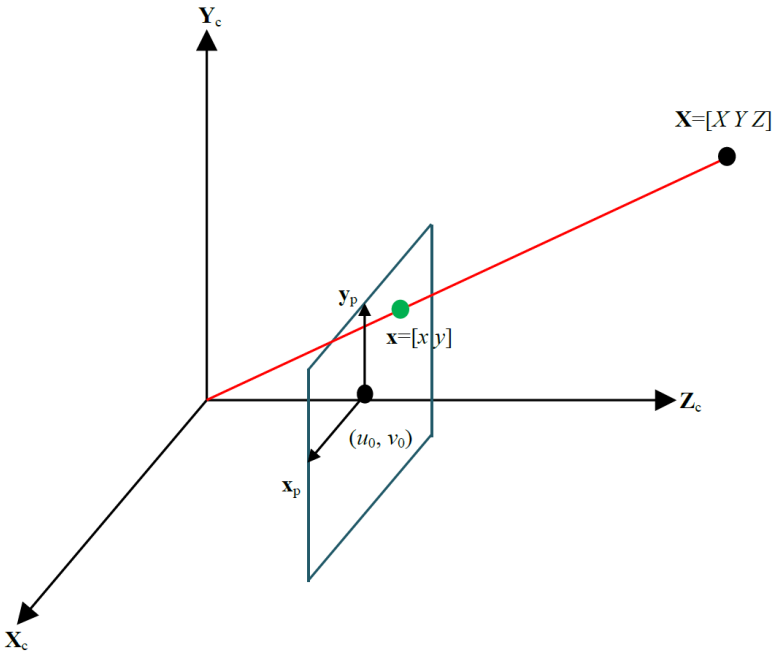


Fig. 1 Pinhole camera model.

The rotation matrix can also be stored as a vector of size 3×1 using Rodrigues' transform [4] [5] in order to save storage space. Rodrigues' transform transforms a 3×3 rotation matrix into a 3×1 vector and vice versa. The three angles, ω, φ and ψ are the angles of rotation around the x, y and z axes, respectively, in a specific predefined manner. The rotations are performed as follows: first, the coordinate system is rotated around the x axis. This rotated system is then rotated around the y axis and, finally, the system that had been rotated twice previously is rotated around the z axis. The parameters of **R** can be defined as [1]:

$$\mathbf{R} = \begin{bmatrix} cos\varphi\ cos\psi & sin\omega\ sin\varphi\ cos\psi - cos\omega\ sin\psi & cos\omega\ sin\varphi\ sin\psi + sin\omega\ sin\psi \\ cos\varphi\ sin\psi & sin\omega\ sin\varphi\ sin\psi + cos\omega\ cos\psi & cos\omega\ sin\varphi\ sin\psi - sin\omega\ cos\psi \\ -sin\varphi & sin\omega\ cos\varphi & cos\omega\ cos\varphi \end{bmatrix} \quad (2)$$

If a 2D planar pattern is used for camera calibration, it can be assumed that the pattern lies in the plane Z=0 without loss of generality. Hence, by convention, each camera plane is assumed to be in Z=0 plane with the origin at the control point on the left top. The 3D coordinates of other control points can be determined accordingly as the distance of separation of the control points on the calibration pattern is known. Thus, if **R** is represented as $[\mathbf{r_1}\ \mathbf{r_2}\ \mathbf{r_3}]$, where $\mathbf{r_1}$, $\mathbf{r_2}$, and $\mathbf{r_3}$ are each column vectors, then (1) can be rewritten as:

$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{A}[\mathbf{r_1} \quad \mathbf{r_2} \quad \mathbf{r_3} \quad \mathbf{t}]\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = \mathbf{A}[\mathbf{r_1} \quad \mathbf{r_2} \quad \mathbf{t}]\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \mathbf{H}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \qquad (3)$$

where $\mathbf{H} = [\mathbf{h_1}\ \mathbf{h_2}\ \mathbf{h_3}]$ is called the homography matrix. As it can be seen, **H** can be defined up to a scale factor. Given a set of 3D coordinates and their corresponding 2D pixel coordinates, the homography matrix can be determined by using a technique of maximum likelihood criterion [2].

Since each 2D image point location is defined using an *x* and *y* coordinate, it can be seen that *N* images each with *K* control points provide us with 2*KN* equations or constraints. If the skew, γ, is assumed to be zero, then we have 4 intrinsic parameters and 6 extrinsic parameters (3×1 rotation vector and 3×1 translation vector) summing up to a total of 10 unknowns. The 6 extrinsic parameters vary for each view whereas the 4 intrinsic parameters remain constant in all views Therefore, solving for these equations imposes the condition that the total number of constraints should satisfy:

$$2KN \geq 6N+4 \quad (4)$$

Hence, from (4), it might seem that, for a calibration object having 5 or more control points, only one image would suffice to calculate all the parameters of camera calibration. However, each plane is only four control points' worth of information as four points are sufficient to define a plane in space [5]. A larger number of control points is used to obtain an over-sampled system in order to get more accurate results. Hence, by substituting a maximum value of 4 for K in (4), one arrives at the condition $N>1$. Therefore, at least two images of a calibration target are needed with a minimum of four control points to determine the camera parameters. However, in order to obtain a higher accuracy, an excess number of constraints are typically used so as to find the solution with minimum error. Therefore, a larger number of images with a number of control points that is greater than four is typically used to solve for camera parameters.

2.2 Solution to linear parameters of a camera using Zhang's method [2]

It can be shown that [2]:

$$\mathbf{h}_1^{\mathrm{T}}\mathbf{A}^{-\mathrm{T}}\mathbf{A}^{-1}\mathbf{h}_2 = \mathbf{0} \quad \text{and} \quad \mathbf{h}_1^{\mathrm{T}}\mathbf{A}^{-\mathrm{T}}\mathbf{A}^{-1}\mathbf{h}_1 = \mathbf{h}_2^{\mathrm{T}}\mathbf{A}^{-\mathrm{T}}\mathbf{A}^{-1}\mathbf{h}_2 \quad (5)$$

where $\mathbf{A}$ is a 3×3 matrix containing the camera intrinsic parameters as defined in (1), and $\mathbf{h_1}$ and $\mathbf{h_2}$ are the first two column vectors of the homography matrix $\mathbf{H}$ as defined in (3).

Let $\mathbf{B} = \mathbf{A}^{-\mathrm{T}}\mathbf{A}^{-1}$. Then,

$$\mathbf{B} = \mathbf{A}^{-\mathrm{T}}\mathbf{A}^{-1} = \begin{bmatrix} \boldsymbol{B_{11}} & \boldsymbol{B_{12}} & \boldsymbol{B_{13}} \\ \boldsymbol{B_{21}} & \boldsymbol{B_{22}} & \boldsymbol{B_{23}} \\ \boldsymbol{B_{31}} & \boldsymbol{B_{32}} & \boldsymbol{B_{33}} \end{bmatrix} = \begin{bmatrix} \frac{1}{{f_x}^2} & -\frac{\gamma}{{f_x}^2 f_y} & \frac{v_0\gamma - u_0 f_y}{{f_x}^2 f_y} \\ -\frac{\gamma}{{f_x}^2 f_y} & \frac{\gamma^2}{{f_x}^2 {f_y}^2} + \frac{1}{{f_y}^2} & \frac{\gamma(v_0\gamma - u_0 f_y)}{{f_x}^2 {f_y}^2} - \frac{v_0}{{f_y}^2} \\ \frac{v_0\gamma - u_0 f_y}{{f_x}^2 f_y} & \frac{\gamma(v_0\gamma - u_0 f_y)}{{f_x}^2 {f_y}^2} - \frac{v_0}{{f_y}^2} & \frac{(v_0\gamma - u_0 f_y)^2}{{f_x}^2 {f_y}^2} + \frac{{v_0}^2}{{f_y}^2} + 1 \end{bmatrix} \quad (6)$$

It can be noticed that **B** is a symmetric matrix. Hence, it can be perfectly evaluated by the determination of a 6D vector given by $\mathbf{b} = [B_{11}\, B_{12}\, B_{22}\, B_{13}\, B_{23}\, B_{33}]^{\mathrm{T}}$. If the ith column vector of the homography matrix **H** is split into $\mathbf{h}_i = [h_{i1}\, h_{i2}\, h_{i3}]$, one obtains:

$$\mathbf{h}_i^{\mathrm{T}}\mathbf{B}\mathbf{h}_j = \mathbf{v}_{ij}^{\mathrm{T}}\mathbf{b} \quad (7)$$

where

$$\mathbf{v}_{ij} = [h_{i1}h_{j1} \quad h_{i1}h_{j2} + h_{i2}h_{j1} \quad h_{i2}h_{j2} \quad h_{i3}h_{j1} + h_{i1}h_{j3} \quad h_{i3}h_{j2} + h_{i2}h_{j3} \quad h_{i3}h_{j3}]^{\mathrm{T}}$$

Therefore, from (5) and (7), it follows that:

$$\begin{bmatrix} \mathbf{v}_{12}^{\mathrm{T}} \\ (\mathbf{v}_{11} - \mathbf{v}_{22})^{\mathrm{T}} \end{bmatrix}\mathbf{b} = \mathbf{0} \quad (8)$$

If there are N images corresponding to N different views of the calibration pattern, this results in $2N$ equations and we need to solve for 6 intrinsic parameters of **b**. Therefore, there is a need for $N \geq 3$ images to find the intrinsic parameters using the set of closed-form equations given by (8). If only two images are present, one can impose a skewless constraint on the image [2], i.e., the two axes of the image need to be perpendicular. However, to get more accurate results a larger number ($N \geq 3$) of images are used to determine the intrinsic parameters of the camera.

Once the system of equations (8) is solved for **b,** then the intrinsic parameters can be calculated as [2]:

$$v_0 = (B_{12}B_{13} - B_{11}B_{23})/\left(B_{11}B_{22} - {B_{12}}^2\right)$$

$$\lambda = B_{33} - [B_{13}^2 + v_0(B_{12}B_{13} - B_{11}B_{23})]/B_{11}$$

$$f_x = \sqrt{\lambda/B_{11}}$$

$$f_y = \sqrt{\lambda B_{11}/\left(B_{11}B_{22} - {B_{12}}^2\right)}$$

$$\gamma = -B_{12}f_x^2 f_y/\lambda$$

$$u_0 = \gamma v_0/f_x - B_{13}f_x^2/\lambda \tag{9}$$

The rotation and translation vectors can be obtained as follows [2]:

$$\mathbf{r_1} = \lambda\mathbf{A}^{-1}\mathbf{h}_1$$

$$\mathbf{r_2} = \lambda\mathbf{A}^{-1}\mathbf{h}_2$$

$$\mathbf{r_3} = \mathbf{r}_1 \times \mathbf{r}_2$$

$$\mathbf{t} = \lambda\mathbf{A}^{-1}\mathbf{h}_3 \tag{10}$$

However, due to errors, the rotation matrix **R**, obtained henceforth would not be orthogonal which is typically required for a rotation matrix, that is $\mathbf{RR}^T=\mathbf{I}$. Hence, to obtain an orthogonal matrix, the singular value decomposition (SVD) of **R** is computed. SVD is a process of factoring a matrix into two orthogonal matrices **U** and **V**, and a diagonal matrix **D**. In the ideal case, **D** would be an identity matrix **I**. Therefore, after SVD, the rotation matrix **R** is calculated as $\mathbf{R} = \mathbf{UIV}^T$.

2.3 Non linear parameters of lens distortion

The methods described above would obtain all the linear parameters of the camera. However, the practical camera is not an ideal pinhole camera; there are a few more intrinsic parameters due to practical issues that arise from the optical lens employed in a camera. These parameters are called lens distortion parameters and can be classified into radial and tangential distortion parameters.

The radial distortion is 0 at the camera optical center and increases symmetrically on either direction as one moves outwards. Rays farther from the center are bent more than the ones that are closer to the center. Hence, radial distortion is more observable towards the edges of the image. Straight lines near the edges of the image appear to be bent due to radial distortion. This bulging phenomenon is termed as barrel or fish eye effect. Radial distortion can be expressed in the form of Taylor series and the first two terms of the series are typically used to approximate the radial distortion phenomenon [5]. For highly distorted cameras, the third term is also considered [5]. The pixels' coordinates of the image (*x, y*) will be distorted according to:

$$x_d = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_d = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \quad (11)$$

The second type of lens distortion is called the tangential distortion. During the manufacturing of the camera, the lens might not be placed exactly parallel to

the imager. Hence, this manufacturing defect usually causes a tangential lens distortion. The coordinates of the pixels are distorted according to [6]:

$$x_d = x + (2p_1 y + p_2(r^2 + 2x^2))$$
$$y_d = y + (p_1(r^2 + 2y^2) + 2p_2 x) \quad (12)$$

The lens distortion parameters are not linear and cannot be estimated in the form of a closed form solution. Solving for distortion parameters is performed using iterative algorithms. But, if the intrinsic parameters are not properly initialized, there is a danger of the optimization technique sticking to a local minimum. Therefore, to solve this problem, in [1], the intrinsic parameters are solved for using closed form equations assuming that there are no distortion parameters present in the model. These values will act as initial values for the intrinsic parameters. The values of the intrinsic parameters are then optimized using a nonlinear optimization technique such as the Levenberg-Marquardt algorithm [1] [2] to minimize the reprojection error.

2.4 Non linear optimization of camera parameters

Once the camera parameters are initialized assuming that there are no distortion parameters present in the model, all the parameters including the distortion parameters can be optimized simultaneously using an iterative nonlinear optimization [1] [2].

Let $\mathbf{k_c} = [k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3]$. If there are a total of N images with K control points in each image then the error function,

$$E(\mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, \mathbf{k_c}) = \sum_{i=1}^{N} \sum_{j=1}^{K} \left\| \boldsymbol{m_{ij}} - \boldsymbol{m'_{ij}} \right\|^2 \quad (13)$$

is minimized in an iterative fashion using a nonlinear optimization technique such as the Levenberg-Marquardt algorithm. In (13), $\mathbf{m}_{ij}$ are the pixel coordinates in the camera plane of the estimated control points, and $\mathbf{m}'_{ij} = f\left(\mathbf{X}_{ij}, \mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, \mathbf{k}_c\right)$ are the coordinates that are obtained from the projection of the known (based on manufacturer's specification of the spacing between the center of the patterns) 3D coordinates $\mathbf{X}_{ij} = [X_{ij}, Y_{ij}, Z_{ij} = 0]$, of the control points using the camera parameters that are obtained from the previous iterations.

3. RELATED WORK

This chapter summarizes the existing work that is related to camera calibration techniques. Section 3.1 describes several popular methods that are used in auto calibration. Section 3.2 summarizes popular methods that are employed in classic camera calibration using calibration patterns.

Auto-calibration is a process of determining the intrinsic parameters of a camera directly from the multiple uncalibrated images taken from different views from the same camera. Such a calibration does not require images of any special calibrated objects in the image. The scene required for 3D reconstruction is captured from different views and the camera parameters are determined also from the same images.

Classic camera calibration is the process of determining the parameters of the camera with the help of a known special calibrated object in the image. Such calibration object can be either a rigid 3D or a 2D object of known geometric specifications. Most 3D calibration objects involve a cube or a cuboid with specific geometric patterns separated by known distances [1] [7]. 2D calibration objects contain geometric patterns on a plane surface. 2D calibration objects are more popular and widely used because of their simplicity [2] [3].

Few of the common patterns used for calibration objects are alternate blocks of black and white squares (popularly known as checkerboard), circles and rings as shown in Fig. 2. Ring patterns consist of concentric circles [3]. Certain features on these patterns are employed for determining the parameters of the camera.

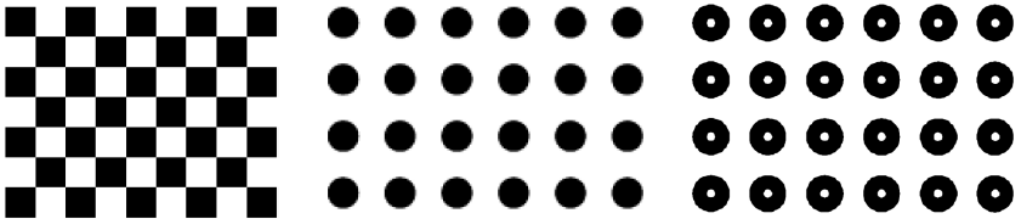


Fig. 2. Various 2D calibration patterns: Checkerboard (Left), Circle (Center), and Ring (Right).

These feature points are also known as control points. Some examples of control points are corners of squares in the checkerboard, centers of circles and centers of ring patterns. If the displacements between the control points on the calibration object are known with respect to a reference coordinate system and these control points are localized accurately in the camera image plane, this information can be used to determine the parameters of the camera.

3.1 Auto-calibration

Faugeras, Luong and Maybank [8] introduced auto-calibration to camera calibration. It is historically known as the first auto-calibration method and employs Kruppa equations [9] to solve for camera parameters. Seo and Heyden [10] presented an iterative algorithm using only linear equations and bundle adjustment by introducing an additional orthogonality constraint. Further, Seo, Heyden and Cipolla also presented a robust algorithm [11] where the algorithm is a hybrid between of a nonlinear optimization and initial linear computation. The most significant property of the algorithm of [11] is the fact that this algorithm is

general in the sense that any type of constraint can be imposed on the initial parameters of the camera.

Sengupta and Das [12] presented a modified algorithm for auto-calibration. If the dual of the image of the absolute conic is not positive definite, metric reconstruction is not possible. Hence, the authors modify the computation of the dual of the image of the absolute conic by approximating its covariance square root guaranteeing its positive definiteness. The results were found to be quite satisfactory in terms of non-ambiguous reconstruction [12]. There have been attempts to use some specific features in the scene for auto-calibration. Pflugfelder and Bischof [13] proposed an auto-calibration method for a stationary camera. The method uses a sequence of images captured from the camera and estimates certain points called vanishing points using the line segments in the image. Kuo, Nebel and Makris [14] used the biomechanical constraints of the human body. The algorithm analyses certain key points on the human body during a sequence and detects frames where the body adopts a specific posture which allows for accurate camera calibration.

