
Fast Computer Vision based Geometry Estimation

Bachelor Thesis

Authors

Cédric Renda, Manuel Tischhauser

Supervisor

Prof. Dr. Guido M. Schuster

Subject

Image Processing

HSR Hochschule Für Technik Rapperswil

May 14, 2020

Abstract

Introduction

Approach

Conclusion

Contents

Abbreviations	1
1 Introduction	3
2 Theory	5
2.1 Camera Calibration in OpenCV	5
2.1.1 The pinhole camera model	5
2.1.2 The distortion model in OpenCV	6
2.1.3 Reprojection error	8
2.1.4 Error propagation as further quality assessment	9
2.2 Measurement with back-light illumination	10
2.2.1 Light source	10
2.2.2 Light ray distribution	10
2.2.3 Light properties	10
2.2.4 Measurement precision changes	10
2.2.5 Image sensor properties	10
3 Development	11
3.1 Calibration	11
3.2 Software	11
3.3 Hardware	11
3.3.1 Demonstrator	11
3.3.2 Back-light	12
3.3.3 Diffusion	12
3.3.4 Computer	12
3.3.5 Camera	12
3.3.6 Production cost	13
4 Results	15
5 Conclusion	17
References	19
List of figures	19
List of tables	21

Statement of Plagiarisms	23
A Requirements	27
A.1 Assignment	27
A.2 Requirement Specification	28

Abbreviations

RMS Root Mean Square

Chapter 1

Introduction

Chapter 2

Theory

This chapter takes a closer look at the theory and technology applied in this thesis.

2.1 Camera Calibration in OpenCV

Camera calibration is needed to obtain camera parameters like focal length and center point. Further, it provides a method to correct distortions caused by the imperfect optics according to a certain distortion models. For that, a set of images of a known pattern (usually a checkerboard) have to be taken from different view-points an angles. These points provide the training data which is needed to estimate all the coefficients.

This section describes the theory behind the calibration process in OpenCV 4.3.0, which is the latest version at the time of writing this thesis.

2.1.1 The pinhole camera model

The functions OpenCV provides to calibrate the camera use the so-called pinhole camera model **cv_calib**. This model describes, how a 3D-point specified in world-coordinates (P_w) is transformed to a 3D-point in camera-coordinates (P_c) and then further projected onto the image plane (p). After this step, the point is described as a 2D-point in pixel coordinates. Figure 2.1 illustrates this setup.

Figure 2.1: Pinhole-model (not up to date).

The transition from the world-coordinates to the camera-coordiantes can be described as

$$\underbrace{\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix}}_{P_c} = \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}}_R \underbrace{\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix}}_{P_w} + \underbrace{\begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}}_t.$$

The vector P_w is first rotated by R and the translated by t . This can be written in one single matrix:

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{pmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \Leftrightarrow P_c = (R \mid t) \begin{pmatrix} P_w \\ 1 \end{pmatrix}. \quad (2.1)$$

As a result of the theorem of intersecting lines, the projection from P_c to p is described as

$$\underbrace{\begin{pmatrix} u \\ v \end{pmatrix}}_p = \begin{pmatrix} f_x \cdot X_c / Z_c \\ f_y \cdot Y_c / Z_c \end{pmatrix} + \begin{pmatrix} c_x \\ c_y \end{pmatrix}.$$

where f_x and f_y are the focal length f (in world units) normalized by their respective pixel size (in world units). Thus f_x and f_y are the same, if the pixels are quadratic.

