

---

# 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

April 21, 2020



# **Abstract**

## **Introduction**

## **Approach**

## **Conclusion**

---

# Contents

<b>Abbreviations</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
<b>2 Theory</b>	<b>5</b>
2.1 Camera Calibration in OpenCV . . . . .	5
2.1.1 The pinhole camera model . . . . .	5
2.1.2 The distortion model in OpenCV . . . . .	6
<b>3 Evaluation</b>	<b>9</b>
<b>4 Development</b>	<b>11</b>
<b>5 Results</b>	<b>13</b>
<b>6 Conclusion</b>	<b>15</b>
<b>References</b>	<b>17</b>
<b>List of figures</b>	<b>17</b>
<b>List of tables</b>	<b>19</b>
<b>Statement of Plagiarisms</b>	<b>21</b>
<b>A Requirements</b>	<b>25</b>
A.1 Assignment . . . . .	25
A.2 Requirement Specification . . . . .	26

---

# Abbreviations

**ASK** Amplitude Shift Keying





# **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

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 [1]. 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-coordinates 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 then 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:

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

The rotation and translation in  $(R \mid t)$  are called the extrinsic parameters. The camera matrix  $K$  contains analogously the linear intrinsic parameters.

## 2.1.2 The distortion model in OpenCV

### Radial Distortion

Radial distortion is caused by the flawed curvature of the lens [2]. It can be modeled with

$$\begin{pmatrix} x'' \\ y'' \end{pmatrix} = \begin{pmatrix} x' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} \\ y' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} \end{pmatrix}, \quad (2.4)$$

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.

### Tangential distortion

### Thin prism distortion

### The combined model in OpenCV

Non linear distortions, which appear before the projection in 2.2 should be considered too. OpenCV implements the following model:

$$\begin{pmatrix} x'' \\ y'' \end{pmatrix} = \begin{pmatrix} x' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} + 2p_1x'y' + p_2(r^2 + 2x'^2) + s_1r^2 + s_2r^4 \\ y' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} + 2p_2x'y' + p_1(r^2 + 2y'^2) + s_3r^2 + s_4r^4 \end{pmatrix} \quad (2.5)$$

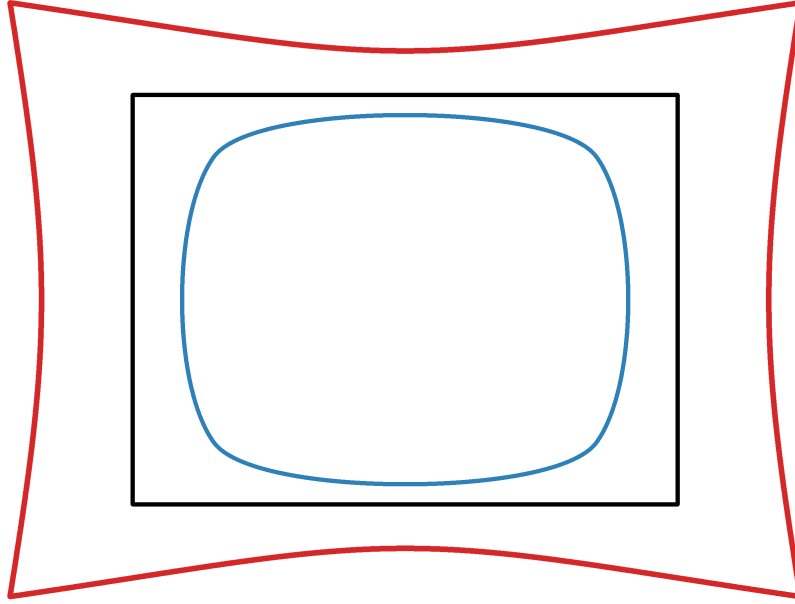


Figure 2.2: Different types of radial distortion. Black: no distortion, red:  $k_1 = 10, k_2 = 11, k_3 = 12, k_4 = 5, k_5 = 6, k_6 = 7$ , blue:  $k_1 = -2.2, k_2 = -1.2, k_3 = -0.8, k_4 = -0.4, k_5 = -0.3, k_6 = -0.2$

[1]

In summary, a point in normalized camera-coordinates  $(x' \ y')^T$  is distorted as modeled in 2.5, 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 2.5 has to be inverted.

---

# **Chapter 3**

## **Evaluation**

---



# **Chapter 4**

## **Development**

This chapter covers the developing process in more detail.

---

# **Chapter 5**

## **Results**

This chapter covers the most important results.

---

## **Chapter 6**

## **Conclusion**

---

# References

- [1] (). Camera Calibration and 3D Reconstruction, [Online]. Available: [https://docs.opencv.org/4.3.0/d9/d0c/group\\_\\_calib3d.html](https://docs.opencv.org/4.3.0/d9/d0c/group__calib3d.html).
- [2] J. Weng, P. Cohen, and M. Herniou, “Camera calibration with distortion models and accuracy evaluation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 10, pp. 965–980, 1992.

---



---

# 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

---

---

## 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.

<b>Place</b>	<b>Date</b>
Rapperswil	April 21, 2020

## Signatures

Cedric Renda

Manuel Tischhauser

---

# **Appendix A**

## **Requirements**

### **A.1 Assignment**

---

## **A.2 Requirement Specification**





---