Fachhochschule Westküste
Hochschule für Wirtschaft & Technik

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterthesis

Christian Heinze

Entwurf und Test eines Kalibrierungsverfahrens zur Berechnung von metrischen Tiefenwerten für 3D-Lichtfeldkameras

Fachhochschule Westküste
Fachbereich Technik

Hochschule für Angewandte Wissenschaften Hamburg
Fakultät Technik und Informatik
Department Informations- und Elektrotechnik

Fachhochschule Westküste - University of Applied Sciences
Faculty of Engineering

Hamburg University of Applied Sciences
Faculty of Engineering and Computer Science
Department of Information and Electrical Engineering

Christian Heinze

Entwurf und Test eines Kalibrierungsverfahrens zur Berechnung von metrischen Tiefenwerten für 3D-Lichtfeldkameras

Masterthesis eingereicht im Rahmen der Masterprüfung
im gemeinsamen Masterstudiengang Mikroelektronische Systeme
am Fachbereich Technik
der Fachhochschule Westküste
und
am Department Informations- und Elektrotechnik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr.Ing. S. Hußmann
Zweitgutachter: Prof. Dr. P. Kölzer

Abgegeben am

Christian Heinze

Thema der Masterthesis

Entwurf und Test eines Kalibrierungsverfahrens zur Berechnung von metrischen Tiefenwerten für 3D-Lichtfeldkameras

Stichworte

Lichtfeldkamera, plenoptische Kamera, Raytrix, metrische Kamerakalibrierung, radiale Tiefendistorsion, Mikrolinsenarray

Kurzzusammenfassung

Ziel dieser Arbeit ist, einen Algorithmus zur metrischen Kalibrierung einer Lichtfeldkamera zu implementieren und zu testen. Für die Kalibrierung wird eine Kalibriertafel mit der Lichtfeldkamera aufgenommen. Aus dem Bild der Tafel werden die Positionen von Merkmalen extrahiert. Die so gewonnenen Punkte werden durch ein geeignes Kameramodell projiziert, so dass eine Fehlerfunktion minimiert werden kann.

Christian Heinze

Title of the paper

Design and test of a calibration method for the calculation of metrical range values for 3D light field cameras

Keywords

Light field camera, plenoptic camera, Raytrix, metric camera calibration, radial depth distorsion, microlens array

Abstract

The goal of this work is to implement and test an algorithm for metrically calibrating a light field camera. For the calibration, an image of a calibration target is taken with the light field camera. The positions of features are extracted from this image. The points obtained in this way are then projected through a suitable camera model, so that an error function can be minimized.

Contents

List of Tables

List of Figures

1 Introduction

Light field cameras are capable of capturing 3D information about a scene with a single shot and through a single main lens. This is achieved by placing a microlens array in front of the image sensor of the camera. The 3D depth estimation process of a light field camera works similarly to a stereo camera system and is based on disparities. With a light field camera, it is possible to generate an intensity image and a corresponding depth map at the same time. Once the depth map is known, the intensity image can be refocused to increase the depth of field.

This work is created in collaboration with Raytrix GmbH[1]. The algorithms and camera models implemented for this work are to be incorporated in to the next release of the Raytrix RxLive software package. The light field cameras manufactured by Raytrix give the distance from the camera in virtual depth units. The goal of this work is to implement a camera calibration algorithm, so that the virtual depth values can be converted to metric depth values. This would for the first time allow metric measurements in 3D space with a Raytrix light field camera.

To allow a precise metric measurement, it is not enough to find the relationship between virtual depth and metric depth. It is also necessary to rectify any geometric distortions, because these would affect the measurements. Therefore, state of the art models for main lens aberrations are analyzed and implemented. For the first time, the rectification of the main lens aberrations is extended to 3D space. Also, a novel way of rectifying geometric aberrations related to the microlens array of the light field camera is introduced.

The metric calibration process is based on calibration targets, which are imaged with the Raytrix light field cameras. Knowledge about the geometry of these targets is leveraged to establish a relation between metric distances on the target and distances between features detected in the images of these targets. The metric calibration process is designed so that it can later be performed by users of the RxLive software. This means that the algorithm must be very robust and tolerate input images taken under adverse conditions.

[1] http://www.raytrix.de

This work is divided into three main parts. First, the theoretical background necessary to implement the calibration algorithm is given. This includes details about main lens projection, geometric aberrations and the image formation in a light field camera. Also, the algorithms necessary to build an automatic camera calibration are introduced.

Next, the implementation is detailed. This includes the design of a calibration target, which must be optimally suited for light field cameras. The target design determines the algorithms, which must be implemented to automatically and reliably detect features in the image of the target. The final part of the implementation is an optimization step to minimize the reprojection error function. For the implemented algorithms, flow diagrams are given and the parameters of the algorithms are explained.

Finally, results are presented. These test the robustness and accuracy of the feature detection, the target design and the projection models chosen for this work, and finally the optimization step to find the parameters for the projection model.

After that, a short conclusion sums up this work and gives an outlook on future work.

2 Theoretical foundations

The next sections give an overview of the theory needed to implement a camera calibration algorithm for a Raytrix light field camera. But first, a short introduction into general optical systems is given. Figure 2.1 shows a schematic diagram of a basic imaging system.

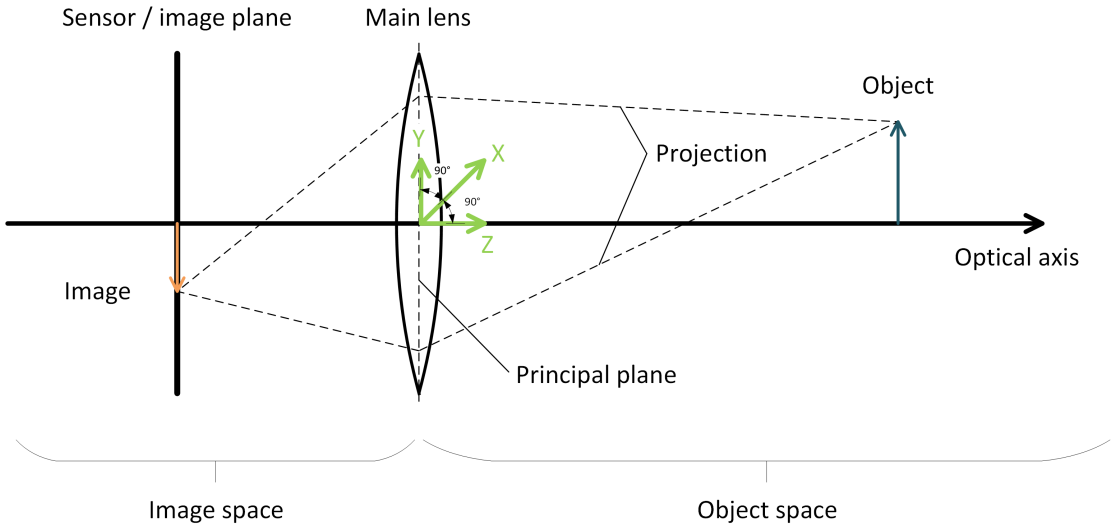


Figure 2.1: Overview of an optical system

This camera consists of a main lens and a sensor. In this work, sensor can mean a digital sensor as well as film, which is used to record the incoming light. When analyzing the projection of light through the main lens, the coordinate system is best placed at the center of the main lens, as seen in figure 2.1. The X and Y axes lie in the plane of rotational symmetry of the main lens, while the Z axis coincides with the optical axis and is perpendicular to this plane. In general, the side of the main lens on which the sensor is located is called the image space, while the other side of the lens is called the object space. When referring to lateral coordinates, these are only the coordinates in X and Y direction, normal to the optical axis (Z axis). Depth values or object/image distances are measured along the Z axis. Depth values can be measured as absolute distances from the main lens, and so can be positive on both the image side and the object side of the lens.

When analyzing images on the sensor or projections into the space behind the sensor, it is more useful to center the coordinate system on the sensor, as in figure 2.2.

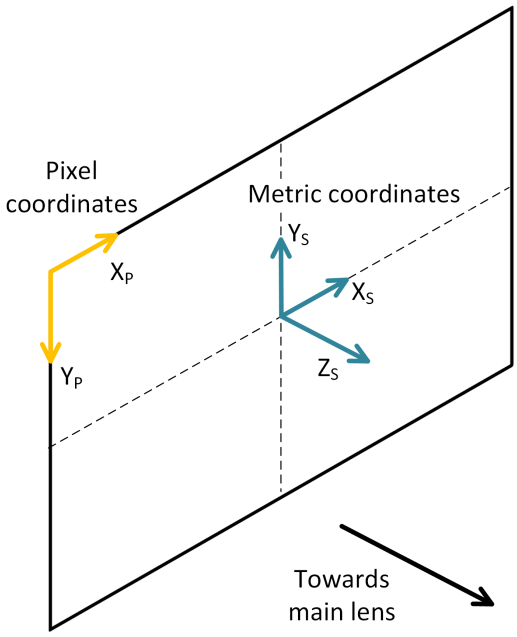


Figure 2.2: Sensor coordinate systems

To measure distances on the sensor, both pixel values and millimeter values can be used arbitrarily, because the pixel size and the number of pixels in X and Y direction on the sensor are generally known.

The size of the pixels in millimeters and the number of pixels in X and Y directions are the first of the so called intrinsic parameters of the camera system. The intrinsic camera parameters describe the camera and the way the images are generated. Other intrinsic parameters are the focal length f_L of the main lens, the focus distance T_L the main lens is set to, but also coefficients which describe projection errors of the main lens. Separate from the intrinsic parameters are the extrinsic parameters of a camera system. These describe the pose of the camera in 3D space. To allow a metric calibration, usually both the intrinsic and the extrinsic parameters of the camera must be known. The next sections will introduce the models which are controlled by these parameters.

2.1 Projection models

To be able to perform the metric calibration as described in this work, a mathematical model of a light field camera with an associated main lens must be found. The three established models for the projection of objects onto a sensor are the pinhole model, the thin lens model, and the thick lens model. These models will be introduced here, based on the book Optics by Eugene Hecht [1].

Since a metrically calibrated light field camera is supposed to measure distances in 3D, monochromatic aberrations of the main lens must be corrected. These aberrations show themselves by distorting the geometry of the imaged object. A new distortion model is introduced here, which allows for the first time to model and correct curvature of field.

2.1.1 Pinhole model

The pinhole model is used in many traditional approaches to camera calibration, since it is the simplest model with which the occurring relationships can be described. This model is based on the assumption that the complex main lens can be approximated by a pinhole in a certain distance from the imaging sensor. This simplifies the actual optical phenomena like diffraction at the boundaries between different optical media or the radii of the involved lenses and replaces them with a proportional equation (see equation 2.1). This equation describes the relationship between the image distance B_P (distance between the imaging sensor and the pinhole) and the object distance G_P (distance between the object and the pinhole). Let g_P be the height of an object, then b_P is the height of projection of this object onto the sensor:

$$\frac{b_P}{B_P} = \frac{g_P}{G_P}. \tag{2.1}$$

Figure 2.3 shows the parameters used in the pinhole model.

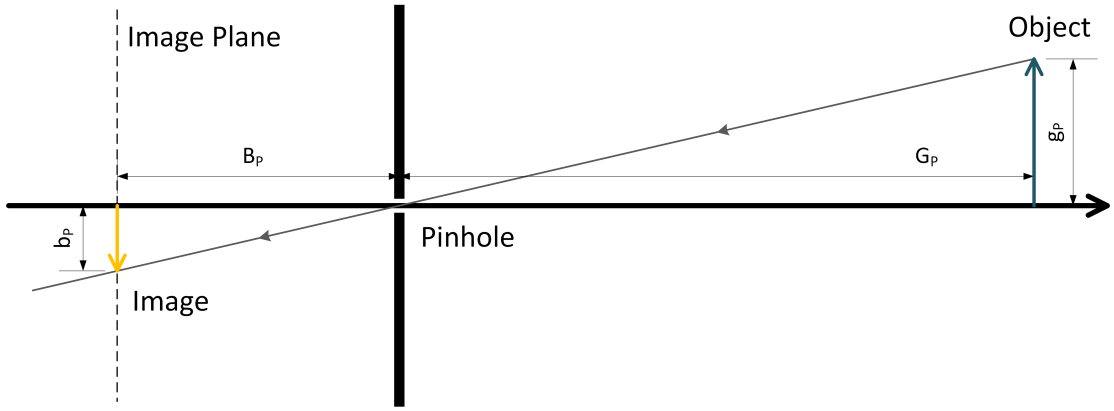


Figure 2.3: Pinhole model

A special property of the pinhole model is that the infinitesimally small pinhole aperture causes an infinite depth of focus. Every point of the object is imaged on the sensor by just a single ray of light. This means that there is no focus point like there would be with a lens, which means that there is no three dimensional projection from the object space to the

image space. The distance of an object to the pinhole only affects the size of it's image on the sensor, but not the sharpness of the image. This can be seen from equation 2.1. Only the lateral (X, Y) position of a projection onto the imaging sensor can be calculated, but not the distance in direction of the Z-axis.

Despite this limitation, the pinhole model can be used for many traditional camera calibration procedures. For objects which are in the depth of field of a real lens, the image distance B_P of the pinhole model behaves similarly to the focal length f_L of a real lens. By using this equivalence, the intrinsic lens parameters can be estimated with the pinhole model in conventional camera calibration approaches. In these approaches, a known 2D position on the image sensor is related to an estimation of a 3D position in the object space.

The calibration procedure detailed in this work is different, because a light field camera can produce 3D information on the image side of the lens. This fact leads to an approach that uses a 3D (object side) to 3D (image side) correspondence for the calibration. Since the pinhole model only yields 2D data on the image side, it is not suited for the approach used in this work.

2.1.2 Thin lens model

The thin lens model generalizes the pinhole model by adding the focal length f_L of the main lens. This main lens parameter is a measure of the refractive power of the modeled lens. The thin lens model ignores the thickness of the lens in direction of the optical axis. The thick lens model takes this thickness into account and will be introduced in the next section.

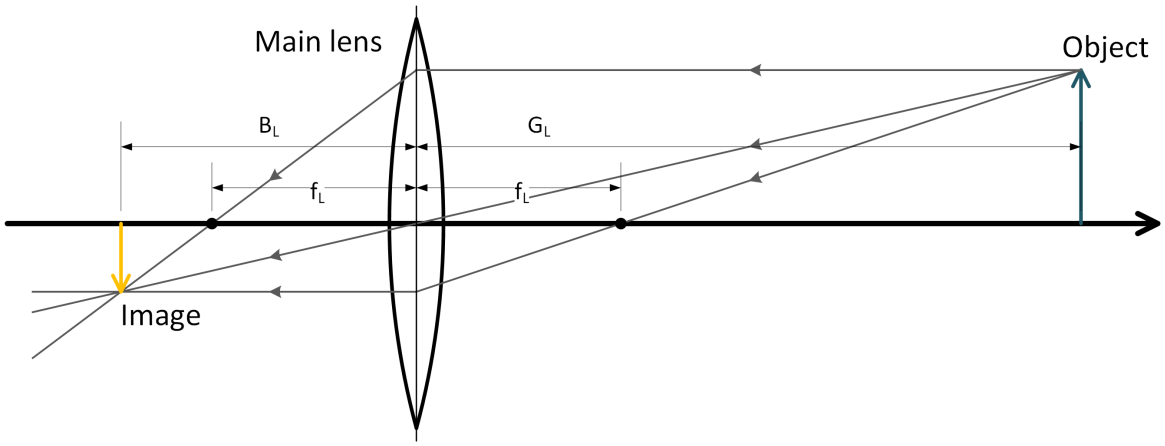


Figure 2.4: Thin lens model

Figure 2.4 shows an overview of the thin lens model with the newly introduced focal length f_L. The focal length is measured from the principal plane H_L in both directions of the optical axis. In contrast to the pinhole model, not only a single ray of light reaches the

imaging sensor for each point on the object. Instead, the main lens refracts a cone of light emitted from the object and focuses it into a single point. The distance B_L where the cone of light emitted from an object at distance G_L converges to a single point can be calculated using equation 2.2:

$$\frac{1}{f_L} = \frac{1}{G_L} + \frac{1}{B_L}. \tag{2.2}$$

The first result this equation yields, is that objects at infinite distance from the lens will be in focus at the focal length of the main lens. With the object distance G_L becoming smaller, the point of focus moves away from the lens as the image distance B_L becomes larger. This means that the thin lens model relates 3D information from the object side of the lens to 3D information on the image side of the lens.

The limited depth of focus that real lenses have is also explained by the thin lens model. Observing the cone of light on the image side of the lens, one can see that at distances different from B_L, the cone of light is not focused to a point, but has a growing lateral extent depending on the distance from the point of focus. This means that the light of a single object point is no longer projected to a single point in image space, but extends over a large area and mixes with light coming from other points of the object. When a sensor is placed at a distance different from B_L, a loss of sharpness can be observed. This can be perceived in many images taken with real lenses. This loss of sharpness is an important factor when building correspondences between points in the image and known points in the object space. The lateral position in the image can no longer be measured precisely, when image sharpness is lost. A robust calibration technique must take this into account.

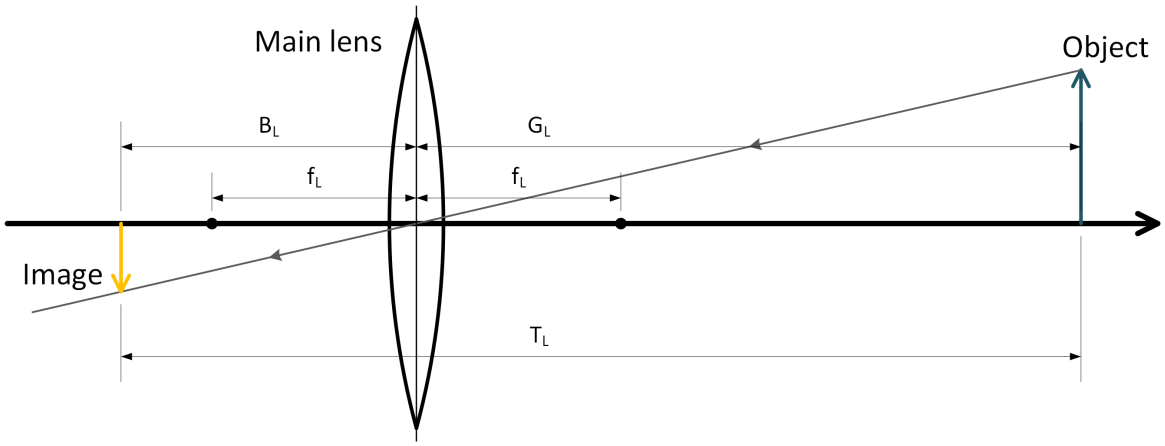


Figure 2.5: Focus distance T_L in the thin lens model

For a real lens, the position of the principal plane of the lens is generally not known. This information is critical for any camera calibration technique and can be calculated with the

thin lens equation. Instead of the position of the principal plane, the focus distance T_L is supplied by the user (figure 2.5). The focus distance can either be measured or read from the focus scale of the lens that is used. It is the sum of the image distance B_L and the object distance G_L and extends from the image sensor to the the plane in object space, which would be in focus.

$$T_L = B_L + G_L \tag{2.3}$$

Calculating the image distance B_L from the focus distance T_L can be done through the following equation:

$$B_L = \frac{T_L}{2}\left(1 - \sqrt{1 - 4\frac{f_L}{T_L}}\right). \tag{2.4}$$

Using the value for T_L together with the focal length imprinted on the lens, a projection from 3D object space to 3D image space can be calculated. This means that the thin lens model satisfies all requirements and can be used in the calibration technique detailed in this work.

2.1.3 Thick lens model

The thick lens model is a further generalization of the thin lens model. As it's name implies, it takes the thickness of the lens into account. It can be shown that any combination of thin lenses can be approximated by a thick lens. Figure 2.6 shows that the thin lens model is extended by splitting a thin lens into two. While the thin lens only had a single principal plane H_L, the thick lens has two principal planes H_{L1} and H_{L2}. The thickness of the lens is the distance between these two planes. For the thick lens model it is assumed that all rays of light travel through this space parallel to the optical axis, without any influence by the other parameters of the lens. This means that the thickness of the lens $H_{L1} - H_{L2}$ is added to any projection as an offset in direction of the optical axis, and has no other effect on the projection.

The thick lens model should be applied when the thickness of a lens is large compared to the focal length f_L. Photographic lenses are usually made up of several glass elements, which can have a significant total thickness when added up. Macro lenses used at $1:1$ magnification and microscope optics must usually be modeled as a thick lens (see Johannsen et al. [2], page 4).

Given these advantages of the thick lens model, it seems to be ideally suited for use in this camera calibration algorithm. Unfortunately, the lens thickness $H_{L1} - H_{L2}$ can not

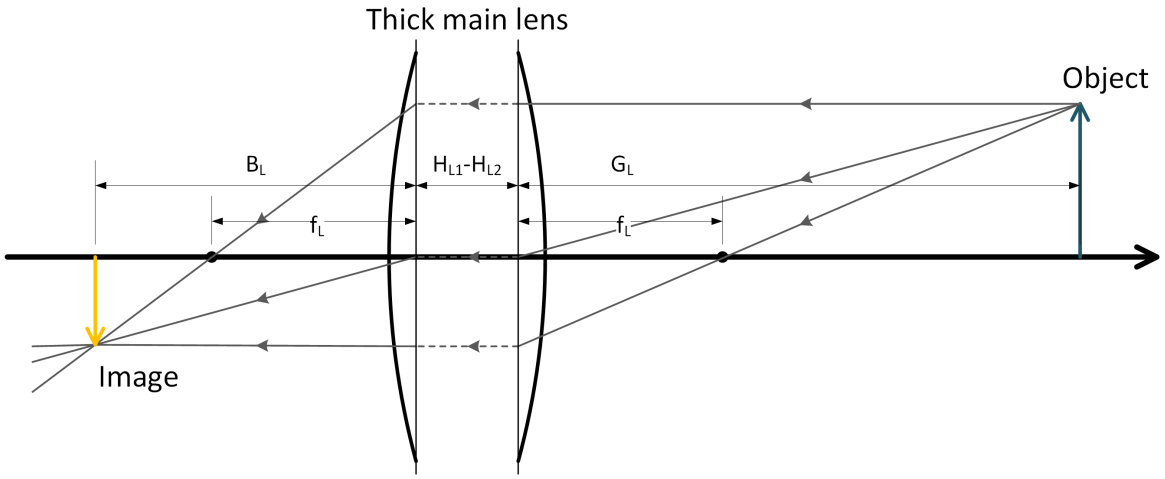


Figure 2.6: Thick lens model

be estimated in a robust way. Without access to the design specifications of a lens, no initial value can be given for a parameter optimization of the lens thickness. Also, the distance $H_{L1} - H_{L2}$ is a constant offset added to depth values during calculation and has no discernible influence on the projected image.

Due to these reasons, the thin lens model is best suited for this calibration algorithm. It allows calculating a metric projection through an ideal lens. Real lenses usually have aberrations, which can distort the geometry of the image. To make precise measurements in the 3D image space possible, the geometric distortions must be taken into account.

2.2 Main lens aberrations

The term monochromatic aberrations specifies a number of optical phenomena, which affect all wavelengths of light in the same way. This must be differentiated from chromatic aberrations, which affect the color reproduction of a lens. Figure 2.7 shows an image of a black and white checkerboard, where chromatic aberration has influenced the imaging process. These kinds of aberrations are not rectified through the calibration technique proposed in this work.

This work is focused on rectifying the geometric distortions introduced by the main lens of an optical system. Monochromatic aberrations have different kinds of effects which can be described by different models. Figure 2.8 shows an example of geometric distortions which are commonly found on lenses.

It can be seen that theses types of distortion must be rectified, before exact metric measurements in the image are possible. The distortions shown in figure 2.8 are of the same type,

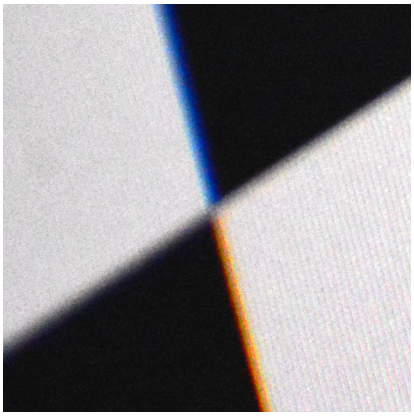


Figure 2.7: Chromatic aberration causes colored fringing on high contrast edges

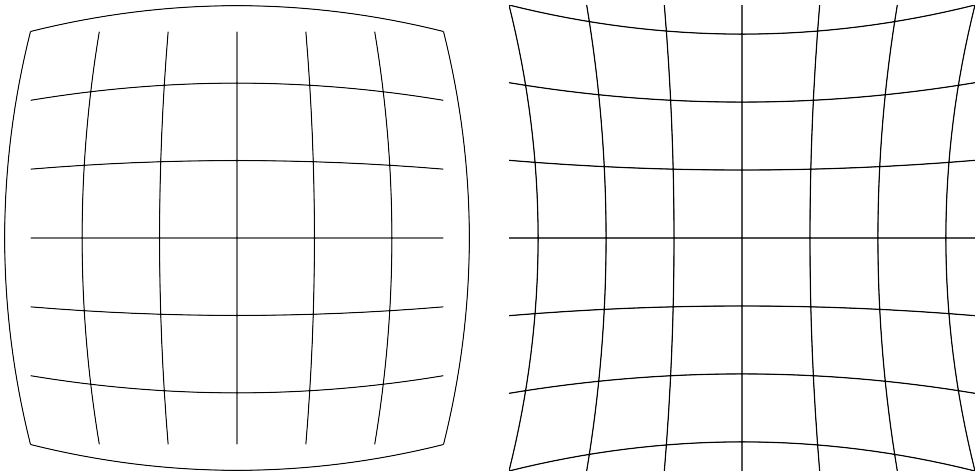


Figure 2.8: Examples of barrel and pincushion lateral distortion from https://en.wikipedia.org/wiki/Distortion_(optics) [28.12.13]

because they affect the lateral position of points in the image. These images show the effect created by applying the lateral distortion model detailed in the next section.

2.2.1 Lateral distortion

The lateral distortion model used in this work is a subset of Brown's distortion model. Brown's model [3] was introduced in 1966 and is widely used to rectify geometric distortion in camera calibration algorithms [4] [5]. Brown's model is also implemented in many

software libraries for camera calibration like for example the OpenCV calib3d module[1] or the MVTec Halcon image processing software[2].

According to Brown's model, the relation between an undistorted point (x_u, y_u) and a distorted point (x_d, y_d) is given by equation 2.5:

$$\begin{aligned} x_u &= (x_d - x_o)(1 + k_1 r^2 + k_2 r^4 + ...) \\ &\quad + (p_1(r^2 + 2(x_d - x_o)^2) + 2p_2(x_d - x_o)(y_d - y_o))(1 + p_3 r^2 + ...) \\ y_u &= (y_d - y_o)(1 + k_1 r^2 + k_2 r^4 + ...) \\ &\quad + (p_2(r^2 + 2(y_d - y_o)^2) + 2p_1(x_d - x_o)(y_d - y_o))(1 + p_3 r^2 + ...) \end{aligned} \tag{2.5}$$

The amount of distortion in radial direction is controlled by the parameters k_i, while the parameters p_i allow for a distortion that is not radially symmetric. These coefficients act depending on the radius r which is the euclidean distance from the distorted point x_d, y_d to the center point of the distortion x_o, y_o. The center of the distortion usually coincides with the image center, but the model allows arbitrary offsets.

The equation can be split into two parts, separating the purely radial distortion (equation 2.6)

$$(1 + k_1 r^2 + k_2 r^4 + ...) \tag{2.6}$$

from the tangential distortion (equation 2.7)

$$(p_1(r^2 + 2(x_d - x_o)^2) + 2p_2(x_d - x_o)(y_d - y_o))(1 + p_3 r^2 + ...), \tag{2.7}$$

which depends on the radius and the lateral position in the image. Both the radial and the tangential distortion only affect the lateral position of points in the image. Radial distortion models the effects that an imperfect lens has on the projection. Tangential distortion on the other hand, can be used to model a perfect lens, which is not perfectly aligned with the sensor. Figure 2.9 shows the effect of tangential distortion on a square grid.

The tangential distortion model proposed by Brown is a heuristic way of modeling lens alignment errors. In traditional 2D camera calibration approaches, it can be used to rectify the effects of a skewed lens. Yet when using a light field camera, which can capture 3D data, a representation based on physics can be found. The tangential distortion which is present in the 2D image is only a side effect of the skewed lens system. When 3D data is available, the lens skew results in skewed 3D points in the image space. When a plane is mounted

[1] http://docs.opencv.org/trunk/doc/tutorials/calib3d/camera_calibration/camera_calibration.html
[2] http://www.halcon.com/

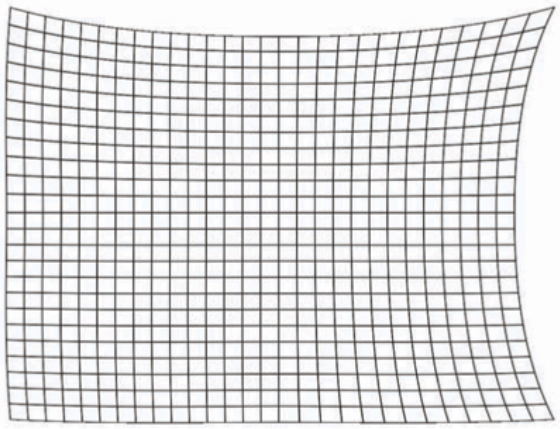


Figure 2.9: Example of tangential distortion caused by a skewed lens from https://ieeexplore.ieee.org/ieee_pilot/articles/06/ttg2009061307/assets/img/article_1/fig_8/large.gif [28.12.13]

parallel to the sensor and imaged with a skewed lens, the resulting 3D data in image space is a skewed plane. This can be seen in figure 2.10. This effect is known from the Scheimpflug principle [6].

Since the skew of the plane can be recorded with a light field camera, it is more useful to explicitly model the skew of the lens, than to use heuristic parameters for tangential distortion.