However, when the camera is static and is used in an environment where a calibration object can be used such as in factories, laboratories, and indoor environment, it is favorable to use camera calibration with the aid of a specific calibration object as it is more accurate and provides more accurate reconstruction results.

3.2 Camera calibration with a calibration object

Sobel [15] introduced a method for solving for the camera parameters using a complex system of nonlinear equations. Both intrinsic and extrinsic parameters of the camera were solved for in his method. However, he assumed a pinhole model of a camera and hence did not model lens distortions. Further, the method required the user to provide the algorithm with initial values of the parameters for the optimization.

Tsai [16] improved on the approach and modeled the lens distortions for the system. His algorithm also provided a method for estimating initial parameters. His method first converts 3D world coordinates into 3D camera coordinates using the rotation and translation matrices that are obtained by introducing inhomogeneity in the system of equations and assuming the magnitude of each vector in the rotation matrix to be unity. The algorithm then calculates the 2D image coordinates using the estimated intrinsic parameters. The model of lens distortion assumes that the distortions are radially symmetric. Gremban, Thorpe and Kanade [17] implemented an algorithm to perform camera calibration using a system consisting only of linear equations and without explicitly calculating lens distortions.

Wei, He and Ma [18] presented a camera calibration method where calibration was performed with or without modeling for lens distortions. The initial parameters were solved for using a linear system of equations. Radial distortions

were considered through refinements made by the invariance properties of cross ratio and collinearity of perspectivity. The algorithm would then use linear methods again to solve for extrinsic parameters. Chatterjee and Roychowdury [19] presented a robust algorithm for camera calibration and proved analytically that if all the camera parameters are solved for using the method of least squares, there would be no error in the computation of the image or world coordinates if there is an error in the image center displacement or the scaling factor, *s*.

In [1], Heikkilä and Silvén proposed that, for a circular calibration pattern, the center of the detected ellipse in the image may not be the actual center due to rotation and translation and they introduced a correction for asymmetric projection. Later, in [20], Heikkilä proposed an improved procedure for camera calibration including procedures for calibrating both the forward and reverse camera models. The forward camera model computes the parameters by minimizing the error between the current 2D pixel coordinates and the projected 2D pixel coordinates that are obtained from the projection of the 3D coordinates onto the camera image plane using the estimated calibration parameters. The reverse camera model computes the parameters by minimizing the error between known 3D coordinates and the 3D coordinates that are obtained from the back projection of the 2D pixel coordinates onto the 3D coordinates using the estimated calibration parameters.

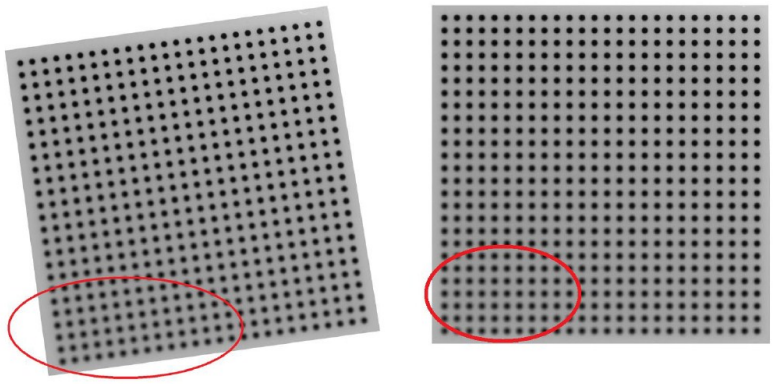


Fig. 3. Drawback of the method of [3]. (Left) Blur in the camera plane (highlighted by the ellipse). (Right) Blur propagated in the fronto parallel plane (highlighted by the ellipse).

Several camera calibration methods were also developed using design of experiments to get more accurate results [21] [22]. However, these algorithms involve complex hardware and experimental set up.

One popular camera calibration method is provided by the *OpenCV Camera Calibration Toolbox* [5] based on Zhang's method [2]. A major improvement over this method was achieved using a scheme proposed by Datta *et al.* [3], which produced 50% better results in terms of a smaller reprojection error than that of the OpenCV method [5] [2]. Datta *et al.* observed that calibration results improve significantly when control points are localized in the fronto parallel view with respect to the camera coordinate system. The scheme of [3] performs camera calibration in a number of iterations. In the first iteration, the camera parameters are calculated as in [2] [5]. The control points in the image of the calibration pattern are localized using curve fitting *only initially,* in the first iteration [23]. In

the following iteration, each view in the image is undistorted and then unprojected onto a canonical fronto parallel plane view using boundary detection of the area containing the calibration pattern and interpolation. The control points in these fronto parallel views are localized using template matching with a blurred circular template followed by quadratic fitting in the neighborhood pixel in order to obtain sub-pixel accuracy. These control points are then projected onto their camera plane using camera parameters from the previous iteration. This is then followed by the refinement of the camera calibration parameters using the Levenberg-Marquardt algorithm to minimize the reprojection error.

However, when each view is unprojected onto the fronto parallel plane using approximate camera parameters from the previous iterations, it is assumed in [3] that each view is converted to a perfect fronto parallel plane and, hence, the control points can be localized by template matching using a blurred circular template. But, in practice, the conversion is approximately fronto parallel as the camera parameters are approximate. Hence, each circle will actually be an ellipse due to a slight angular projection. Therefore, matching an ellipse with a circular template does not provide optimum results.

Also, another issue with the scheme of [3] is that it is not resilient to blur in the imaged circular pattern. Blur can occur, for example, in applications requiring high calibration precision. For such applications, relatively small circles need to be used as calibration patterns, say to tune for a precision in the fraction of

millimeters, and the images are typically captured from a close distance in order to capture the small circular patterns. This can result in blurred circles in the captured images due to focusing problems as shown in Fig. 3. Even if the images are not blurred by the acquisition process, reprojection of the captured images onto fronto parallel planes involves interpolation which causes a blurred effect. In the case of an initial blur in the image, the problem is aggravated. In these cases, different circles in the same image are blurred to different extents and using the same circular template for all the circles might produce erroneous results. Our proposed algorithm overcomes these drawbacks and results in more accurate estimates of the camera calibration parameters, especially for high-precision visual applications requiring calibration patterns consisting of small-size patterns (in the millimeter and nanometer range) in order to limit the reconstruction errors to the desired small ranges.

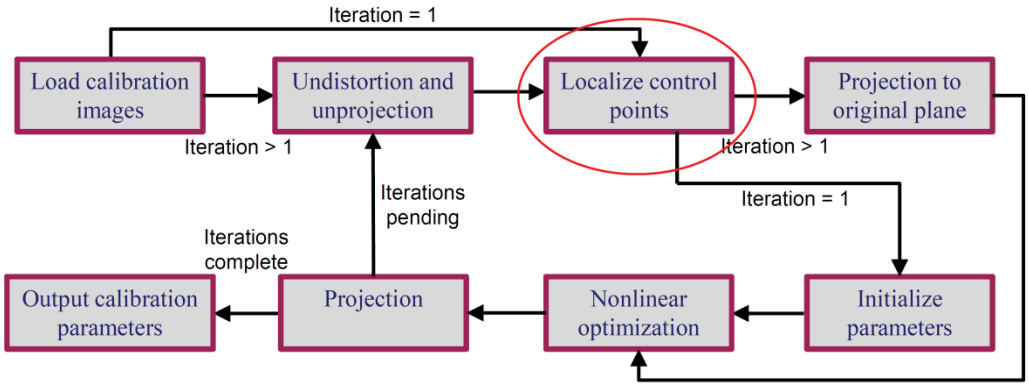
4. PROPOSED CAMERA CALIBRATION ALGORITHM

This chapter describes the proposed camera calibration method. Section 4.1 presents an overview of the proposed camera calibration method. Details about the main components of the proposed camera calibration method are described in Sections 4.2 to 4.6. Section 4.7 describes the calculations of the reprojection error and uncertainty.

4.1 Overview of the proposed camera calibration method

Fig. 4 shows the block diagram of the proposed camera calibration method. There are five main steps in the calibration algorithm: localization of control points in the images, initialization of intrinsic parameters, nonlinear optimization of all camera parameters, undistortion and unprojection of images, and projection of 3D points on the camera image plane. The major contribution of this thesis lies in the localization of control points which is highlighted in Fig. 4(a). A block diagram for the proposed localization of control points is given in Fig. 4 (b).

The proposed camera calibration algorithm requires, as input, images corresponding each to a different view of a 2D circular pattern. All the input images of the calibration pattern at different views are read into the memory. For each image, the user is asked to select a bounding box that encapsulates the region of the image containing the calibration pattern in a quadrilateral in that image. For this purpose, the user is asked to select four points on the image starting from the top left of the image in the clockwise direction. This procedure is performed on all the calibration images.



(a)

(b)

Fig.4. Flowchart of the proposed algorithm. (a) Block diagram of the proposed camera calibration algorithm. (b) Block diagram of the proposed procedure for localizing control points.

The next step of the algorithm is based on the iteration under execution. In the first iteration, the selection of the bounding box is followed by the automatic localization of control points. In the subsequent iterations, the calibration images are first undistorted and unprojected onto a canonical fronto parallel plane and this is followed by the localization of control points.

In the first iteration, the localized control points in all images are used to initialize the intrinsic parameters of the camera by the method of planar vanishing points [24]. All parameters including the nonlinear lens distortion parameters are

then refined and estimated by nonlinear optimization using the Levenberg-Marquardt (LM) algorithm [1] [2].

At each iteration, the estimated camera parameters are used to project the actual (known, manufacturer-specified) and estimated (localized) coordinates of the control points using the camera intrinsic parameters and the extrinsic parameters corresponding to each view. The accuracy is determined by computing the differences between the projected coordinates of the localized control points and those of the actual control points in both the x and y directions in all images, and their mean squared error is calculated as in (13). The procedure stops in the second iteration if the desired accuracy is reached after optimization using the LM algorithm or if the maximum allowed number of LM iterations is reached.

4.2 Localizing control points

The proposed algorithm localizes control points in the calibration images during the first iteration as in [3] by calculating, for each circle in the imaged circular pattern, the mean of all points contained in a considered circle. This mean is taken to be an initial estimate of the center of that circle. Though it is not an accurate localization, it serves well as an initial guess.

In order to locate the circles in the calibration pattern at the first iteration, the bounding box of each image is divided such that each segment consists of a circular pattern. Each segment is thresholded into a binary image with a threshold of 0.5 times the mean intensity in the segment as in the implementation of the

code obtained from the authors of [3]. The mean of the locations of all the black pixels in the binary image is taken to be the control point for that circle.

In the subsequent iteration, the calibration images are first undistorted and unprojected onto a canonical fronto parallel plane using the values of the camera parameters from the previous iteration. The undistorted and unprojected fronto parallel plane images are represented using a discrete grid of points consisting of 40 samples per d in each of the x and y direction, where d is the spacing between the centers of the circular patterns. Consequently, each circular pattern is represented using a region of 40×40 pixels. In other words, after the camera plane has been converted to a fronto parallel plane, each 40×40 pixel area in the unprojected image encapsulates a single circle and hence a single control point. This eases our control point localization task by narrowing the search area. The control points are then localized by converting each unprojected image of the fronto parallel plane into a binary image using adaptive thresholding followed by ellipse fitting.

The proposed adaptive thresholding of the image brings an advantage of removing outlier pixels that are introduced due to blur and focusing problems which lead to incorrect calculations of the control points. The boundary of each segmented circle is extracted using the Canny edge detector, and ellipse fitting is performed on these extracted boundaries to localize the control points in the fronto parallel plane. The control points are then projected onto the camera plane

using the calibration parameters from the previous iteration. More details about the adaptive thresholding and ellipse fitting components of the proposed scheme are presented below.

4.2.1 Adaptive thresholding

Various algorithms exist for the selection of an adaptive threshold for a grayscale image. However, most existing algorithms typically select an adaptive threshold for the entire image and, since the considered image consists of significant bright regions (background) and dark regions (circles), the selected threshold would separate the background from the foreground but not blur in the image. One of the efficient methods for segmentation in the presence of low contrast is the local adaptive segmentation scheme of [25]. But, this latter algorithm yields best results when a local region of interest in the image, which has low contrast, is chosen. The algorithm also involves a manual scaling of the threshold which is a drawback for automated calculations. This algorithm is modified in this work to provide a segmentation threshold that best segments the pixels that belong to the circle.

The proposed algorithm can be summarized as follows [26]:

Step 1: Calculate the average intensity μ of the considered image and round it off to the nearest integer.

Step 2: Determine the histogram of the entire image using M bins for an image with M gray levels (M=256 in our case).

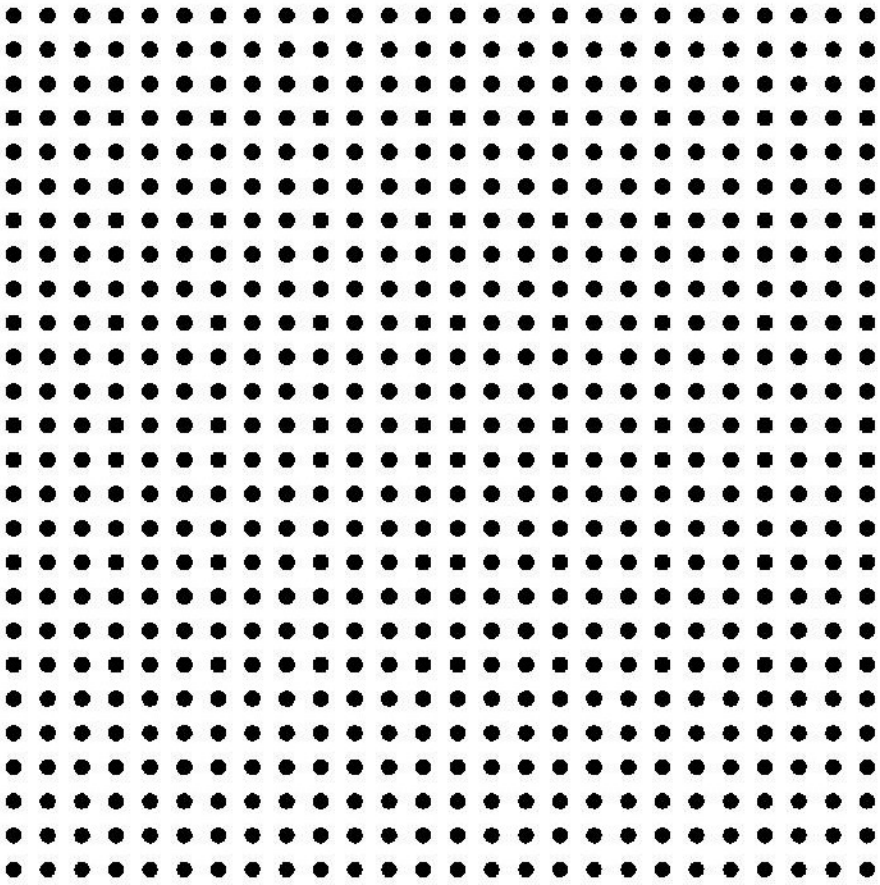


Fig. 5. Fronto parallel plane image in Fig. 3 converted into a binary image using adaptive thresholding.

Step 3: Extract the first μ bins and discard the remainder of the histogram. This modified histogram with only μ bins is used in the subsequent steps.

Step 4: Calculate the average intensity μ' of all the pixel intensities in the modified histogram. This serves as the initial threshold *T*.

Step 5: Calculate the means $\mu 1$ and $\mu 2$ of the pixels below and above the threshold *T*, respectively.

Step 7: The threshold *T* is updated to be the mean of $\mu 1$ and $\mu 2$.

Step 8: Repeat from Step 5 until the value of the threshold T is no longer changing as compared to its value from the previous iteration.

This procedure is performed on an interpolated image by a factor 4 so as to obtain sub-pixel accuracy. Fig. 5 shows the resulting binary image of the unprojected fronto parallel plane image of Fig. 3.

4.2.2 Ellipse fitting

Ellipse fitting takes in a set of input points, typically along a boundary or closed curve, and determines all the parameters of an ellipse that best fits the input set of points.