By adding the principal point $(c_x \ c_y)^T$, which is usually close to the image center, it is taken into account, that pixel-coordinates are specified with respect to the upper left corner of the image plane. . It is now simpler to write this in homogeneous coordinates:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim s \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}}_K \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} \Leftrightarrow s \begin{pmatrix} p \\ 1 \end{pmatrix} = K \cdot P_c, \quad (2.2)$$

where s is an arbitrary scaling factor and K is called the camera matrix. The overall transition from world- to pixel-coordinates is the result of combining 2.1 and 2.2:

$$s \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{pmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} \Leftrightarrow s \begin{pmatrix} p \\ 1 \end{pmatrix} = K \begin{pmatrix} R & | & t \end{pmatrix} \begin{pmatrix} P_w \\ 1 \end{pmatrix} \quad (2.3)$$

The rotation and translation in $\begin{pmatrix} R & | & t \end{pmatrix}$ are called the extrinsic parameters. The camera matrix K contains analogously the linear intrinsic parameters.

2.1.2 The distortion model in OpenCV

Non linear distortions, which appear before the projection in 2.2 should be considered too. OpenCV takes the effects of radial, tangential and thin prism distortion into account **cv_calib**.

Radial Distortion

Radial distortion is caused by the flawed curvature of the lens **weng**. It can be modeled with

$$\begin{pmatrix} x'' \\ y'' \end{pmatrix} = \begin{pmatrix} x' \frac{1+k_1 r^2+k_2 r^4+k_3 r^6}{1+k_4 r^2+k_5 r^4+k_6 r^6} \\ y' \frac{1+k_1 r^2+k_2 r^4+k_3 r^6}{1+k_4 r^2+k_5 r^4+k_6 r^6} \end{pmatrix}, \quad (2.4)$$

where x' and y' are coordinates, described in camera-coordinates, normalized with Z_c

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} X_c / Z_c \\ Y_c / Z_c \end{pmatrix}$$

from and r is the radius as taken with respect to the principal point $(c_x \ c_y)^T$

$$r^2 = x'^2 + y'^2.$$

This type of distortion is symmetrical about the optical axis. Figure 2.2 illustrates the effect on a rectangle.

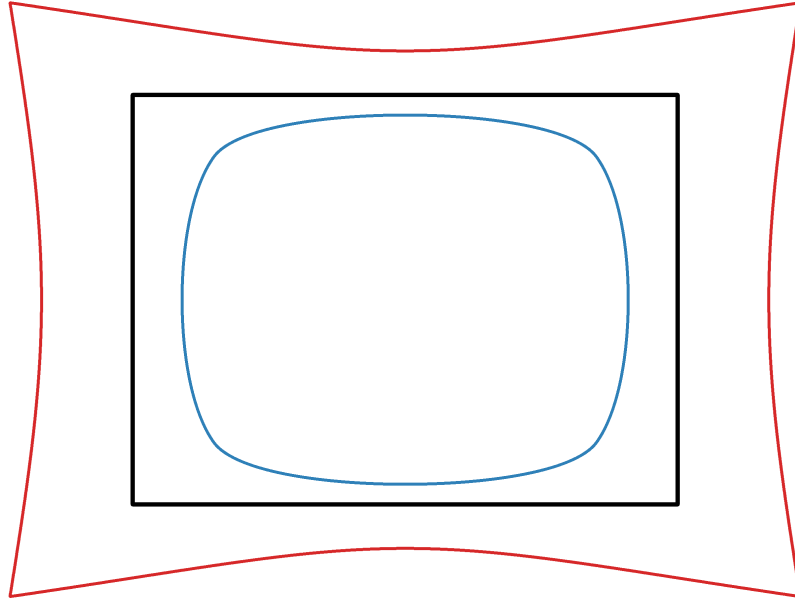


Figure 2.2: Different types of radial distortion. Black: no distortion, red: $k_1 = 10, k_2 = 11, k_3 = 12, k_4 = 5, k_5 = 6, k_6 = 7$, blue: $k_1 = -2.2, k_2 = -1.2, k_3 = -0.8, k_4 = -0.4, k_5 = -0.3, k_6 = -0.2$

Tangential distortion

Real optical systems are also subject to tangential distortion. This type of distortion has a decentering effect and occurs, if the line through the optical center of the lens and the principal point is not col-linear with the optical axis **weng**. The model

$$\begin{pmatrix} x'' \\ y'' \end{pmatrix} = \begin{pmatrix} x' + 2p_1x'y' + p_2(r^2 + 2x'^2) \\ y' + 2p_2x'y' + p_1(r^2 + 2y'^2) \end{pmatrix} \quad (2.5)$$

takes this into account. Figure 2.3 shows the distortion this model introduces.