Therefore the lateral distortion model used for this work is reduced to the radial distortion component (equation 2.6), while the skew of the lens can be dealt with in 3D space, which means it is no longer only a lateral form of distortion.

To model the skew of a lens, the pose of the lens in 3D space relative to the sensor must be known. In general, this means that six degrees of freedom must be parametrized: the translation from the sensor center to the lens center X_L, Y_L, Z_L and three rotation angles σ_L, θ_L, ρ_L. Figure 2.11 shows that the three parameters for the translation are already known. Equations 2.8 to 2.10 give these equivalences:

$$X_L = x_o \tag{2.8}$$

$$Y_L = y_o \tag{2.9}$$

$$Z_L = b_L = \frac{T_L}{2}\left(1 - \sqrt{1 - 4\frac{f_L}{T_L}}\right) \tag{2.10}$$

The offsets of the lens center in X and Y directions are the same parameters, which are used in the lateral distortion model for the offset of the distortion center (see equation 2.5). The

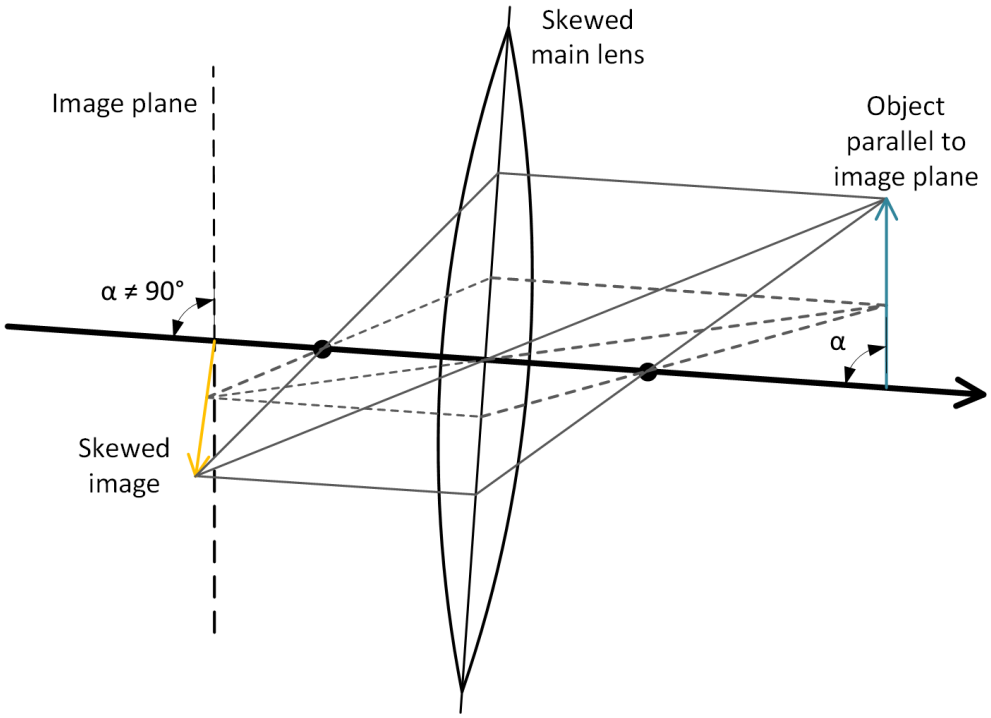


Figure 2.10: Projection through a skewed lens

translation of the lens in direction of the optical axis Z_L is equal to the image distance b_L (see figure 2.4).

To describe the orientation of the lens, only two new parameters are needed. The rotation ρ_L about the optical axis Z does not need to be modeled, because lenses are generally rotationally symmetric. This means that a rotation about the optical axis has no effect on the image. Only the angles θ_L and σ_L must be added to the main lens model. The angle between the optical axis of the main lens and the normal vector of the sensor in the $X - Z$ plane is θ_L, and the respective angle in the $Y - Z$ plane is σ_L.

When these angles are used for projecting through the main lens, they will not only have an effect on the lateral position of the projection, but also on the depth. The next section deals with radial depth distortion, which only has an effect on the depth values.

2.2.2 Radial depth distortion

The final distortion that is rectified with this calibration algorithm is the curvature of field of the main lens. This effect is also known as Petzval field curvature [7]. Curvature of field means that a plane parallel to the sensor is projected to a curved surface in image space (see

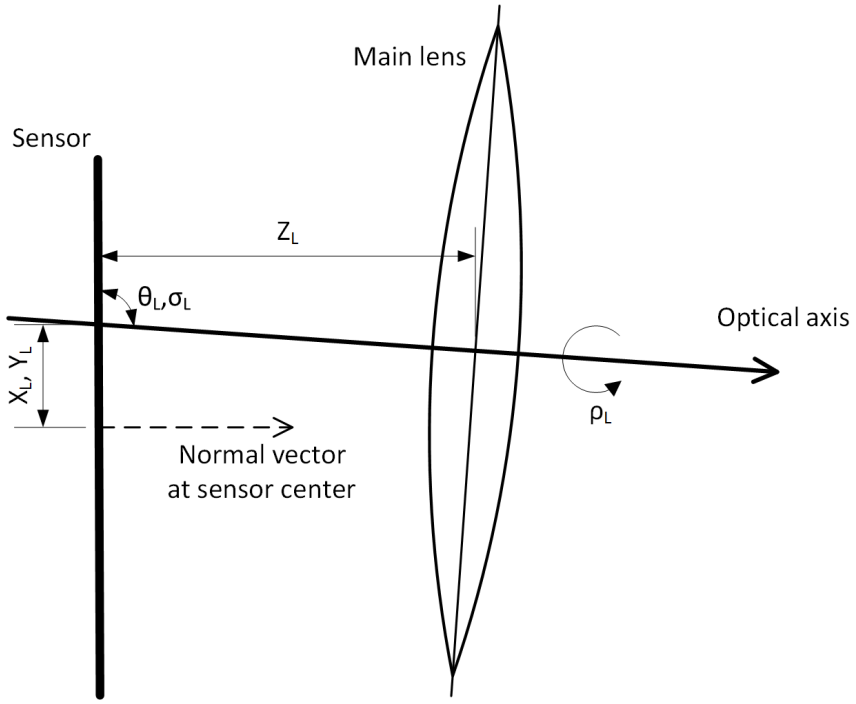


Figure 2.11: Lens pose parameters

figure 2.12). In traditional 2D camera systems, this effect leads to a loss of sharpness when the curved surface diverges too much from the sensor plane.

The field curvature has to be corrected during lens design, and no form of 2D calibration can rectify this aberration. But when using a light field camera, this distortion can be measured in 3D space and undistorted with an appropriate model. In a recent work on calibrating Raytrix light field cameras, the authors proposed a novel approach to this problem (see [2], page 7).

Viewing the projection of a lens with residual curvature of field in 3D space (figure 2.13), it becomes obvious that this kind of distortion can be modeled similarly to the radial distortion (equation 2.6). The amount of distortion depends on the radial distance r from the distortion center (x_o, y_o) and is controlled by the coefficients d_i. It must be noted that this distortion does not act upon the lateral position of a point, but on the point's depth value along the optical axis.

In addition to this radial influence, the amount of distortion also depends on the distance of the object from the main lens. This means that for example a plane at infinity distance is projected to a flat plane in the image space, but a plane closer to the main lens is projected to a curved surface. To account for this, an additional parameter d_d is added to the radial depth distortion model. This new model allows calculating the undistorted depth value of a point z_u from it's radial position and it's current (distorted) depth z_d:

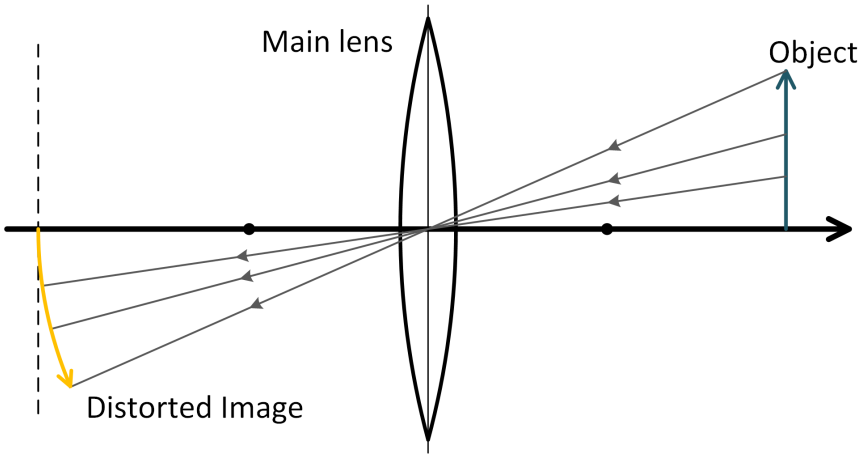


Figure 2.12: Petzval field curvature distortion in direction of the optical axis

$$z_u = z_d + (1 + d_d z_d)(d_1 r^2 + d_2 r^4 + ...). \tag{2.11}$$

As has been shown by Johannsen et al., the influence of the distorted depth z_d is purely linear. Therefore only one parameter d_d is needed to model the relationship between the depth and the amount of depth distortion (see [2], page 8). Using the correct parameters for this new model, the main lens depth distortion can be rectified.

When using all the projection and distortion models as described in this chapter with the correct values for the parameters, it is possible to measure correct metric values in 3D space with a light field camera. The next chapter explains how the Raytrix light field camera is able to record the 3D data.

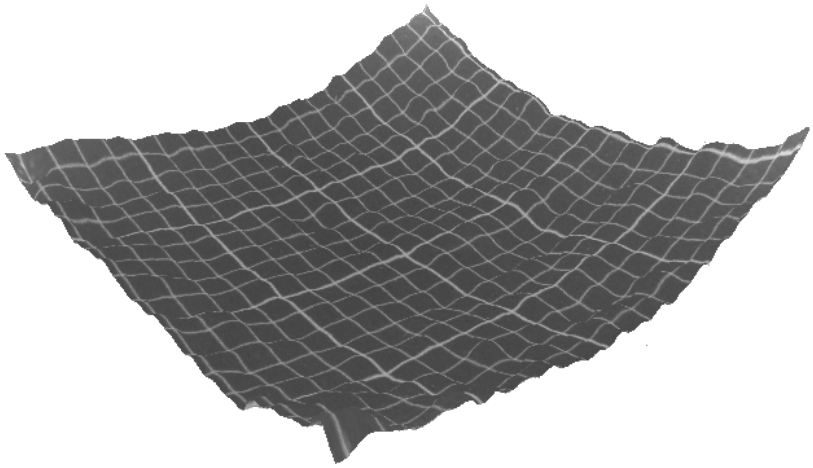


Figure 2.13: 3D image of a flat surface showing Petzval field curvature

2.3 3D light field cameras

In the next sections, the plenoptic or light field camera will be introduced in detail. First, a short account of the history of the light field camera is given. Then the principle behind the Raytrix light field cameras is explained, with respect to both depth estimation and computation of intensity images. Finally, an in-depth look is taken at distinctive features of the Raytrix light field cameras, and how they can be used to improve the results of this calibration algorithm.

2.3.1 History of light field cameras

The first descriptions of light field cameras date back to the early 1900s. The underlying principle which links all light field cameras is to capture not only the intensity of the light hitting each point on the sensor, but also the direction where the light is coming from. This is usually done by recording a multitude of views from slightly different perspectives. The earliest design by Yves in 1903 [8] proposes the use of an array of pinholes directly in front of the film plate. Each pinhole would then project a small sub-image onto the film, with the perspective differing a little between neighboring pinholes. In 1908, Lippmann proposed using an array of small lenses [9] and from there the idea was developed further until today.

With the advent of fast, low-cost computers and high resolution digital sensors, a computational analysis of these light field images was feasible for the first time. A recent effort was the introduction of the first portable consumer light field camera by Lytro[1], based on work by Ng published in 2005 [10]. This camera is based on the light field 1.0 principle, where the resolution of the computationally reconstructed image is equal to the number of lenses on the microlens array (540 x 540 microlenses for the Lytro camera).

There have also been efforts to build light field cameras by using arrays of many 2D cameras[2]. The methods required to calibrate these camera arrays are unfortunately quite different to what is needed for a monocular light field camera. Therefore these camera arrays are not discussed here any further.

This work is based on a light field 2.0 camera developed by the company Raytrix based in Kiel, Germany. A detailed explanation of the principle behind light field 2.0 is given in the work of Perwass published in 2010 [11]. The next sections will give a short introduction into the topic.

[1] https://www.lytro.com/
[2] http://graphics.stanford.edu/projects/array/

2.3.2 Depth estimation

In a plenoptic 2.0 camera like the ones produced by Raytrix, a microlens array is placed directly in front of the image sensor. The placement of this microlens array (MLA) can be seen in figure 2.14.

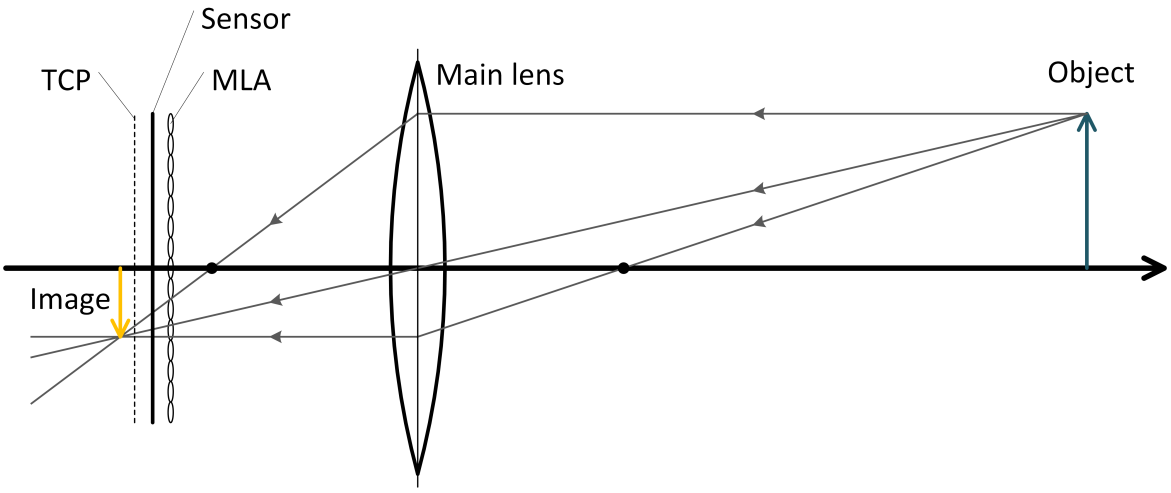


Figure 2.14: Schematic drawing of Raytrix light field camera

The total covering plane (TCP) is the plane, on which the main lens must be focused. If the projection of the main lens is any closer to the sensor, no depth estimation is possible. When the MLA is placed in the correct distance from the sensor, the microlenses will project small sub-images onto the sensor. As can be seen in figure 2.15, each of these sub-images shows a slightly different view of the object. When an object point can be seen in at least two sub-images, the so called virtual depth of the object point can be estimated. This depth estimation works similarly to stereo vision approaches [12] [13].

To identify a pair of points in the sub-images, areas with high local contrast are selected. For these areas, the cross correlation of small pixel patches is computed across lens pairs. If the cross correlation is above a certain threshold, the two patches can be assumed to show the same object point. It is important, that only areas with high local contrast are used. Comparing two patches without any contrasting features will result in a high cross correlation in many places, and would therefore give ambiguous results. This means that for good depth estimation results, the surface texture of an object must have high local contrast.

To be able to calculate the virtual depth of an object point P_V, the positions i_1 and i_2 of the point pair as seen in two microlenses must be known. For these microlenses, the centers of projection c_1 and c_2 must be known as well. Since only the 2D position of the points on the sensor can be known, the pinhole model is sufficient to project the points behind the sensor

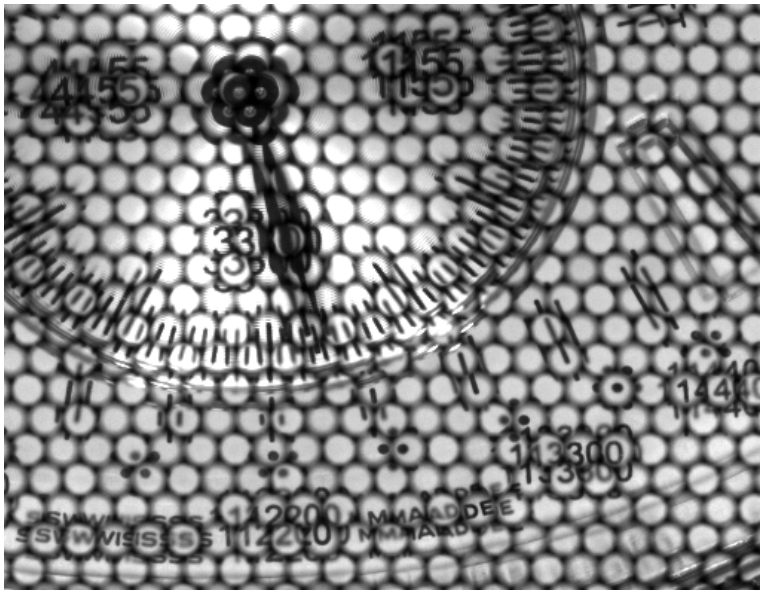


Figure 2.15: Raw image captured with Raytrix light field camera

and into virtual 3D space. The depth where the two projected rays would intersect in 3D space is the virtual depth v which corresponds to the distance of the object from the camera. A schematic overview of this is given in figure 2.16.

The virtual depth v is defined as the ratio between the object distance a and the image distance b:

$$v := \frac{a}{b}. \tag{2.12}$$

These distances are measured in the same direction from the principal plane of the microlens array. Although the distance b is constant but generally not known and a is the metric distance of the projection behind the MLA which shall be found, it is still possible to calculate the virtual depth v. Given the distance D between the centers of the microlenses c_1 and c_2 and the distance $i_1 - i_2$ between images of the same point in two microlenses, the virtual depth v can be calculated with the intercept theorem (see [2], page 6):

$$v := \frac{a}{b} = \frac{D}{D - (i_1 - i_2)}. \tag{2.13}$$

From this equation (2.13) it follows, that to get metric information about the projection behind the MLA, the virtual depth v and the distance from the MLA to the sensor b must be known. This means that the value of b has to be determined by this calibration algorithm.

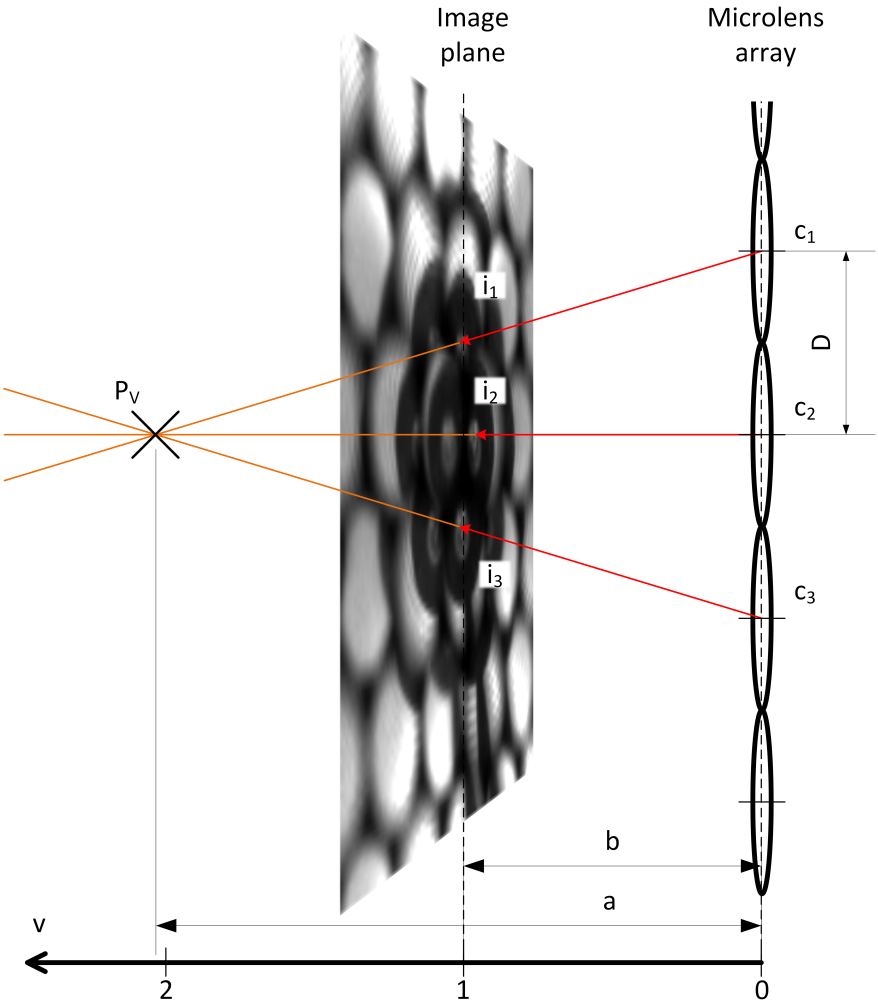
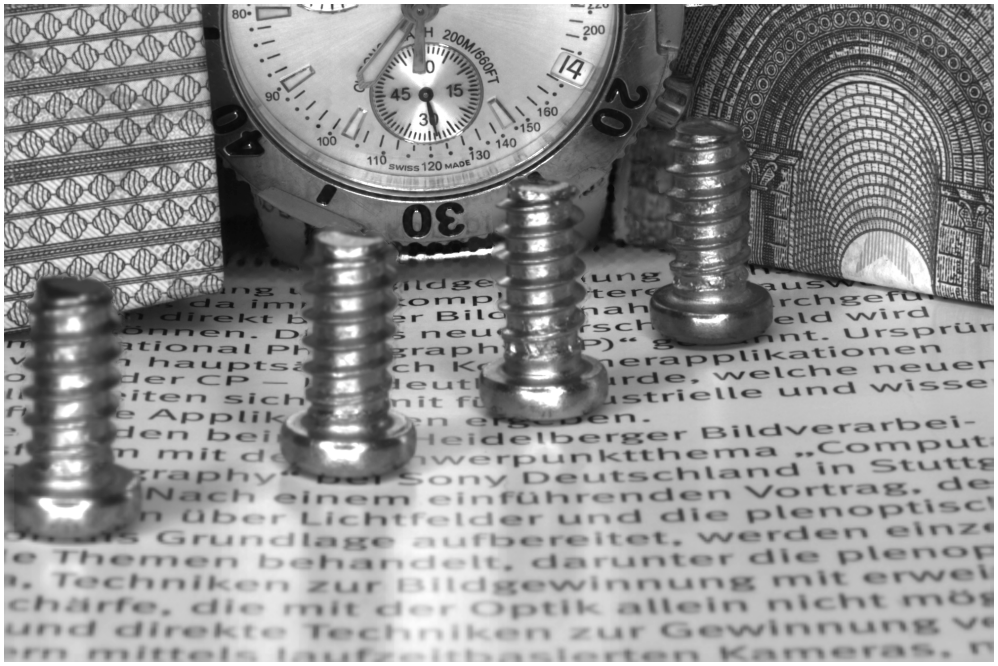
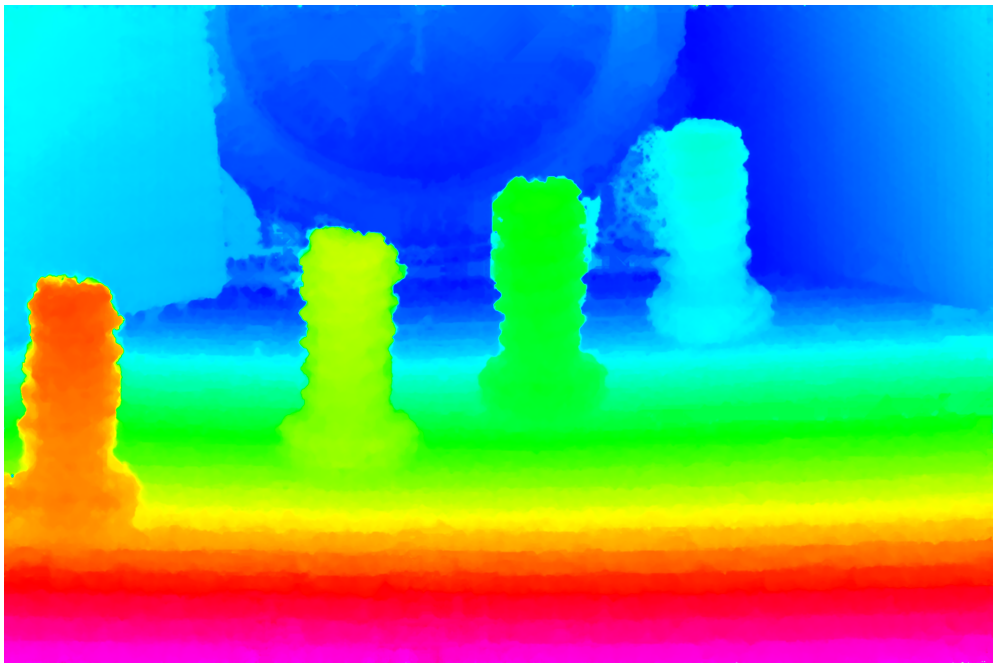


Figure 2.16: Schematic drawing of depth estimation principle

When this depth estimation process is applied to the whole raw image, a depth map can be generated. Figure 2.17 shows a scene with the associated depth map. The depth values are represented as color coded virtual depth units. Blue means further away from the camera and magenta is closer. Since depth estimation is only possible at some points of the image, the sparse depth map was filled using a propagation algorithm.



Total focus intensity image



Color coded depth image

Figure 2.17

2.3.3 Image reconstruction

The Raytrix light field cameras are multi-focus plenoptic cameras. This means, that not every microlens on the MLA is the same, but that there are different lens types with different focal lengths. The multi-focus approach is detailed in greater depth in the works of Perwass [11] and Lumsdaine [14]. The main motivation behind this approach is the extended depth of field that can be achieved with a multi-focus MLA. First of all, depth estimation is only possible when the object is in focus, because high local contrast is needed. This means that an extended depth of field greatly increases the applicability of the camera. Then, after the depth has been estimated, an image with a computationally enlarged depth of field can be synthesized. This is done by combining the sub-images from the different lens types, according to the object distance for which the lens type gives a sharp image.

To illustrate this, an example image with three different lens types is shown in figure 2.18.

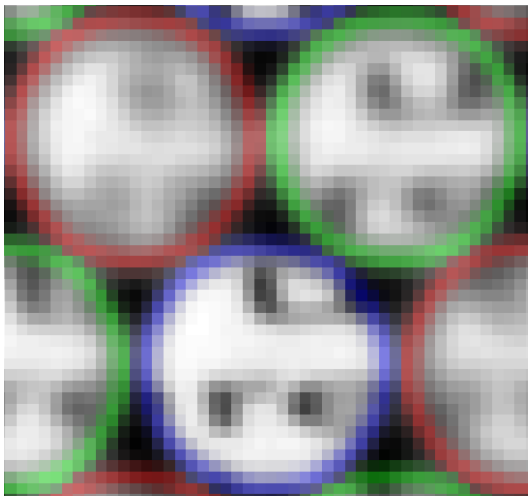


Figure 2.18: Raw image showing focused microlens design

The lens type highlighted in blue can focus objects which are close to the camera, while the red lens type focuses objects onto the sensor which are further away. The illustration shows, that using only the red lens type as in a single-focus MLA makes depth estimation impossible at this distance. The local contrast of the red lens type sub-images is too low to allow contrast based feature matching.

When the depth map for the image has been calculated, the total focus image can be synthesized. The pinhole model is used to project the points in 3D virtual space back to the raw image. From the locations in the raw image, the intensity or color values can be extracted. The synthesized image can be based on the 3D shape of the object, but also on any other shape. Using a plane parallel to the sensor makes it possible to create defocus effects after the image has been taken. Figure 2.19 shows a computationally defocused image. The raw image information is the same that was used to create the images in figure 2.17.

Figure 2.19: Computationally defocused image

The focus was put on the third screw from the left after the image was taken. By using a multi-focus MLA, the image quality can be greatly enhanced as compared to a conventional MLA. To make optimum use of multi-focus MLAs, the next section will introduce a novel model for projection errors of multi-focus MLAs.

2.3.4 Microlens array aberrations

As in section 2.2, the most important aberrations concerning the metric calibration of a light field camera are geometric distortions. A microlens array can generally exhibit different kinds of chromatic aberrations. Additionally, the vignetting of individual microlenses must be corrected to allow a robust depth estimation and image refocusing. Yet these effects are not discussed here, since they must be dealt with regardless of whether a metric calibration is performed or not.

There are two causes for geometric distortion in the virtual depth space behind the sensor. First is a deviation of the ideal shape of the MLA on a large scale. This might be due to a manufacturing error, when the MLA is not flat but curved on a large scale. Figure 2.20 shows an example of such a deformation.

This curvature will have a pronounced effect on the depth estimation, because the distance b between the MLA and the sensor varies, and the depth estimation is very sensible to this

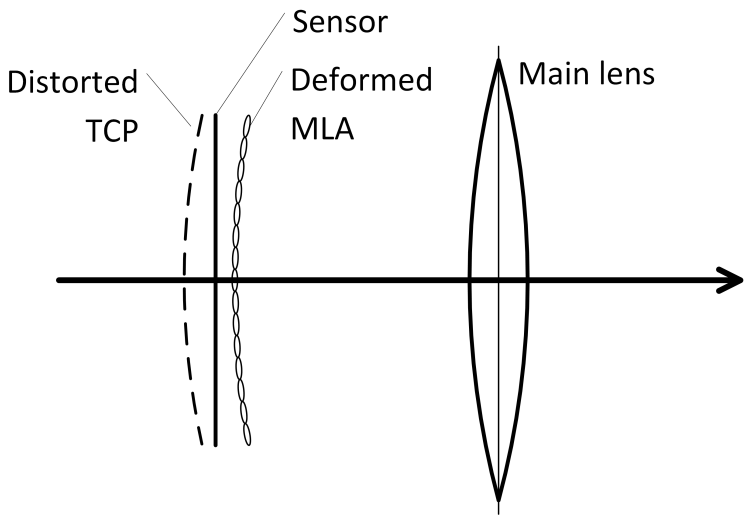


Figure 2.20: A deformed MLA will also distort the TCP

parameter. Unfortunately, this defect can only be measured during the production of the light field camera, by using either a mechanical profilometer or a special optical measurement technique developed by Raytrix. When the camera is used with a lens, these large scale effects can not be reliably separated from the effects of lens skew or radial depth distortion. This means that large scale distortion can be rectified through the main lens undistortion model, but not explicitly by an MLA undistortion model.