An ellipse belongs to the class of conics. A conic is a curve that is obtained as the intersection of a cone with a plane. Let $F(\mathbf{a},\mathbf{p})$ represent a two-dimensional second-order polynomial given by:

$$F(\mathbf{a}, \mathbf{p}) = \mathbf{a}.\mathbf{p} = ax^2 + bxy + cy^2 + dx + ey + f \qquad (14)$$

where $\mathbf{a} = [a\ b\ c\ d\ e\ f]$ and $\mathbf{p} = [x^2\ xy\ y^2\ x\ y\ 1]^{\mathrm{T}}$. A point (x,y) on the conic satisfies the equation $F(\mathbf{a},\mathbf{p}) = 0$. $F(\mathbf{a},\mathbf{p})$ is known as the algebraic distance of the point (x,y) to the conic $F(\mathbf{a},\mathbf{p})=0$ [23]. To reinforce the conic to be an ellipse, the conic parameters, $\mathbf{a}$, need to satisfy the constraint $4ac - b^2 = 1$ [27]. Let J be the number of points to be fitted into an ellipse. Ellipse fitting can thus be performed by finding the parameters of the conic, $\mathbf{a}$, that minimize the following sum of algebraic distances [23]:

$$\sigma = \sum_{j=1}^{J} F(\mathbf{a}, \mathbf{p_j}) \qquad (15)$$

subject to the constraint $\mathbf{a}^T\mathbf{C}\mathbf{a}= \mathbf{1}$, where C is a constraint matrix given by

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{16}$$

In [23], Fitsgibbon, Pilu and Fisher proposed an algorithm to determine the parameters of the ellipse for a set of data points, $\mathbf{p}_j$, $j=1,...,J$. A design matrix D is constructed as $\mathbf{D} = [\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3\ \mathbf{p}_4 \ldots..\ \mathbf{p}_J]^T$ and its scatter matrix is obtained by $\mathbf{S} = \mathbf{D}^T\mathbf{D}$. The square matrices **S** and **C** are used to obtain a diagonal matrix **geval** whose diagonal elements represent generalized eigenvalues, and a full matrix **gevec** whose columns are the corresponding eigenvectors such that **S**(**gevec**) = **C**(**gevec**)(**geval**). The parameter vector **a** is given by the negative values of **geval.** Once the parameter vector, **a**, is obtained. The center of the ellipse is calculated from the parameters as follows:

$$x_{center} = 2cd - be/(b^2 - 4ac)$$

$$y_{center} = 2ae - bd/(b^2 - 4ac) \tag{17}$$

Maini [28] showed that the scatter matrix can be ill conditioned in the case of high-resolution cameras with typical values in the range of mega pixels, which would prevent solving for the eigenvalues.. Hence, he modified the scatter matrix by scaling and recentering the input data points. The input data points are shifted and scaled so as to be transformed into an ellipse located at the origin with the

points at the maximum distance of 1 from the center according to the following normalizing equations:

$$x_m = min_{j=1}^{J}(x_i) \quad \text{and} \quad y_m = min_{j=1}^{J}(y_i)$$

$$s_x = max_{j=1}^{J}(x_i) - min_{j=1}^{J}(x_i)/2 \quad \text{and} \quad s_y = max_{j=1}^{J}(y_i) - min_{j=1}^{J}(y_i)/2$$

$$x_i' = x_i - x_m/s_x - 1 \quad \text{and} \quad y_i' = y_i - y_m/s_y - 1 \tag{18}$$

Ellipse fitting is performed on the normalized data points (x'_i, y'_i) as described above and the obtained parameter vector **a'** of the ellipse is denormalized to fit the original input data points according to [28]:

$$K_x = x_m + s_x$$

$$K_y = y_m + s_y$$

$$a = a' s_y^2$$

$$b = b' s_x s_y$$

$$c = c' s_x^2$$

$$d = d' s_x s_y^2 - b' s_x s_y K_x - 2a' s_y^2 K_y$$

$$e = e' s_x^2 s_y - b' s_x s_y K_x - 2c' s_x^2 K_y$$

$$f = a' s_y^2 K_x^2 + b' s_x s_y K_x K_y + c' s_x^2 K_y^2 - d' s_x s_y^2 - e' s_x^2 s_y K_y + f' s_x^2 s_y^2 \tag{19}$$

The aforementioned methods [23, 28], however, suffer from numerical instability when the data points lie exactly on the ellipse [29]. Hence, in [29], a perturbing function, e.g., a sinusoidal function, is used to intentionally perturb the points. The input data points are first recentered and shifted using the normalization equations. The normalized coordinates (x_i', y_i') , are converted into

polar coordinates (ρ_i', θ_i'). The perturbing functions are applied to these polar coordinates according to:

$$\rho_i'' = \rho_i' + Asin(2\pi f \theta_i') \tag{20}$$

where A and f are, respectively, the amplitude and frequency of the sinusoidal perturbation function

These coordinates are remapped to Cartesian coordinates and the ellipse is fitted to the modified data points. The resulting ellipse parameters are denormalized according to (19) in order to obtain the final ellipse parameters.

Table.1 Sum of algebraic distances for each method after ellipse fitting.

Center/radius	**Only least squares [23]**	**Least squares with rescaling [28]**	**Least squares with rescaling and perturbation [29]**
[0,0]/1	3.1678e-030	1.9845e-030	1.9845e-030
[0,25]/8	-	2.8843e-030	3.8341e-009
[50,50]/85	4.0218e-027	-	3.1062e-006
[40,40]/13	-	-	9.1605e-010
[40,40]/16	-	1.0231e-030	2.2280e-009
[40,40]/17	-	-	2.9042e-009
[40,40]/18	-	1.1217e-030	3.7397e-009
[40,40]/20	1.0724e-030	-	6.0135e-009
[20,30]/5	-	-	1.1328e-010

However, as part of this work, it was observed that the technique used in [28] produces less error when it converges to a solution as compared to that in [29], as the scheme in [29] employs a perturbing function to avoid instability. Table 1 shows the accuracy of ellipse fitting for the three aforementioned methods [23, 28, 29] for a given center and circle radius. A set of points on the circle of a given center and radius, spaced at 5 degrees apart from each other, were generated and the methods of [23], [28], and [29] were used to fit an ellipse to the data. From Table 1, it can be seen that, while both the methods of [28] and [29] result in a higher error as compared to [23], the method of [29] results in a higher error as compared to [28] but is the most stable among the three methods.

The proposed algorithm uses a decision-based ellipse fitting which combines the advantages of the methods of [23] and [29]. Ellipse fitting is first performed using the technique in [28]. If the method of [28] fails to fit an ellipse to the curve, in which case the calculated parameters of the ellipse take the form of a null vector, or if there is numerical instability, the algorithm switches over to the technique of perturbing function [29]. The center of the ellipse is calculated using the obtained ellipse parameters as given by (17), followed by a scaling with a factor of 1/4 to compensate for the interpolation by 4 as discussed in Section 4.2.1. The obtained ellipse centers give the localized control points. The proposed ellipse fitting method ensures that the control points are localized in the image with a high accuracy. The localized control points are then used to initialize the

linear parameters of the camera after the first iteration as explained in Section 4.3, and later to estimate the camera calibration parameters in the subsequent iteration as described in Section 4.4.

4.3 Initialization of linear parameters using planar vanishing points

A vanishing point for a set of parallel lines is the point at which a set of parallel lines in world coordinates intersect on the image plane. A vanishing point for a set of parallel lines is shown in Fig. 6. As shown in Fig. 6, $\mathcal{L}_1$ and $\mathcal{L}_2$ are two parallel lines in plane π. When these lines are projected onto a camera image plane π' with an optical center O, the resulting projections $\mathcal{L}_1'$ and $\mathcal{L}_2'$ of these two lines are not parallel to each other and seem to be converging at a point v. Hence, v is known as the vanishing point on π' for lines $\mathcal{L}_1$ and $\mathcal{L}_2$.

If $M_1 = [1\ 0\ 0]$ and $M_2 = [0\ 1\ 0]$ represent points at infinity in the x and y directions, respectively, then, from (3), $\mathbf{h_1}$ and $\mathbf{h_2}$ represent the vanishing points in the x and y directions, respectively. Two important properties of vanishing points are the fact that the line connecting the vanishing point and the optical center O has the same direction with the corresponding lines in the 3D space and the fact that the vanishing points are independent of the camera translation [24].

Using these properties, it can be shown [24] that if $\mathbf{v_1}$ and $\mathbf{v_2}$ are the vectors after subtracting the principal point from $\mathbf{h_1}$ and $\mathbf{h_2}$, respectively, one obtains [24]:

$$(\mathbf{K}\mathbf{v}_1)^{\mathrm{T}}(\mathbf{K}\mathbf{v}_2) = \mathbf{0} \quad \text{where} \quad \mathbf{K} = \begin{bmatrix} 1/f_x & 0 & 0 \\ 0 & 1/f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{21}$$

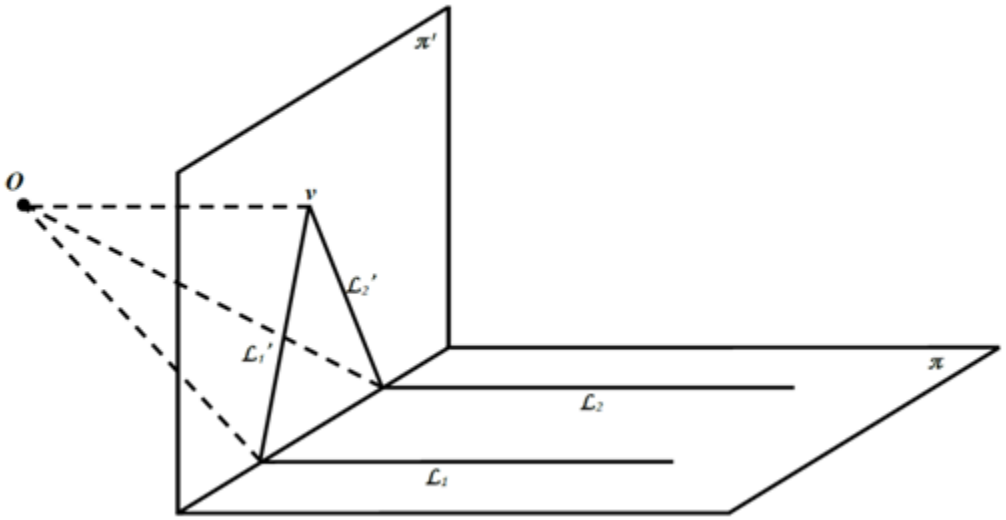


Fig. 6. Vanishing point of a pair of parallel lines.

It can also be shown that $\mathbf{v_3} = \mathbf{v_1} + \mathbf{v_2}$ and $\mathbf{v_4} = \mathbf{v_1} - \mathbf{v_2}$ represent the vanishing points in the diagonal directions of the image. Let the principal point of the camera be initialized to the center of the image, and assume that the skew γ is 0 and that the vanishing points in pixel coordinates are given by

$$\mathbf{v_i} = [a_i \quad b_i \quad c_i] \quad \text{for} \quad i = 1, 2, 3, 4 \tag{22}$$

where i = 1, 2, 3, 4, represent, respectively, the 4 directions along the x, y, and the two diagonal directions.

From (21), it can be seen that

$$(\mathbf{K}\mathbf{v}_1)^{\mathrm{T}}(\mathbf{K}\mathbf{v}_2) = \mathbf{v}_1(\mathbf{K}^{\mathrm{T}}\mathbf{K})\mathbf{v}_2 = \mathbf{0}$$

$$(\mathbf{K}\mathbf{v}_3)^{\mathrm{T}}(\mathbf{K}\mathbf{v}_4) = \mathbf{v}_3(\mathbf{K}^{\mathrm{T}}\mathbf{K})\mathbf{v}_4 = \mathbf{0} \tag{23}$$

From (22) and (23), one gets [24]

$$a_1a_2/f_x^2 + b_1b_2/f_y^2 + c_1c_2 = 0$$

$$a_3a_4/f_x^2 + b_3b_4/f_y^2 + c_3c_4 = 0 \tag{24}$$

The disadvantage of this method is that the skew and principal point has to be assumed to be zero and at the center of the image, respectively. However, this method is chosen over Zhang's method [2] for implementation as it was experimentally observed in [24] that this method requires less iterations as compared to [2] for the camera parameters to converge to their final values during optimization Also, the principal point converges to a value comparable to that obtained using Zhang's method [2] after nonlinear optimization using the Levenberg Marquardt algorithm. Skew is assumed to be zero in the implementation. The extrinsic parameters are calculated according to (10).

4.4 Nonlinear optimization using Levenberg Marquardt (LM) algorithm

The optimization of the camera parameters using the LM algorithm is performed in a way similar to [3]. The Levenberg-Marquardt (LM) algorithm, also known as the damped least squares method (DLS), provides a numerical solution to minimizing a nonlinear function over multiple parameters. If there are multiple local minima, it results in the minimum that is closest to the provided initial value. Hence, in this latter case, it is important to provide the algorithm with initial values that are relatively closer to the true global minimum.

If there are a total of N images with K control points on each image then the error function in (13) is minimized in an iterative fashion using a nonlinear optimization technique such as the Levenberg-Marquardt algorithm where $\mathbf{m}_{ij}$ are pixel coordinates of the control points in the image of the calibration pattern

and $\mathbf{m}'_{ij} = f\left(\mathbf{X}_{ij}, \mathbf{A}, \mathbf{R}_i, \mathbf{t}_i, \mathbf{k}_c\right) = \boldsymbol{f}(\mathbf{X}_{ij}, \boldsymbol{\kappa})$ are the coordinates that are obtained from the projection of the 3D coordinates of the control points $\mathbf{X}_{ij} = [X_{ij}, Y_{ij}, Z_{ij} = 0]$, using the camera parameters obtained from the previous iteration and $\boldsymbol{\kappa} = [\mathbf{A} \quad \mathbf{R}_i \quad \mathbf{t}_i \quad \mathbf{k_c}]$ is the vector containing all the camera parameters. In the remainder of this section, for simplicity of notation, without loss of generality, $\boldsymbol{f}(\mathbf{X}_{ij}, \boldsymbol{\kappa})$ will be simply denoted by $f(\boldsymbol{\kappa})$. If $\boldsymbol{\kappa}$ is to be replaced by $\boldsymbol{\kappa} + \boldsymbol{\delta}$ then, $f(\boldsymbol{\kappa} + \boldsymbol{\delta})$ is approximated to

$$f(\boldsymbol{\kappa} + \boldsymbol{\delta}) = f(\boldsymbol{\kappa}) + \mathbf{J}\boldsymbol{\delta} \quad \text{where} \quad \mathbf{J} = \partial f(\boldsymbol{\kappa}) / \partial (\boldsymbol{\kappa}) \tag{25}$$

$\mathbf{J}$ is also known as the Jacobian of $f(\boldsymbol{\kappa})$. Let $E(\boldsymbol{\kappa} + \boldsymbol{\delta})$ denote the error between $\mathbf{m}_{ij}$ and $\mathbf{m}'_{ij}$. $E(\boldsymbol{\kappa} + \boldsymbol{\delta})$ can be expressed as [30]:

$$E(\boldsymbol{\kappa} + \boldsymbol{\delta}) = \left\| \boldsymbol{m}_{ij} - \boldsymbol{f}(\boldsymbol{\kappa}) - \mathbf{J}\boldsymbol{\delta} \right\|^2 \tag{26}$$

To minimize (26), the derivative of the error is taken with respect to $\boldsymbol{\kappa}$ and is set to zero resulting in

$$\mathbf{J}^{\mathrm{T}}\mathbf{J}\boldsymbol{\delta} = \mathbf{J}^{\mathbf{T}}[\boldsymbol{m}_{ij} - \boldsymbol{f}(\boldsymbol{\kappa})] \tag{27}$$

Levenberg introduced an additional term λ to (27) in order to control the minimization of (26), as follows:

$$\mathbf{J}^{\mathrm{T}}\mathbf{J}\boldsymbol{\delta} + \lambda\mathbf{I} = \mathbf{J}^{\mathbf{T}}[\boldsymbol{m}_{ij} - \boldsymbol{f}(\boldsymbol{\kappa})] \tag{28}$$

where $\mathbf{I}$ is the identity matrix. The term λ helps control the rate of change in the variables.

Solving (28) results in an estimate for $\boldsymbol{\delta}$ at each iteration. If the change in the mean square error is large, then λ is reduced for the next iteration. If the change is small, then λ is increased.

As indicated before, the camera parameters are initialized according to the method of vanishing points [24] using (24) and (10). For optimizing the extrinsic parameters, the extrinsic parameters are computed alone in an iterative fashion to refine and correct only the extrinsic parameters. In the adopted implementation [3], the optimization is run either for a maximum of 20 iterations or until the change in the extrinsic parameters is less than 10^{-10} times the magnitude of their previous values. λ is kept at a constant of 1 in all iterations.

After refining the extrinsic parameters, the optimization is performed for all the camera parameters. In our simulations, the optimization is performed either for a maximum of 50 iterations for minimization or until the change in the camera parameters is less than 10^{-9} times the magnitude of their previous values. As in [3], λ is calculated for the i^{th} iteration as follows:

$$\lambda_i = 1 - (0.6)^{i+1} \tag{29}$$

The nonlinear optimization of all the parameters is followed by another run of optimization for only the extrinsic parameters either for a maximum of 20 iterations or until the change in the extrinsic parameters is less than 10^{-10} times the magnitude of their previous values. λ is kept at a constant of 1 in all iterations.

4.5 Projection

The estimated camera parameters are used to project the 3D coordinates of the control points onto the pixel coordinates of the image.

The 3D coordinates ($\mathbf{x_w}$, $\mathbf{y_w}$, $\mathbf{z_w}$) are converted into camera coordinates ($\mathbf{x_c}$, $\mathbf{y_c}$, $\mathbf{z_c}$) as follows:

$$\begin{bmatrix} \mathbf{x}_\mathrm{c} \\ \mathbf{y}_\mathrm{c} \\ \mathbf{z}_\mathrm{c} \end{bmatrix} = \mathbf{R} \begin{bmatrix} \mathbf{x}_\mathrm{w} \\ \mathbf{y}_\mathrm{w} \\ \mathbf{z}_\mathrm{w} \end{bmatrix} + \mathbf{t} \tag{30}$$

where $\mathbf{R}$ is the rotation matrix and $\mathbf{t}$ is the translation matrix and $\mathbf{z}_\mathrm{w} = 0$.

As described in Section 2.3, the obtained 2D camera coordinates are distorted as follows [5]:

$$\begin{aligned} \mathbf{x}_d &= \mathbf{x}_\mathrm{c}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + (2p_1 \mathbf{y}_\mathrm{c} + p_2(r^2 + 2\mathbf{x}_\mathrm{c}{}^2)) \\ \mathbf{y}_d &= \mathbf{y}_\mathrm{c}(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + (p_1(r^2 + 2\mathbf{y}_\mathrm{c}{}^2) + 2p_2 \mathbf{x}_\mathrm{c}) \end{aligned} \tag{31}$$

Finally, the coordinates ($\mathbf{x_d}$, $\mathbf{y_d}$) are converted into pixel coordinates ($\mathbf{x}_\mathrm{p}$, $\mathbf{y}_\mathrm{p}$) as follows:

$$\begin{aligned} \mathbf{x}_\mathrm{p} &= \mathrm{f_x} \mathbf{x}_d + u_0 \\ \mathbf{y}_\mathrm{p} &= \mathrm{f_y} \mathbf{y}_d + v_0 \end{aligned} \tag{32}$$

The projection function is used to project the 3D planar control points on to pixels in the camera image plane during the calculation of the reprojection error in order to determine the accuracy of the camera calibration. This same projection function is also used to calculate the pixel coordinates during the construction of the fronto parallel plane images by projecting the 3D coordinates $\mathbf{X}_{ij} =$

$(X_{ij}, Y_{ij}, Z_{ij} = 0)$ of the fronto parallel plane grid points onto the camera image plane in order to determine the image intensity at these points. The projection function is also used to project the determined ellipse centers (control points) in the fronto parallel plane onto the pixel coordinates of the image in the camera plane.