Thin prism distortion

Thin prism distortion is partially caused by lens imperfections and camera assembly **weng**. It introduces additional radial and tangential distortion, modeled with

$$\begin{pmatrix} x'' \\ y'' \end{pmatrix} = \begin{pmatrix} x' + s_1r^2 + s_2r^4 \\ y' + s_3r^2 + s_4r^4 \end{pmatrix}. \quad (2.6)$$

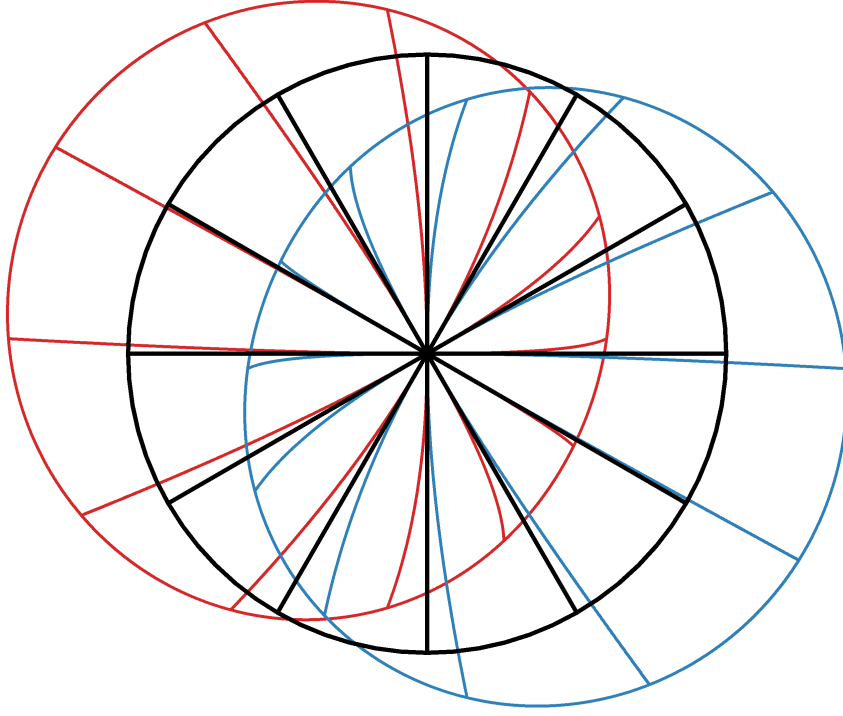


Figure 2.3: Different types of tangential distortion. Black: no distortion, red: $p_1 = -0.15$, $p_2 = -0.4$, blue: $p_1 = 0.15$, $p_2 = -0.4$

The combined model in OpenCV

The models in 2.4, 2.5 and 2.6 combined result in

$$\begin{pmatrix} x'' \\ y'' \end{pmatrix} = \begin{pmatrix} x' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} + 2p_1x'y' + p_2(r^2 + 2x'^2) + s_1r^2 + s_2r^4 \\ y' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} + 2p_2x'y' + p_1(r^2 + 2y'^2) + s_3r^2 + s_4r^4 \end{pmatrix}. \quad (2.7)$$

In summary, a point in normalized camera-coordinates $(x' \ y')^T$ is distorted as modeled in 2.7, which leads to the point $(x'' \ y'')^T$. To get the distorted pixel-coordinates (subscript d), the projection has to be applied

$$\begin{pmatrix} u_d \\ v_d \end{pmatrix} = \begin{pmatrix} x'' f_x + c_x \\ y'' f_y + c_y \end{pmatrix} \Leftrightarrow \begin{pmatrix} u_d \\ v_d \\ 1 \end{pmatrix} = K \begin{pmatrix} x'' \\ y'' \\ 1 \end{pmatrix},$$

which completes the model so far. Keep in mind, that when correcting the distortion, equation 2.7 has to be inverted.