However for aberrations on a small scale, this work proposes a novel rectification approach. When an image of a plane target parallel to the sensor is taken, it is expected that there is a single plane of depth estimations in virtual space. Yet measurements have shown that there are actually several distinct planes, one for each type of microlens in the MLA (see figure 2.21).

The difference in depth between these planes is constant over the image and varies linearly with the virtual depth. Therefore a novel model is proposed, which assumes not one distance b between the MLA and the sensor, but several different values b_i, one for each lens type index i. Figure 2.22 gives an insight into why there might be a systematic offset for each lens type of an MLA. It shows an enlarged view of an idealized MLA.

If the microlenses are approximated by the pinhole model, one can see that the image distances b_i are different for lens types with different curvatures. The different curvatures are necessary to realize the different focal lengths for a multi-focus MLA. Calibrating and rectifying this error can be done by using not one value for b, but i different values b_i for the projection from virtual depths to metric depths behind the sensor. Using the correct b_i would rectify the offset between depth estimations from different lenses. This novel approach could not only reduce the geometric distortion and allow for more precise measuring,

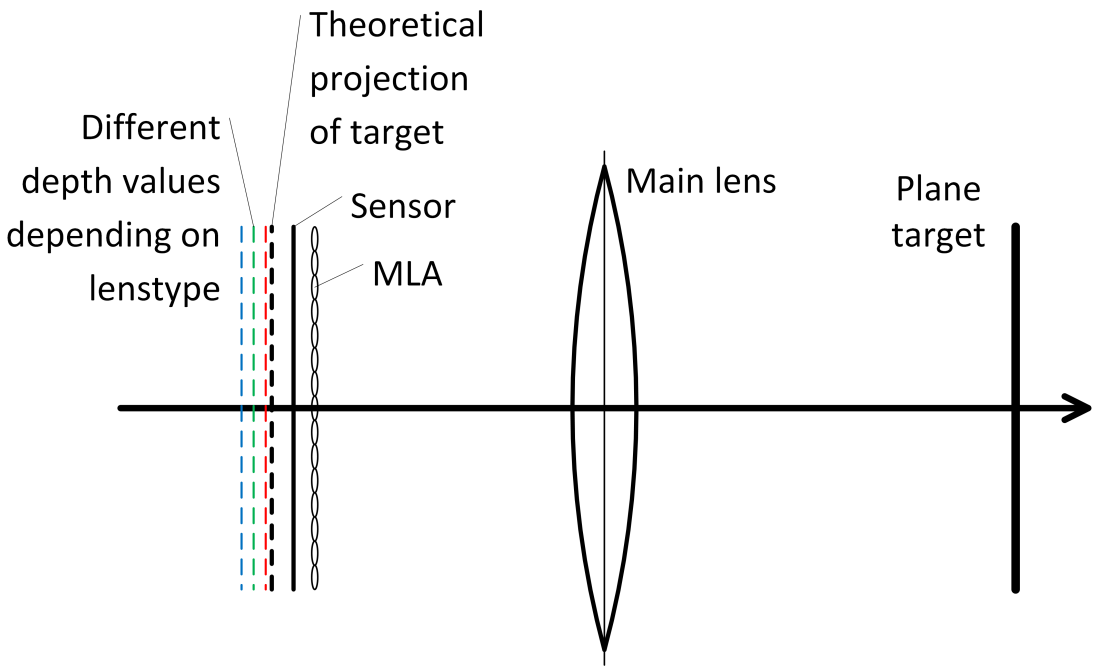


Figure 2.21: Real MLA projects plane target to different depth planes, depending on lens type

it could also reduce noise in the depth map. For object distances in which two or more lens types provide overlapping depth estimations, there is no longer a depth offset between estimations from different lens types. In an uncorrected image, there can be neighboring depth values from different lens types. The systematic offset between these depth values adds noise to the output depth map.

Now the projection model for a light field camera with main lens used for the calibration algorithm is complete. The next section will show how the parameters of this model can be optimized to give correct projection results.

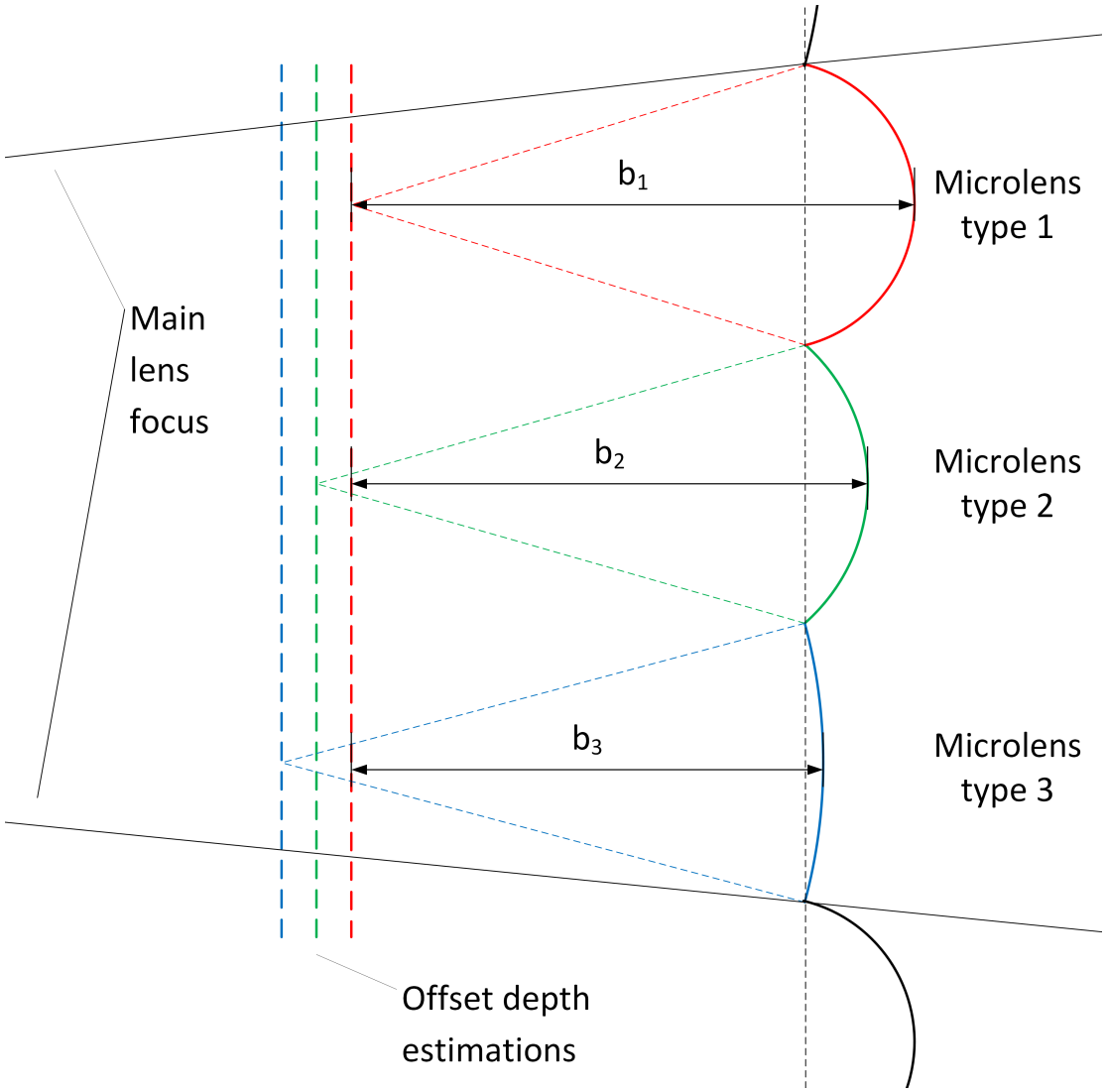


Figure 2.22: Influence of MLA geometry on projection

2.4 Camera Calibration

The basic idea behind camera calibration can be summed up in three steps. A mathematical model for the camera has to be found, which accounts for all relevant effects. Second, images are taken and some additional knowledge about the objects in the images is leveraged. Finally, the parameters of the mathematical model are tuned until the information extracted from the images matches with the prior knowledge about the imaged objects.

An example for this kind of calibration is the former procedure used for metrically calibrating Raytrix light field cameras. The camera is placed on a linear stage facing a noise target. A noise target is a plane target, on which a random structure is printed to give optimum depth estimation at different image sizes. Figure 2.23 shows an image of the setup at Raytrix.

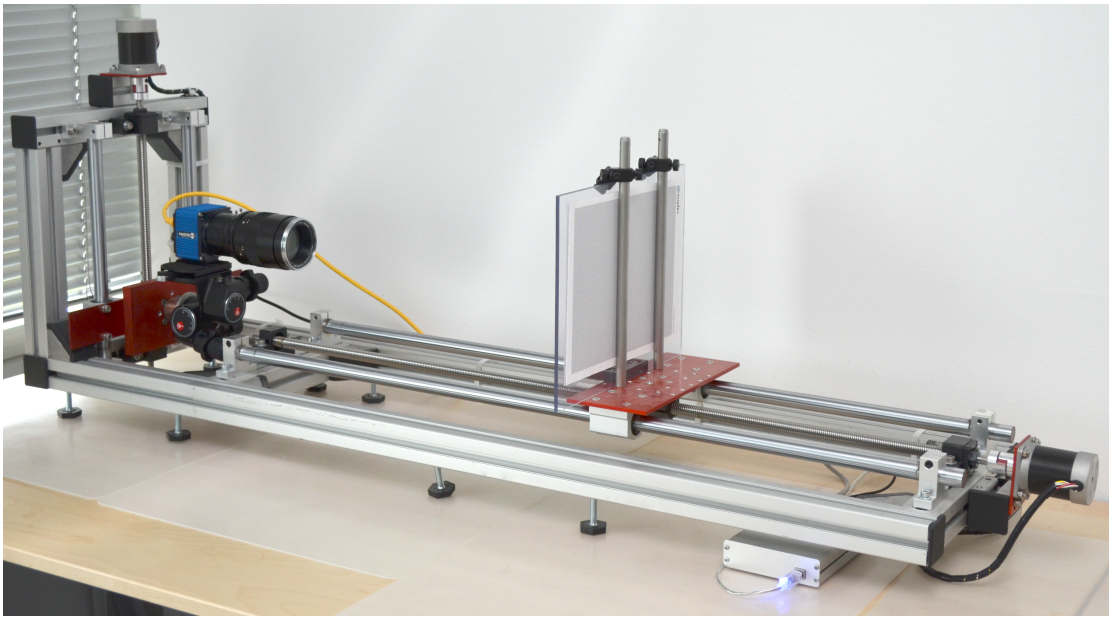


Figure 2.23: Camera and target on linear stage

The distance between the camera and the target can be controlled by a step-motor. The calibration model used is a direct conversion from virtual depths behind the sensor to metric distances in front of the sensor. This conversion is done by multiplying a virtual depth with a polynomial, which directly yields the metric distance from the sensor.

The coefficients of the polynomial are found by observing the virtual depths for the known distances to the target on the linear stage. Corresponding virtual and metric depths are found by moving the target to known distances and then taking an image with the camera. Then the virtual depth of the noise target is calculated. Figure 2.24 shows an example of virtual depth values plotted over corresponding metric depth values. As a last step, a polynomial

is fitted to the data points to allow converting virtual depth to metric depth at an arbitrary point over the range.

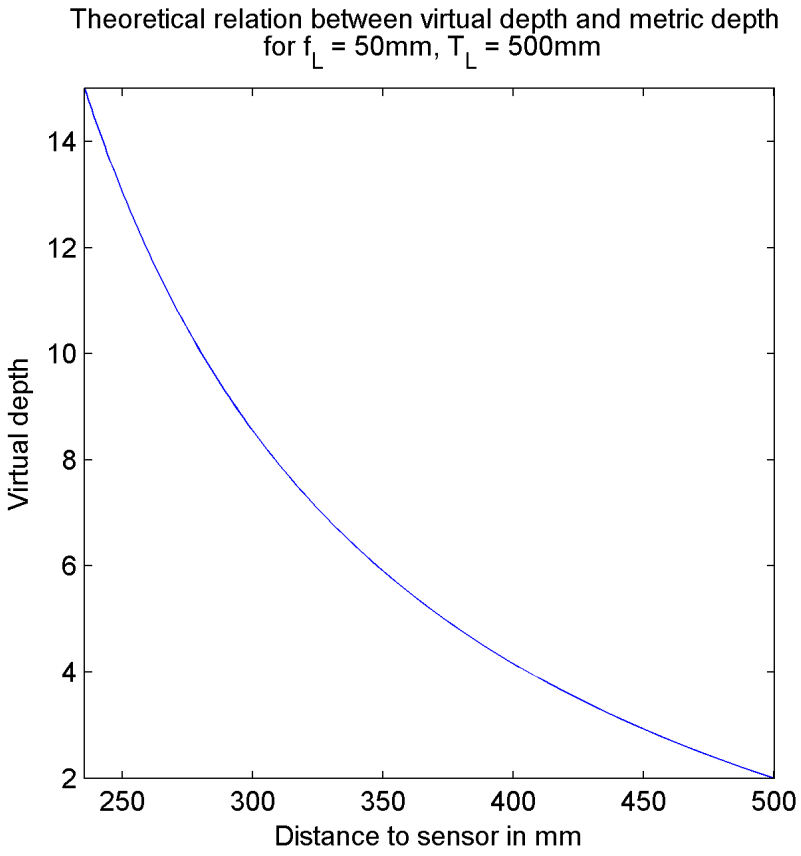


Figure 2.24: Theoretical correspondence between virtual and metric depth

This example serves as an introduction into a basic camera calibration technique, but it also shows the shortcomings of the former Raytrix camera calibration. Calibrating a camera required expensive equipment which had to be operated very precisely. The calibration is only applicable for the range over which depth estimations were calculated. Also, no form of aberration can be rectified, and no values for the intrinsic camera parameters can be obtained. Finally, the calibration procedure is time consuming and can only be performed by someone instructed about the operation of the linear stage.

Ideally, it should be possible to perform a camera calibration in the field without any specialized equipment. This is possible by using target based camera calibration techniques. The next section gives a short overview of such techniques.

2.4.1 Target based camera calibration

For target based camera calibration, a target with a known geometry is imaged. Reference points are then extracted from the image and put into correspondence with the known reference points on the target. Some common types of targets are shown in figure 2.25.

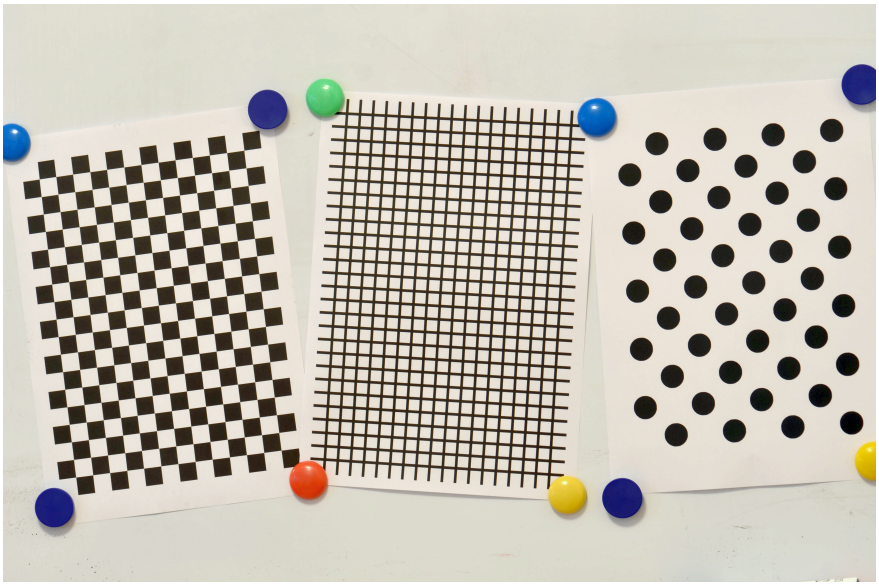


Figure 2.25: Checkerboard target, line target, circle pattern target

With checkerboard targets, the known geometry features which are leveraged for calibration are usually the distances between the corners of the squares, and the fact that the lines on the target are straight. A typical calibration algorithm will find corner locations in the image and associate the locations of these corners on the sensor with the known geometry of the target. Then the extrinsic pose of the camera in relation to the target is found by parameter variation. A virtual 3D model of the target is generated. This model is then projected into the image according to the camera calibration model. After the projection, the 2D distances between the points in the image and the projection of the model are calculated. The pose of the model is varied until this distance error becomes minimal. The same process is then applied to the intrinsic parameters of the camera calibration model. If the model fits the real camera close enough, the final parameters should minimize the distance error to below an acceptable threshold. This means both the pose of the target in front of the camera as well as the intrinsic parameters of the camera have been found.

A detailed description of a target based camera calibration algorithm is given by Tsai [4], with comments and improvements by Horn [15]. A recent publication by Danserau et al. [5] proposes using checkerboard targets to calibrate lenslet-based plenoptic cameras. Danserau et al. use a Lytro plenoptic camera, which is based on light field 1.0 technology. The calibration model is based on a correspondence between 4D rays on the image side of the

camera and 3D points on the object side. The approach is generally similar to the one presented in this work, as for example the main lens is also modeled as a thin lens and a similar lens distortion model is used, but unfortunately not much else is applicable for the problem described in this work.

Regardless of the specific approach used to calibrate a camera, many approaches face the same problems which can be solved by employing the same algorithms. The next section presents some applicable algorithms from the fields of image processing, classification, regression and optimization.

2.4.2 Image processing algorithms for camera calibration

A target based camera calibration algorithm must be able to recognize the target features in the image taken by the camera. Some calibration methods rely on user input to solve this problem. The user is presented with the image of the target and has to click on the feature locations in the right order. This solves both the problem of identifying the feature locations in the image, as well as the correspondence of these features to the virtual model of the target.

There are some ways to solve these problems automatically. For checkerboard targets, the Hough transformation can be used to find the straight lines in the image[1]. The intersections of these lines are candidates for feature points of the target. The coordinates of the corners can be calculated with subpixel accuracy. A paper published in 2010 by Escalera and Armingol [16] gives a comprehensive overview over a state of the art automatic calibration technique using checkerboard targets.

Figure 2.26 shows a checkerboard target where the corners identified by using the Hough transformation have been refined, classified and drawn as an overlay on the image.

When using targets with a circle pattern like in figure 2.27, the features are identified by using a blob detection algorithm. A simple method would be to use a global threshold to binarize the image. For the remaining regions, the centroids can be calculated. This gives feature locations with subpixel accuracy. Using a global threshold is not a robust way of segmenting the image, because any variation in lighting will interfere with the algorithm. More sophisticated implementations use the MSER (maximally stable extremal region)[2] algorithm for detecting feature locations and will be more flexible and robust. Even further, Mateos [17] proposes considering the effects of perspective on the circular features and computes feature locations using knowledge about the local neighborhood of circles.

Image 2.27 shows a target with a circle pattern, with the final location and classification data overlaid on the image.

[1] http://planetmath.org/HoughTransform
[2] http://www.vlfeat.org/overview/mser.html

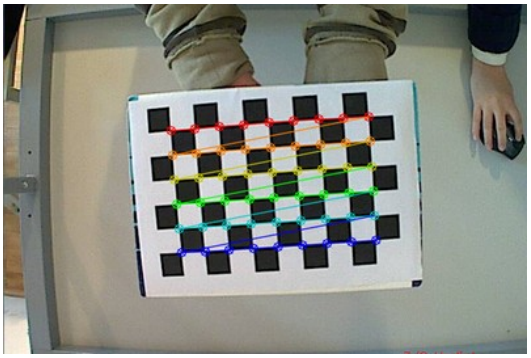


Figure 2.26: Checkerboard target

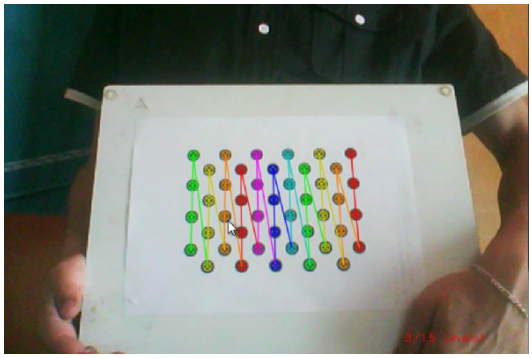


Figure 2.27: Circle pattern target

Pictures of targets with detected features highlighted from
http://docs.OpenCV.org/trunk/doc/tutorials/calib3d/camera_calibration/camera_calibration.html

Having identified the locations of features in the image, the features must now be brought into correspondence with the features on the target. Most approaches do this by leveraging additional knowledge. There may be special markers used on the target, which indicate the absolute orientation of the target in the image. With this information, a target coordinate system can be built.

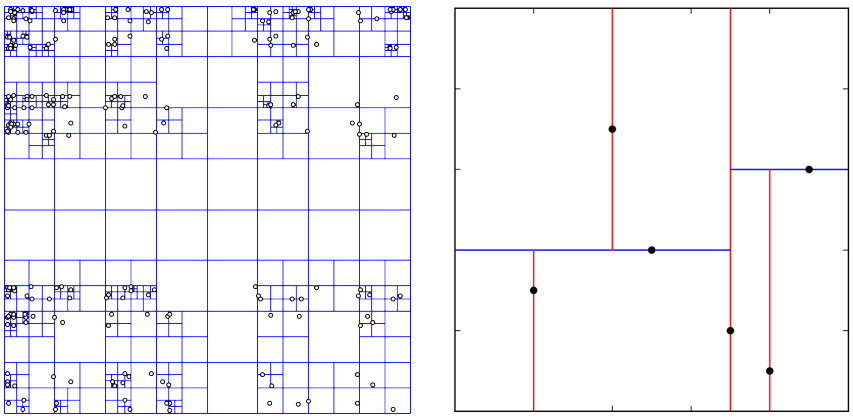
Now the feature coordinates found in the image can be transformed into the target coordinate system, for example by applying a homography[1]. The feature locations in this coordinate system are ordered into rows and columns just like on the actual target. Because the distance between features on the real target is known, a virtual model of this target can be generated with the geometry of the real target. This virtual target can then be rotated, translated, distorted and projected into the image and distances between corresponding points of the model target to the image of the real target can be calculated.

2.4.3 Classification algorithms for camera calibration

When classifying points for building these correspondences, it is often necessary to find the k nearest neighbors to a certain point. The point coordinates are usually stored in an unordered list. Explicitly calculating the euclidean distances to every other point and then selecting only those other points with the k smallest distance values is very ineffective. A more efficient way to do this calculation in two dimensions is by using a quadtree or a k-d tree. Figure 2.28 shows examples of a quadtree and a k-d tree partitioning of 2D space.

To build a quadtree, the 2D points are sorted into smaller and smaller subdivisions of the original area. For each step, the area is divided into four rectangles of equal size. The areas

[1] http://docs.opencv.org/doc/tutorials/features2d/feature_homography/feature_homography.html



Quadtree k-d tree

Figure 2.28: Area partitioning structures from
https://en.wikipedia.org/wiki/Quadtree [28.12.13]
https://en.wikipedia.org/wiki/K-d_tree [28.12.13]

in which the points lie are stored in a tree structure, where every node has four children. When searching for the nearest neighbor of a point, only the distances to points in adjacent nodes have to be calculated.

Another variation of 2D space partitioning algorithms is the k-d tree. To build a k-d tree in two dimensions, the space is partitioned into two parts, which means any k-d tree is always a binary tree. This can be done by taking the median of all X coordinates. Those points with smaller X coordinates go into one leaf, while those with greater X coordinates go into the other leaf. When subdividing these new leaves, the points are split according to their Y coordinate. The axis used for splittings keeps toggling between X and Y until the number of points in any one leaf is low enough. To search for the nearest neighbor of a point, the binary tree must be traversed recursively while performing checks to make sure that the closest point is not actually in a different part of the tree.

While building the data structure for these algorithms can take up some time, the faster searching speeds up the nearest neighbor search when there are hundreds or thousands of points to consider. Further information on nearest neighbor search can be found in the documentation of the ANN[1] library for C++. Apart from the ANN library, there are many other implementations of nearest neighbor search available, for example in the FLANN toolbox of OpenCV[2].

For building a 3D-to-3D camera calibration, it may be necessary to eliminate outliers in 3D space. An easy way to do this for a planar target is to fit a plane to the 3D point cloud. Figure 2.29 shows a cloud of points in 3D space. It can be seen that the distribution of the

[1] http://www.cs.umd.edu/∼mount/ANN/Files/1.1.2/ANNmanual_1.1.pdf
[2] http://docs.opencv.org/modules/flann/doc/flann_fast_approximate_nearest_neighbor_search.html

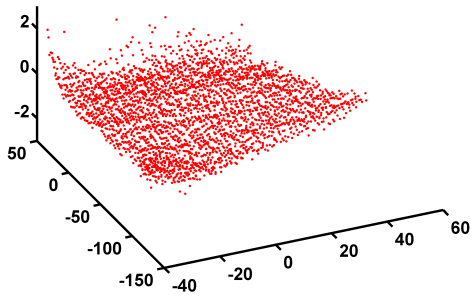


Figure 2.29: 3D point cloud extracted from plane target image

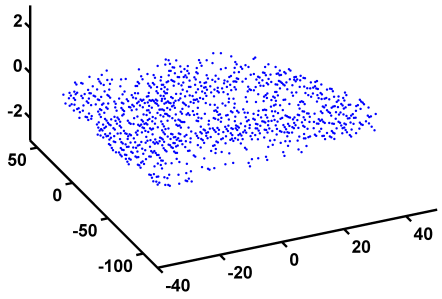


Figure 2.30: Point cloud after RANSAC outlier rejection

points is generally planar, but with some noise and outliers present. To remove the outliers, a plane can be fit to the 3D data and points too far away from this plane can be eliminated.

An explicit plane fit would calculate the plane, which would minimize the sum of squared distance errors from this plane to all points in the set. For a high number of points, this is very time consuming. Instead, the RANSAC algorithm can be used. The RANSAC algorithm was introduced in 1981 by Fischler and Bolles [18]. RANSAC is an abbreviation for *RANdom SAmple Consensus*. A simplified version of the RANSAC algorithm can be described as follows: instead of fitting a plane to all points, a random subset of points is chosen. A plane is fitted to this small set of points, and the sum of squared distance errors for all points is calculated. Since it can not be guaranteed that this small set of randomly chosen points is a good representation of the plane, this process is repeated many times. Each time, the parameters of the fitted plane and the total error are saved. When all iterations are complete, the plane parameters with the lowest total error are chosen. The advantage in speed is derived from the fact, that fitting a plane generally involves some form of matrix inversion. For large data sets, this inversion is much slower than randomly sampling many small sets of data. Figures 2.29 and 2.30 shows how the outlier rejection performs for an example data set.

2.4.4 Optimization algorithms for camera calibration

The final class of algorithms needed for camera calibration is the optimization or minimization algorithm. Though it is sometimes possible to find solutions for part of the calibration problem explicitly, most approaches rely on an optimization step. Horn [15] calculates the first steps of his camera calibration technique using an explicit approach by solving linear equation systems. But to refine the parameters and adapt to the more complex errors terms, he proposes switching to a non-linear optimization method. A good primer on optimization techniques is an article published by Shewchuk in 1994 [19]. It gives an intuitive and richly illustrated introduction into linear optimization algorithms.

The algorithm of choice for many camera calibration algorithms (for example [2] [5] [15]) is the nonlinear Levenberg-Marquardt method [20] [21]. The free C++ implementation *levmar*[1] of the Levenberg-Marquardt method was documented by Lourakis [22].

As was mentioned in section 2.4, the optimization algorithm tunes parameters until the minimum output value of an error function is found. The output of the error function is also called the residuum. For most camera calibrations, the residuum is a list of euclidean distances. The distances are calculated between each feature found in the image and it's associated model point. Each model point has been rotated, translated, distorted and projected according to the model parameters, and these model parameters are automatically tuned by the Levenberg-Marquardt algorithm. Figure 2.31 shows a flow diagram of the Levenberg-Marquardt algorithm.

The theoretical background necessary for this work has now been introduced, and the next chapter will look at the implementation details.