4.6 Undistortion and unprojection

The origin of the coordinates in the fronto parallel plane is assumed to be at the left top control point (center of the top left circle). Hence, a grid of points uniformly spaced located starting at $-d$ and ending at $k \times d$ with a spacing of $d/40$ pixels is created in the x and y directions, where d is the spacing between two control points and k is the number of control points in each direction. This implies that each control point is encapsulated in a region consisting of 40×40 points. Hence, when these grid points are projected to the 2D camera plane, each control point is bounded by a bounding box of 40×40 pixels.

The pixel intensities at these fronto parallel plane coordinates in the 3D coordinate system are required to construct the image in the fronto parallel plane. These coordinates are converted into pixel coordinates $(\mathbf{x}_p, \mathbf{y}_p)$ using projection as described in Section 4.5. The image coordinates obtained henceforth might not necessarily be an integer. Hence, the intensities at these non-integer coordinates in the image are calculated using bilinear interpolation. The intensities obtained from interpolation are assigned to the location corresponding to their respective

coordinate in the grid, hence constructing the images in the canonical fronto parallel plane. Fig. 7 depicts sample calibration pattern images (left) and their corresponding projections onto the fronto parallel plane (right).

The undistorted, unprojected images in the canonical fronto parallel plane are calculated at each iteration (except for the first iteration) using the camera calibration parameters that are obtained from the previous iteration. The images in the canonical fronto parallel plane are converted into a binary image and an ellipse is fit for each pattern using the methods described in Section 4.2.1 and Section 4.2.2, respectively. The centers of the ellipses are calculated using (17). The coordinates of the centers of the ellipses are back-projected using the camera parameters onto the camera plane so that the obtained projected pixel coordinates represent the coordinates of the centers of the ellipses in the camera plane. These coordinates act as control points for further calculations in the Levenberg Marquardt algorithm and for calculating the reprojection error.

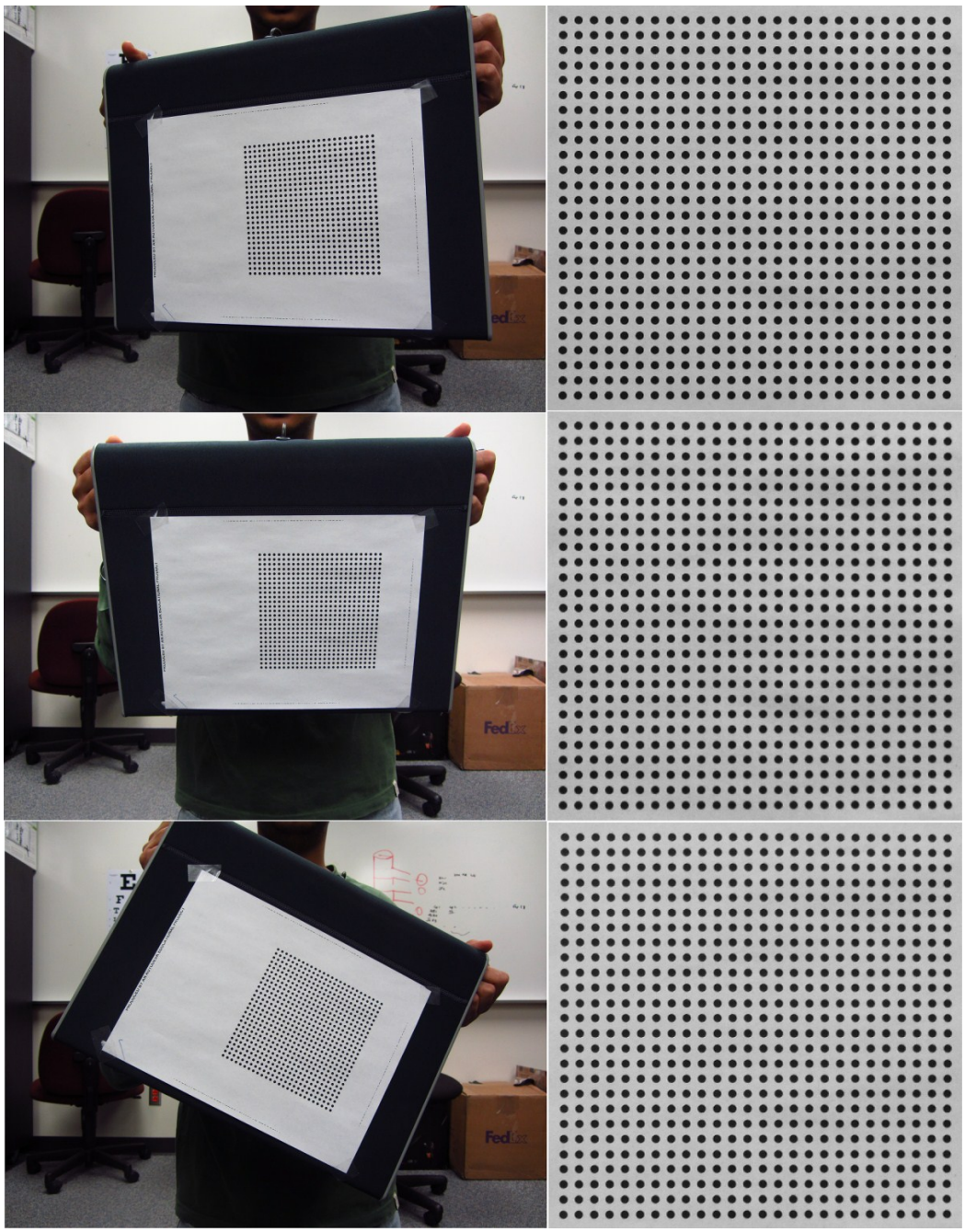


Fig. 7. Unprojection and undistortion. (Left) Calibration images. (Right) Undistorted, unprojected images in the canonical fronto parallel plane.

4.7 Calculation of the reprojection error

Calculation of the reprojection error is performed at every iteration in order to verify the improvements due to conversion of the image to a canonical fronto parallel plane. The coordinates of the control points obtained after they are projected from the fronto parallel plane to the camera plane serve as the detected coordinates of control points $\mathbf{m}_{ij}$. The known 3D coordinates of the control points are projected onto the pixels coordinates as explained in Section 4.5. These coordinates serve as the determined coordinates of the image $\mathbf{m}'_{ij}$.

The reprojection error is calculated in both the x and y directions according to (13) as the standard deviation of the difference between the $\mathbf{m}_{ij}$ and $\mathbf{m}'_{ij}$ in their respective directions. The average error is calculated as follows:

$$Average\ Error = \sqrt{\sum_{i=1}^{N}\sum_{j=1}^{K}\left\|\boldsymbol{m}_{ij} - \boldsymbol{m}'_{ij}\right\|^2}/NK \tag{33}$$

where N is the number of acquired images for the considered calibration pattern and K is the number of control points in each image.

The uncertainty of focal length determination in the x and y directions respectively, d_x and d_y are given by [31]:

$$d_x = 3 \times \sqrt{\mathbf{J}'_{11}} \times reprojection\ error\ in\ x\ direction$$

$$d_y = 3 \times \sqrt{\mathbf{J}'_{22}} \times reprojection\ error\ in\ y\ direction \tag{34}$$

where $\mathbf{J}'_{11}$ and $\mathbf{J}'_{22}$ are the first and the second diagonal elements of the inverse of the Jacobian of $f(\boldsymbol{\kappa})$.

5. EXPERIMENTAL RESULTS

In this chapter, the experimental results of the proposed camera calibration algorithm are presented and analyzed. Section 5.1 introduces the image sets that are used to evaluate the performance of the proposed camera calibration method. Section 5.2 illustrates the proposed adaptive thresholding in the fronto parallel plane. Section 5.3 presents a performance analysis of the obtained camera calibration parameters in terms of the reprojection error. Section 5.4 presents the camera calibration parameters calculated for the different data sets and Section 5.5 discusses the effect of control points on the reprojection error and uncertainty.

5.1 Data set description

Calibration results are obtained using 6 sets of calibration images. Datasets 1 to 5 were generated by taking images of select regions of the circular calibration pattern available from [32] using an 8-megapixel camera with an accuracy of 5.5 microns per pixel, while dataset 6 consists of the images used in the calibration scheme of [3].

The number of images in datasets 1 to 6 are 20, 25, 25, 15, 15 and 5, respectively. Datasets 1 to 3 consist of a calibration pattern of 26×26 circles spaced apart from each other at a distance of 0.5 mm and were captured at different lighting conditions and different angles. Dataset 4 consists of a calibration pattern of 6×14 circles spaced apart from each other at a distance of 1mm. Dataset 5 consists of a calibration pattern of 6×6 circles spaced apart from

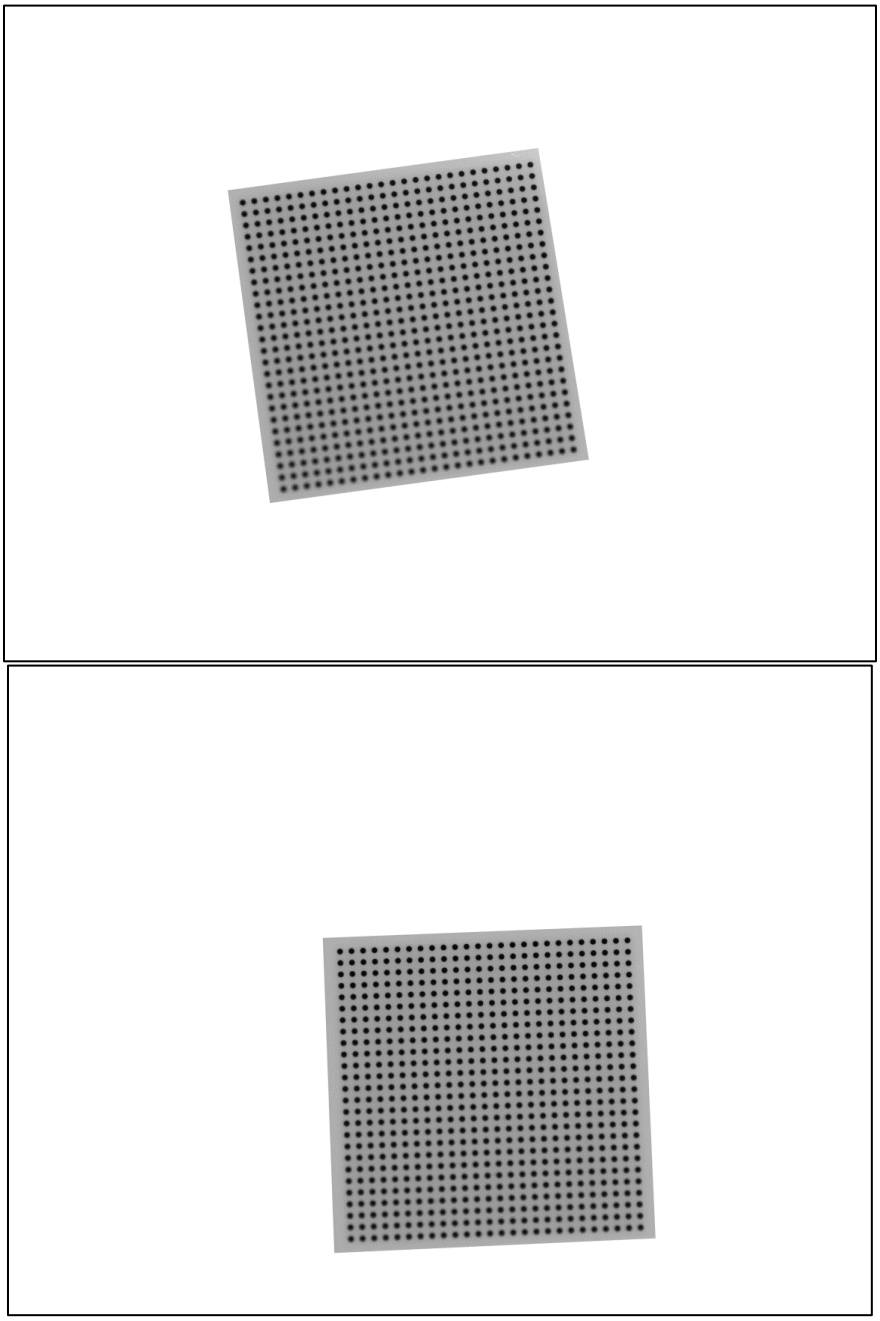


Fig. 8. Example of images in Dataset 1.

each other at a distance of 2 mm. Dataset 6 consists of a calibration pattern of 9×6 circles

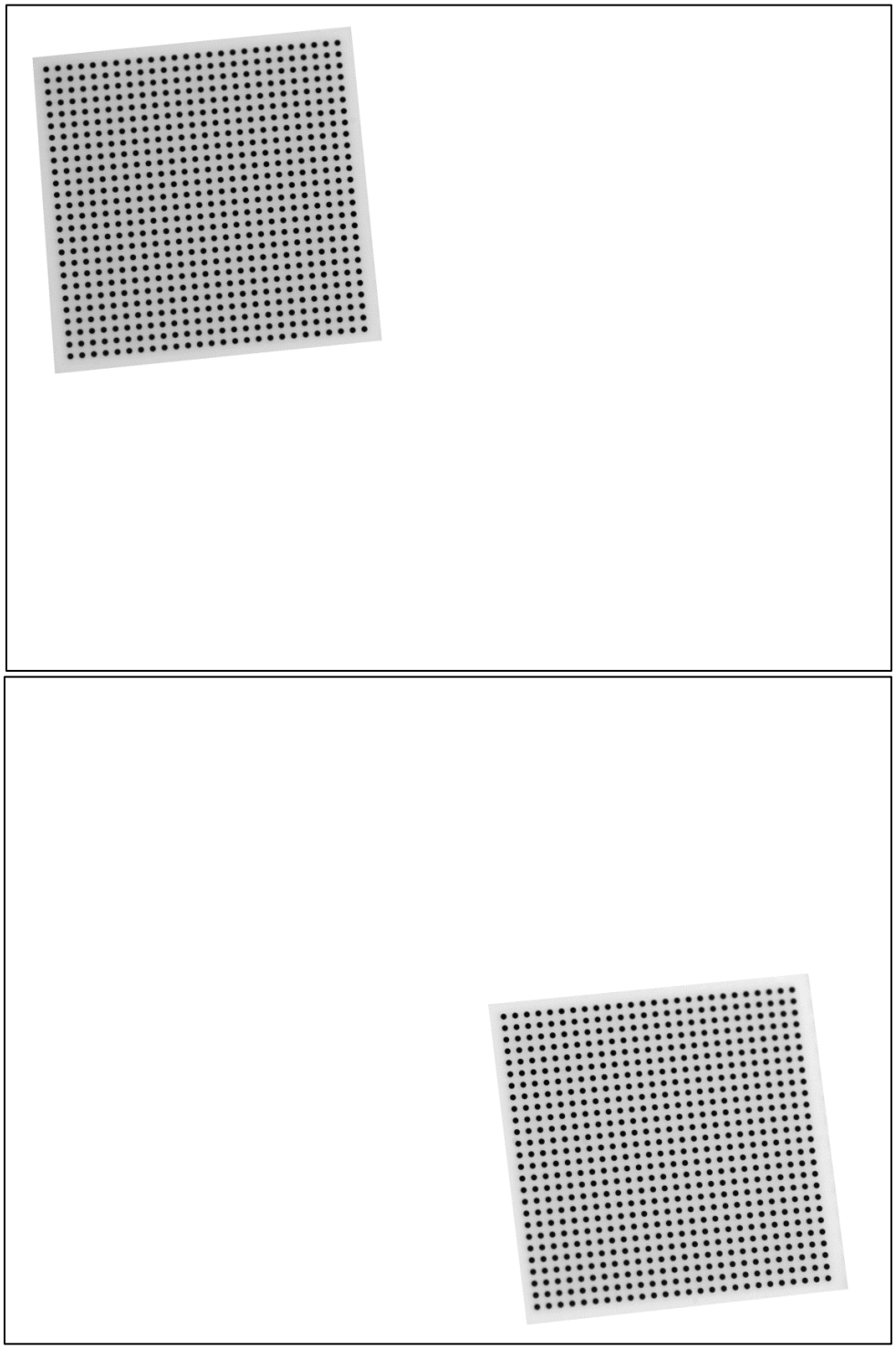


Fig. 9. Example of images in Dataset 2.

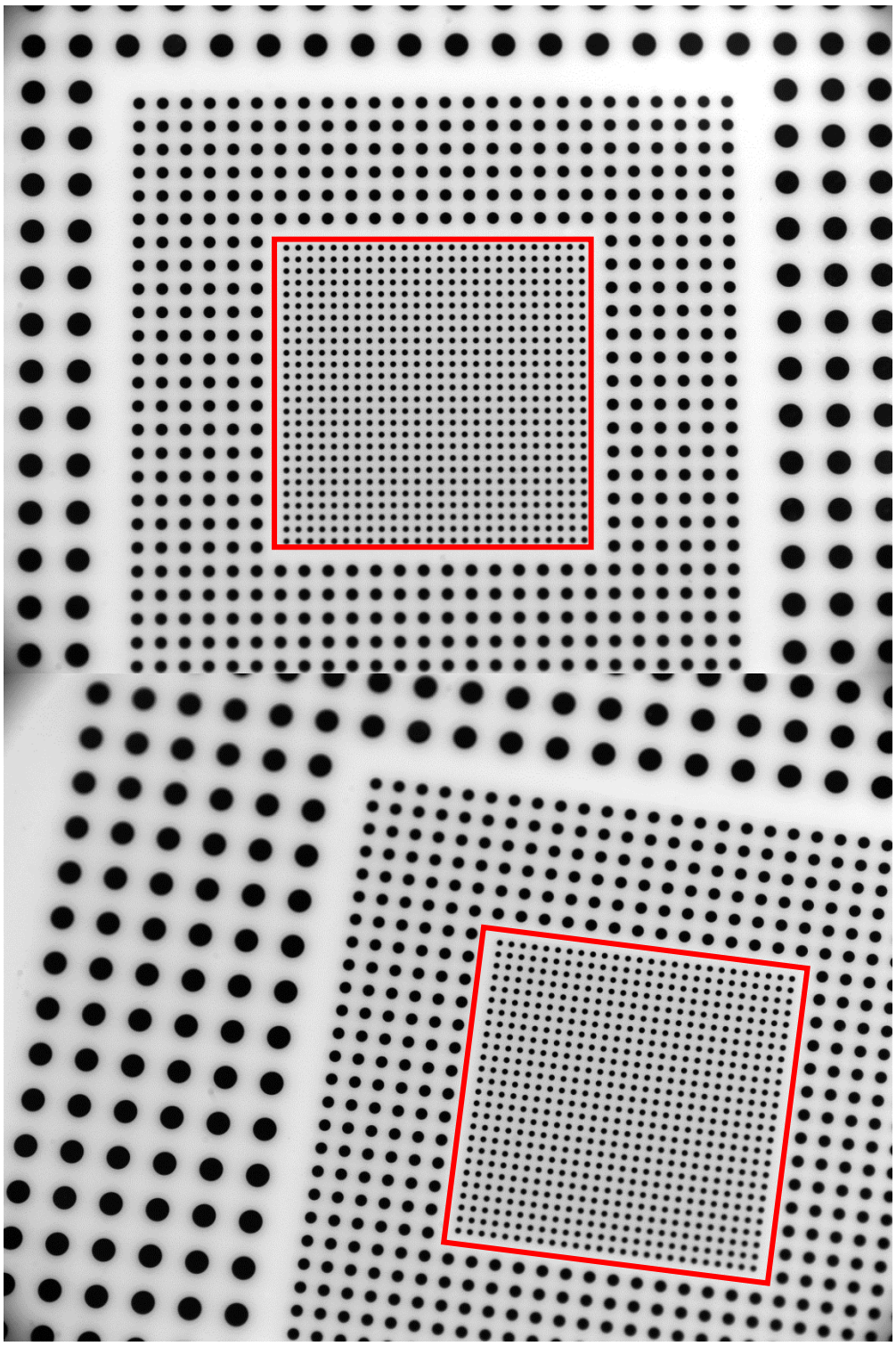


Fig. 10. Example of images in Dataset 3. Highlighted circles are considered.