2.1.3 Reprojection error

The reprojection error is a value to assess the quality of a calibration. It is the error between the detected image point and the projection (using the obtained coefficients) of its corresponding

world point onto the image plane. Since it only uses the points, which were previously used to obtain the calibration-parameters, it can be interpreted as a training error rate.

Calibration functions in OpenCV return the overall RMS reprojection error **cv_calib**.

2.1.4 Error propagation as further quality assessment

As argued before, the reprojection error is a training error rate. A low reprojection error is therefore not necessarily enough to estimate, how good the calibration really performs.

OpenCV returns with its extended calibration functions not only the camera intrinsics, but also an estimated standard deviation of each coefficient **cv_calib**. Under assumption that low standard deviations suggest a good calibration, it is valid to combine all standard deviations into one value using the propagation of uncertainty.

The propagation of error states that influence of each error of the inputs x_i on the output value y of a function

$$y = f(x_1, x_2, \dots, x_n)$$

can be linearly approximated and summed up to:

$$\Delta y = \sqrt{\sum_{i=1}^n \left(\frac{\partial y}{\partial x_i} \Delta x_i \right)^2}. \quad (2.8)$$

Applied to 2.7, this leads to

$$\Delta x'' = \sqrt{\sum_{i=1}^6 \left(\frac{\partial x''}{\partial k_i} \Delta k_i \right)^2 + \sum_{i=1}^2 \left(\frac{\partial x''}{\partial p_i} \Delta p_i \right)^2 + \sum_{i=1}^4 \left(\frac{\partial x''}{\partial s_i} \Delta s_i \right)^2}$$

and

$$\Delta y'' = \sqrt{\sum_{i=1}^6 \left(\frac{\partial y''}{\partial k_i} \Delta k_i \right)^2 + \sum_{i=1}^2 \left(\frac{\partial y''}{\partial p_i} \Delta p_i \right)^2 + \sum_{i=1}^4 \left(\frac{\partial y''}{\partial s_i} \Delta s_i \right)^2}.$$

It is at this point not intended to go into further computations of the derivatives. If we look at pixel coordinates from the center of the image, btw. $c_x = 0$ and $c_y = 0$, the radius (distance from the centerpoint) can be expressed as

$$r = \sqrt{(f_x \cdot x'')^2 + (f_y \cdot y'')^2}$$

and therefore, if errors in f_x and f_y are neglected, with the propagation in 2.8 applied

$$\Delta r = \sqrt{\left(\frac{\partial r}{\partial x''} \Delta x'' \right)^2 + \left(\frac{\partial r}{\partial y''} \Delta y'' \right)^2}. \quad (2.9)$$

This value Δr can now be plotted over half the diagonal of an image to compare calibration coefficients of different calibration approaches.

2.2 Measurement with back-light illumination

To measure the size and form of an object there are multiple ways. A popular way to get the contour is to place a light source behind the object desired to measure, and aim on the object with an camera. With this method the optical sensors get the silhouette of the object.

2.2.1 Light source

2.2.2 Light ray distribution

2.2.3 Light properties

2.2.4 Measurement precision changes

2.2.5 Image sensor properties

Chapter 3

Development

This chapter covers the developing process in more detail.