[1] http://users.ics.forth.gr/∼lourakis/levmar/

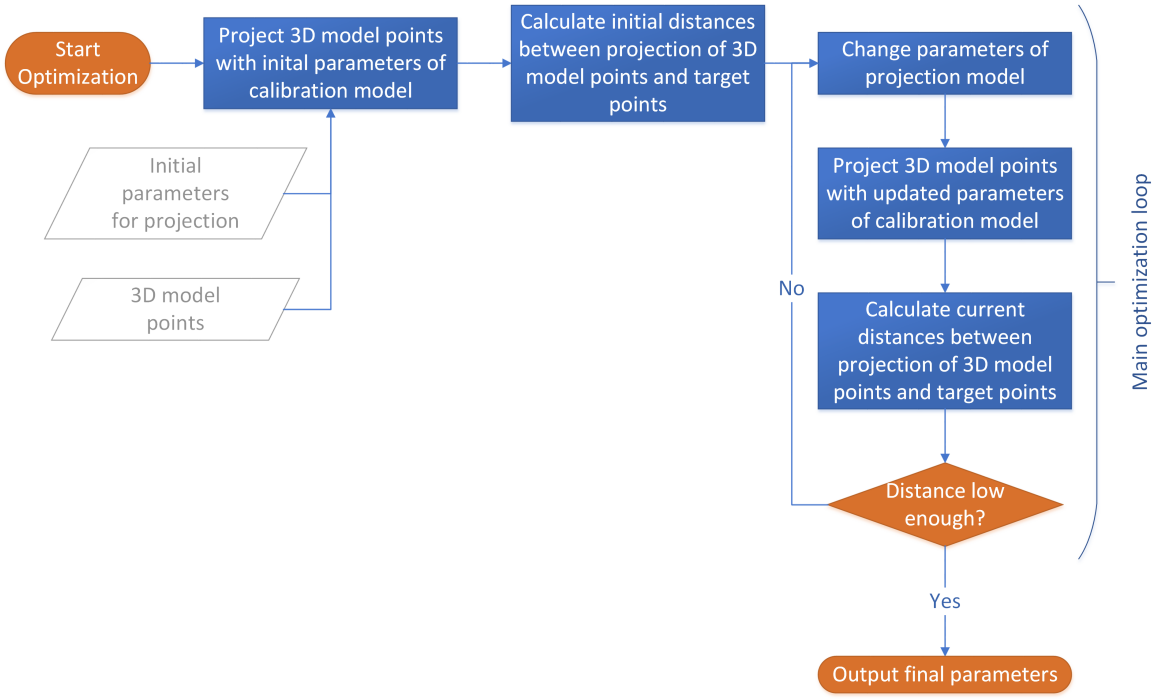


Figure 2.31: Flow diagram for generic calibration model optimization algorithm

3 Implementation

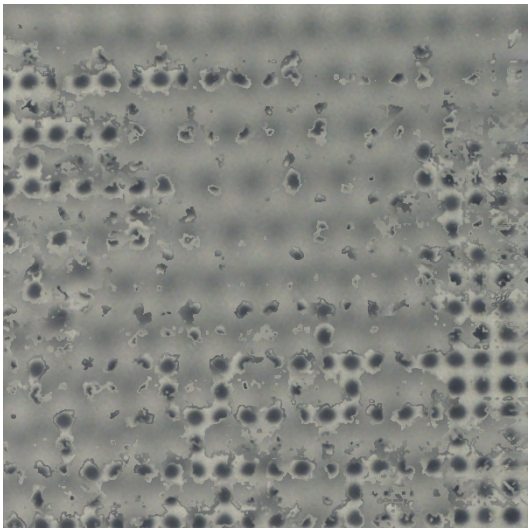
This chapter comprises not only the implementation of algorithms necessary for the metric calibration of a light field camera, but also details the design of the calibration target. Generally, the algorithms have been built to be modular. The optimization for example, could work with any set of target and model points for which an error can be calculated. This means, that different camera models can be implemented easily. This calibration algorithm could also be modified to calibrate standard 2D cameras or other types of 3D cameras, with which a feature detection can be realized. Finally, the software environment in which this algorithm will be used is presented.

3.1 Target design

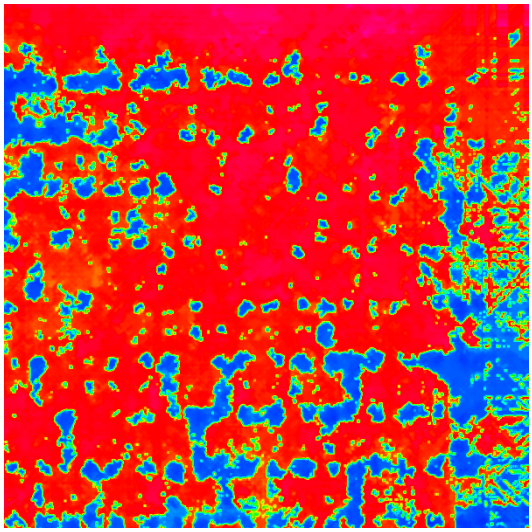
The design of the target used for calibration should be one of the first steps when implementing a camera calibration algorithm. The type of target used will determine the algorithms needed to identify the features on the target. A feature may be anything that can be reliably located with image processing algorithms. Examples for features would be intersections of lines, the centroid of a circle or the corners of a checkerboard pattern (compare figure 2.25).

In general, the target should have as many detectable features as possible, and the detection should work without producing any errors. Increasing the number of features means reducing their individual size. A balance must be found here, between the number of features on the target and the size, at which the camera can reliably detect these features. While the line target and the checkerboard target are generally quite comparable, the line target is not well suited to being used at different image scales. Regardless of the type of camera used or the resolution available, at some point the imaging system will no longer be able to resolve a line of a certain thickness. Making the line thicker in the first place will reduce the number of features which can be detected. For this reason, the line target is rarely used in established camera calibration algorithms.

Whether a conventional or a light field camera is to be calibrated, economic aspects must also be considered when designing a calibration target. Using a target which is not planar, but has a known 3D structure may yield better results. But the cost of manufacturing a 3D target with the needed precision is prohibitively high at the moment. Compared to that, a



Defective total focus image of circle pattern



Defective depth image of same circle pattern

Figure 3.1

planar target printed with a office printer will generally give very good results for a very low cost. Because of this, 3D targets are not considered any further.

With a Raytrix light field camera, special care has to be taken so that the target has enough local contrast for the depth estimation to work (see section 2.3.2). Without a good depth estimation, not only the depth map will be corrupted, but also the total focus image used for finding the 2D feature locations will be defective. The pattern of the target should allow for a reliable depth estimation with low noise in the calculated depth values. This noise can be reduced by increasing the density of the points, where depth estimation is possible. If the target has a high number of feature points, a single erroneous lateral or depth value will not influence the total result by very much.

But care must also be taken, so that the self-similar nature of the target does not interfere with the depth estimation. All target types proposed so far are made up of repeating patterns. The Raytrix Light field camera estimates depth by measuring distances in the raw image between features that match in two or more microlens images. If the spatial frequency of the features on the target and the spatial frequency of the microlenses are a certain ratio, the image in a neighboring microlens may not show the same target feature, but the next one on the target. Figure 3.1 illustrates this issue with a circle pattern target.

The total focus image shows serious defects, which can be explained by looking at the depth image. The depth estimation fails, because a correspondence in the microlenses is found for two different circles on the target. Usually depth estimation works by finding the same point of the object in several microlenses. But if different points of the object look exactly the

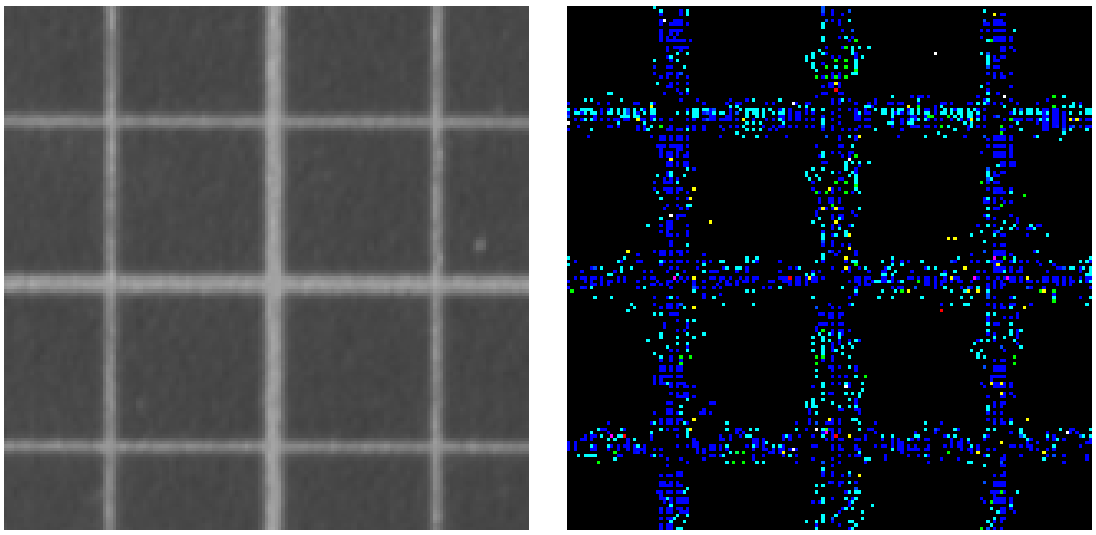


Figure 3.2: Noisy depth estimation of a line target

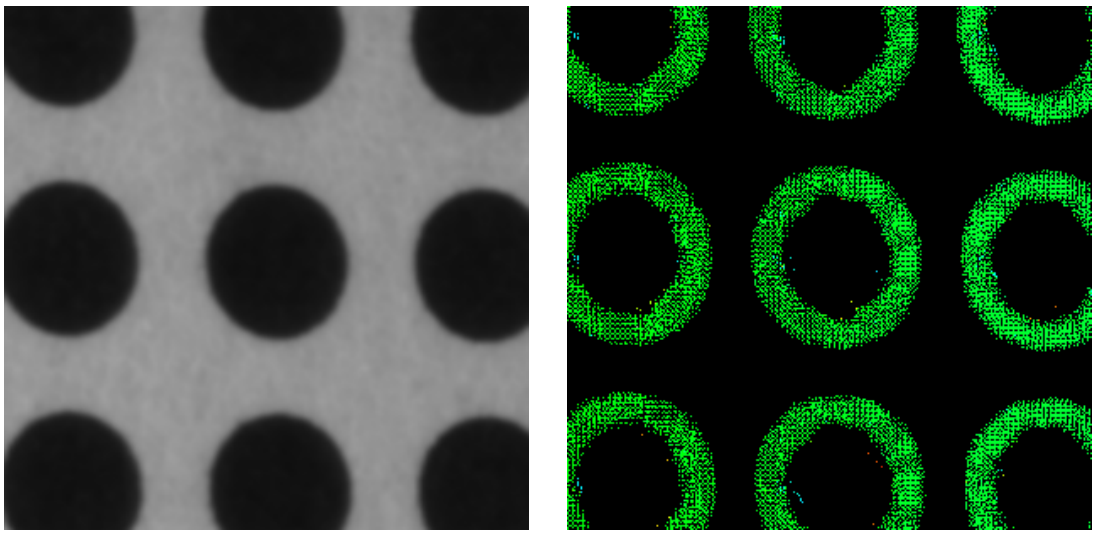
same in the microlens images, the depth estimation can produce the systematic error that is seen in figure 3.1. This effect can happen with line, checkerboard and circle pattern targets, and is very noticeable due to the defective total focus image.

Another more subtle effect related to self-similarity can be observed with line or checkerboard targets. These targets are not only self-similar because the features form a repeating pattern, but also the straight lines themselves are self-similar. Figure 3.2 shows a refocus and a depth image of a line target. The green clusters of noise half between the intersections are wrong depth estimations caused by the self-similarity of the line.

The depth values for the line are noisy, because the self-similarity produces many matches with a high correlation. Since this effect is so subtle, it will still produce an acceptable total focus image, while increasing the noise in the depth data. Due to this added noise, the depth estimations of the target features will be randomly distorted, depending on the virtual depth and the angle of the lines to the MLA.

To reduce random noise as much as possible, the target should be designed so that a number of depth values can be associated with a single feature location. The simplest way to do this, would be to take the arithmetic mean of all depth values in a certain radius around that feature location. Figure 3.3 shows a detail view of a circular feature and the depth values associated to it.

For all target types mentioned so far, it is also possible to use knowledge about the underlying geometry of the target to provide a better depth extraction than by just taking the mean of all values in a neighborhood. For line and checkerboard targets, it is possible to fit a line to the depth values in 3D space. Then the depth value at the feature location can be extracted by calculating the intersection between the two lines which define the feature location. For



Total focus image Color coded depth image

Figure 3.3: Detail views of a circular target feature

a circular feature, a circle can be fit to the 3D depth data. After calculating the depth value at the center of this circle, this value can be used for the depth of the feature. Both line fit and circle fit reduce the noise of the final depth value by using many depth values and rejecting outliers through the initial geometry fit.

The final criterion with which to assess a calibration target is the complexity of the algorithms needed to automatically detect the features. For line or checkerboard targets, this usually means using an edge enhancing filter and a Hough Transform to get the initial candidates for feature locations. When using a circle pattern target, a blob-detection must be performed. The resulting centroids must then be filtered and ordered on a regular grid so that a virtual model of the target can be built. The algorithmic complexity is moderately high in both cases. At Raytrix, a sophisticated blob-detection algorithm is already required for some other tasks. That makes using a calibration method based on blob-detection more reasonable.

The requirements introduce have shown the strengths and weaknesses of different target types, and the conditions under which targets give the best results for metric calibration. Table 3.1 outlines the aspects covered so far.

The table shows that using a circle pattern target is a good choice for this light field camera calibration algorithm. The most important advantage of the circle pattern is the robustness against problems with self-similarity. With line or checkerboard targets, this problem can not be detected or rectified. The amount of depth noise introduced by the false correlations depends on the pose of the target relative to the camera. Even small rotations of the target can increase the noise of the depth values greatly, but without any obvious influence on

Table 3.1: Comparison of different target types

	Lines	Checkerboard	Circle pattern
Prior work	few	many	many
Handling of different image scales	bad	good	very good
Problems with self-similarity	bad	bad	good
Extracting feature depth value	easy	easy	very easy
Algorithms reusable in house	no	no	yes

the refocus image. Also, some amount of noise in the depth data will always be present, regardless of the image scale. With circle pattern targets, the effects of false correlation can be seen very easily. Only images with certain discrete depth values will suffer from these defects and a small movement of the target will remedy the problem.

The choice of a circle pattern target is favorable when using a light field camera, yet it also means that a lot of work has to be done in developing the algorithms needed. Especially building the target model automatically means taking a new approach different to what is currently used in other camera calibration techniques. The next section will detail the algorithms implemented to realize this circle pattern based camera calibration, starting with the blob-detection algorithm.

3.2 Overview of implemented algorithms and data flow

To allow the use of a circle pattern target for the calibration of a light field camera, a number of algorithms must be applied to the input images. Figure 3.4 gives an outline of the algorithms and the data flow implemented for this work.

The calibration procedure begins by using the standard Raytrix depth estimation algorithms to calculate a number of depth images and a refocus image. More than one depth image is calculated to allow calibrating the offset between different microlens types of the MLA (see section 2.3.4). If the MLA has three lens types, three depth images will be calculated. Each depth image contains depth values calculated by using only one lens type.

The total focus image is then used to perform the blob detection. This gives a list of centroids, which is further refined in the next steps. The centroids not only contain 2D pixel coordinates for the centroid location in the image, but also the parameters of an ellipse fitted to the blob. Since the blob detection will detect some very large or very small blobs, a filtering step is then applied. The rejection criteria are based on the 2D pixel-coordinates, the radius, and the eccentricity of the ellipse. If a centroid is filtered out, it is not removed from the list but flagged as an invalid 2D point, so that no information is lost.

Next the depth values for all centroids are extracted from the image. The centroids are now 3D points with depth values in virtual depth units. The next step filters the 3D centroids by fitting a plane in 3D space and flagging all outliers as invalid 3D points.

Now the correspondences between points in the 3D space and the known target must be found. This means building a virtual model of the target with metric distances between points on a regular grid. For every valid centroid in 3D space, a corresponding point in the virtual model must be generated.

All the steps above are repeated for several images of the calibration target in different poses. This gives a set of points extracted from the image of the target and corresponding model points for each of the images. All these point sets are then passed to the optimization algorithm.

The initial parameters for the projection model consist of a common set of intrinsic camera parameters, and one set of extrinsic parameters for each image of the target. The 3D centroids are projected with the camera model and their respective parameters. Then the residual distances between each 3D centroid and it's associated model point are calculated. When the sum of distances falls below a certain threshold, the optimization is terminated. The set of intrinsic parameters giving the smallest reprojection error is the output of the calibration algorithm.

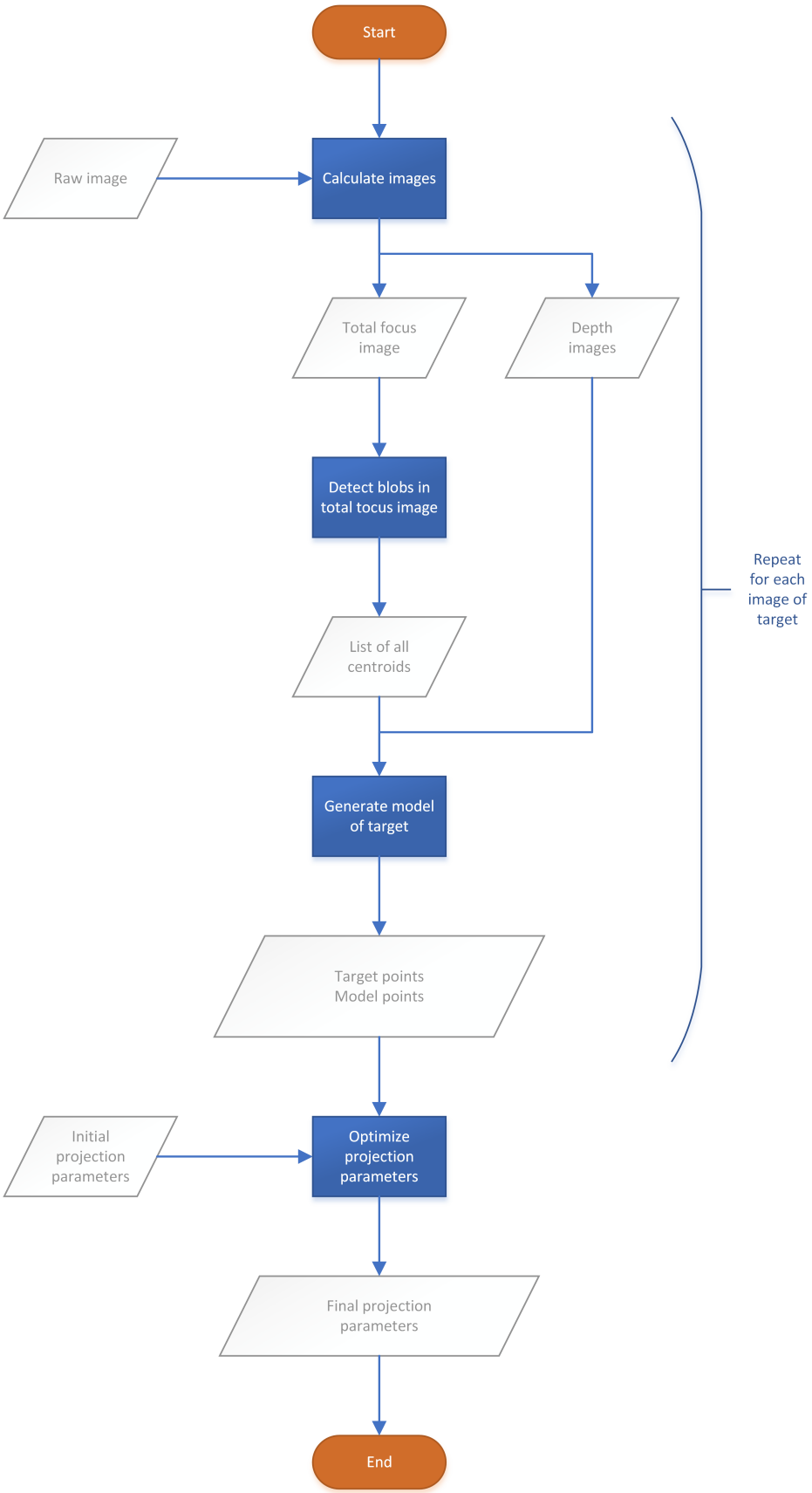


Figure 3.4: Flow diagram of implemented algorithms

3.3 Blob detection and filtering

An overview of the blob detection and filtering is given in figure 3.5. For blob-detection, the MSER algorithm implemented by OpenCV is used[1]. Input data to the MSER algorithm is the 2D total focus image and the minimum area threshold value. Regions with an area smaller than this value are discarded by the MSER algorithm. This is necessary to reduce the number of false positive regions.

The output generated by the MSER algorithm is an unordered list of ellipses. Each entry consists of the 2D pixel-coordinate of the detected centroid, the brightness value of the total focus image at the centroid location, the approximate radius of the ellipse (major axis) and the covariance matrix of the ellipse. The ellipses are then filtered to reject any outliers. To do this, the mean of all radius values and all values for eccentricity are calculated. The user can then supply thresholds that reject outliers which differ by more than a certain factor from the mean. This will for example reject a small number of very large centroids, or centroids that are highly eccentric while the majority of the ellipses are more circular.

As a last step, the ellipses are filtered according to the intensity value of the central pixel. The targets used so far have black circles on white background. This check can eliminate ellipses which are located over a bright area, and are therefore probably not valid. Entries in the list of ellipses are labeled depending on how they performed in these filtering steps. No entries are discarded completely, so that the user can get feedback on what the input data was, and what caused rejection of ellipses. After these steps, it can be assumed that the data is reasonably free from detection errors and can be processed by the following algorithms.

[1] http://code.opencv.org/projects/opencv/wiki/MSER

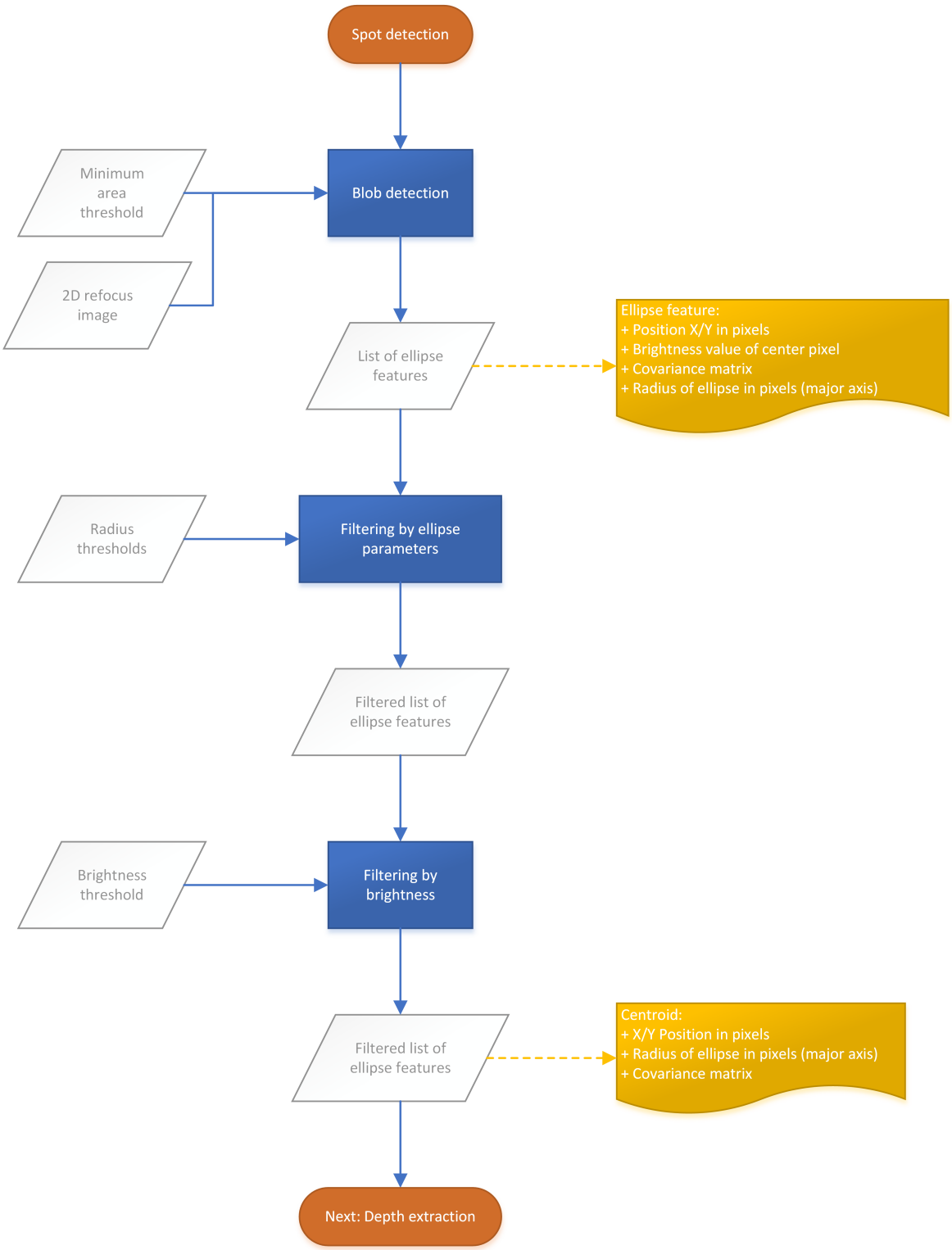


Figure 3.5: Flow diagram of spot detection and filtering

3.4 Depth Extraction

After the candidates for 2D feature locations have been found, depth estimation has to be added to these features and they have to be filtered once again. Figure 3.6 gives an overview of the data flow and the algorithms involved to do this.

The first step is to check if the 2D locations in the image are also valid 3D points. To do this, the sparse depth estimations from the depth map have to be associated with the 2D locations from the ellipse list. This is done by calculating the median value of all depth estimations in a square region around the centroid of each ellipse. The size of the square region depends on the radius of the ellipse. Figure 3.7 shows a total focus image, the associated sparse depth map and the region from which the depth values are extracted.

After the median value has been found, a test is performed to see if there are enough valid depth estimations in the square region. Figure 3.8 shows a case where ellipses can be detected in 2D, but some ellipses only have very few associated depth estimations.

This case will always occur, when the target is held at an angle to the sensor plane and the depth values leave the valid region of one lens type. In figure 3.8, the depth estimations of the near lens type are shown. Since the target was angled away from the camera, the top of the image is too far away for the near lens type. To counter this, ellipses are marked as invalid when they have too few depth estimations in the square region.

This step is very important when several MLA distances b_i are to be determined as described in chapter 2.3.4. It can not be assumed that an ellipse will have a valid depth estimation from all lens types, and this is the only way to reliably reject values from outside the lens type's valid depth region.

Finally, even if the depth estimation is dense, there can still be outliers in the depth estimation process. To reject these outliers, the same methods as in other 3D classification approaches can be used. Since the target is known to be planar, a plane can be fit to the 3D point cloud using a RANSAC algorithm as described in chapter 2.4.3. Points which lie too far away from this plane are marked as invalid and will not be used for calculating the error function in the optimization step. But since the rejected points are valid 2D locations, they may still be used for building the model of the target. This is detailed in the next section.