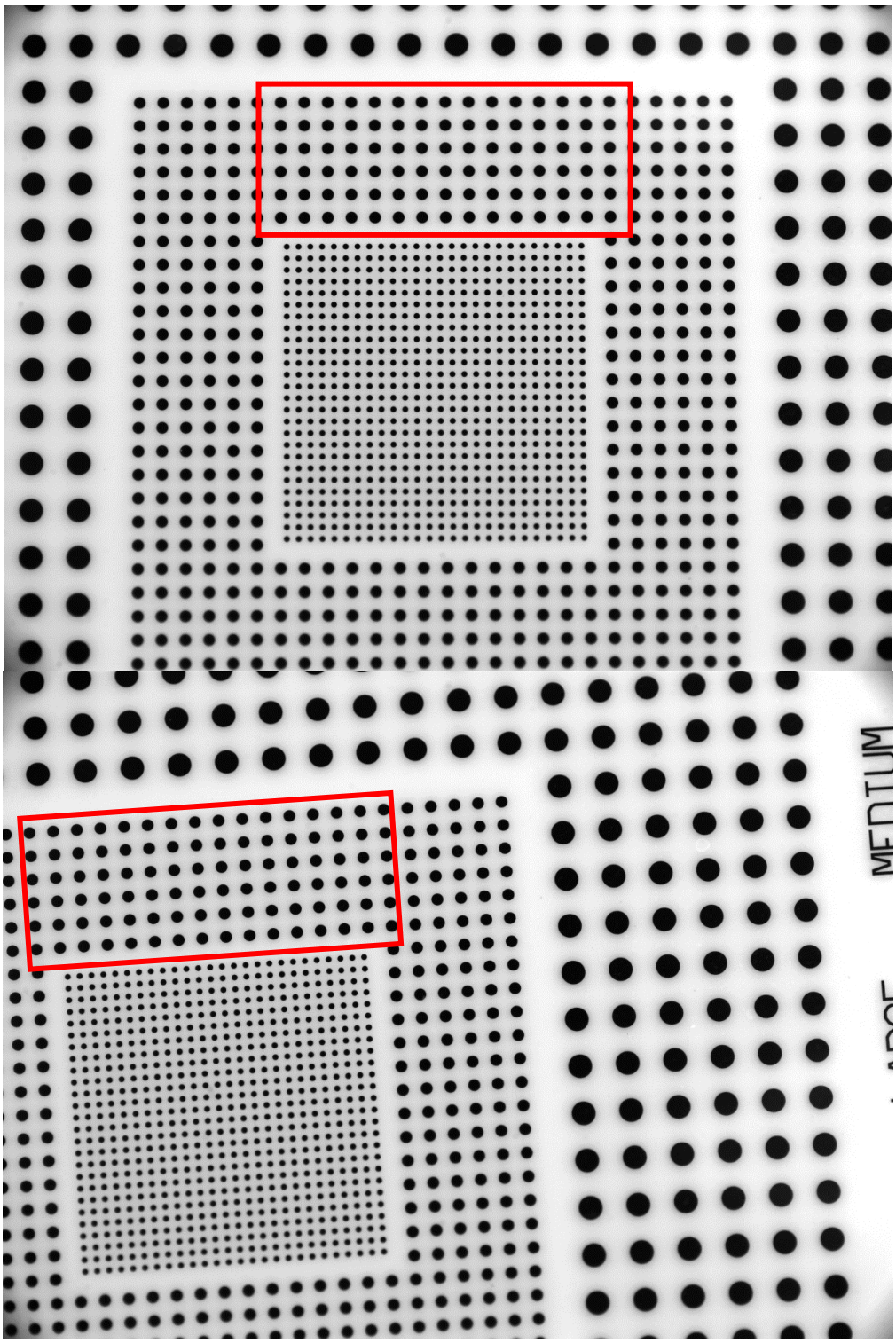


Fig. 11. Example of images in Dataset 4. Highlighted circles are considered.

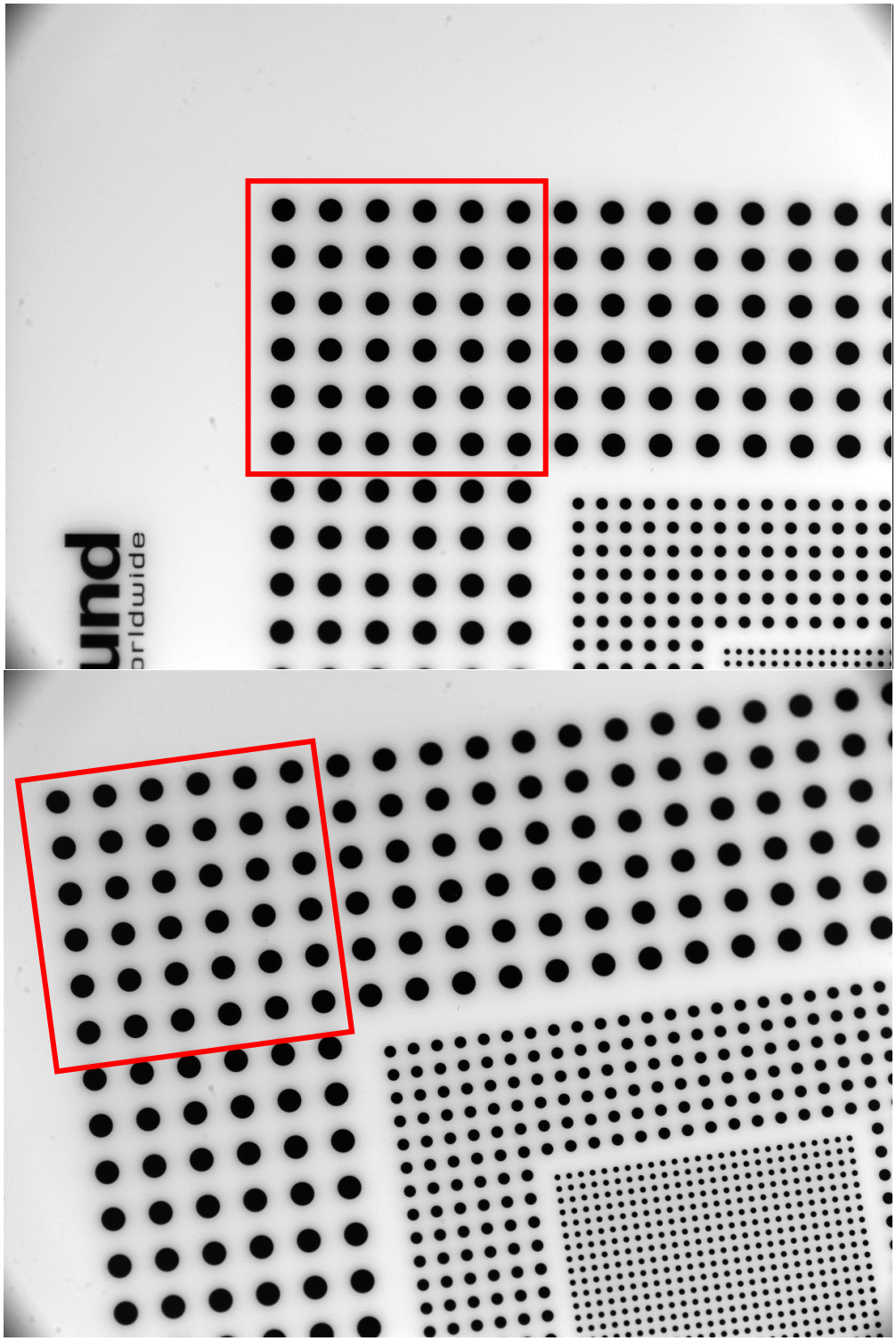


Fig. 12. Example of images in Dataset 5. Highlighted circles are considered.

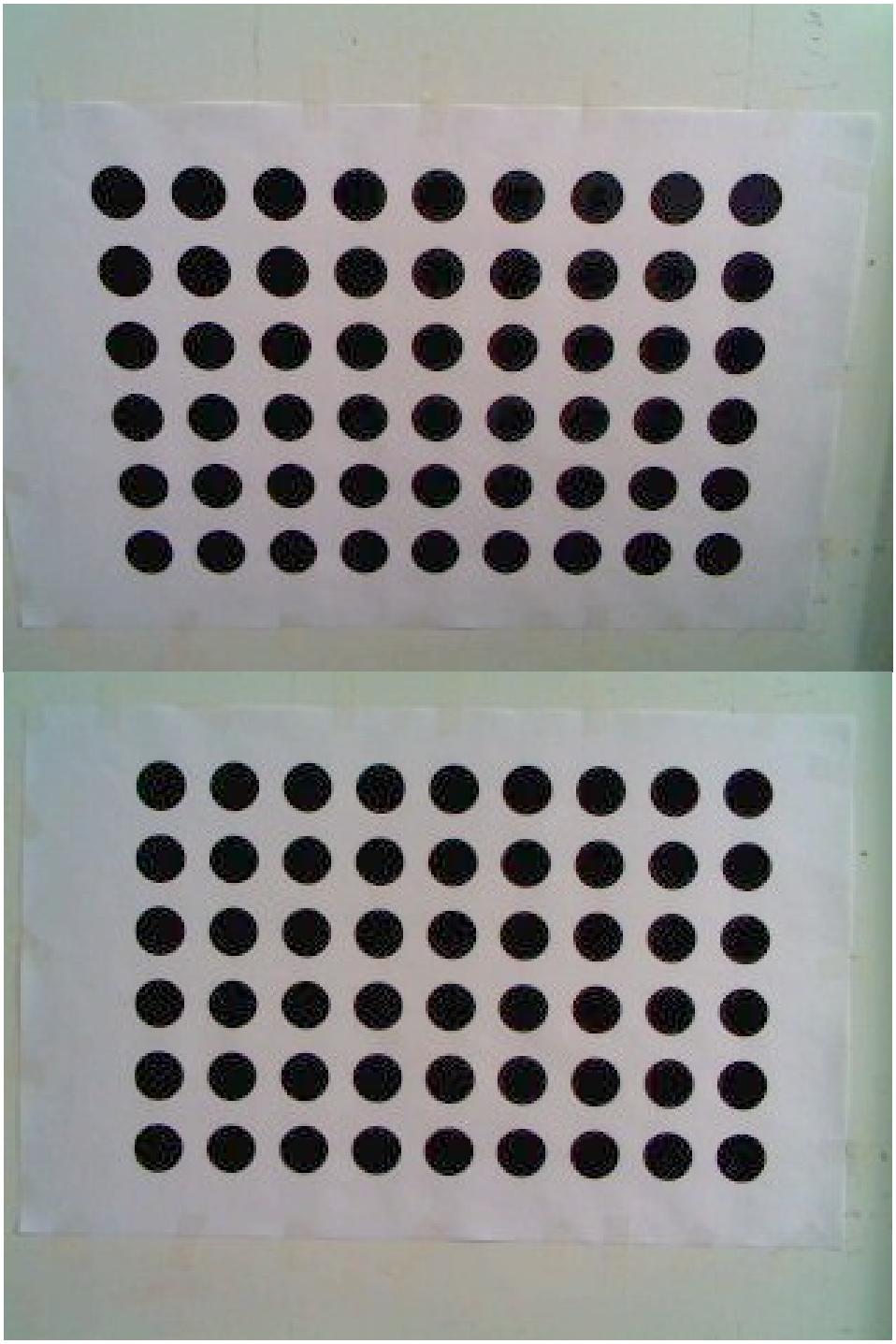


Fig. 13. Example of images in Dataset 6.

5.2 Adaptive thresholding in fronto parallel plane

The algorithm was run on each dataset using the proposed scheme. For comparison, results were also obtained using the scheme of [3] on these datasets. Skew was assumed to be 0. As indicated previously in Section 4.2, for both scheme of [3] and the proposed scheme, in the first iteration, the control points were localized in the camera plane by converting the image in the camera plane into a binary image with a threshold of 0.5 times the mean intensity of the image to form an initial estimate of the camera calibration parameters. The control points were localized, from the second iteration onwards, in the fronto parallel plane which was constructed from the camera calibration parameters obtained in the previous iteration. The images in the fronto parallel planes were sampled such that each circle in the calibration pattern occupies an area of 40×40 pixels. In the proposed algorithm, the images in the fronto parallel plane were subsequently interpolated by a factor of 4 and converted into a binary image using adaptive thresholding. Figs. 14 to 19 show, respectively, for the images in Figs. 8 to 13, the resulting images in the fronto parallel plane and their corresponding thresholded binary images.

The boundary pixels of each circle in the resulting binary images were detected using the Canny edge detector and were used for fitting ellipses and localizing control points.

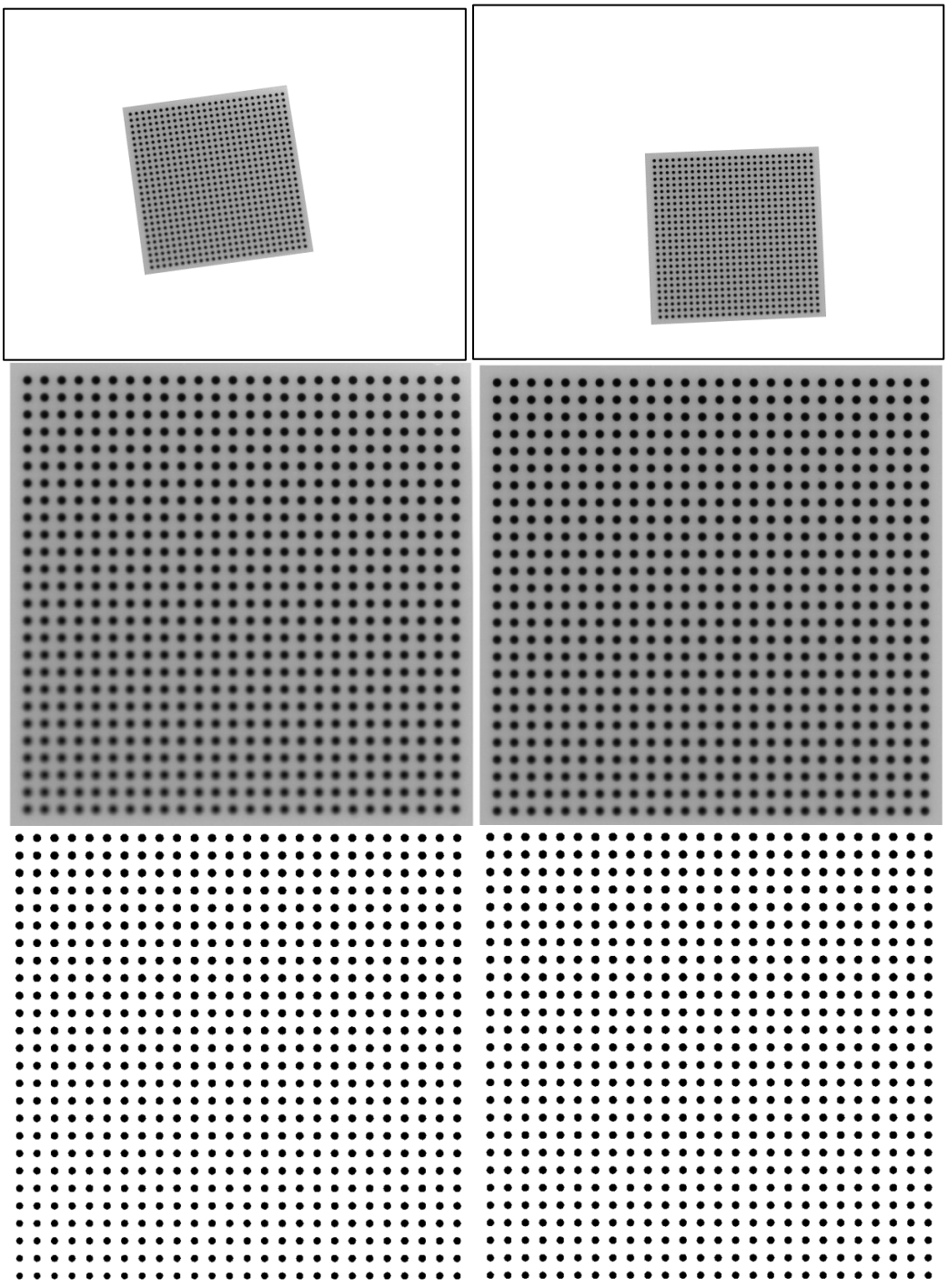


Fig. 14. Calibration images of Dataset 1 (Top). Images in fronto parallel plane (Middle). Corresponding binary images (Bottom).