3.1 Calibration

3.2 Software

3.3 Hardware

3.3.1 Demonstrator

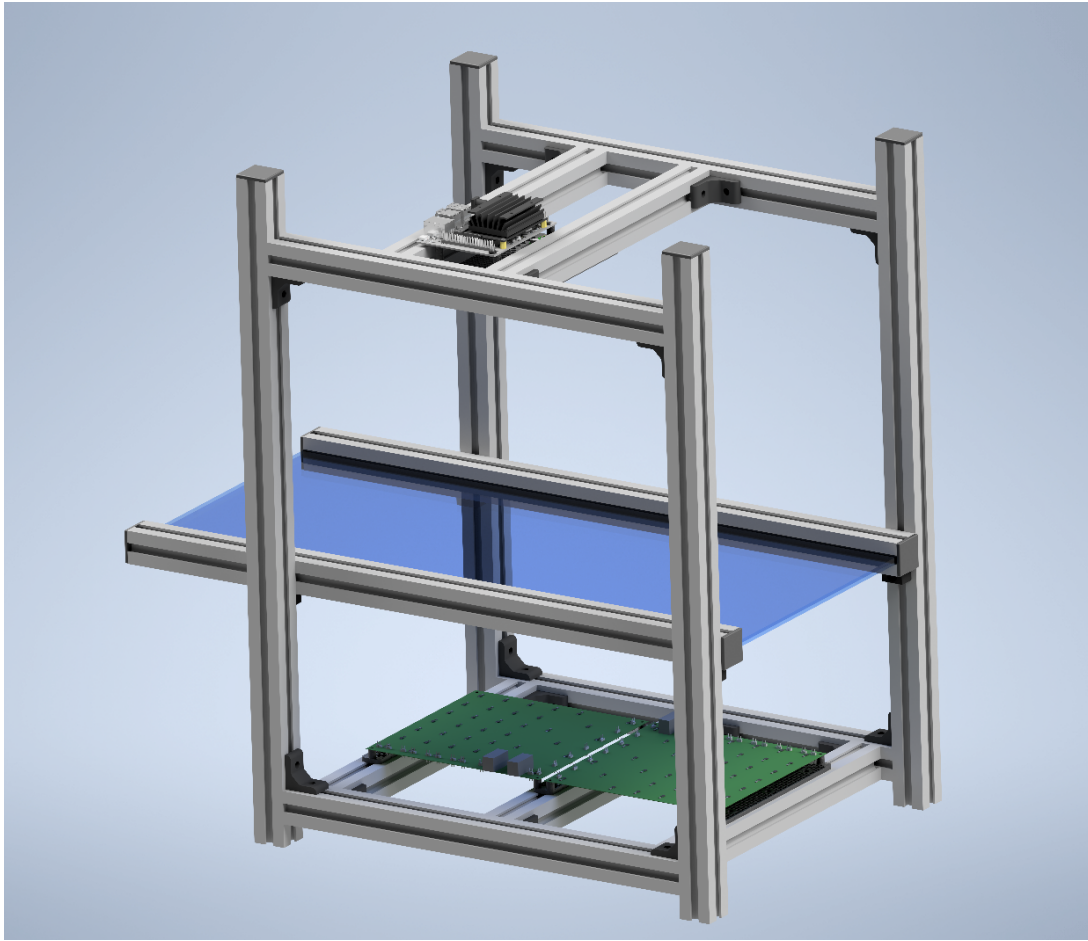


Figure 3.1: 3D-Model of the demonstrator

3.3.2 Back-light

3.3.3 Diffusion

3.3.4 Computer

3.3.5 Camera

The camera used for this project had to following evaluation criteria to meet:

- compatible with the Jetson Nano enviroment
- a good developer community
- open source drivers
- low cost
- minimum frame rate of 20[fps]
- measurement accuracy of $\frac{1}{10}$ [mm]