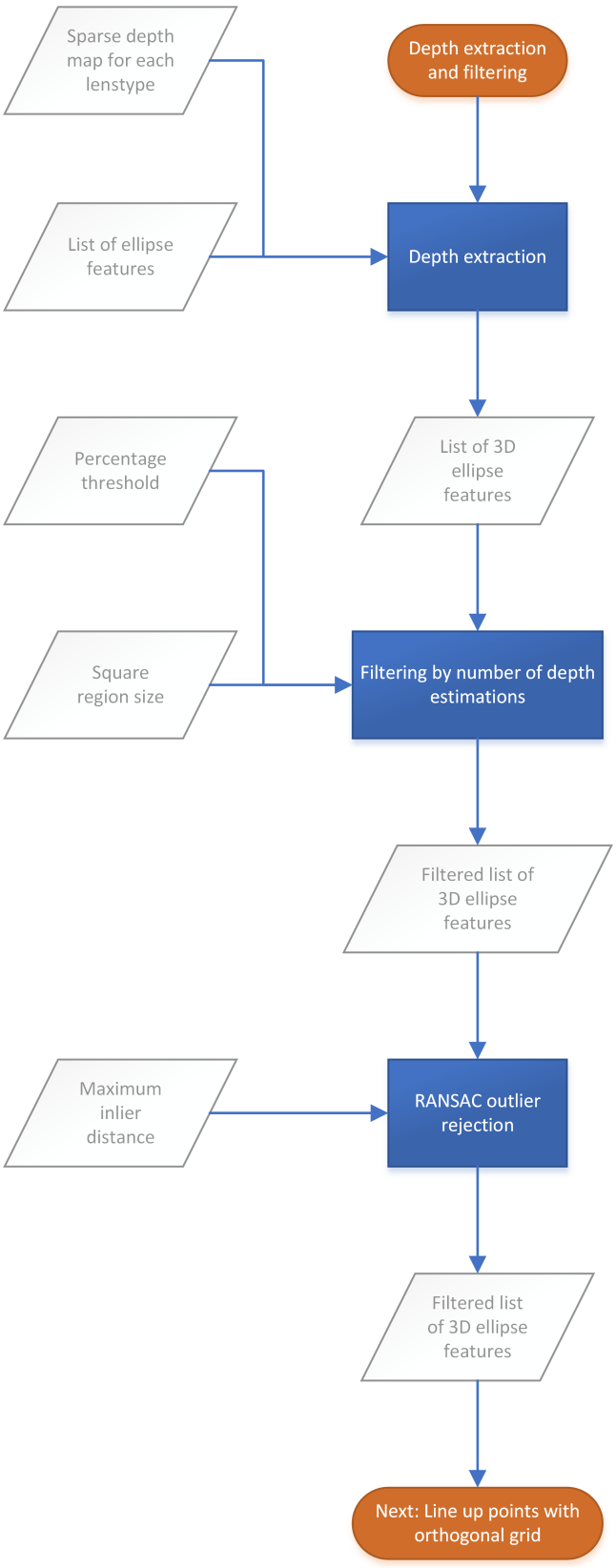
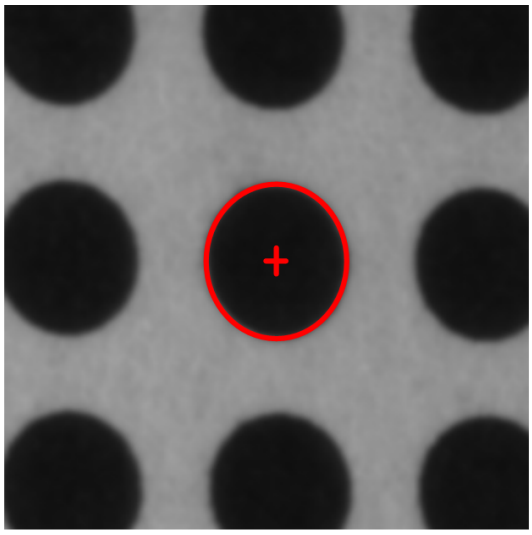
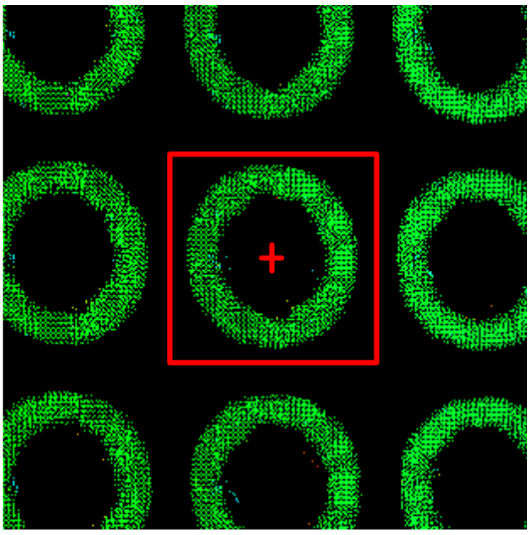


Figure 3.6: Flow diagram of depth extraction algorithm

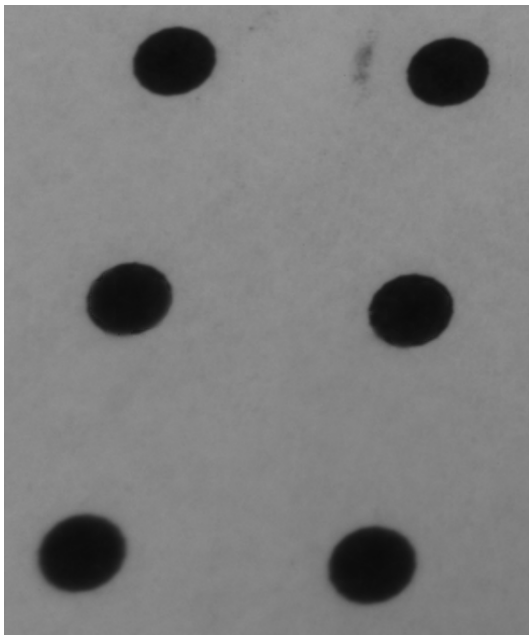


Total focus image with fitted ellipse overlaid

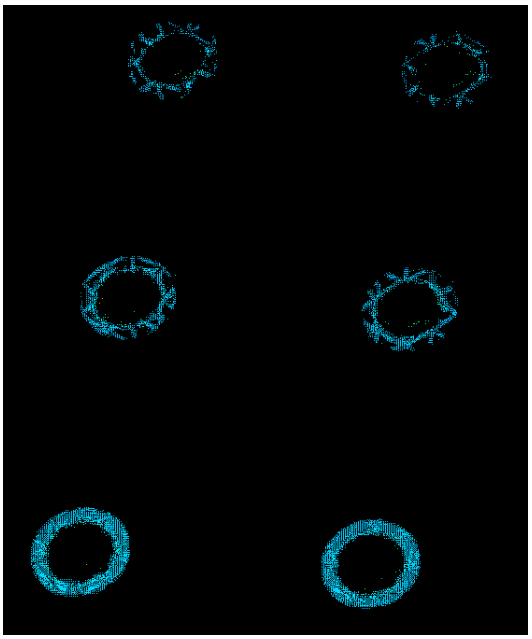


Depth image with square depth extraction region marked

Figure 3.7



Flawless total focus image



Number of depth estimations varies

Figure 3.8

3.5 Building the target model

This calibration algorithm relies on an optimization step to find the optimum parameters for the projection model. The optimum set of parameters is the set, which minimizes the projection error. To calculate the projection error, all valid 3D points from the section before are projected from virtual space behind the MLA to metric space in front of the main lens. Then the position of each 3D point is compared to the position of it's corresponding point on the model of the target. The steps described in this section detail how this model is generated.

From the steps before, the positions of the features in the image are known. Also, the original metric distance D_M between circles on the printed checkerboard target is known. Therefore, it is also known that two points which are next to each other in the image must have had the distance D_M between them before they were imaged with the camera. What must be found now, is the correspondence of every point in our image to it's original metric position on the printed target.

Image 3.9 shows an example of point locations extracted from a total focus image using the steps described above. For human eyes, the underlying rectangular structure of the points is easy to see. Knowing that the original distance from point to point was for example $4mm$, it is easy to align the points on an orthogonal equidistant grid like in figure 3.10. These virtual model points describe, what parts of the target have been seen by the camera. The target used to create the image was of course much larger, but there were only model points created for those points of the target, which were seen and detected. This is the desired behavior, as it makes the calibration process much easier and more flexible for the user.

Figure 3.11 gives an overview of the first steps necessary to automatically perform this assignment of model points to features in the image.

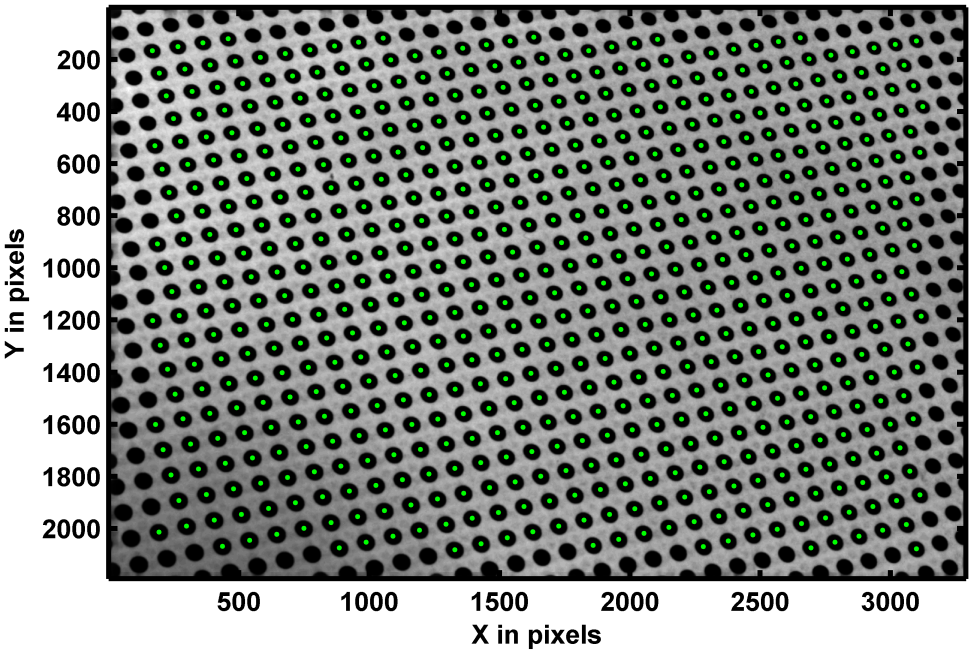


Figure 3.9: Example of 2D circle locations (target points) overlaid on a total focus image

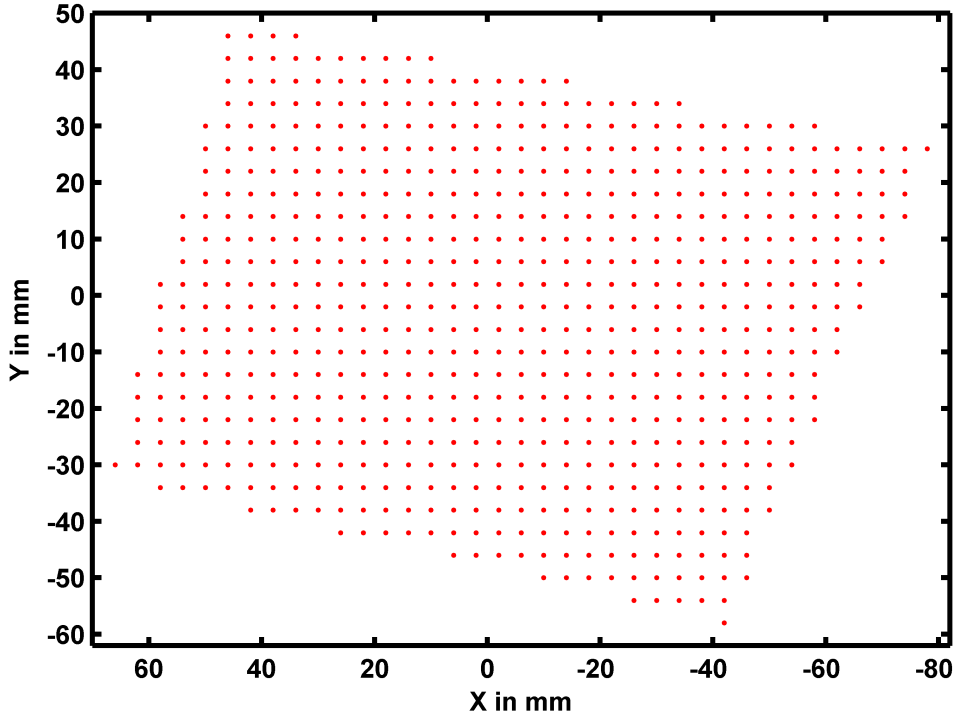


Figure 3.10: Example of orthogonal equidistant grid (model points) created from target points above

Table 3.2: Parameters used in spot detection and assignment algorithms

Parameter	Value range	Explanation
Minimum blob area threshold	16...1024	Blobs with fewer pixels are discarded
Radius divergence threshold	2...10	Ellipses with radius diverging by more than this factor from mean radius are discarded
Brightness threshold	32...192	Ellipses with brightness values higher than this are discarded
Square region size factor	1...3	Factor controlling the size of square region for depth extraction, internally multiplied with ellipse radius
Depth estimation percentage	5%...20%	Ellipses must have at least this percentage of the square region with valid depth estimations
Maximum inlier distance	0.01...0.1	Points further away from fitted plane are rejected, value is in virtual depth units
Candidate location tolerance	5...100	After calculating a candidate location, the nearest neighbor must be closer than this distance in pixels

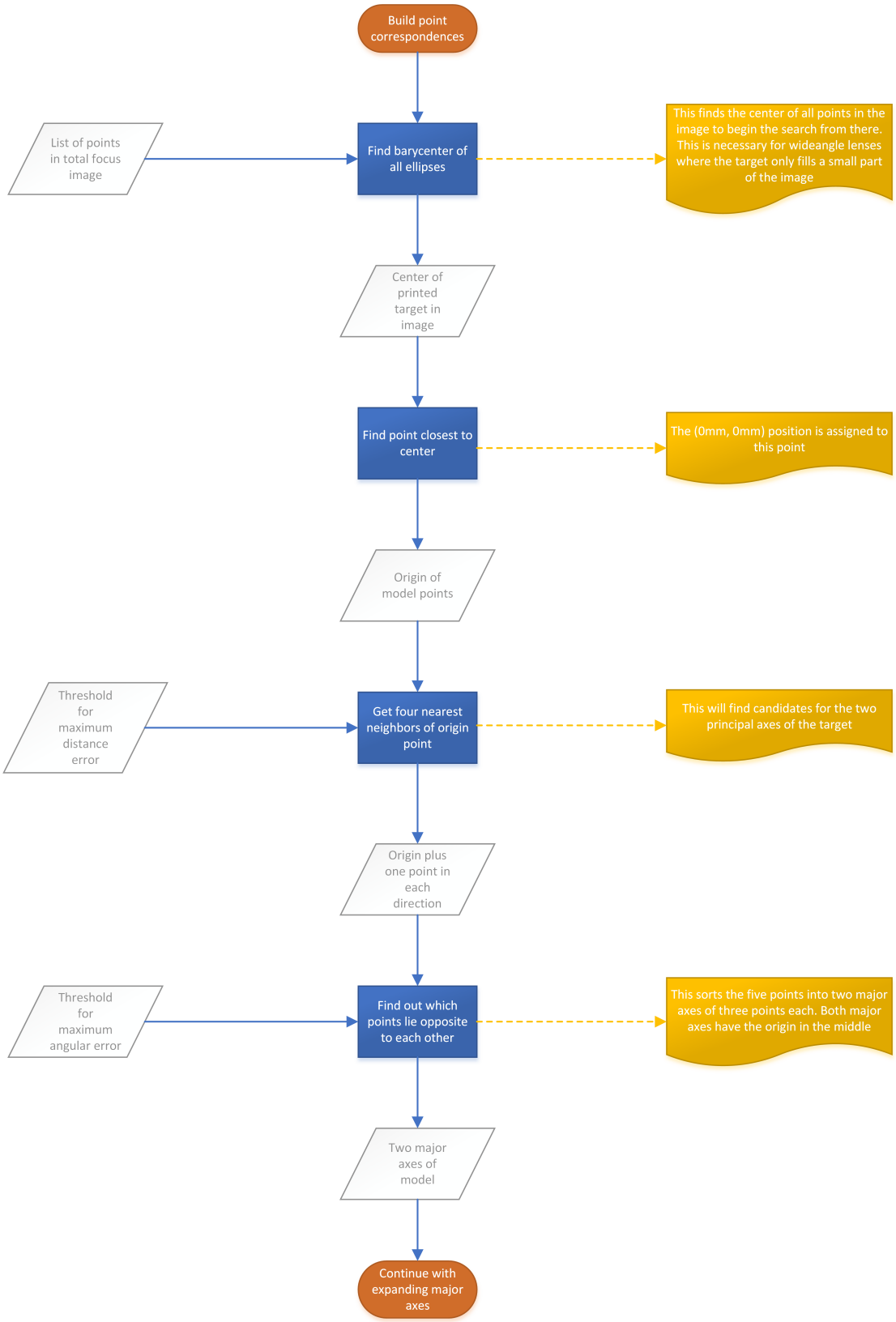


Figure 3.11: First part of algorithm to automatically assign model points to features found in total focus image

The algorithm starts by finding the middle of the target in the image. For images where the target fills the whole frame, this is not important. But this calibration is also capable of calibrating wide angle lenses, where the printed target only fills a small part of the frame. Therefore the target center is detected, by calculating the barycenter of all detected ellipses. Figure 3.12 shows an example image taken with a wide angle lens that can be used with this calibration algorithm.

Figure 3.12: Calibration target imaged with wideangle lens

When the barycenter has been calculated, the point from the ellipse list nearest to it must be found. This is done by using the k-nearest-neighbors algorithm as described in section 2.4.3. The input to the k-nearest-neighbors algorithm is the X, Y pixel coordinate of the barycenter and it returns the one point from the ellipse list that is closest to the barycenter. This point is set as the origin of the target model. The metric coordinate $(0mm, 0mm)$ on the target model is added to this point's entry in the ellipse list. This is the first point for which a one-to-one correspondence between the pixel coordinate and the metric coordinate in the target model has been found. The origin of the target model can be chosen at will because the extrinsic pose of the camera allows for an arbitrary translation in three dimensions. Only the distances between points of the target model must be correct.

To find the correspondences for the other points in the ellipse lists, the two principal axes of the orthogonal target grid must be found. The k-nearest-neighbors algorithm is used to search for the four points closest to the origin. These are candidates for defining the direction of the two principal axes. Figure 3.13 illustrates this.

The four neighboring point are put into an unsorted list. The first point in this list is chosen as the starting point P_S. For each of the remaining points, the distance from a line connecting P_S and the currently chosen endpoint P_{Ei} to the origin is calculated.

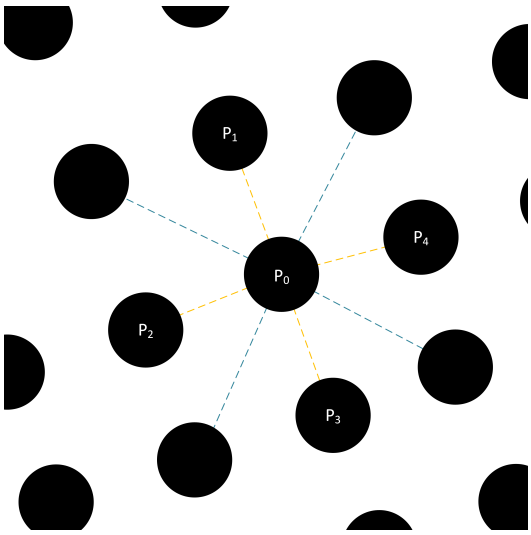


Figure 3.13: Origin of target model with nearest neighbor points.
The orange lines are generally shorty than the blue lines.

With the starting point

$$P_S = \begin{pmatrix} X_S \\ Y_S \end{pmatrix} \tag{3.1}$$

and the end point

$$P_{Ei} = \begin{pmatrix} X_{Ei} \\ Y_{Ei} \end{pmatrix} \tag{3.2}$$

the vector a_{SEi} is the difference

$$a_{SEi} = P_S - P_{Ei} \tag{3.3}$$

between the points. To calculate the distances d_i from the line to the origin P_0, the following equation is used:

$$d_i = \frac{\|a_{SEi} \times (P_0 - P_{Ei})\|}{\|a_{SEi}\|}. \tag{3.4}$$

For each pair of points, the distance of the line to the origin is saved. After all combinations of points have been dealt with, the two points with the lowest value for d_i form a line that runs through the origin. If the value for d_i is below a threshold supplied by the user, the first principal axis is found. The second principal axis must now consist of the two other nearest neighbors. A check using equations 3.3 and 3.4 and the threshold is performed for these points as well. If it passes, the second major axis has been found. Figure 3.14 shows the lines connecting the starting point to each of the possible end points, and the distances to the origin. The correct principal axis which goes through the origin is drawn in green, the two other lines with too large distances d_i are drawn in red.

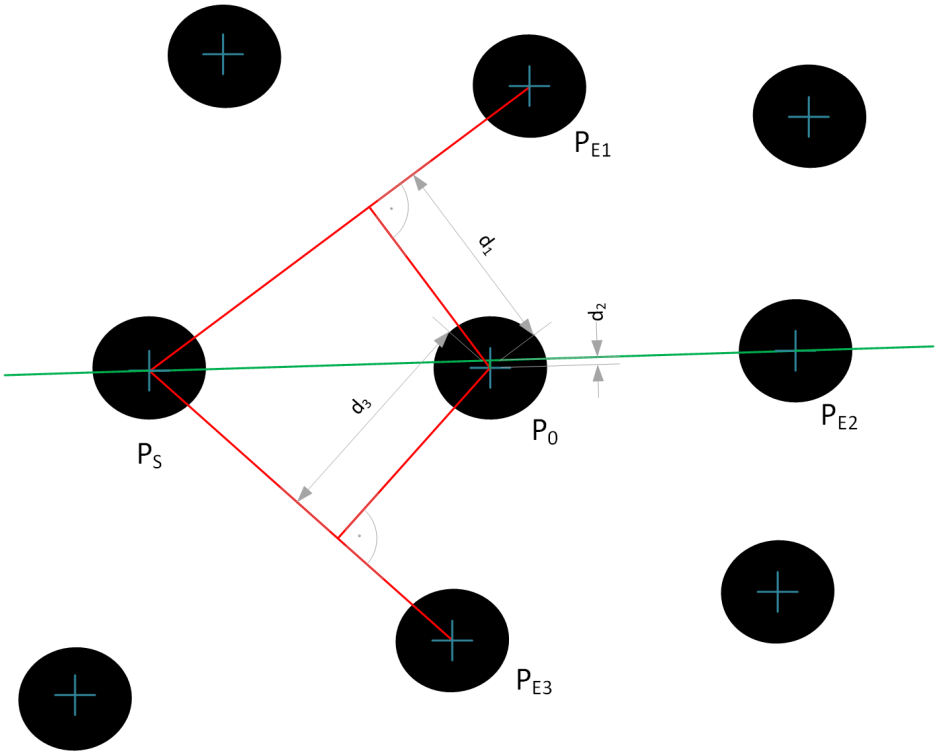


Figure 3.14: Calculating point-to-line distances d_i to find first principal axis

There are some cases, in which the sanity checks will fail. If for example there is a false positive detection closer to the origin than the four points which would form the principal axes, no consistent set of axes could be found. To counter this, the algorithm actually searches for more than just four nearest neighbors of the origin, and then tries to find the set of points which gives correct principal axes. If the sanity check is still not passed with this larger neighborhood, the algorithm is restarted with another point in the vicinity of the former origin. This point is then used as a new candidate to find the origin, the nearest neighbors and the two principal axes. This is repeated until the sanity checks pass and the two principal axes are found.

Each line now consists of three points - the start, the origin and the end point. The new points are now assigned metric values according to the known distance between features on the printed target and the direction. Once again, the directions can be chosen arbitrarily and do not have to line up with the X and Y axes of the image coordinate system. Now the origin of the target model and the starting points for the two principal axes are known. This concludes the first part of the algorithm overview (figure 3.11). The second part of the algorithm is shown in figure 3.15.

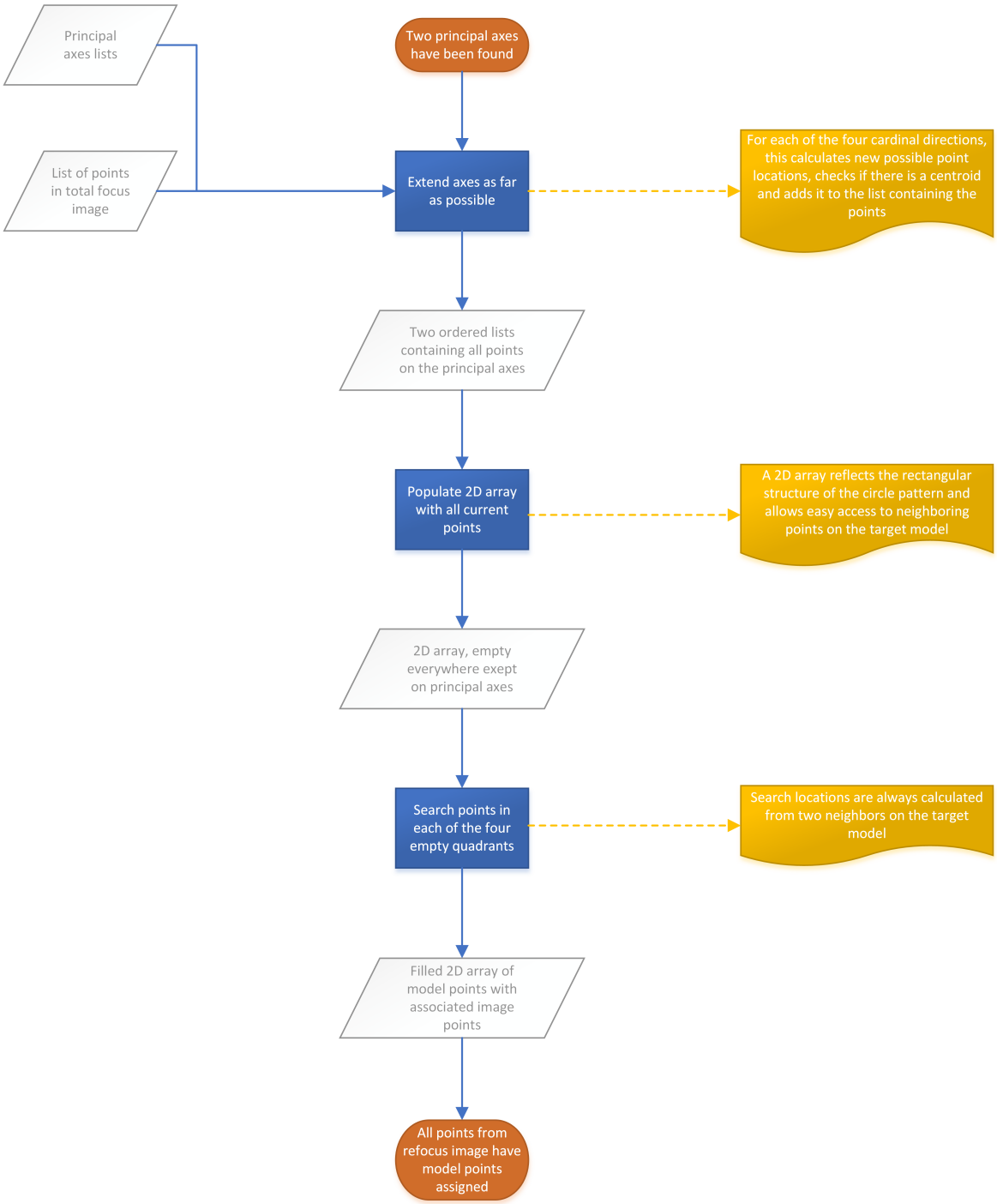


Figure 3.15: Second part of algorithm to automatically assign model points to points found in total focus image

So far, only the starting directions of the two principle axes have been found. Now the axes need to be expanded over the whole target. First, the start, origin and end point of the two lines are copied to two ordered lists. Since both the pixel coordinates in the image and the metric coordinates on the target model are known for every point in the lists, it is now possible to extend the lines by adding points at the end. The principle of this algorithm is illustrated in figure 3.16. To add points at the current end of the line, the vector between the point at the very end of the line and the point one before it is calculated. When looking at the upper end of the list containing the points of a principal line, this can be written as

$$P' = P_E - P_0 \tag{3.5}$$

or in general with the current line length i as

$$P'_{i+1} = P_i - P_{i-1}. \tag{3.6}$$

The point P' is only a candidate location for a new point. From this location, a nearest neighbor search in the list of ellipses is performed. It must be checked, if the returned ellipse coordinates are close enough to the candidate point that was calculated. If such a point is found, this centroid is added to the end of the line list as element P_{i+1}. In the same step, the correct metric values on the target model are added to this centroid. Before the next iteration, the line length i is of course incremented to account for the new element. If there is no centroid in the centroid list near the candidate location P', the calculated coordinates of P' are added to the line list with a flag marking this as an invalid point. This way the line can be extended further, even if there are points missing. Stopping criteria for the addition of points are either when the calculated location P' is located outside of the image, or when there have been a certain number of consecutive points which were invalid.

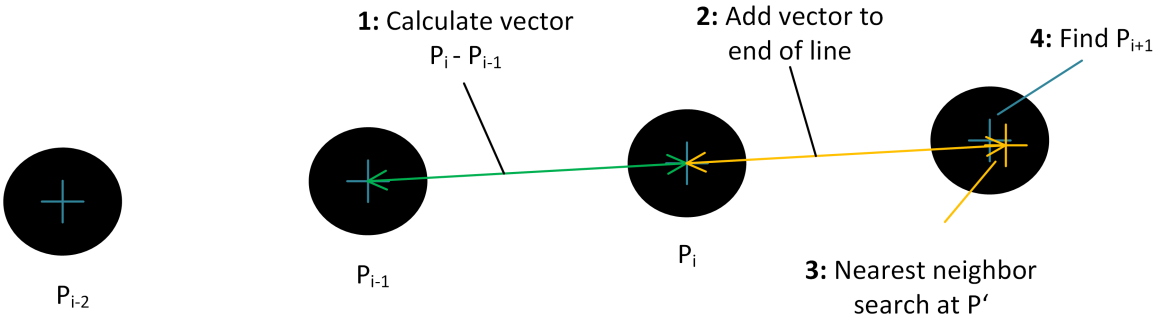


Figure 3.16: Algorithm to extend principal line lists at the ends

After applying this to the start and end of both line lists, the two principal axes are fully populated. The two lines form a cross, intersecting orthogonally at the origin of the target model grid. This cross partitions the points from the total focus image into four quadrants. Figure

3.17 shows the pixel coordinates of the points of the principal axes overlaid on the total focus image. The next step is to assign model points to the points in the four quadrants.

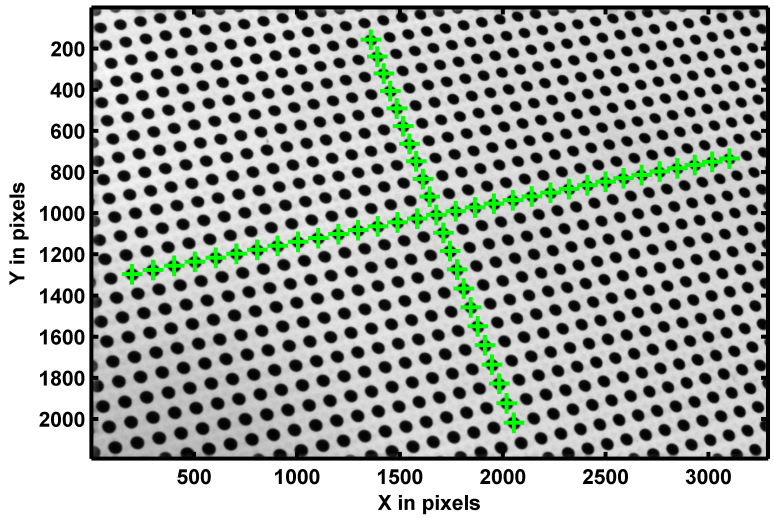


Figure 3.17: Pixel coordinates of the points of the principal axes

To populate the four quadrants, the points from the two principal axes are copied into a 2D array. It is useful to either use an array that is larger than the extent of the axes, or one that can be resized dynamically. A 2D array is suitable because the points of the target model also form a 2D array. This data structure makes searching for neighbors in the target model coordinate system very easy, because the known neighbors can be addressed by incrementing or decrementing the array indices.

To find points in the four quadrants, a similar approach to the one detailed above is used. To make the algorithm more robust, the coordinates of two neighboring points are used. To illustrate this, the fill algorithm for the upper-right quadrant is described. All points known up to now have 2D coordinates by which they can be addressed. The origin for example is $P_{0,0}$, the point just above it on the Y axis is $P_{0,1}$ and the point right of the origin on the X axis is $P_{1,0}$. All points on the principal axes ($P_{0,y}$ and $P_{x,0}$) are already known. Now the candidate location for point $P_{1,1}$ can be calculated. To do that, the vectors between the known points are calculated:

$$v_X = P_{1,0} - P_{0,0}, \tag{3.7}$$

$$v_Y = P_{0,1} - P_{0,0}. \tag{3.8}$$

Then the two vectors are added to the base point $P_{0,0}$:

$$P' = P_{0,0} + v_X + v_Y. \tag{3.9}$$

This gives a candidate location P', which depends on the position of two points relative to the origin. This makes the algorithm more robust, since the local grid can be distorted differently in X and Y directions. When the candidate location is known, once again a nearest neighbor search is performed, and the result is checked against a threshold, exactly the same way as above for expanding the principal lines. Figure 3.18 illustrates, that the algorithm takes both the differences in X and in Y direction into account to calculate the new candidate location.