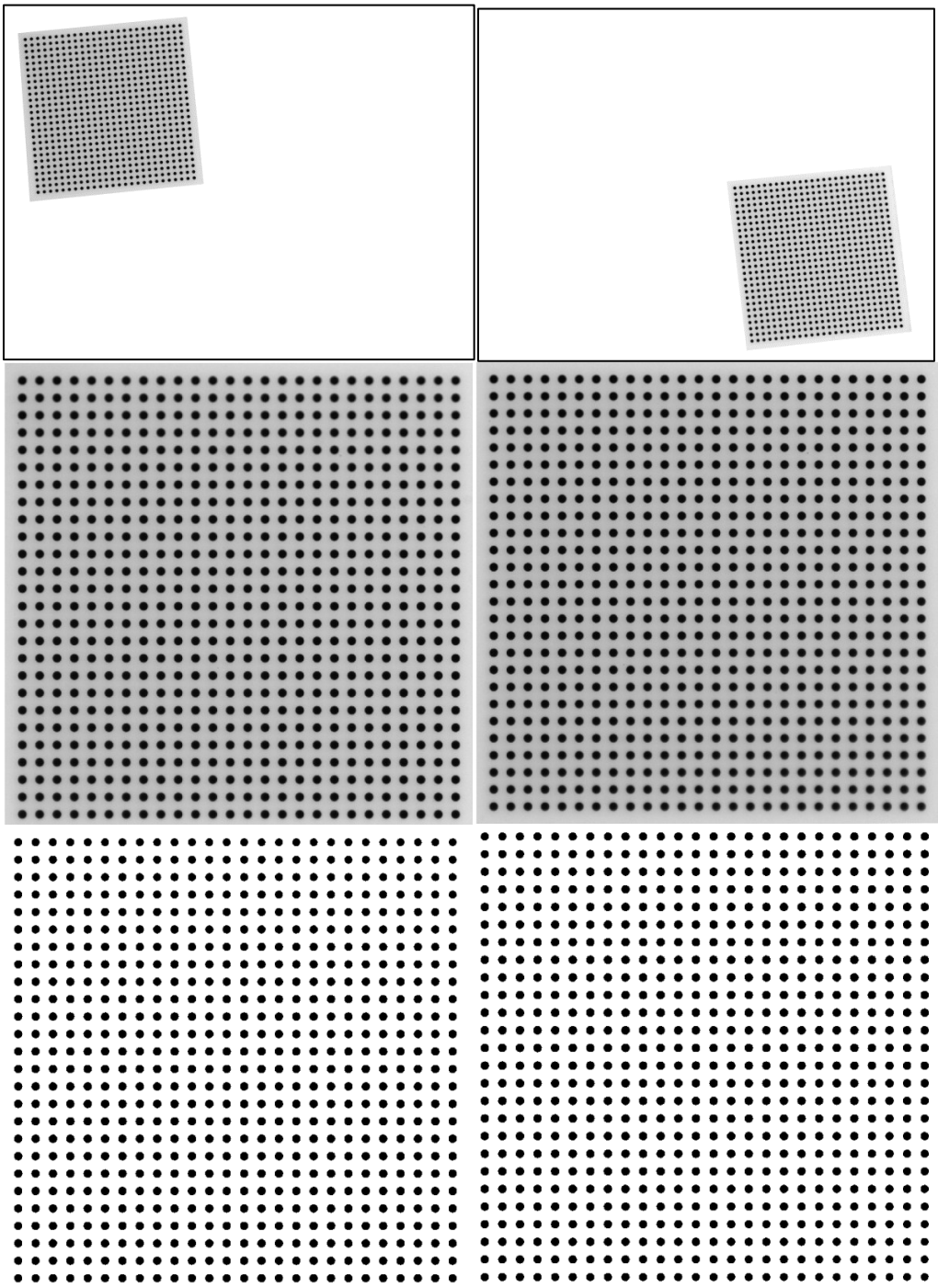


Fig. 15. Calibration images of Dataset 2 (Top). Images in fronto parallel plane (Middle). Corresponding binary images (Bottom).

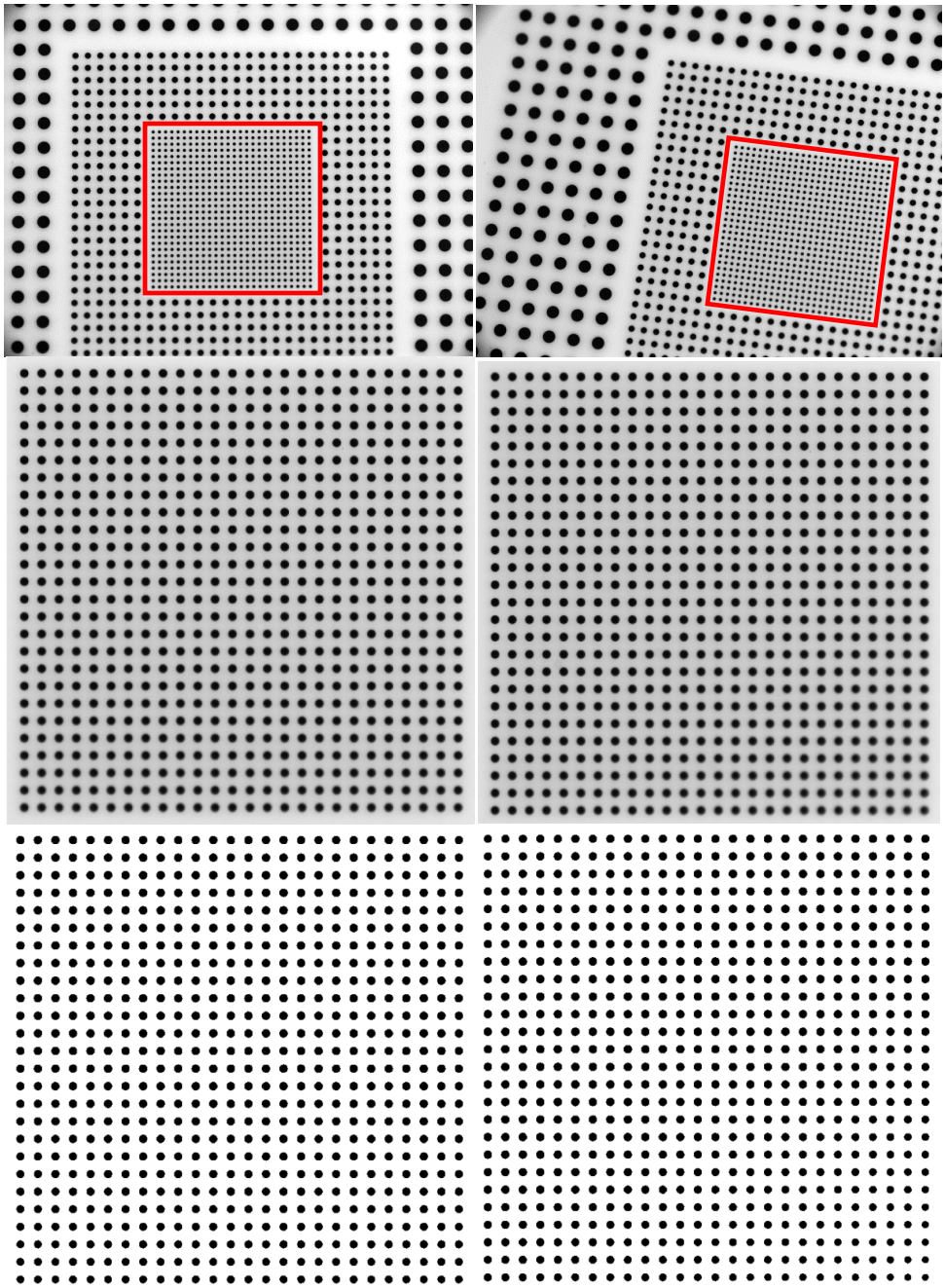


Fig. 16 Calibration images of Dataset 3 (Top). Images in fronto parallel plane (Middle). Corresponding binary images (Bottom).

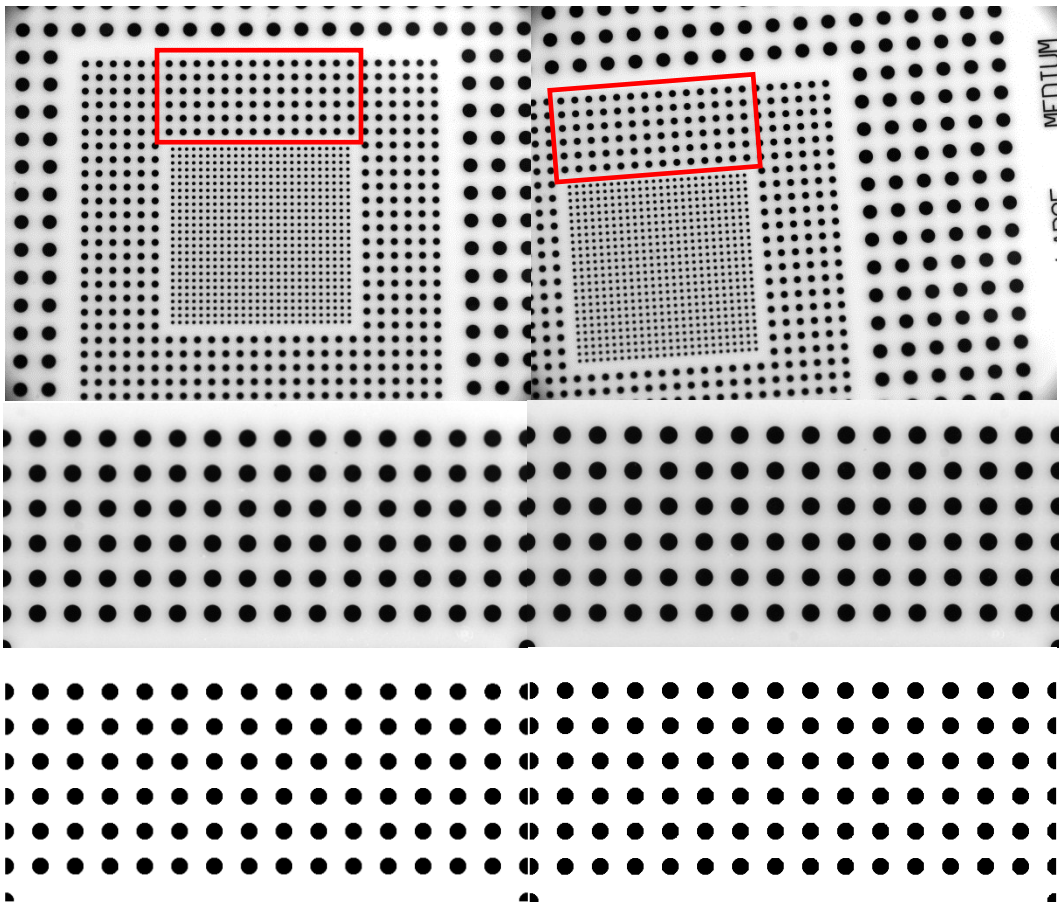


Fig. 17. Calibration images of Dataset 4 (Top). Images in fronto parallel plane (Middle). Corresponding binary images (Bottom).

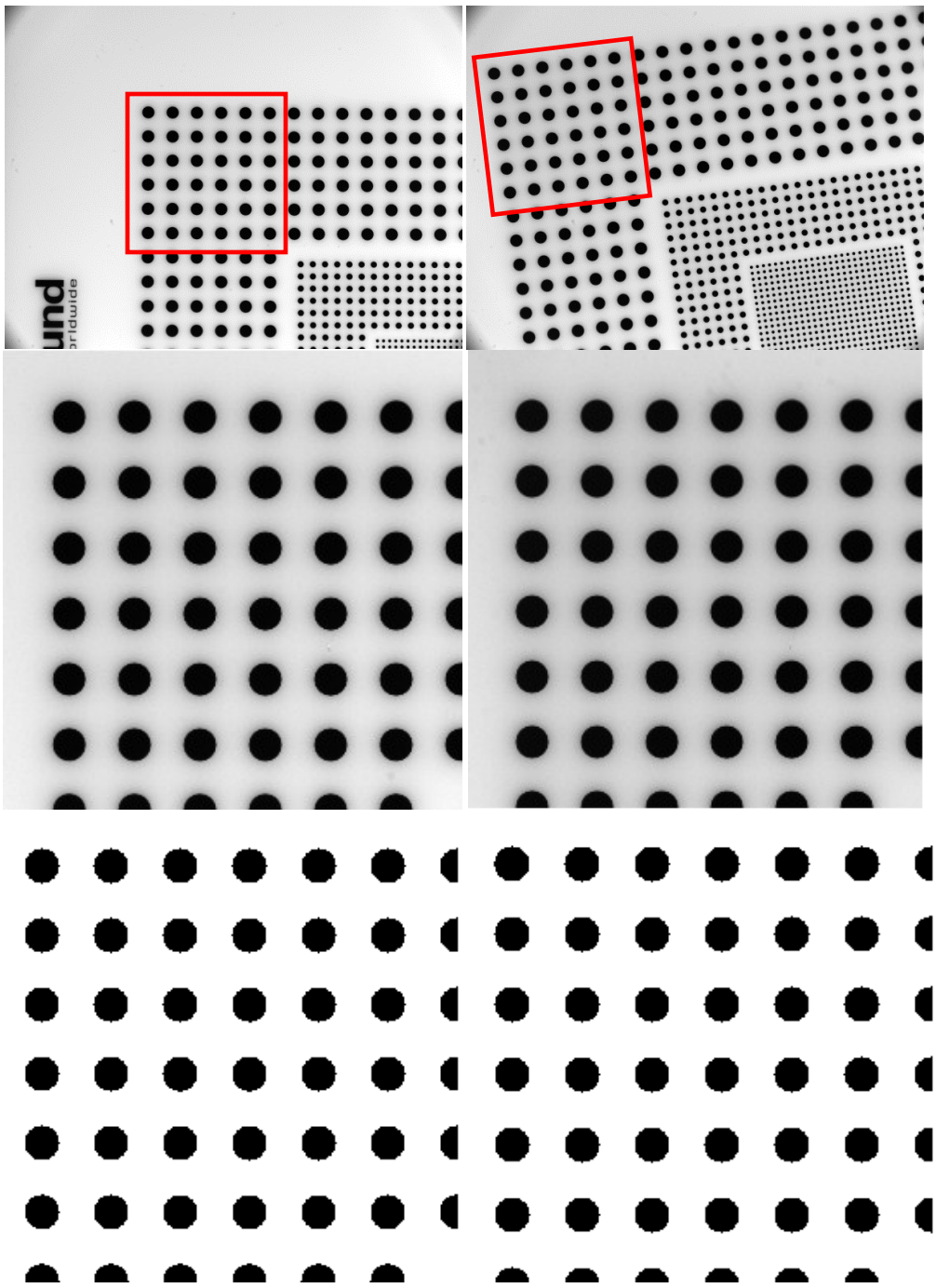


Fig. 18. Calibration images of Dataset 5 (Top). Images in fronto parallel plane (Middle). Corresponding binary images (Bottom).

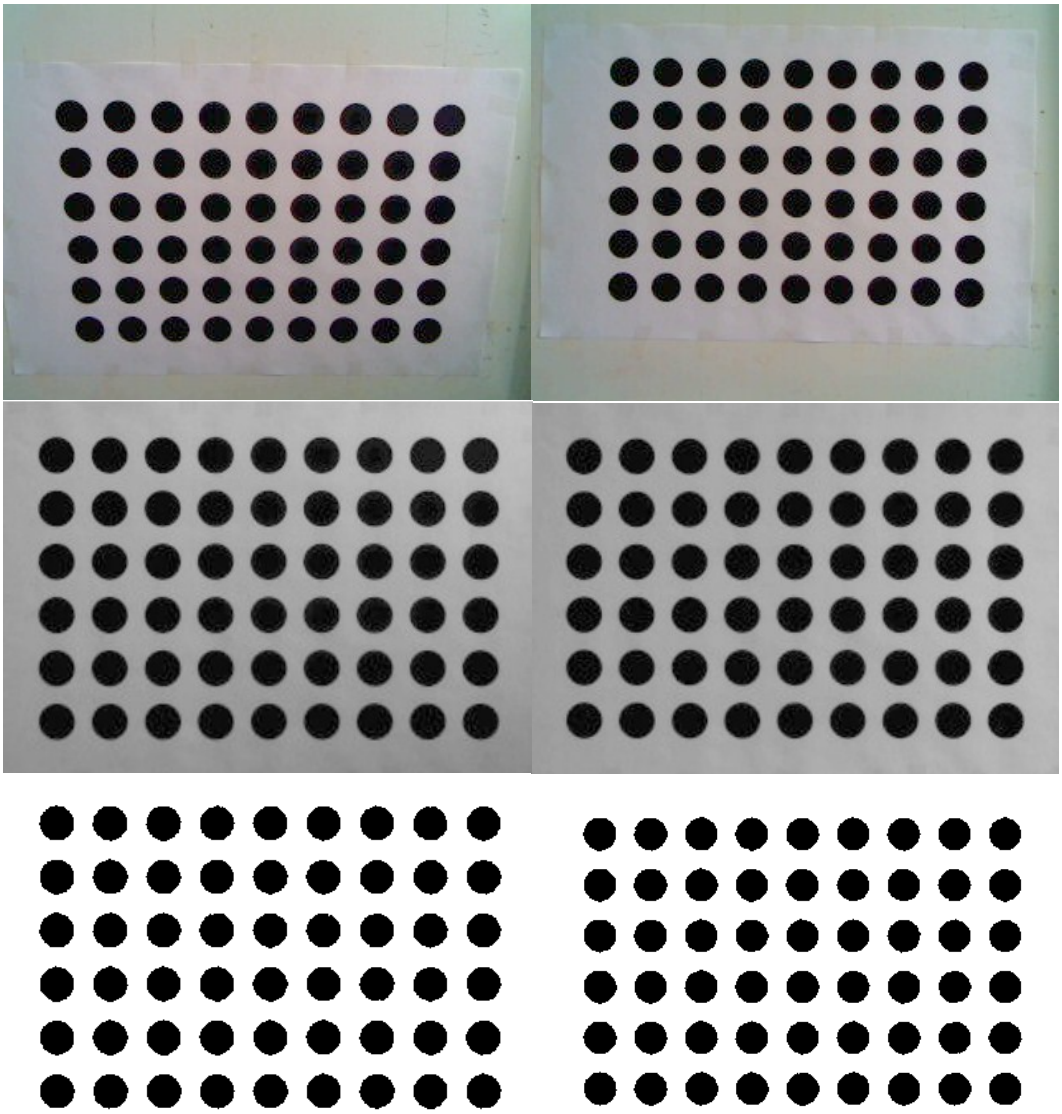


Fig. 19. Calibration images of Dataset 6 (Top). Images in fronto parallel plane (Middle). Corresponding binary images (Bottom).