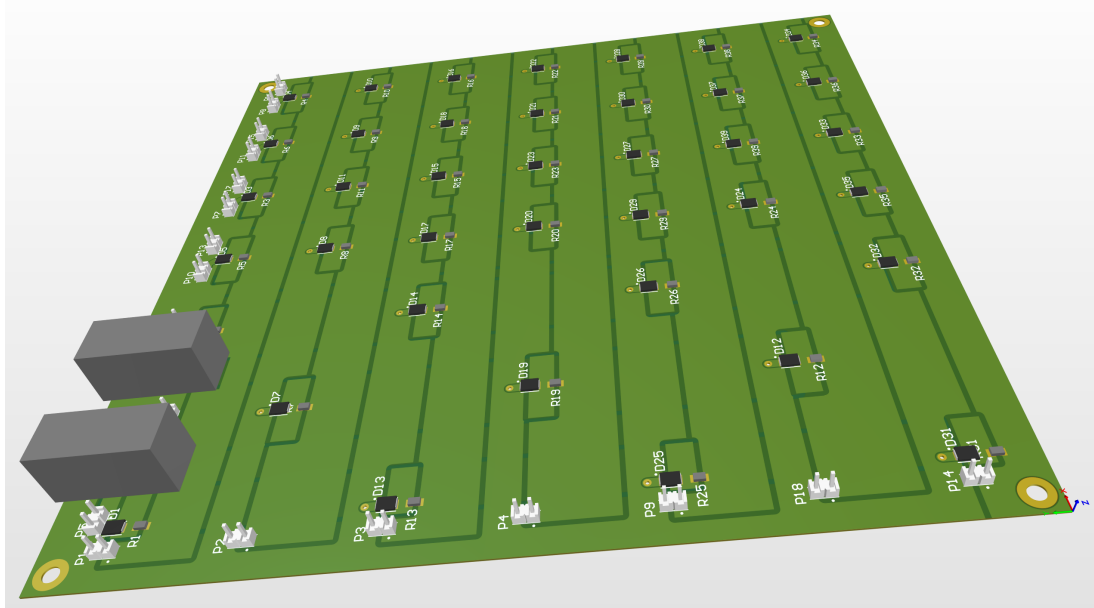


Figure 3.2: 3D-Model of the back-light PCB

The following is a list of the cameras that have been studied and used in the project:

List of possible Cameras			
Company Camera Name	Raspberry Pi Camera Module V2	Raspberry Pi HQ Camera	Arducam Arducam IMX219 Low Distortion M12 Mount Camera Module
Price	\$25	\$50	\$39.99
Camera Sensor	Sony IMX219	Sony IMX477	Sony IMX219
Still resolution	8M	12M	12.3M
Sensor resolution	3280 x 2464	4056 x 3040	3280 x 2464
Linux integration	V4L2 driver	V4L2 driver	V4L2 driver
Pixel size	1.12 x 1.12 μm	1.55 x 1.55 μm	1.12 x 1.12 μm
Focal length	3.04[mm]	Depends on lens	2.8[mm]
Focal ratio	2.0	Depends on lens	2.8
View angle Horizontal	62.2 deg.	Depends on lens	75 deg.

3.3.6 Production cost

Chapter 4

Results

This chapter covers the most important results.

Chapter 5

Conclusion

References

List of Figures

2.1	Pinhole-model (not up to date).	5
2.2	Different types of radial distortion. Black: no distortion, red: $k_1 = 10$, $k_2 = 11$, $k_3 = 12$, $k_4 = 5$, $k_5 = 6$, $k_6 = 7$, blue: $k_1 = -2.2$, $k_2 = -1.2$, $k_3 = -0.8$, $k_4 = -0.4$, $k_5 = -0.3$, $k_6 = -0.2$	7
2.3	Different types of tangential distortion. Black: no distortion, red: $p_1 = -0.15$, $p_2 = -0.4$, blue: $p_1 = 0.15$, $p_2 = -0.4$	8
3.1	3D-Model of the demonstrator	12
3.2	3D-Model of the back-light PCB	13

List of Tables

Statement of Plagiarism

We declare that, apart from properly referenced quotations, this report is our own work and contains no plagiarism; it has not been submitted previously for any other assessed unit on this or other degree courses.

Place	Date
Rapperswil	May 14, 2020

Signatures

Cedric Renda

Manuel Tischhauser

Appendix A

Requirements

A.1 Assignment

A.2 Requirement Specification