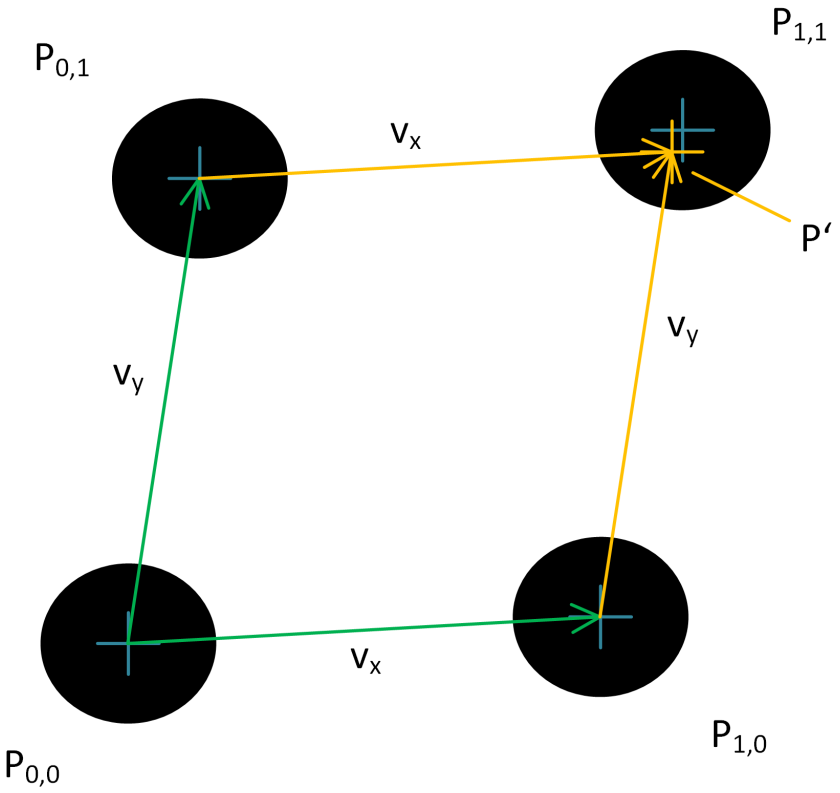


Figure 3.18: Calculation of candidate locations in upper-right quadrant

In the same fashion, the whole line of points $P_{x,1}$ can be found. To calculate the next candidate location for point $P_{2,1}$, the base point is now $P_{1,0}$, and the differences are calculated to points $P_{2,0}$ and the newly calculated $P_{1,1}$. Also, the same algorithm applies to any row parallel to the principal axes. They are filled with points one row after a time, until the whole quadrant has model points assigned to the points in the total focus image. This algorithm can be used on the other quadrants as well, when adapting the calculation of the difference

vectors v_X and v_Y. When all four quadrants have been searched in this fashion, the result looks like in figure 3.9 with the corresponding model points shown in figure 3.10.

Ideally, the result of this algorithm is, that each point visible in the image has a virtual model point with metric coordinates assigned to it. This concludes the description of the algorithm illustrated in figure 3.15. Since all points now have a pixel coordinate in the image, a virtual depth value and an associated metric coordinate on the target model, the optimization step can begin.

3.6 Projection model

The goal of the optimization is to minimize the reprojection error. To be able to calculate this error, the points found in the total focus image have to be projected into metric depth behind the MLA, as described in section 2.3.2. From there, the points are undistorted according to the models introduced in section 2.2. After that, the undistorted points are projected through the main lens according to the model detailed in section 2.1.2. Now the points are located in 3D metric space in front of the camera. Finally, the extrinsic pose (section 2.4.1) is applied to the points, and they ideally lie very close to the points of the target model. Then the distances in 3D space between the points projected from the total focus image and the points of the target model can be calculated.

Figure 3.19 gives an overview of the projection from virtual depth values behind the MLA to metric depth values in front of the camera. Shown in the illustration is also the naming convention for spaces involved in the projection. Points in space I are given in lateral pixel positions and virtual depth units. Space II contains points projected from Space I into metric coordinates. Points in space III have been undistorted and are given in metric coordinates relative to the principal plane of the main lens. This is equivalent to the image space as mentioned in chapter 2. Finally, space IV is located on the other side of the main lens and contains points in the object space. The coordinates of these points are metric, relative to the sensor center.

The first step is the projection from virtual depth values z_{I} to metric depth values z_{II} (compare equation 2.12 in chapter 2.3.2). To calculate this distance, the definition of the virtual depth is used (see equation 2.12):

$$v := \frac{a}{b_i}. \tag{3.10}$$

The equation is solved for the metric distance a and the variables are replaced:

$$z_{\mathrm{II}} = z_{\mathrm{I}} \cdot b_i. \tag{3.11}$$

To do the conversion, from virtual depth to metric depth, a value for the distance b_i between sensor and MLA depending on the lens type is needed. This is set to a user supplied value

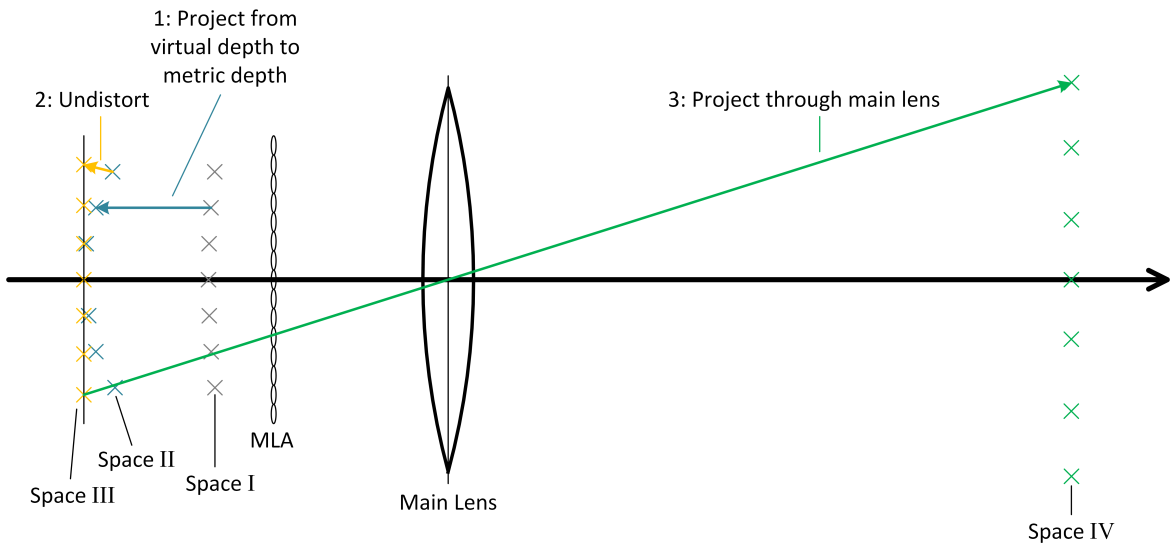


Figure 3.19: Projection from virtual depth to metric depth - undistortion - projection through main lens

initially, which is the same for all lens types. The initial value for b_i is derived from light field camera design theory and is known for each type of Raytrix camera.

After that, the pixel positions of the points in the total focus image $x_{\mathrm{I}}, y_{\mathrm{I}}$ must be converted to metric values $x_{\mathrm{II}}, y_{\mathrm{II}}$. These values are measured from the center of the sensor in millimeters. The conversion can be done with help of the constant intrinsic camera parameters for pixel size s_{pixel} and resolution R_x and R_y:

$$x_{\mathrm{II}} = -(x_{\mathrm{I}} - \frac{R_x}{2}) \cdot 2 \cdot s_{pixel}, \tag{3.12}$$

$$y_{\mathrm{II}} = -(y_{\mathrm{I}} - \frac{R_y}{2}) \cdot 2 \cdot s_{pixel}. \tag{3.13}$$

Now each point detected in the total focus image has a position in 3D space, which can be put into relation with the main lens. But first, the undistortion models must be applied:

$$x_{\mathrm{III}} = x_{\mathrm{II}} \cdot (1 + k_1 r^2 + k_2 r^4), \tag{3.14}$$

$$y_{\mathrm{III}} = y_{\mathrm{II}} \cdot (1 + k_1 r^2 + k_2 r^4). \tag{3.15}$$

These two equations shift the lateral position of the points to counter radial distortion (see equation 2.6 in chapter 2.2.1). Following this, the radial depth undistortion is applied (compare equation 2.11 in chapter 2.2.2):

$$z'_{\mathrm{III}} = z_{\mathrm{II}} + (1 + d_d z_d) \cdot (d_1 r^2 + d_2 r^4). \tag{3.16}$$

For all these calculations, the radius r is the euclidean distance relative to the distortion center x_o, y_o. The polynomials used for lateral and depth undistortion only use coefficients for second and fourth degree. This gives sufficiently exact results without the effects of overfitting the data. The degree of the polynomials could be increased at any time in the future, if a higher precision is needed.

For the current implementation of the calibration algorithm, the explicit model for lens skew is not realized. Tests during development showed, that the model will cause instability in the optimization process. The cause of this is probably, that the lens pose parameters are interdependent with projection and the two undistortion steps mentioned before.

Now the points have been undistorted in lateral and depth directions. The metric depth values are measured from the MLA. To project the points through the main lens, the image distance of each point must be calculated. To do this, the image distance B_L of the main lens must be calculated with the current values of the focal length f_L and the focus distance T_L:

$$B_L = \frac{T_L}{2}\left(1 - \sqrt{1 - 4\frac{f_L}{T_L}}\right). \tag{3.17}$$

The image distance is measured from the principal plane of the main lens to the TCP of the light field camera (see section 2.3.2). Therefore, the distance from the TCP to the MLA must be subtracted before adding the metric depth of the points to the image distance:

$$z_{\text{III}} = (z'_{\text{III}} - 2 \cdot b_i) + B_L. \tag{3.18}$$

Light field camera design theory states that the distance from the TCP to the sensor is equal to the distance from the sensor to the MLA b_i for the types of light field cameras considered here. This means the distance from the MLA to the TCP is $2 \cdot b_i$. Subtracting that gives the distance from each point z_{III} to the principal plane of the main lens.

Now the projection through the main lens can be applied. The distance z'_{IV} to which a point is projected in front of the principal plane of the main lens is calculated with the thin lens model (compare equation 2.2 in chapter 2.1.2):

$$z'_{\text{IV}} = \frac{z_{\text{III}} \cdot f_L}{z_{\text{III}} - f_L}. \tag{3.19}$$

The total metric distance z_{IV} from the point z'_{IV} to the TCP is then:

$$z_{\text{IV}} = z'_{\text{IV}} + B_L. \tag{3.20}$$

To get the lateral position of the point in front of the main lens, the pinhole model is used

(refer to equation 2.1 in chapter 2.1.1):

$$x_{\text{IV}} = \frac{x_{\text{III}} \cdot z_{\text{III}}}{G_P}, \tag{3.21}$$

$$y_{\text{IV}} = \frac{y_{\text{III}} \cdot z_{\text{III}}}{G_P}. \tag{3.22}$$

Now that the points have been projected to space IV, all projections involving the intrinsic camera parameters are finished. As a last step before calculating the distance error values, the extrinsic pose of the points has to be applied. Figure 3.20 shows how the points in space IV are rotated and translated to coincide with the points of the target model. The model points were created with metric coordinates for X and Y, and an implicit Z value of 0. Since coordinates in space IV are measured relative to the sensor center, this means the model points are in the same plane as the sensor.

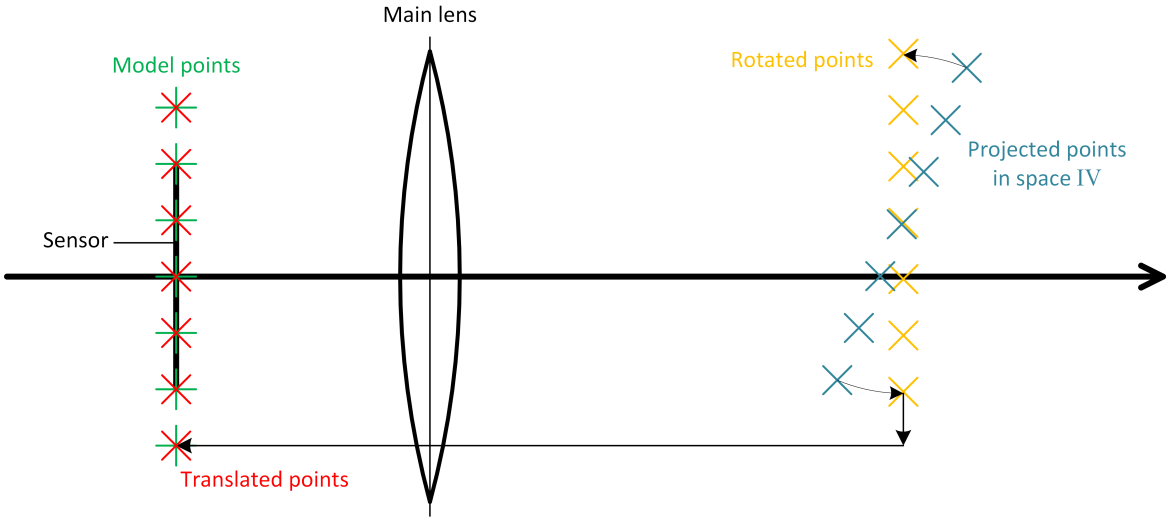


Figure 3.20: Applying the extrinsic pose

The rotation and translation are governed by the extrinsic pose. The effect of translation and rotation which define the extrinsic pose are illustrated in figure 3.20. For the translation, three extrinsic parameters T_x, T_y, T_z are needed. The rotation is expressed through quaternions[1], therefore four parameters q_1, q_2, q_3, q_4 are required. These extrinsic parameters are unique to each image of the target. As this calibration is designed to use multiple images of the target at different distances and angles, the extrinsic parameters must be provided for each image separately. In contrast to that, the intrinsic parameters apply to all projections in the same way. Table 3.3 gives an overview of all parameters of the camera calibration model.

[1] http://www.j3d.org/matrix_faq/matrfaq_latest.html

Table 3.3: Parameters of camera calibration model

Parameter	**Symbol**	**Type**	**Used for**	**Parameter count**
Pixel size	s_{pixel}	Intrinsic, constant	Projection from Space I to II	1
Resolution of sensor	R_x, R_y	Intrinsic, constant	Projection from Space I to II	2
Focal length	f_L	Intrinsic	Thin lens model	1
Focus distance	T_L	Intrinsic	Thin lens model	1
MLA-Sensor distance	b_i	Intrinsic	Projection from Space I to II	1...4
Distortion center offset	x_o, y_o	Intrinsic	Radial undistortion, depth undistortion	2
Radial distortion coefficients	k_1, k_2	Intrinsic	Radial undistortion	2
Radial depth distortion coefficients	d_1, d_2	Intrinsic	Depth undistortion	2
Depth dependent radial distortion coefficient	d_d	Intrinsic	Depth undistortion	1
Translation	T_x, T_y, T_z	Extrinsic	Target pose	3 per image
Rotation	q_1, q_2, q_3, q_4	Extrinsic	Target pose	4 per image

Figure 3.20 shows the points extracted from the total focus image perfectly coinciding with the model points. This means that the intrinsic and the extrinsic parameters for the projection model already have the optimum value. This of course can only be achieved by optimizing the intrinsic and extrinsic parameters until the distance error is minimized as far as possible. To do that, the calculations described in this section are performed many times, with the parameters varied by the optimization algorithm. The calculations above are therefore called the error function, because they allow calculating the error between our model and the real camera system. This error function can now be used in the optimization algorithm described in the next section.

3.7 Optimization algorithm

The algorithm used to minimize the error function is the Levenberg-Marquardt nonlinear least square solver. The implementation used for this work is available in the open source *cminpack* library[1]. The use of this algorithm is straightforward. A list of parameters that shall be optimized is passed to the algorithm. The algorithm then calls the error function supplied by the user whenever the effects of a change to the parameters must be evaluated. This process runs until an exit condition is reached. The desired exit condition is that the sum of squared error values is below a user-supplied threshold. This means that the parameters fit the data very well and the camera calibration was successful. Other exit conditions are that the maximum number of iterations has been exceeded or that changing the parameters has no influence on the value of the error function. In these cases, the optimization has either diverged or has been set up incorrectly. The user can influence the behavior of the algorithm by setting the parameters highlighted in table 3.4

The error function returns not the sum of all squared distance errors, but the error in X, Y and Z individually for each point. This way, the optimization algorithm can better sense the influence of the parameters. For example, the radial distortion coefficients mainly influence the X and Y errors, while the translation T_z only affects the error component in Z direction. Due to this, a high number of residual error values is returned by the error function.

The implementation of the algorithm was chosen as a C++ library, because this calibration algorithm has been implemented in C++. The next section gives some more details about the software environment.

[1] http://mirror.umd.edu/roswiki/cminpack.html

Table 3.4: Parameters of Levenberg-Marquardt algorithm

Parameter	Function	Value range
Number of parameters	Number of intrinsic and extrinsic parameters, changes with number of images used for calibration	Minimum 20 for one image of the target, add seven for each further image
Number of residuals	Values returned by the error function, three per detected point	100...10000
Maximum iterations	Exit optimization after this number of iterations	50...1000
Tolerance	Precision used for internal calculations	10^{-4}...10^{-6}

3.8 Software environment

The algorithms developed for this work are implemented in C++. Software developed in-house at Raytrix is generally developed either in C++ or C#. This software uses many functions from the extensive Raytrix Lightfield SDK (Software development kit), which has APIs (Application programming interfaces) for C++ and C. The Lightfield SDK provides all functions relating to depth estimation and image generation used in this work. Therefore the SDK is deeply integrated into this camera calibration algorithm, and the algorithm can not run without it. In fact, this algorithm was developed to be integrated into the Raytrix Lightfield SDK itself. A license for the SDK is priced at several thousand euros and can therefore can not be distributed with this work. To showcase the functionality of Raytrix light field cameras, a demo version of RxLive is included on the DVD supplied with this work. Figure 3.21 shows a screenshot of the RxLive user interface. RxLive can be used to open light field images, calculate depth and total focus images and visualize many of the topics mentioned in this work. There are a number of light field images provided on the DVD, ranging from images of calibration targets to real world application images.

RxLive will eventually incorporate a way to metrically calibrate light field cameras. Yet for the development of this algorithm, a separate test program was written. A screenshot of this development tool is shown in figure 3.22. Unfortunately, this test software can not run without the Raytrix light field SDK, so it can not be supplied on the DVD attached to this work. The data analyzed in the next chapter was created using this development tool.

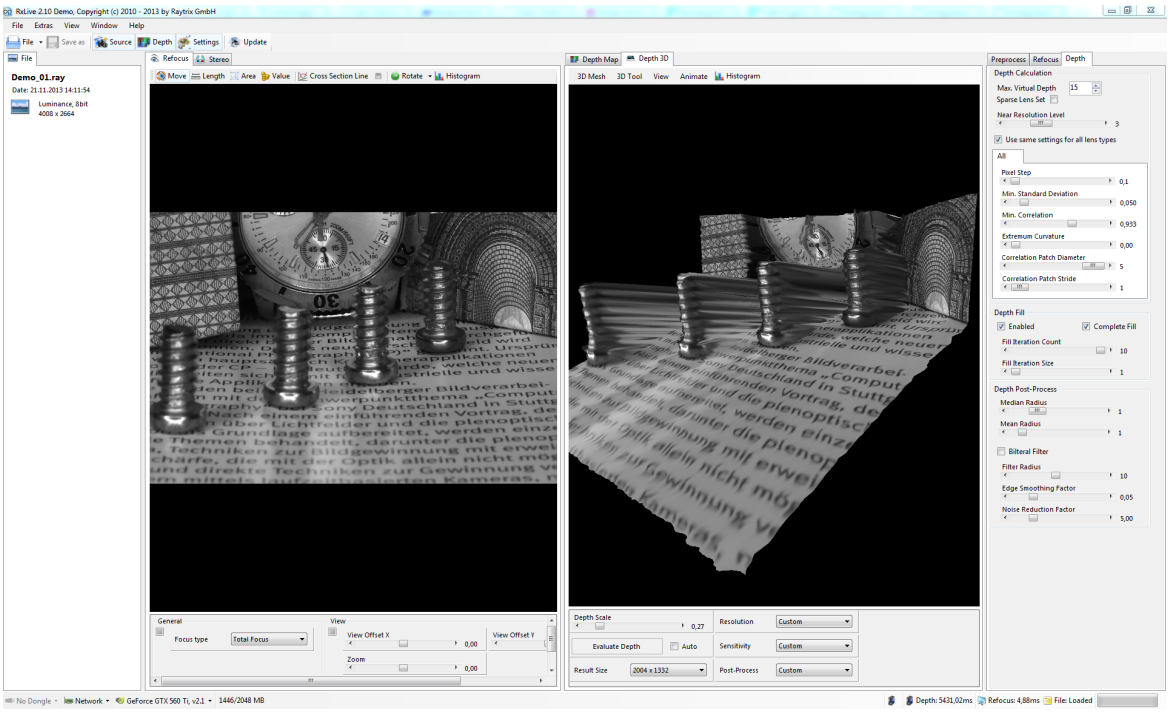


Figure 3.21: Screenshot of the Raytrix RxLive user interface

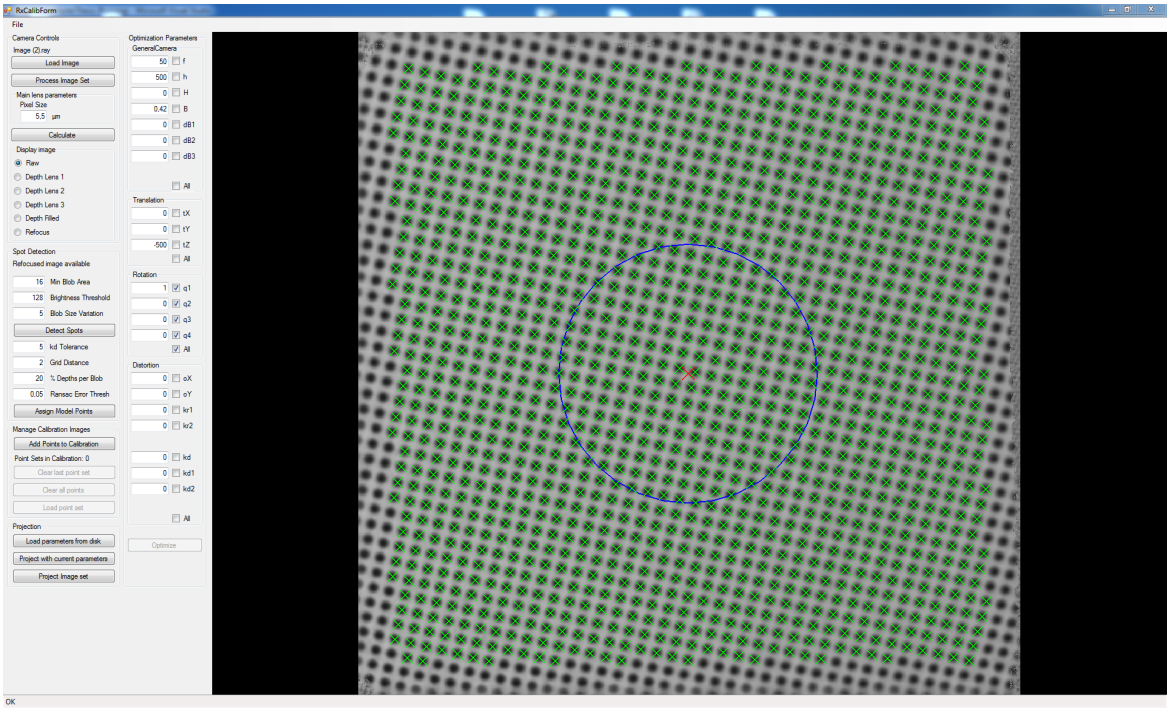


Figure 3.22: Screenshot of the metric calibration development tool

4 Results

The goal of this calibration algorithm is to provide a robust automated way for a customer to metrically calibrate a Raytrix light field camera. To evaluate the quality of the implementation, it is useful to start with the automatic detection of points in the image and the assignment of model points. This first step not only tests the image processing algorithms, but also makes a statement about the choice of the calibration target type. The focus of the analysis is on precision of the detection and on the robustness of the process.

After that, the data gained from the automatic detection should be used to test the optimization step. This again makes both a statement about the implementation of the algorithms, but also about the projection models that were chosen. Indeed, this is a test of the complete calibration algorithm, as the detection and assignment from before also influence the results. Quality can be measured by comparing the intrinsic parameters found through optimization with known ground truth values. Also, the effectivity of the undistortion models should be checked. Finally, the reprojection into metric space gives results which can also be compared with ground truth values.

4.1 Spot detection and spot assignment

In this section, the spot detection algorithm and the assignment of model points are evaluated. To do this, real world images taken with a variety of Raytrix light field cameras are used as input. The output of the algorithms is visualized by overlaying the calculated coordinates on the total focus images.

First, an ideal case is shown to explain the methodology. Figure 4.1 shows a total focus image, which was processed by the algorithm. In figure 4.2, the results of the spot detection and filtering are shown. Note that circles are drawn instead of the ellipses which were actually fitted. These centroid locations are then fed into the algorithm, which assigns the model points and builds the model target coordinate system. Figure 4.3 shows all locations, for which a model point could be correctly assigned. The larger marker in the middle shows where the center of the target was found. Finally in figure 4.4, the model points are plotted on a square grid to verify the assignment.

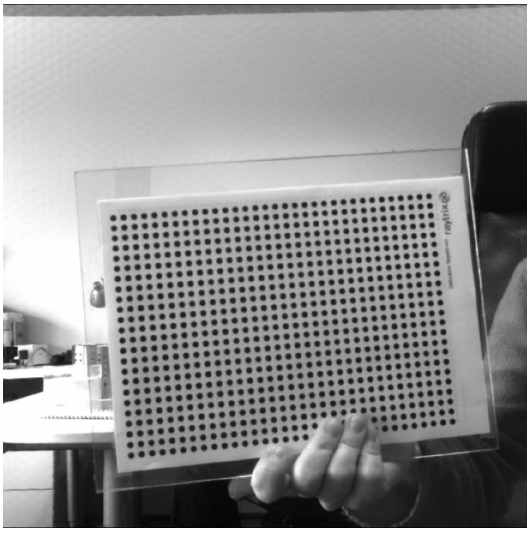


Figure 4.1: Total focus image of target

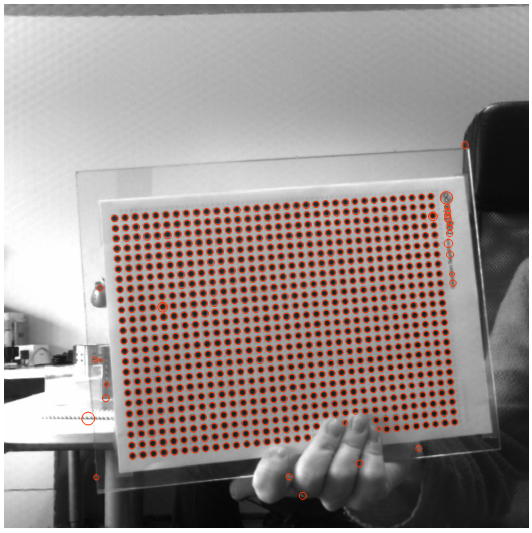


Figure 4.2: Spot detection result

This series of images shows the typical result of the spot detection and model point assignment algorithms. While the first step of detecting spots still produces some false positives, the second step of aligning the points on a grid leaves only correct points. It should be noted, that the spot detection performs very well, although there is a lot of structure in the background of the image, which could lead to false positive detections. Model points are assigned to every possible spot location, without producing a single false positive. A false positive which is included in the following optimization process could influence the results in a very negative way.

Although the bottom area of the target is obscured in some places, the combination of calculating candidate locations and then doing a nearest neighbor search is very robust. The algorithm does not lose count of the correct metric coordinates and the model points are assigned without flaws. The target was printed with a standard office printer at paper size A4. With the user-supplied distance of $8mm$ between spots, it can be seen that the extent of the model point coordinates corresponds to an A4 paper size of roughly $200mm$ by $300mm$. This result is absolutely fit to be used in the following optimization step.

A great advantage of this detection and assignment algorithm, is that it works even if only parts of the target are seen. It is not necessary to for the user to specify the amount of points in the image as in some other state of the art camera calibration algorithms.

Figure 4.5 shows the spot detection result on a target that was placed very close to the camera, so that only a part of the printed points could be seen. Again it should be noted, that not the actually fitted ellipses are overlaid, but circles with the radius equal to the major axis of each ellipse. The result of model point assignment for this image is seen in figure 4.6.