5.3 Error plots

This section shows the error plots of all datasets for the initial and second iterations. The initial iteration is performed using the same method as mentioned in Section 4.2 for both the scheme of [3] and the proposed algorithm. In the second iteration and in the LM iterations that are included within this second iteration, the localization of control points is performed in the proposed algorithm using ellipse fitting, while in the scheme of [3] the localization of control points is performed using template matching with a circular template. Figs. 20 to 25 show, respectively for each dataset, the error plots in both the x and y directions for all the control points in the considered dataset. The error in the x and y direction was computed for each control point using (13) and the average error is calculated for each dataset using (33) as discussed in Section 4.7. Figs. 20 to 25 (Top) show the error plots resulting after the initial iteration. Figs. 20 to 25 (Bottom) show the error plots that are obtained using the proposed scheme at the end of the second iteration. For comparison, Figs. 20 to 25 (Middle) show the error plots that are obtained at the end of the second iteration using the scheme of [3]. Compared to the scheme of [3], it can be seen that the proposed algorithm results in more compact error plots and thus in a better performance in terms of average reprojection error for all the considered datasets, except for Dataset 6. This is because the images in Dataset 6 are low in resolution. The images are highly down sampled resulting in heavily distorted circular patterns.

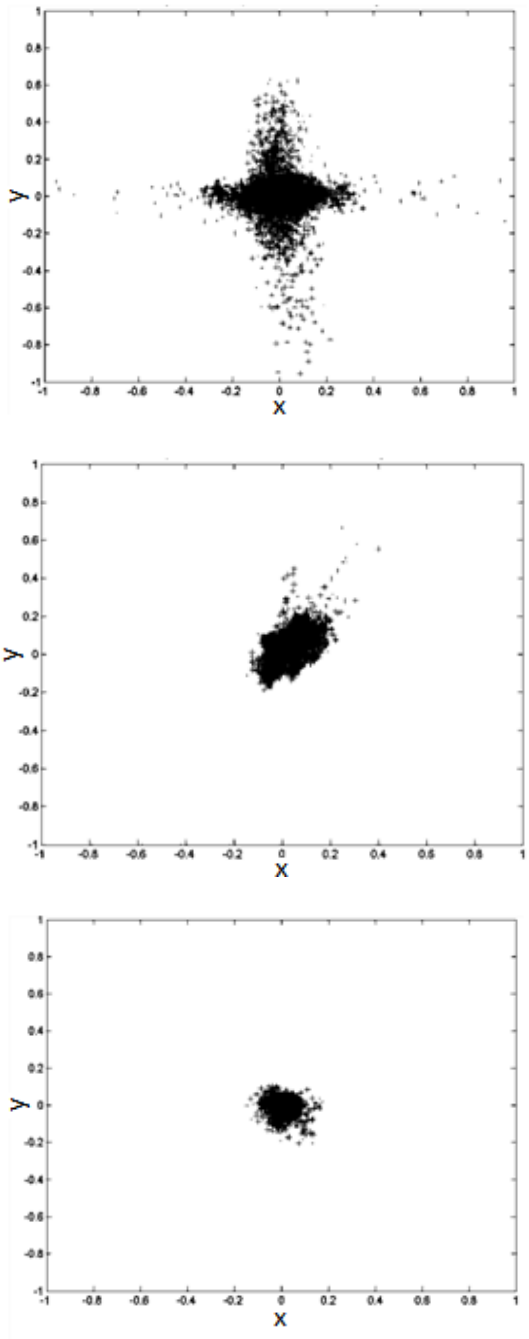


Fig. 20. Error plots for Dataset 1. Error plot after the initial iteration (Top). Error plot after the second iteration using the scheme of [3] (Middle) and the proposed algorithm (Bottom).

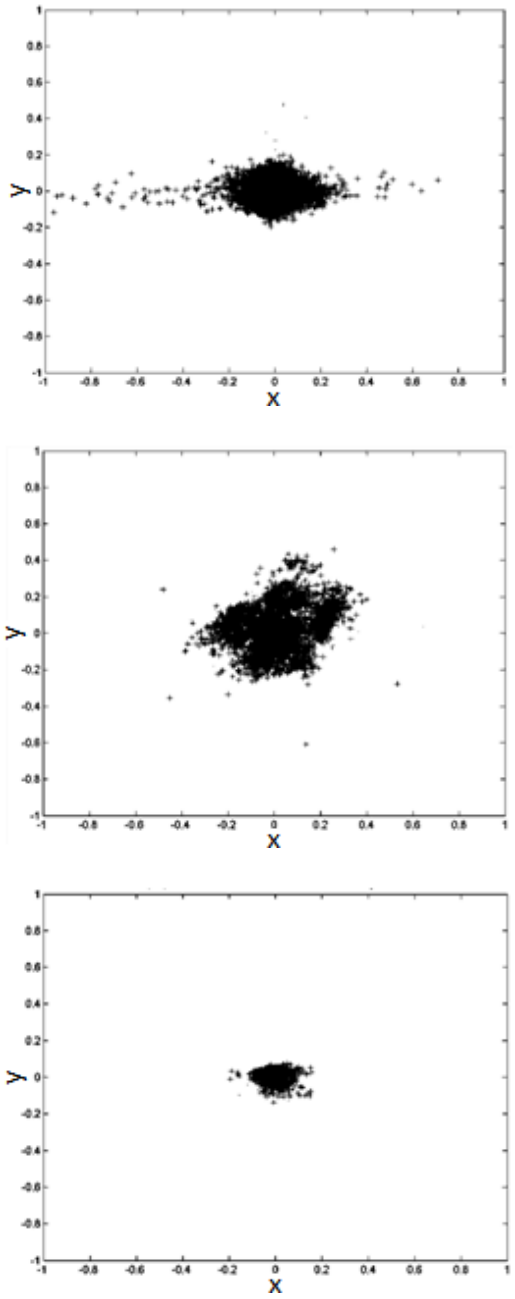


Fig. 21. Error plots for Data set 2. Error plot after the initial iteration (Top). Error plot after the second iteration using the scheme of [3] (Middle) and the proposed algorithm (Bottom).

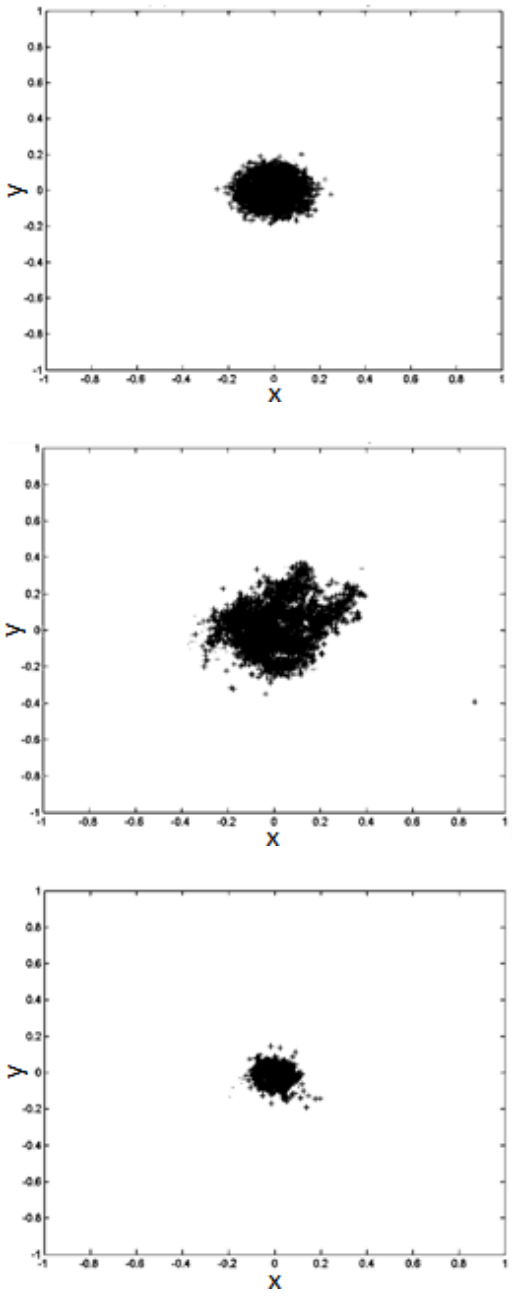


Fig. 22. Error plots for Dataset 3. Error plot after the initial iteration (Top). Error plot after the second iteration using scheme of [3] (Middle) and the proposed algorithm (Bottom)

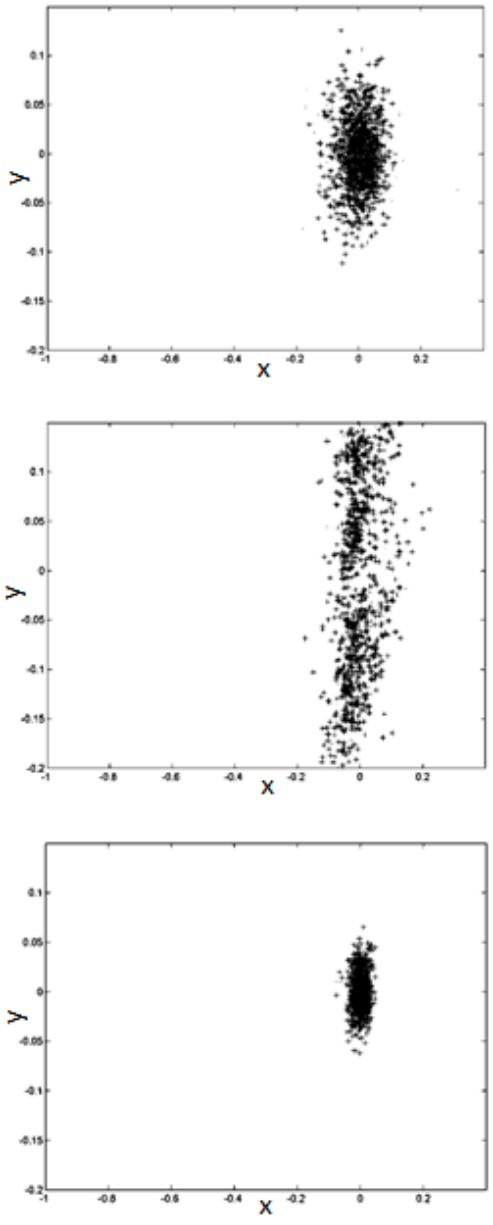


Fig. 23. Error plots for Dataset 4. Error plot after the initial iteration (Top). Error plot after the second iteration using scheme of [3] (Middle) and the proposed algorithm (Bottom)

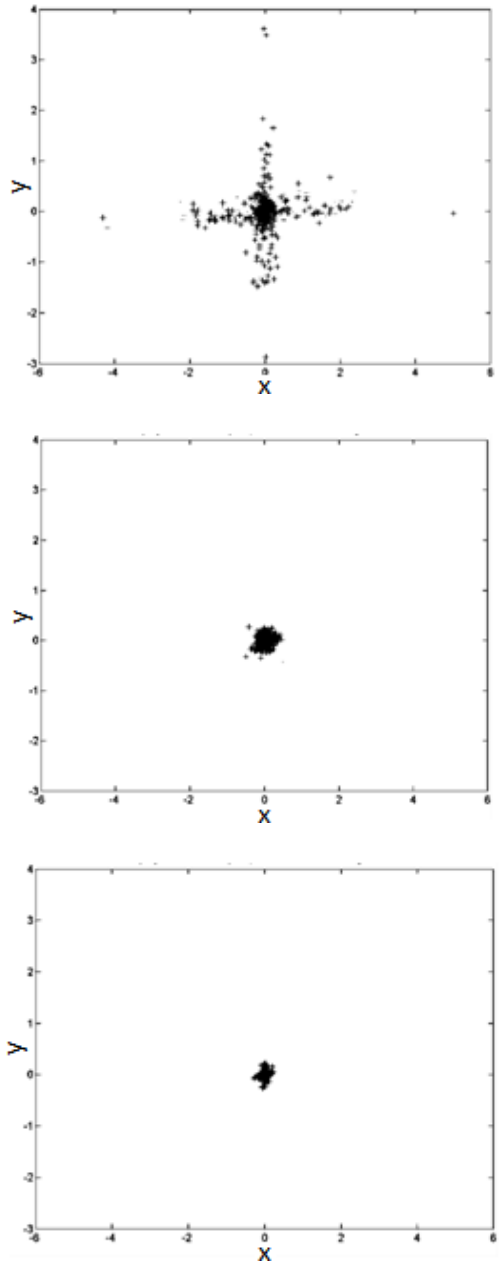


Fig. 24 Error plots for Dataset 5. Error plot after the initial iteration (Top). Error plot after the second iteration using scheme of [3] (Middle) and the proposed algorithm (Bottom)

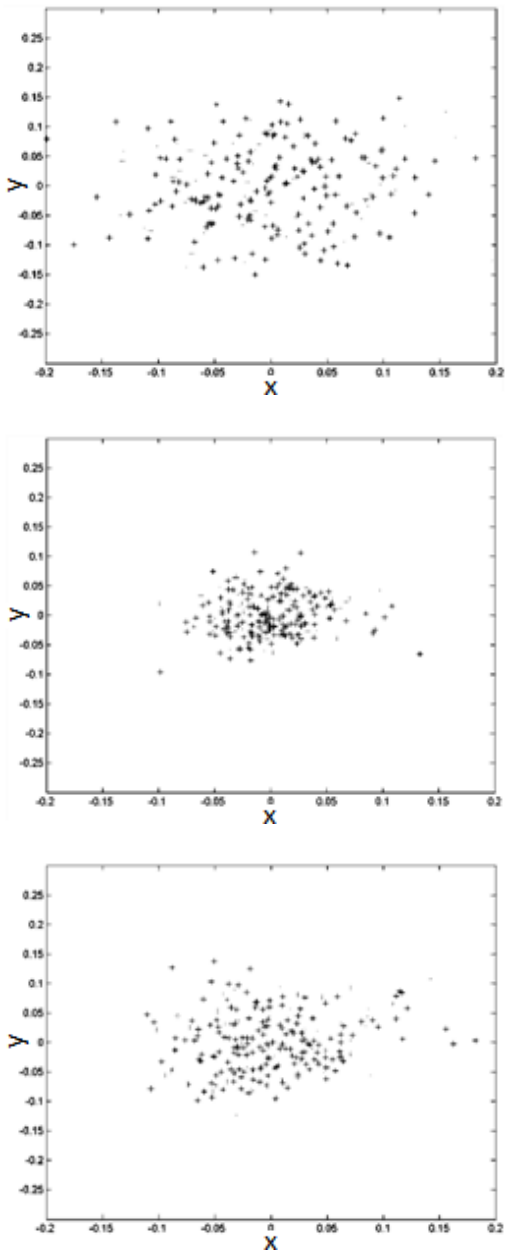


Fig. 25. Error plots for Dataset 6. Error plot after the initial iteration (Top). Error plot after the second iteration using scheme of [3] (Middle) and the proposed algorithm (Bottom)

5.4 Camera calibration parameters and average errors

Tables 2 to 7 show the obtained camera parameters using the proposed algorithm and the scheme of [3], including focal lengths f_x and f_y, deviations in focal lengths d_x and d_y, principal point u_0 and v_0, lens distortion parameters k_1 and k_2, and tangential parameters p_1 and p_2. The average reprojection errors are also shown in these tables.

As it can be seen from Tables 2 to 7, the results obtained using the proposed scheme produce far less reprojection error as compared to the scheme of [3] for all the image datasets, except for Dataset 6. It should be noted that while Datasets 1 to 5 were all obtained by the author of this thesis using a high-resolution camera and the professional calibration pattern from [32], Dataset 6 was downloaded online from the authors of [3]. It was observed that while Datasets 1 to 5 exhibit blur due to focusing problems that arise when high resolution cameras with micron pixel accuracy are employed to focus at a particular specific distance from the camera. The images in Dataset 6 exhibit distorted circular patterns due to down sampling resulting in noisy distorted circular patterns.

Table 2. Camera calibration parameters and average reprojection error for Dataset 1.

Parameters	**Scheme of [3]**	**Proposed scheme**
f_x	22279.21346	22090.17777
f_y	22237.09931	22068.23944
d_x	86.35965	38.55928
d_y	87.98418	39.40207
u_0	1683.32276	1623.81378
v_0	1080.33609	1020.10617
k_1	-0.02655	0.01595
k_2	-0.93514	-7.60280
p_1	0.00427	-0.00009
p_2	0.00100	-0.00053
e	0.046696	0.023234

It should also be noted that the proposed scheme with a circular pattern performs better for Datasets 1 to 5 than the calibration scheme of [3] using a ring pattern which is shown in [3] to produce for Dataset 6 around a 13% reduced reprojection error as compared to a circular pattern. Also, for all the datasets except Dataset 6, the deviations d_x and d_y are significantly less than those produced using the scheme of [3]. This implies that the focal lengths were determined more accurately and with less uncertainty using the proposed scheme.

Table 3. Camera calibration parameters and average reprojection error for Dataset 2.

Parameters	**Scheme of [3]**	**Proposed scheme**
f_x	22495.35230	21668.50730
f_y	22494.99051	21671.52162
d_x	62.00239	14.45032
d_y	61.88834	14.42153
u_0	1637.29917	1618.75375
v_0	1236.25990	1221.70857
k_1	-4.80223	0.01711
k_2	0.00134	-0.56403
p_1	-0.00022	-0.00013
p_2	0.00000	-0.00080
e	0.077935	0.021350

Table 4. Camera calibration parameters and average reprojection error for Dataset 3.

Parameters	**Scheme of [3]**	**Proposed scheme**
f_x	22498.36237	21835.59509
f_y	22495.07816	21836.32356
d_x	61.02624	18.74047
d_y	60.58802	18.59735
u_0	1690.18413	1675.56625
v_0	1274.28067	1230.10968
k_1	0.02846	0.01520
k_2	-5.45367	-0.46903
p_1	0.00247	-0.00013
p_2	0.00028	0.00000
e	0.077054	0.028054

Table 5. Camera calibration parameters and average reprojection error for Dataset 4.

Parameters	**Scheme of [3]**	**Proposed scheme**
f_x	22009.82137	22521.10968
f_y	21976.86463	22505.50304
d_x	373.93271	74.41609
d_y	374.01785	74.40108
u_0	1619.53946	1627.57246
v_0	1044.23160	1061.08447
k_1	-0.01001	0.00241
k_2	2.06366	0.97474
p_1	-0.00087	-0.00082
p_2	-0.00110	-0.00082
e	0.110367	0.020158

Table 6. Camera calibration parameters and average reprojection error for Dataset 5.

Parameters	**Scheme of [3]**	**Proposed scheme**
f_x	23047.78511	22121.17981
f_y	22952.18854	22101.71821
d_x	1069.38927	471.79760
d_y	1075.30680	479.96649
u_0	1747.25549	1755.78360
v_0	1027.08095	1019.73184
k_1	-0.26535	-0.09988
k_2	27.15547	12.80087
p_1	0.01182	-0.00219
p_2	0.00596	0.00180
e	0.152554	0.063248

Table 7. Camera calibration parameters and average reprojection error for Dataset 6.

Parameters	**Scheme of [3]**	**Proposed scheme**
f_x	287.56758	286.85250
f_y	285.61605	284.06077
d_x	0.88390	1.36138
d_y	0.85942	1.32455
u_0	155.97877	156.41917
v_0	120.44190	120.35511
k_1	0.08862	0.08743
k_2	-0.23298	-0.23179
p_1	0.00806	0.00731
p_2	-0.00266	-0.00247
e	0.045308	0.069741

5.5 Effect of number of control points on reprojection error and uncertainty

The number of control points was varied by selecting in each view a square region that consists of the first n control points in the x direction and the first n control points in the y direction where n was varied as shown by the values along the x axis in Figs. 26 to 29.

Fig. 26. And Fig. 27 show the effect of the number of control points n on the reprojection error. These figures plot the number of control points n on the x axis and the average reprojection error on the y axis for Datasets 3 and 6 respectively, for illustration purposes as Dataset 1 was captured by the author of this thesis and Dataset 6 was downloaded online from the authors of the scheme of [3].

Fig. 28. and Fig. 29. show the effect of the number of control points on the mean uncertainty in determining the focal length. These figures plot the number of control points n along the x axis and the mean of the uncertainties d_x and d_y in determining the focal length, along the y axis for Datasets 3 and 6, respectively. The mean uncertainties were computed using (34) as discussed in Section 4.7.

From Figs. 26 and 27, it can be observed that the reprojection error increases with an increase in the number of control points for both the proposed scheme and the scheme of [3]. On the other hand, from Figs. 28 and 29, it can be seen that, as expected, the uncertainty decreases as the number of control points increases for both the proposed scheme and the scheme of [3].

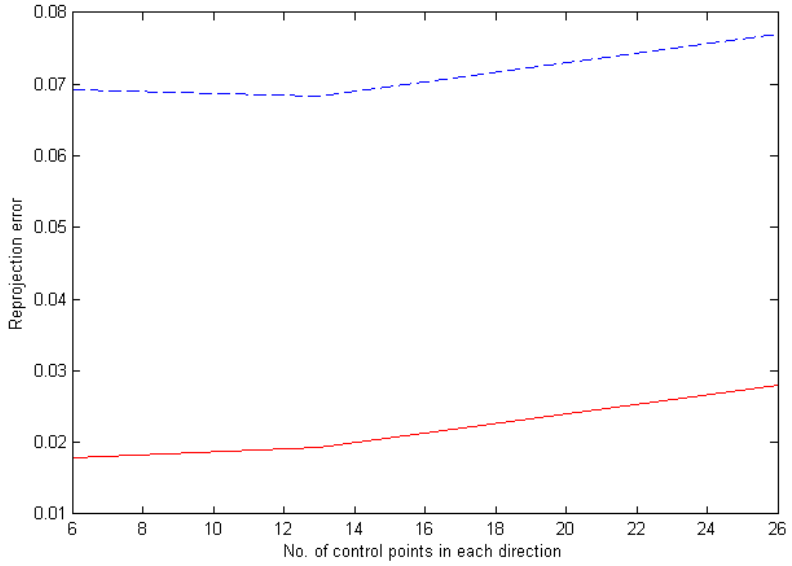


Fig. 26. Number of control points versus reprojection error for Dataset 3 using the proposed scheme (red solid line) and the scheme of [3] (blue dashed line).

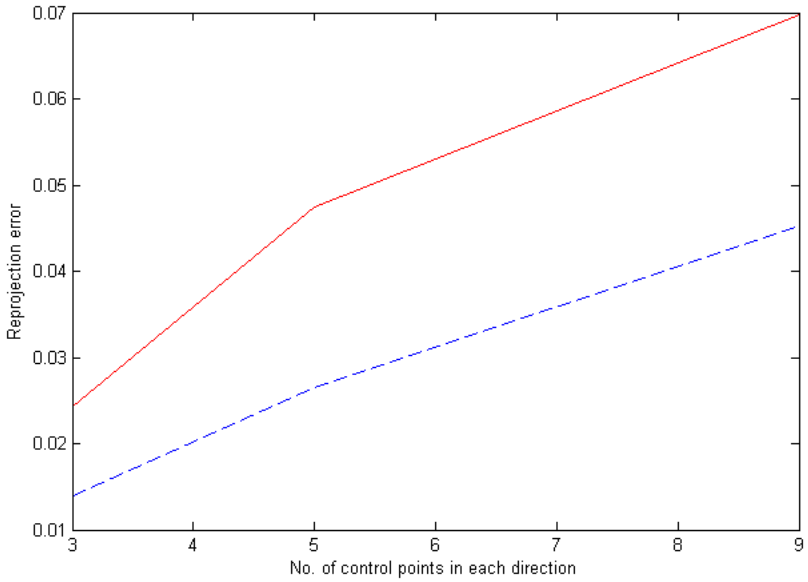


Fig. 27. Number of control points versus reprojection error for Dataset 6 using the proposed scheme (red solid line) and the scheme of [3] (blue dashed line).

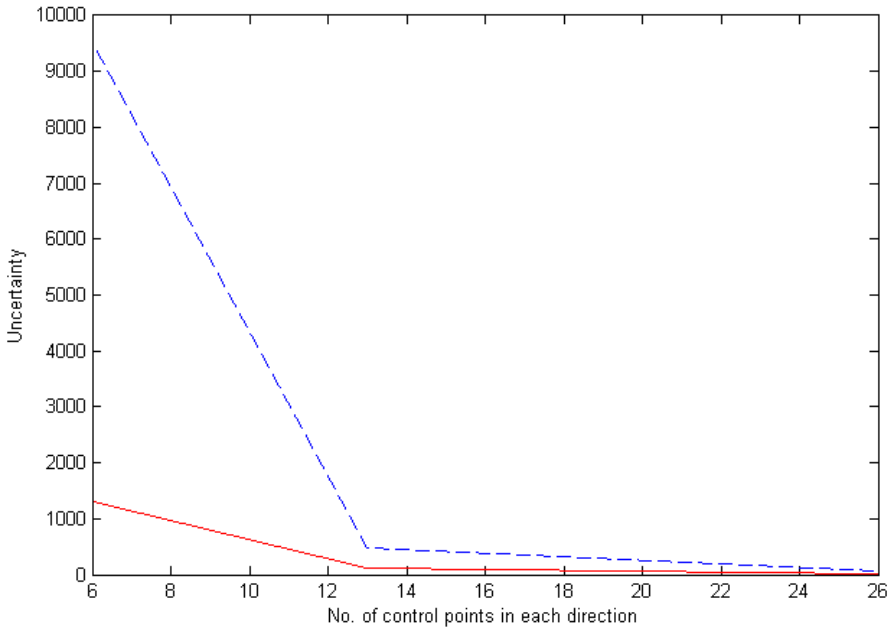


Fig. 28. Number of control points versus uncertainty for Dataset 3 using the proposed scheme (red solid line) and the scheme of [3] (blue dashed line).

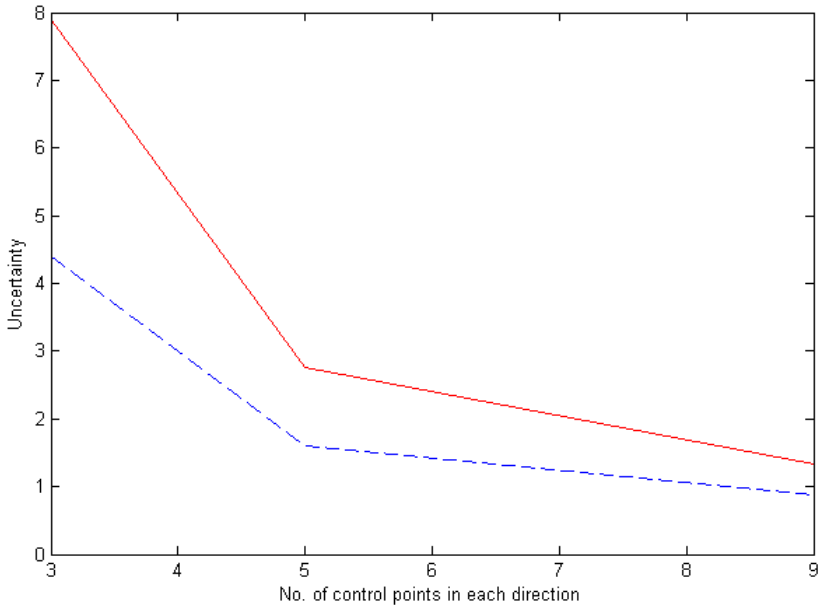


Fig. 29. Number of control points versus uncertainty for Dataset 6 using the proposed scheme (red solid line) and the scheme of [3] (blue dashed line).

6. CONCLUSION

This thesis contributes to the field of image acquisition and processing in general and to the area of camera calibration in particular. This chapter summarizes the contributions of this thesis and proposes several directions for future research.

6.1 Contributions

In this thesis, a novel camera calibration system that utilizes multiple views of a 2D circular calibration pattern, is developed and implemented. The contributions of the thesis can be summarized as follows:

- A novel adaptive thresholding procedure is developed to remove outlier pixels that are introduced due to blur and focusing problems.
- A decision-based ellipse fitting is developed for fitting an ellipse with the minimum possible error in the least square sense.
- This work develops a scheme for the improved localization of the calibration control points on circular calibration patterns by means of adaptive segmentation and decision-based ellipse fitting in the fronto parallel planes of calibration images.
- The proposed camera calibration algorithm can achieve a reduction in the reprojection error as compared to the existing state-of the-art and is shown to be more resilient to blur.
- The proposed method not only can produce a lower reprojection error than the state-of-the-art for high-resolution imaging devices, but can also

determine the camera calibration parameters more accurately, with less deviation and uncertainty.

6.2 Future Research Directions

Possible future directions of research include the following:

- Incorporate calibration using ring pattern – The current work focuses on camera calibration using only circular calibration patterns. The work needs to be extended in the future to also include camera calibration using ring patterns.
- Improve the algorithm to calibrate cameras with low resolution accurately- The current algorithm works better than the state-of-the-art method for high resolution cameras. The work could be extended to improve the calibration of lower resolution cameras as well.
- Optimize the execution time – The current work uses the Levenberg-Marquardt (LM) algorithm for non linear optimization of the camera calibration parameters. The LM algorithm can be replaced by faster algorithms like the dog leg algorithm for faster execution times [33].
- Application in 3D reconstruction – The camera calibration parameters obtained from the present work could be used for 3D reconstruction.

7. REFERENCES

[1] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," *Proc. IEEE CVPR*, pp. 1106-1112, 1997.

[2] Z. Zhang, "A flexible new technique for camera calibration," IEEE Trans. PAMI, vol. 22, no. 11, pp. 1330-1334, Nov. 2000.

[3] A. Datta, J-S Kim and T. Kanade, "Accurate camera calibration using iterative refinement of control points," Proc. IEEE ICCV Workshop, pp.1201-1208, 2009.

[4] S. Belongie, Rodrigues' rotation formula,, [online] Available: http://mathworld.wolfram.com/RodriguesRotationFormula.html.

[5] G. Bradski, A. Kaehler, and V. Pisarevsky, "Learning-based computer vision with Intel's open source computer vision library," INTEL- TECH-J, 9(2), May 2005.

[6] D. C. Brown, "Decentering Distortion of Lenses," Photometric Engineering, 32(3) pp. 444–462, 1966.

[7] B. Caprile and V. Torre, "Using vanishing points for camera calibration," Int'l J. Computer Vision, vol. 4, no. 2, pp. 127-140, Mar. 1990.

[8] O. D. Faugeras, Q. Luong, and S. Maybank, "Camera self-calibration: Theory and experiments," Proc. European Conference on Computer Vision, Lecture Notes in Computer Science vol. 588, pp. 321-334. Springer-Verlag, 1992.

[9] E. Kruppa, "Zur Ermittlung eines Objektes aus zwei Perspek-tiven mit innerer Orientierung," Sitz.-Ber. Akad. Wiss., Wien, math. naturw. Abt. IIa., vol. 122, pp. 1939–1948, 1913.

[10] Y. Seo and A. Heyden, "Auto-calibration from the orthogonality constraints," Proc. International Conference on Pattern Recognition, vol. 1, pp. 1067-1071, 2000.

[11] Y. Seo, A. Heyden and R. Cipolla, "A linear iterative method for auto-calibration using DAC equation," Proc. IEEE CVPR, vol. 1, pp. 880-885, 2001.

[12] S. Sengupta and S. Das, "Modified auto-calibration for 3D reconstruction from multiple views of an object," Proc. IEEE TENCON, pp. 1-6, 2008.

[13] R. Pflugfelder and H. Bischof, “Online auto-calibration in man-made worlds,” Proc. IEEE DICTA, pp. 75-75, 2005.

[14] P. Kuo, J-C. Nebel and D. Makris, “Camera auto-calibration from articulated motion,” Proc. IEEE Advanced video and signal based surveillance, pp. 135-140, Sept 2007.

[15] I. Sobel, “On calibrating computer controlled cameras for perceiving 3D scenes,” Journal of Artificial Intelligence, vol. 5, pp.185-198, 1974.

[16] R. Y. Tsai, “An efficient and accurate camera calibration technique for 3D machine vision,” Proc. IEEE CVPR, pp. 364-374, 1986.

[17] K. Gremban, C. Thorpe and T. Kanade, “Geometric camera calibration using system of linear equations,” Proc. IEEE Conf. Robotics and Automation, vol. 1, pp. 562-567, 1988.

[18] G-Q Wei, Z. He and S. D Ma, “Camera calibration by vanishing points and cross ratio,” Proc. IEEE International Conf. ASSP (ICASSP), pp. 1630-1633, May 1989.

[19] C. Chatterjee and V. Roychoudhury, “Efficient and robust methods of accurate camera calibration,” Proc. IEEE CVPR, pp. 664-665, June 1989.

[20] J. Heikkilä, “Geometric camera calibration using circular control points,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 10, pp.1066-1077, Oct. 2000.

[21] P. Swapna, N. Krouglicof and R. Gosine, “The question of accuracy with geometric camera calibration,” Proc. IEEE CCECE, pp. 541-546, May 2009.

[22] P. Swapna, N. Krouglicof, and R. Gosine, “A novel technique for estimating intrinsic camera parameters in geometric camera calibration,” Proc. IEEE CCECE, pp. 1–7, May 2010.

[23] A.W. Fitzgibbon, M. Pilu and R.B Fisher, “Direct least-squares fitting of ellipses,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 21, no. 5, pp. 476-480 May 1999.

[24] Y. Liu, Y. Wu, M. Wu and X. Hu, “Planar vanishing points based camera calibration,” Proc. IEEE ICIG, pp. 460- 463, Dec. 2004.

[25] J. Dehmeshki, H. Amin, M. Valdivieso, and Xujiong Ye, “Segmentation of pulmonary nodules in thoracic CT scans: A region growing approach,” IEEE Trans. on Medical Imaging, vol. 27, no. 4, pp. 467-480, April 2008.

[26] C. D. Prakash and L. J. Karam, “Camera calibration using adaptive segmentation and ellipse fitting for localizing control points,” Proc. IEEE ICIP, Sept. 2012. (Accepted)

[27] A.W. Fitzgibbon and R.B. Fisher, “A buyer’s guide to conic fitting,” Proc. British Machine Vision Conf., Birmingham, England, 1995.

[28] E. S. Maini, “Robust ellipse-specific fitting for real-time machine vision,” Proc. International Symposium on Brain, Vision, and Artificial Intelligence, pp. 318–327, 2005.

[29] N. Greggio, A. Bernardino, C. Laschi, P. Dario and J. Santos-Victor, "An algorithm for the least square-fitting of ellipses," Proc. IEEE ICTAI, vol.2, pp.351-353, Oct. 2010.

[30] D. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters." SIAM J. Appl. Math. vol. 11, 431-441, 1963.

[31] J. Y. Bouguet, Camera calibration toolbox for Matlab, [online] Available: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

[32] Edmund Optics, Multi frequency grid distortion targets, [online] Available: http://www.edmundoptics.com/testing-targets/test-targets/distortion-test-targets/multi-frequency-grid-distortion-targets/1948

[33] M.L.A. Lourakis and A.A. Argyros, "Is Levenberg-Marquardt the most efficient optimization algorithm for implementing bundle adjustment?," Proc. IEEE ICCV, vol. 2, pp.1526-1531, Oct. 2005.