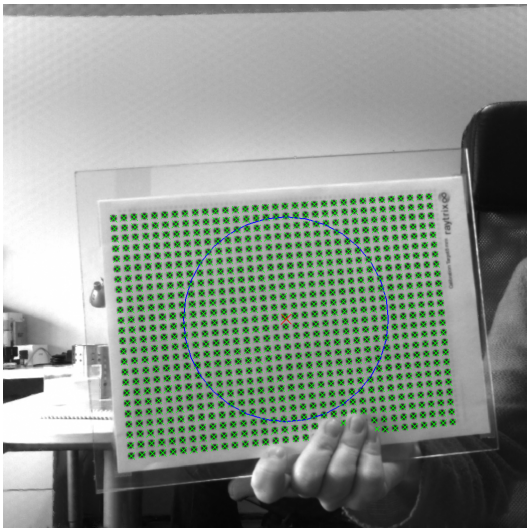


Figure 4.3: Spot assignment result

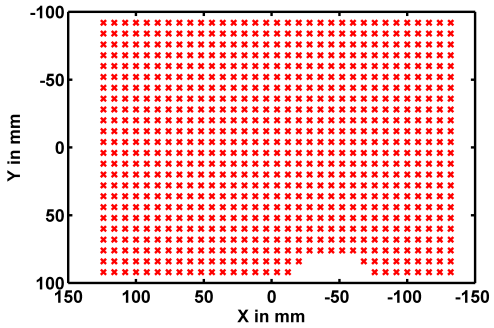


Figure 4.4: Model points

Once again, the results are very good, despite the fact that the lens used was of poor quality regarding sharpness and radial distortion, the lighting was very uneven and the spots are much larger than those in the image series before. Spot detection and assignments for the first target image (figure 4.1) and the second target image (figure 4.5) were performed with the same parameters (see table 3.2).

This means that the algorithms are very robust and work well even under adverse conditions. Figures 4.7 to 4.13 show some more results of the algorithm with a great range of different input images, also taken under adverse conditions. Sometimes not all points in the image are assigned a model point, for example if the spot detection found no centroid for some locations or the calculated candidate location for a point was too far off the real coordinates. Given the number of valid points that are usually found, these missing points have only small influence on the result of the calibration. As mentioned before, it is more important that no false positive matches are introduced, because these could severely disrupt the optimization.

Figure 4.13 shows an image that is on edge of being unusable. Many points on the target have not been detected due to the very uneven lighting. It is not recommended to use an image like this for calibration, though once again there were no false positive point assignments. The first results of customers taking images of targets for metric calibration show, that image quality in the field will probably be superior to the examples shown here. Therefore, the algorithm is robust enough to be used by the customer.

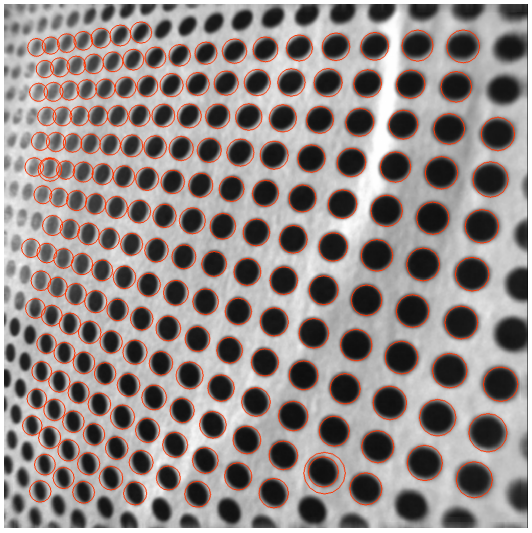


Figure 4.5: Spot detection result, only part of target visible

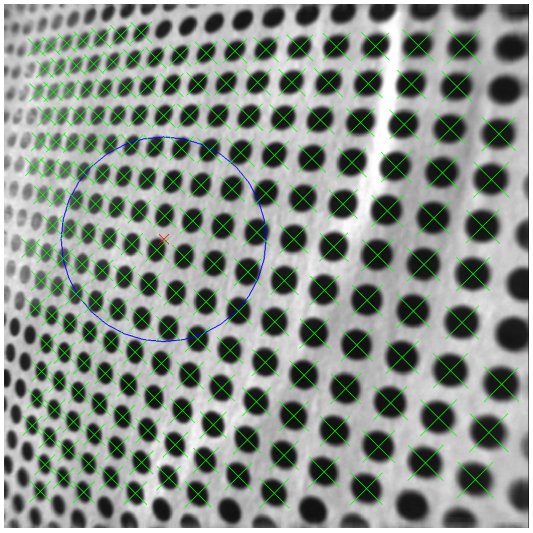


Figure 4.6: Spot assignment result, only part of target visible

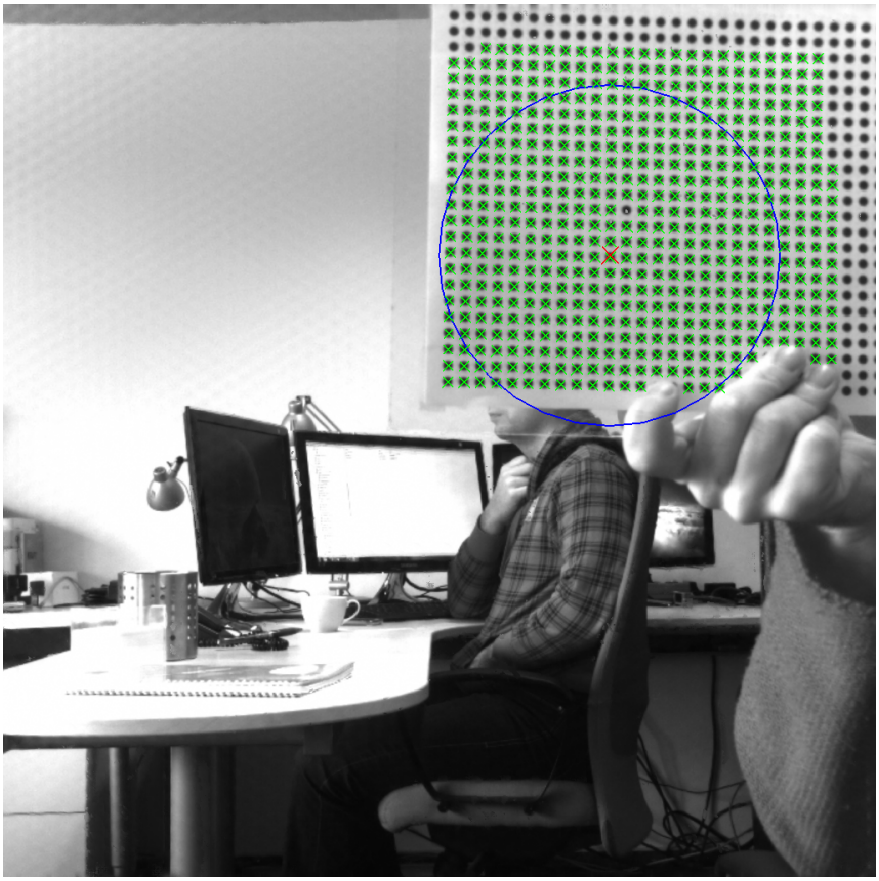


Figure 4.7: Target off center, busy image background

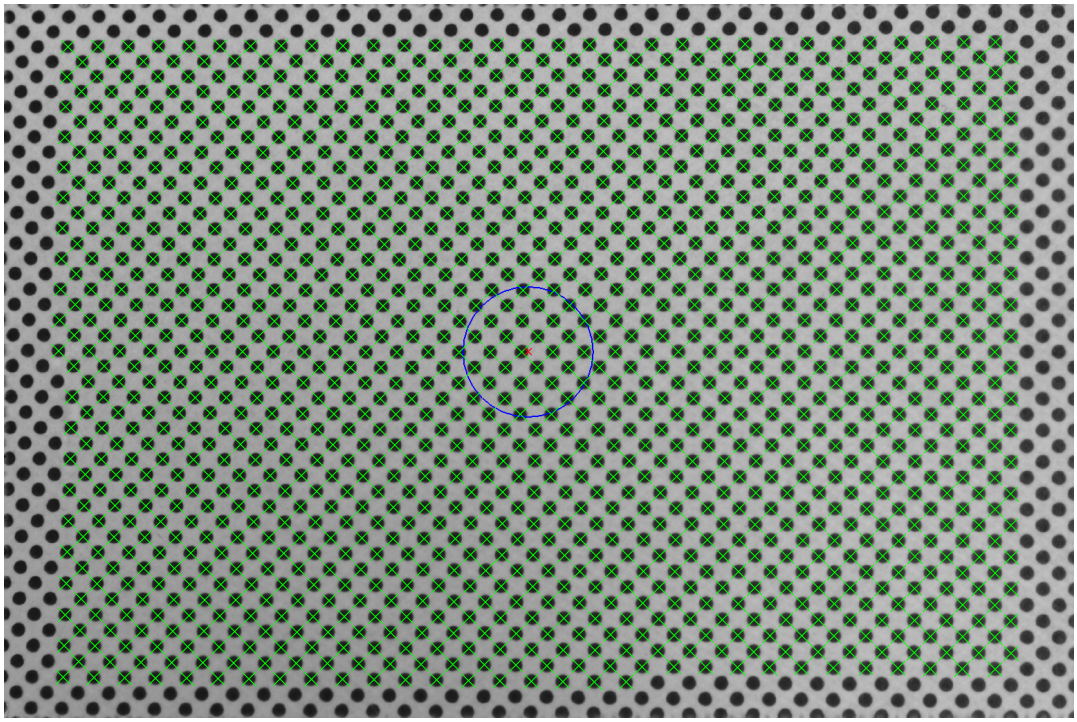


Figure 4.8: Target rotated

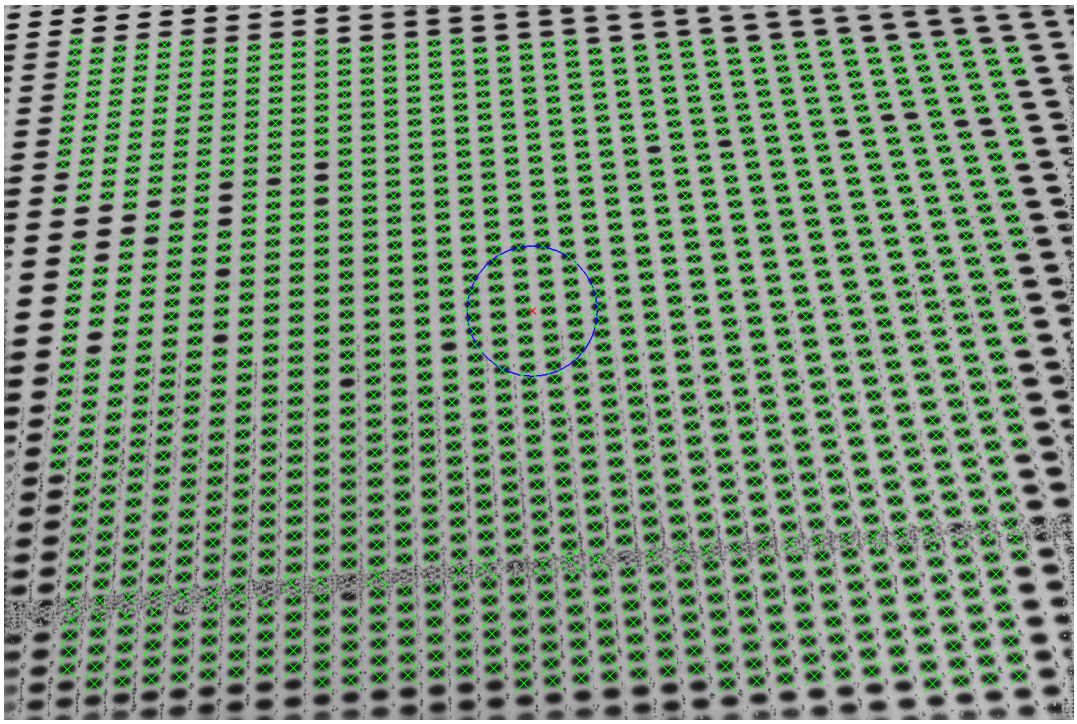


Figure 4.9: Defective refocus image

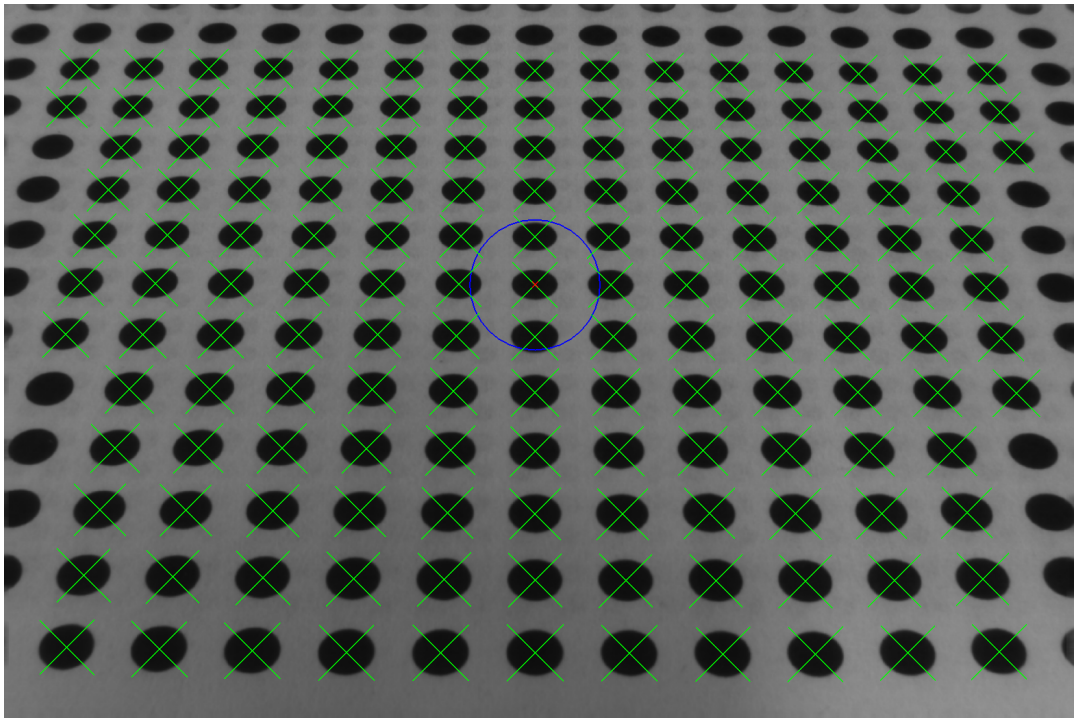


Figure 4.10: Large spots

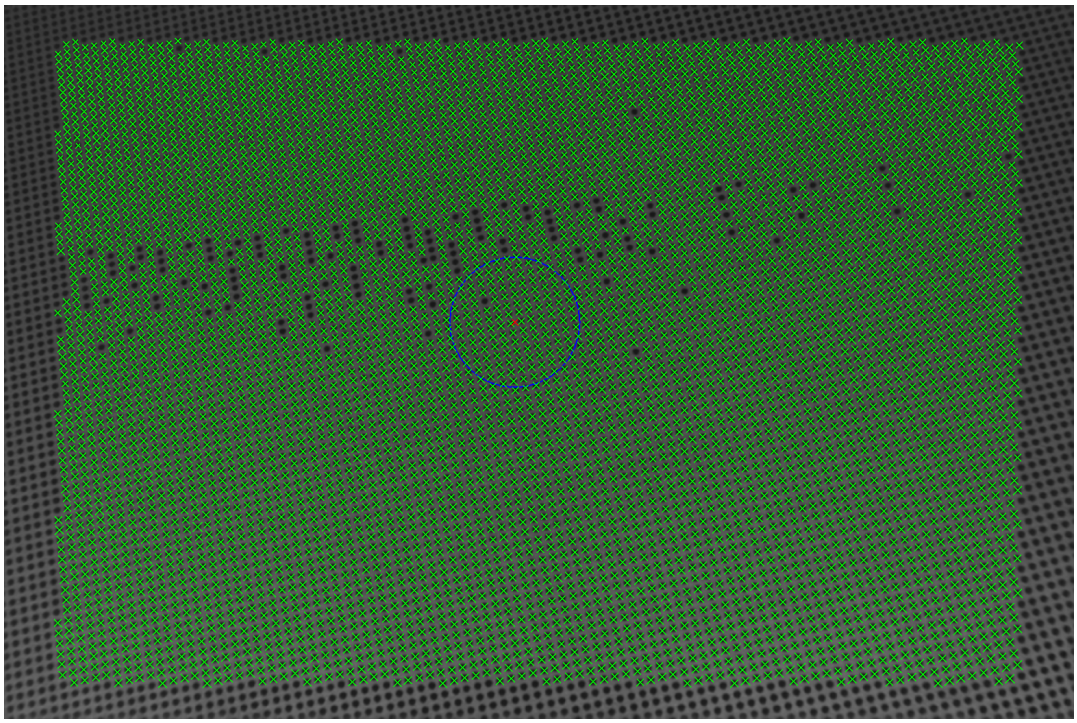


Figure 4.11: Small spots, underexposure

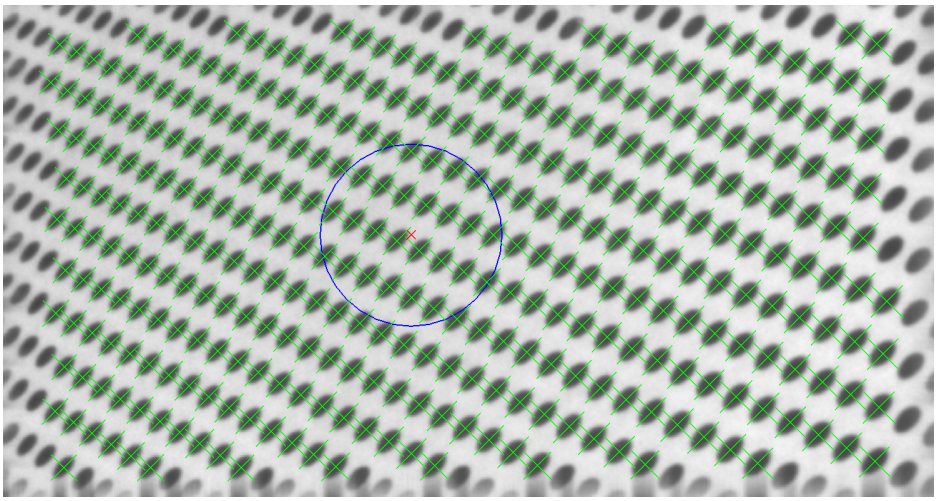


Figure 4.12: Perspective distortion

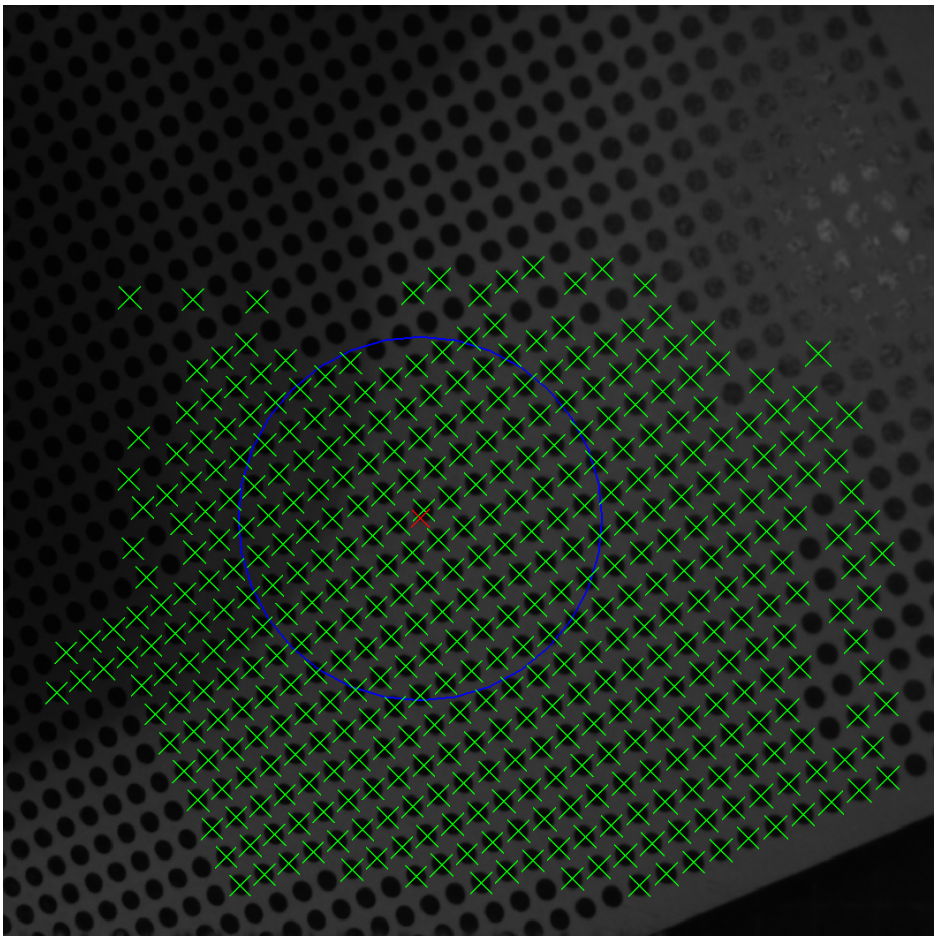


Figure 4.13: Underexposure, uneven lighting

4.2 Comparison of ground truth main lens parameters and optimization results

For most lenses, ground truth values for some intrinsic parameters are known. The focal length f_L and the current focus distance T_L are usually printed on the lenses themselves. The distance b from MLA to sensor is also known for all Raytrix light field cameras, but can not be published here.

Comparing these values with the output of the optimization algorithm is a first way to check the results for plausibility. To do this, a full calibration cycle for a certain camera-lens combination is performed. Table 4.1 compares the values for a Raytrix R29 camera with a Zeiss 100mm f/2 macro lens. This comparison of parameters is not meant to provide a

Table 4.1: Parameter comparison for Raytrix R29 with 100mm Zeiss f/2 macro lens at $700mm$ focus distance

Parameter	Ground truth value	Output of optimization
Focal length f_L	$100mm$	$119.728mm$
Focus distance T_L	$700mm$	$796.185mm$
MLA-Sensor distance b	100%	103.43%

measurable statement about the quality of the optimization. It is, as mentioned above, only a check for plausibility. Also, if the values were exactly equal to the known ground truth values, the camera calibration would not be necessary.

The deviations from the ground truth values are generally below 20%. Some macro lenses tend to increase their effective focal length, the closer they are focused. This behavior makes the lens design easier, since macro lenses are usually designed to give $1:1$ or $1:2$ magnification factors. The magnification of a lens depends on both the focus distance and the focal length. So if the focal length increases while focusing closer, it is easier to achieve a high magnification. This explains the deviation of roughly 20% for the focal length. The difference in focus distance can stem from the fact, that only the thin lens model is used for projection. As mentioned before, most real lenses and especially macro lenses at close focus distances should be approximated by the thick lens model. Most results of the calibration algorithm will show a focus distance value higher than the ground truth. It should be investigated, how the thick lens model can be applied to improve the results. Finally the distance from MLA to sensor is accurate to about 5%. This shows, that it is possible to estimate this distance using a camera calibration technique. Prior to this, the distance could only be measured during the assembly of the light field camera in a clean room.

The following tables 4.2 to 4.5 show some more results for different camera-lens combinations. The results are generally plausible for all combinations.

Table 4.2: Parameter comparison for Raytrix R29 with 100mm Zeiss f/2 macro lens at $440mm$ focus distance

Parameter	Ground truth value	Output of optimization
Focal length f_L	$100mm$	$152.071mm$
Focus distance T_L	$440mm$	$480.340mm$
MLA-Sensor distance b	100%	92.75%

Table 4.3: Parameter comparison for Raytrix R29 with 50mm Nikkor f/1.8 lens at $500mm$ focus distance

Parameter	Ground truth value	Output of optimization
Focal length f_L	$50mm$	$57.188mm$
Focus distance T_L	$500mm$	$592.706mm$
MLA-Sensor distance b	100%	99.94%

Table 4.4: Parameter comparison for Raytrix R5 with 75mm Kowa f/2.4 lens at $900mm$ focus distance

Parameter	Ground truth value	Output of optimization
Focal length f_L	$75mm$	$75.613mm$
Focus distance T_L	$900mm$	$1045.10mm$
MLA-Sensor distance b	100%	78.21%

Table 4.5: Parameter comparison for Raytrix R5 with 25mm Kowa f/2.4 lens at $500mm$ focus distance

Parameter	Ground truth value	Output of optimization
Focal length f_L	$25mm$	$23.465mm$
Focus distance T_L	$500mm$	$543.717mm$
MLA-Sensor distance b	100%	117.73%

To analyze the robustness of the calibration algorithm, some calibrations were performed with initial parameters deliberately set to a wrong value. Tables 4.6 and 4.7 show the parameter comparisons with the value of the initial parameters.

Table 4.6: Parameter comparison for Raytrix R5 with 25mm Kowa f/2.4 lens at $500mm$ focus distance and wrong initial values

Parameter	Ground truth value	Initial value	Output of optimization
Focal length f_L	$25mm$	$50mm$	$23.464mm$
Focus distance T_L	$500mm$	$500mm$	$543.708mm$
MLA-Sensor distance b	100%	200%	82.27%

Table 4.7: Parameter comparison for Raytrix R29 with 100mm Zeiss f/2 macro lens at $700mm$ focus distance and wrong initial values

Parameter	Ground truth value	Initial value	Output of optimization
Focal length f_L	$100mm$	$50mm$	$119.745mm$
Focus distance T_L	$700mm$	$700mm$	$796.266mm$
MLA-Sensor distance b	100%	50%	96.58%

The results presented in this section show, that the calibrations is very robust, even when the initial parameters are far from the ground truth. To get a more detailed look at the calibration process, the next section shows the results of the projection from the total focus image to metric space.

4.3 Analysis of reprojection into metric space

The examination of the projection into metric space begins by visualizing the results of the calibration model projection. The visualization uses data from an image taken with a Raytrix R5 camera and a Kowa 50mm f/2.4 lens focused at $500mm$. A metric calibration was run for this camera with a set of eleven images. After completion, the points extracted from one total focus image were projected with the calibration model and the optimized parameters. Figure 4.14 shows the result of this projection (compare figure 3.19 and 3.20 in chapter 3.6). The points plotted in black are the model points in the $Z = 0$ plane. The colored points are the points extracted from the total focus image. Ideally every colored point would lie exactly on top of it's associated black model point. Instead of this ideal result, this illustration of a real projection shows some small deviations. It should be noted that the Z axis has a different scale than the X and Y axes.

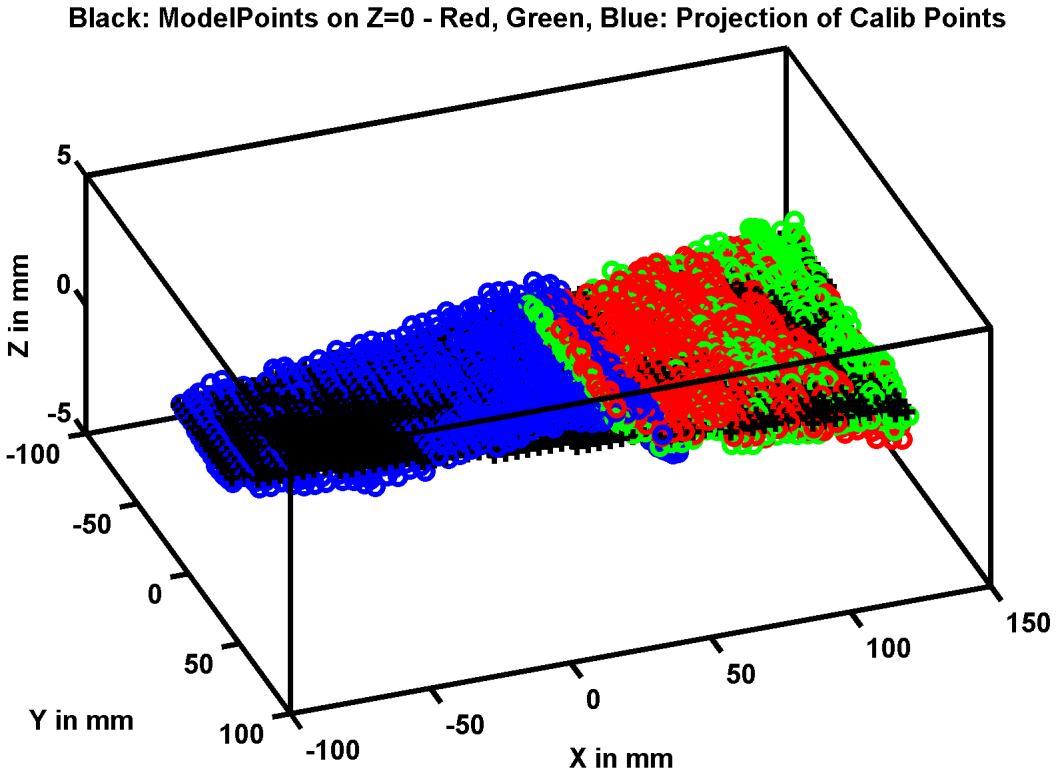


Figure 4.14: Points after projection into metric space with their associated model points

To better see the actual residual errors, figure 4.15 shows a heat map of the 3D distance error between each point from the target and it's associated model point. The heat map

shows that the error in most locations is below $1mm$. The area with the highest errors is found, where the valid depth estimations transition from the blue lens type to the green and red lens types. The average error for all points is $1.1997mm$. It is important to note, that there is no systematic error apparent in the heat map. If for example the undistortion models were not appropriate for the lens that was used, a radially symmetric structure could be seen in the heat map. The fact that the heat map for this example appears quite random means, that the projection and distortion models were well chosen.

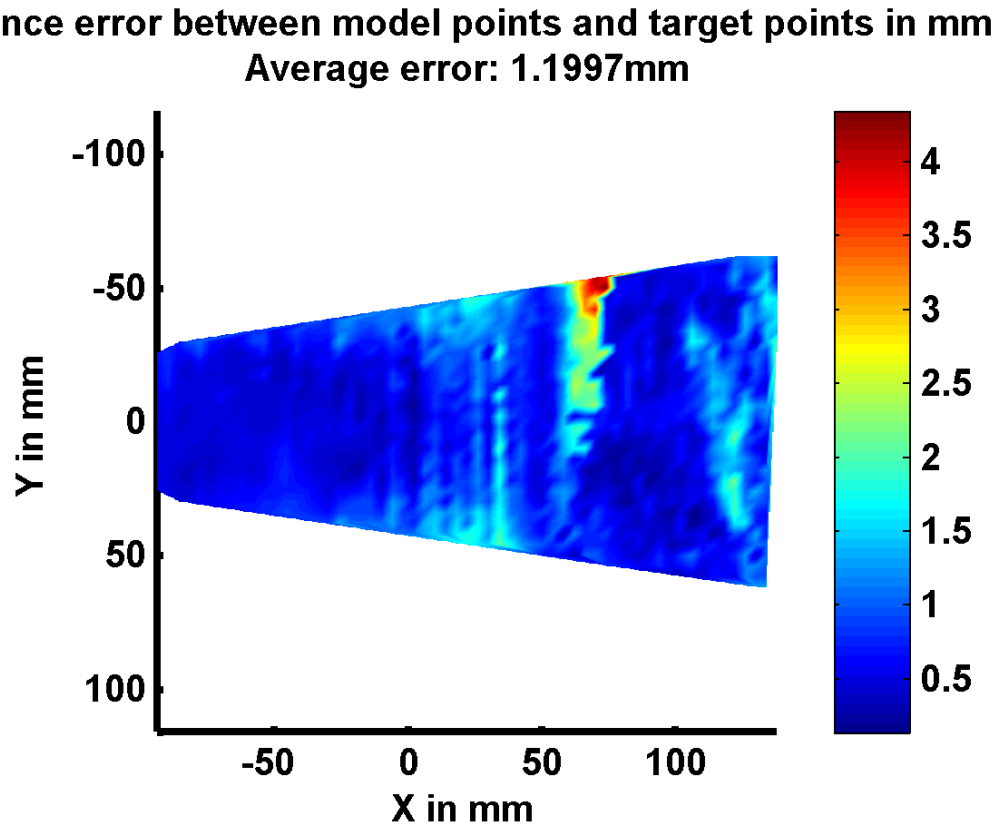


Figure 4.15: Heat map of 3D distance error between points from target and model points

This visual check of the projection error was often used during development. On the DVD attached to this work are a number of metric calibrations for different cameras and lenses, where the data used to plot these point clouds is available. MATLAB scripts to visualize the data are supplied as well.

More important for the actual application of this algorithm however, is the accuracy of the reprojection into metric depth. The metric calibration is supposed to allow measurement in metric 3D space.

To analyze the accuracy, a metric calibration for a camera/lens combination is performed.

Then a plane target is put parallel to the sensor at different metric depths. These distances are the ground truth. Images of the plane target are taken with the same camera/lens combination and the standard depth estimation is performed. Next, the calibration model is applied with the optimized intrinsic parameters. This means that all depth estimations are projected from virtual depth values behind the sensor to metric depth values in front of the camera. The positions in object space should now be as close as possible to the real target distance known for each image.

Figure 4.16 shows this projection for a Raytrix R5 camera with a Kowa 25mm f/2.4 lens focused at $300mm$. The projection parameters used were the initial values as written on the lens or measured by the user. This means that no metric calibration was performed for this. The graph shows, that by just using the initial values without the optimization step, no metric measurement is possible.

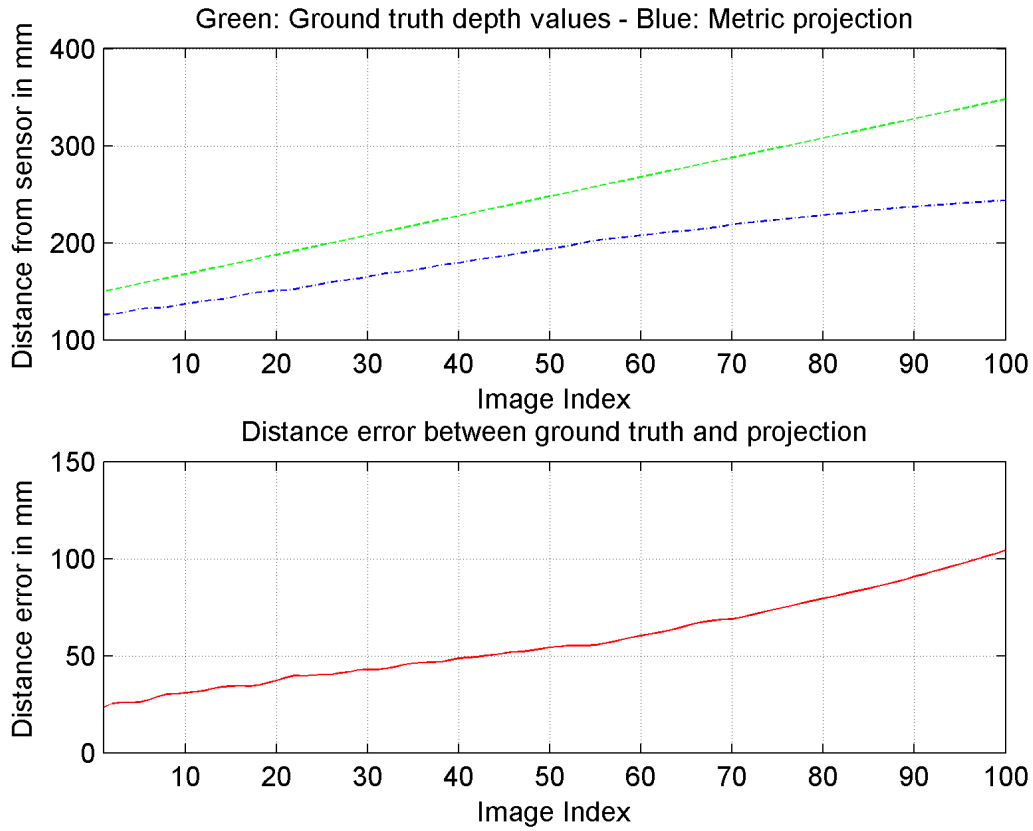


Figure 4.16: Error analysis of metric reprojection with initial parameters

The upper graph shows the distance from sensor plotted over the image number. For this analysis, 100 images of the plane target were taken. The target was moved from $150mm$ distance away from the camera to $350mm$ distance. The green line shows the ground truth

distance value of the target for each image. The goal of metric calibration is, that the output of the metric projection matches the ground truth data as good as possible. The output after metric reprojection is plotted in blue. As can be seen, there is a large distance between the two graphs. For example, the reprojected depth values of the last images differ from the ground truth by more than $100mm$.

The graph at the bottom gives the distance error between the blue and the green line from above. This line should ideally be as close to zero as possible. Any remaining error should appear random and unsystematic. But in this case, the graph shows that a large error remains. The error distance never falls below $30mm$ and increases to more than $100mm$ as the target is moved further away from the camera. More important than the absolute distance error is, that the error is systematic. The two lines have different slopes, which means that one of the projection parameters like the focal length, the focus distance or the MLA-sensor distance is not correct.

In contrast to this, figure 4.17 shows the same raw data, which is now projected with parameters obtained through this metric calibration algorithm. It can be seen that the two lines now show only a very small difference. The distance error graph shows that the value of the metric projection differs no more than $5mm$ from the ground truth over almost the whole depth range. The error is distributed randomly across most of the range, though for the highest distances an increasing depth error can be seen. This result shows that it is possible to measure absolute metric depth from the sensor, when the camera has been calibrated with the algorithm presented in this work.

For the next example, a Raytrix R29 camera with a Nikkor 50mm f/1.8 lens focused at $440mm$ was used. This time, the target was moved away from the camera from $210mm$ to a maximum distance of $500mm$ in steps of $5mm$. Figure 4.18 shows the results of the projection with optimized parameters. The upper plot shows, that the ground truth values and the reprojected values differ by a constant offset. The two lines are parallel to each other over the whole range of depth values. This can also be seen in the lower plot. The error value never drops below $50mm$, but is almost constant over a large part of the depth range. For the first 30 images, the distance between ground truth and metric reprojection is between $58mm$ and $59mm$. This means, if a constant offset of $58mm$ is added to the metric reprojection, the error is actually only $1mm$ over a large part of the depth range.

This constant offset is caused by the use of the thin lens model. As mentioned in chapter 2.1.3, only the thick lens model can account for these constant offsets. It appears, that in contrast to the Kowa 25mm lens used for the first example, the 50mm Nikkor lens can not be approximated as a thin lens. In practice, the constant offset is not problematic. The calibration only has to be performed one time. After that, the user can set any arbitrary constant offset. Even if the camera could measure the correct distance from sensor to object, it might still be useful to set an additional offset after that. If for example the height of boxes should be measured, it is useful to set an offset, so that the floor level is the reference point at a Z-value of $0m$.

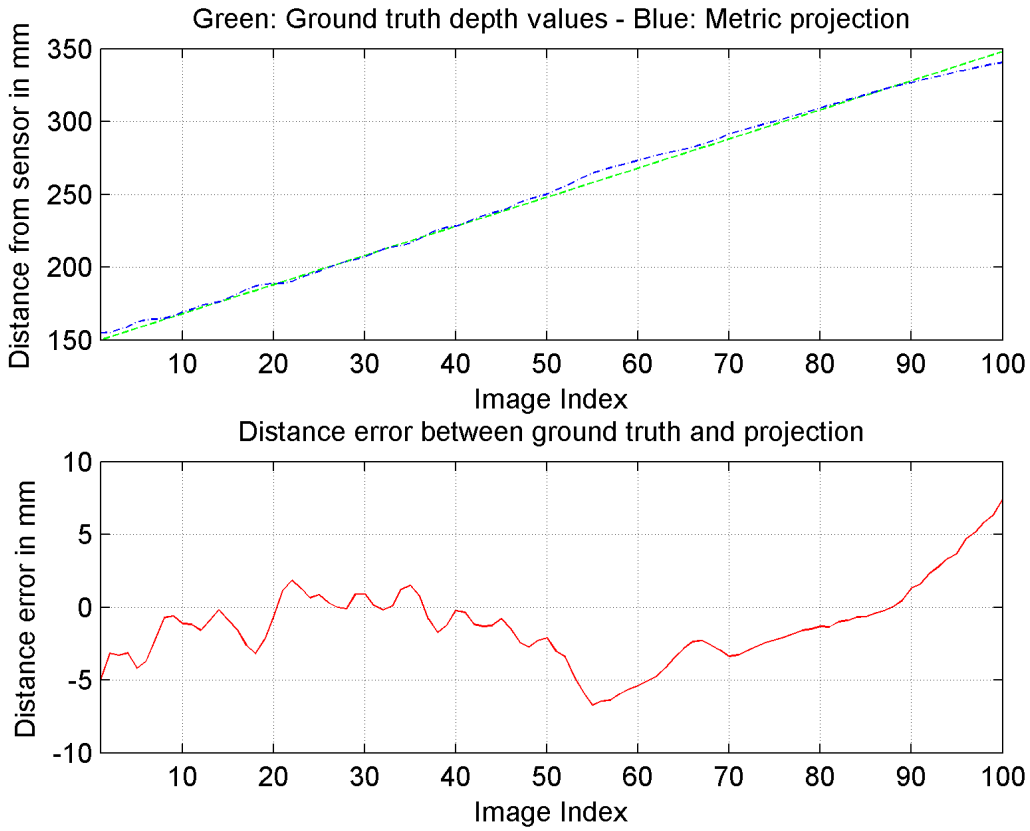


Figure 4.17: Error analysis of metric reprojection with optimized parameters - R5 camera

The metric calibrations for these examples were performed with sets of about ten images of the target. To see how the number of images affects the reprojection error, a series of metric calibrations with an R29 camera and a Zeiss 50mm f/1.4 lens focused at $500mm$ was carried out. At first, only a single image of the target was used for calibration. Then the reprojection error was calculated as described above and the error was calculated by taking the mean of the absolute distance error for all reprojections. This process was repeated with more and more images of the calibration target, until a set of ten images was used. Figure 4.19 plots the average reprojection errors over the number of calibration target images which were used.

It can be seen that the metric calibration did not work when only one image was used. The average distance from the reprojection to the ground truth was over $300mm$. For this reason, a multi image calibration technique is used in this work. Starting at two images, the average error drops considerably. For this test, the input images were chosen so that the points were spread evenly over all virtual depth values. With images 1 and 2, a range from virtual depth

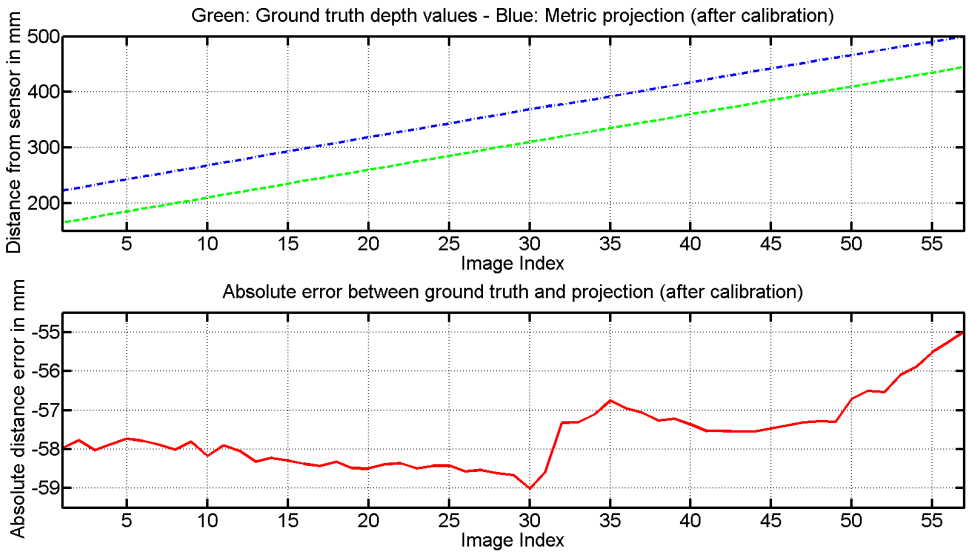


Figure 4.18: Error analysis of metric reprojection with optimized parameters - R29 camera

2.3 to virtual depth 6.5 was covered with a total of 5080 valid points extracted from the target images.

Once four images with carefully chosen point distributions are used, the average error can hardly be improved by using more images. With four images, a total of 10923 points from virtual depth 2.1 to virtual depth 7.3 were used. As mentioned before, the calibrations usually use a greater number of images, generally above ten different images. This is to ensure that the distribution of points is as even as possible. When a user takes images of the calibration target in different poses, it can not be expected that such a uniform distribution is given in the first two or four images. As the previous results show, ten or more images are usually sufficient to cover the range of depth values needed for a valid metric calibration.

Another analysis can be done regarding the efficiency of the depth undistortion model. For this test, a R5 camera with a Kowa 25mm f/2.4 focused at $300mm$ was used. Ten images of the calibration target were taken to perform a metric calibration. The calibration was run twice, once with the depth undistortion model enabled and once without it.

Images of a noise target at different distances from the camera were taken and reprojected into metric space as described above. These points should come to lie in a plane parallel to the sensor. To visualize any deviation from this plane, a paraboloid was fitted to the point cloud. Figure 4.20 shows two contour plots side by side. On the left, the depth undistortion model was turned off, while on the right the undistortion was turned on. The target was at a distance of $154mm$ from the sensor. It can be seen, that the reprojection without depth undistortion results in a deviation of up to $25mm$ from the theoretical plane. On the left, the

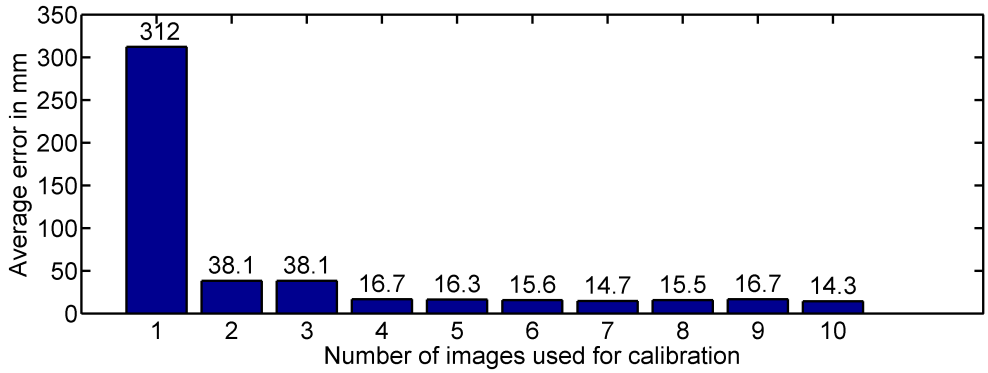


Figure 4.19: Average reprojection error over number of images

Table 4.8: Metric depth values of reprojected plane target with and without correction for MLA aberration

Lens type	Correction off absolute depth	Correction off relative error	Correction on absolute depth	Correction on relative error
Far	616.3255	0%	592.3340	0%
Middle	612.5430	-0.62%	590.4700	-0.32%
Near	609.4920	-1.12%	590.3000	-0.34%

activated undistortion model has produced a point cloud with only minor deviation from the expected plane.

In figure 4.21, the same comparison is shown for an image of a noise target taken at $326mm$ distance. It should be noted, that the image on the left without the undistortion model active shows only up to $20mm$ of deviation from the plane. This variation of the Petzval field curvature depending on the distance from the camera was discussed in section 2.2.2. The contour plot on the right shows that the undistortion also works at a greater distance from the camera. This means that the novel undistortion model is well fitted to the data and can be used for this metric calibration.

Finally, the effectivitiy of the newly introduced offsets b_i for the MLA-Sensor distance should be tested. To do this, calibrations were performed for an R5 camera with a Kowa 50mm f/2.4 lens focused at $500mm$. In the first calibration, only one MLA-Sensor distance b was estimated. For the second calibration, three values b_i were used in the calibration process. After that, a plane target was placed at a distance from the sensor where all three lens types produce valid depth estimations. The depth values for this target were reprojected to metric space in front of the camera. The desired result is, that the points in metric space form a plane. Yet table 4.8 shows, that without correction, the projection results in three planes at different distances from the sensor. The distance values were obtained by fitting a plane to the point cloud and evaluating the value in the center.

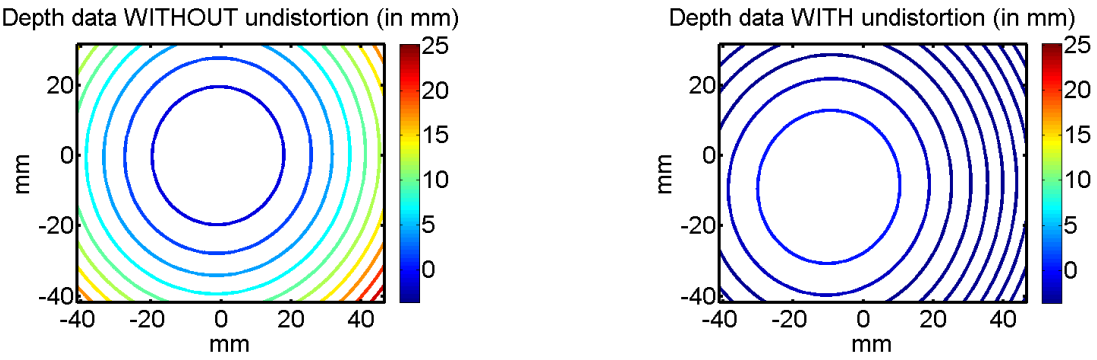


Figure 4.20: Contour plot of depth error values for plane target at $154mm$ distance from sensor

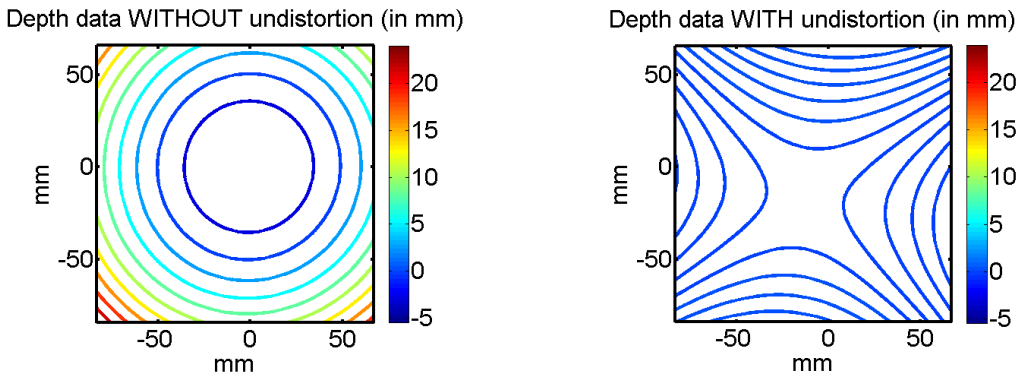


Figure 4.21: Contour plot of depth error values for plane target at $326mm$ distance from sensor

The percentual errors are calculated relative to the plane projected from the data of the far lens type. When looking at the relative error with the new correction method active, it can be seen that the distance between the planes has be reduced compared to before. For example, points with depth estimations from the middle and near lens type would now just lie $0.17mm$ apart, while this error without correction would be over $3mm$. If a systematic error of $3mm$ exists for depth estimations of the same point of the object with different lens types, reducing this error should greatly reduce the noise of the resulting depth map of the object. Unfortunately, it can not yet be analyzed if this corrections has an influence on the noise of the depth values when calculating a full depth map, because the camera model developed for this metric calibration algorithm has not yet been fully incorporated into the projection algorithms of the Raytrix RxLive software.

But as the results above show, this implementation allows modelling a light field camera with a main lens accurately, so that a metric calibration can be performed. It has been shown that the calibration process is very robust. Calibrations were performed for many different cameras, lenses and lens settings. No special care was taken with regard to the images of the calibration targets. The lighting, pose and scale of the targets varied greatly from shot to shot, yet all calibration processes have converged to a plausible set of parameters. The detection algorithms performed good, while not introducing any false positive points. Finally, the newly introduced correction for MLA aberration and the depth undistortion have produced the expected results. This means that this implementation is fit to be incorporated into the Raytrix RxLive software package. The next chapter will give an outlook to further possiblities and sums up this work.

5 Conclusion

As the results in the previous chapter show, the camera calibration algorithm implemented for this work performs very well. The desired robustness of the spot detection and spot assignment has been achieved. The calibration target designed for this work allows a relatively simple set of algorithms to perform the automatic feature location extraction. Also, the calibration target is low-cost, since it can be printed with sufficient accuracy using an office printer.

With a finished valid calibration, metric 3D values can be calculated from virtual depth values. In some cases, even a reconstruction of the absolute distance from sensor to the object is possible. With the undistortion models applied during reprojection, measurements in 3D space give accurate results. For the first time, it is possible to rectify the radial depth distortion caused by Petzval field curvature. This allows the use of low-cost lenses, which are not corrected for these types of distortion. The camera model has proven to be accurate and it will be incorporated in the Raytrix RxLive software package.

For the first time, this algorithm enables metric calibration by the user in the field. Formerly, a metric calibration was only possible by using a large and expensive motorized linear stage. Now the images needed for a metric calibration can be taken in minutes. The actual metric calibration algorithm can be performed with these images at any time. This means that when images of a scene or object have been taken without a metric calibration, it is possible to project the depth values of these images into metric space afterwards, when a metric calibration has been performed.

Future work could be done on the spot detection algorithms to further increase the quality and robustness of the algorithm. In other state of the art camera calibration methods, more advanced algorithms are used to find the feature locations. This algorithm was designed to be modular, so a new spot detection algorithm can easily be inserted into the existing framework. Simultaneously, it should be checked if commercially available high precision calibration targets increase the quality of the calibration.

This algorithm could also be adapted to support multiple calibration targets in the image. If for example, three targets orthogonal to each other are imaged, a better distribution of points could be achieved, while minimizing the effort for the user.

More analysis could also be done on including the full pose of the main lens in the intrinsic parameters. It is possible that a different set of parameters to describe the pose could improve the stability of the optimization. It could also be beneficial to include new error terms based on the lens pose parameters in the optimization to prevent it from diverging.

Another possibility for improving the projection model is to analyze how the thick lens model can be used. Finding the initial values for such a model will be especially difficult, since the lens thickness is generally not known. An iterative approach to the real lens thickness might also be difficult, because the optimization may diverge instead of reaching a valid minimum. None the less, using the thick lens model would allow the calibration of microscope optics, which can not be approximated as a thin lens. Since microscopy is a very interesting field of application for light field cameras, this should be further pursued.

Finally, the biggest step to increase the quality of the calibration would be to extend the calibration to the raw image. In this work, only the total focus image and the corresponding depth map have been used. These are computed from the raw image, which still shows the structure of the microlens array. When using this raw image for calibration, the intrinsic parameters of each microlens could be found. This would provide a much better way to rectify aberrations of the microlens array, and also aberrations of individual microlenses. This step would unfortunately require many changes to the algorithms implemented here and is therefore left for future work.

Bibliography

[1] Hecht, E.: *Optik*, verbesserte Auflage 2009, Oldenbourg Wissenschaftsverlag, 2001, ISBN-13: 978-3486588613

[2] Johannsen, O. et al.: *On the Calibration of Focused Plenoptic Cameras* http://hci.iwr.uni-heidelberg.de/Staff/bgoldlue/papers/JHGP13_gcpr_inm.pdf [28.12.13]

[3] Brown, D. C.: *Decentering distortion of lenses*, Photometric Engineering 32(3), 444-462 (1966)

[4] Tsai, R. Y.: *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, Vol. RA-3, No. 4, August 1987

[5] Danserau, D., Pizarro, O. Williams, S.: *Calibration and rectification for lenselet-based plenoptic cameras*, Proc. International Conference on Computer Vision and Pattern Recognition (2013)

[6] Scheimpflug, T.: *Improved Method and Apparatus for the Systematic Alteration or Distortion of Plane Pictures and Images by Means of Lenses and Mirrors for Photography and for other purposes*, GB Patent No. 1196. Filed 16 January 1904, and issued 12 May 1904

[7] Van Walree, P.: *Astigmatism and field curvature*, http://toothwalker.org/optics/astigmatism.html, [28.12.13]

[8] Yves, F.: *Parallax stereogram and process of making same*, (04 1903)

[9] Lippmann, G.: *Épreuves réversibles donnant la sensation du relief*, J. Phys. Theor. Appl. 7(1), 821-825 (1908)

[10] Ng, R.: *Digital lightfield photograpy*, https://www.lytro.com/downloads/resources/renng-thesis.pdf, [28.12.13]

[11] Perwass, C., Wietzke, L.: *Single lens 3D camera with extended depth-of-field*, Proc. SPIE 8291, Human Vision and Electronic Imaging XVII (2012)

[12] Kolmogorov, V., Zabih, R.: *Multi-camera scene reconstruction via graph cuts*, Proc. European Conference on Computer Vision. pp. 82-96 (2002)

[13] Bishop, T., Favaro, P.: *The light field camera: Extended depth of field, aliasing, and superresolution*, IEEE Transactions on Pattern Analysis and Machine Intelligence 34(5), 972-986 (2012)

[14] Lumsdaine, A., Georgiev, T.: *The focused plenoptic camera*, In Proc. IEEE International Conference on Computational Photography. pp. 1-8 (2009)

[15] Horn, B. K. P.: *Tsai's camera calibration method revisited*, http://people.csail.mit.edu/bkph/articles/Tsai_Revisited.pdf, [28.12.13]

[16] De la Escalera, A., Armignol, J. M.: *Automatic Chessboard Detection for Intrinsic and Extrinsic Camera Parameter Calibration*, Sensors 2010, 10, ISSN 1424-8220

[17] Mateos, G. G.: *A camera calibration technique using targets of circular features*, http://dis.um.es/∼ginesgm/files/inv/siarp2000/siarp2000.pdf, [28.12.13]

[18] Fischler, M. A., Bolles, R. C.: *Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography*, Comm. of the ACM 24 (6): 381-395

[19] Shewchuk, J. R.: *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*, http://www.cs.cmu.edu/∼quake-papers/painless-conjugate-gradient.pdf, [28.12.13]

[20] Levenberg, K.: *A Method for the Solution of Certain Non-Linear Problems in Least Squares*, Quarterly of Applied Mathematics 2: 164-168

[21] Marquardt, D.: *An Algorithm for Least-Squares Estimation of Nonlinear Parameters*, SIAM Journal on Applied Mathematics 11 (2): 431-441

[22] Lourakis, M. I. A.: *Levenberg-Marquardt nonlinear least squares algorithms in C/C++*, http://users.ics.forth.gr/∼lourakis/levmar/levmar.pdf, [28.12.13]

Glossary

Aberration Error in the projection of a lens

API Application programming interface, specifies how software components interact with each other

Centroid The center of a region. Can be calculated in 2D or 3D as the barycenter of a distribution of points

Depth map A 2D image where the values represent distance from the camera

Disparity The difference in image location of an object seen by two cameras or two microlenses

Extrinsic parameters Parameters of the camera model that describe the pose of the camera relative to a global coordinate system

False positive A false positive is a wrong detection of a feature, when actually no feature is present

Feature In image processing a feature is anything that can be detected in an image

Image space The space from the principal plane of the main lens in direction of the sensor

Intensity image A 2D grayscale or color image showing the brightness of the scene

Intrinsic parameters Parameters of the camera model that describe the camera itself

Minimization Also known as optimization, tuning of parameters of a model until the value of an error fuction becomes minimal

MLA Microlens array, placed in front of the sensor of a light field camera

Noise target Plane target with a random high contrast pattern designed to give optimal depth estimations when imaged with a light field camera

Object space The space from the principal plane of the main lens in direction of the object

Optical axis Virtual axis through the center of a lens, coaxial with rotationally symmetric lens elements

Optimization Also known as minimization, tuning of parameters of a model until the value of an error fuction becomes minimal

Petzval field curvature Causes the main lens to project a plane object to a curved surface in image space

Principal plane Virtual plane at the center of a lens, normal to optical axis

Profilometer (Mechanical) device to measure surface roughness

Quaternions Quaternions can represent a rotation in 3D space using four parameters

Reprojection Virtual projection through the main lens from image space to object space

SDK Software development kit

TCP Total covering plane at virtual depth 2.0 on which the main lens must be focused

Virtual depth Native unit for depth estimations with Raytrix light field cameras

Versicherung über die Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen habe ich unter Angabe der Quellen kenntlich gemacht.

Ort, Datum	Unterschrift