Technische Universität München
Lehrstuhl für Umformtechnik und Gießereiwesen
Univ.-Prof. Dr.-Ing. Wolfram Volk

# Konzeption, Implementierung und Evaluierung eines Softwaretools zur teilautomatisierten optischen Bauteilvermessung

konstruktive Bachelorarbeit

| | |
|---|---|
| Bearbeitet von: | Fabian Küster |
| Matrikelnummer: | 03674367 |
| Betreuer: | Christoph Hartmann |
| Ausgabedatum: | 15.05.2020 |
| Abgabedatum: | 13.11.2020 |

# Erklärung

Die Bachelor's Thesis

*Konzeption, Implementierung und Evaluierung eines Softwaretools zur teilautomatisierten optischen Bauteilvermessung*

entstand während meiner Tätigkeit als Bachelorand am Lehrstuhl für Umformtechnik und Gießereiwesen an der Technischen Universität München. Die Ergebnisse wurden zusammen mit Herrn Hartmann erarbeitet und ausgewertet. Die schriftliche Ausarbeitung erfolgte von mir selbstständig. Es wurden dabei keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Die Richtlinien der Technischen Universität München zur Sicherung guter wissenschaftlicher Praxis und für den Umgang mit wissenschaftlichem Fehlverhalten habe ich erhalten und bei der Erstellung der vorliegenden Arbeit eingehalten.

Ich gestatte dem Lehrstuhl für Umformtechnik und Gießereiwesen diese Studienarbeit bzw. Teile davon nach eigenem Ermessen an Dritte weiterzugeben, zu veröffentlichen oder anderweitig zu nutzen. Mein persönliches Urheberrecht ist über diese Regelung hinaus nicht beeinträchtigt.

Eventuelle Geheimhaltungsvereinbarungen über den Inhalt der Arbeit zwischen mir bzw. dem Lehrstuhl für Umformtechnik und Gießereiwesen und Dritten bleiben von dieser Vereinbarung unberührt.

Unterhaching, 13.11.2020

Fabian Küster

# Inhaltsverzeichnis

# Verwendete Konventionen

| **Formelzeichen** | **Bedeutung** |
|---|---|
| $R = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$ | Matrix |
| $\underline{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$ | Vektor |
| $\overline{x}$ | Arithmetisches Mittel |
| Var | Varianz |
| E | Erwartungswert |
| RMS | Reprojektionsfehler |
| `CodeInText` | Python Befehle und Bibliotheken |

# 1 Problemstellung

Der Wunsch eines jeden Produktionsunternehmens ist das ständige Senken der Produktionszeit sowie der Produktionskosten bei gleichzeitiger Steigerung der Qualität. Es ist leicht ersichtlich, dass diese Ziele einen Konflikt darstellen. So kann etwa eine Reduzierung der Durchlaufzeit eine Verringerung der Qualität zur Folge haben. Einerseits natürlich, da bei steigendem Zeitdruck die Fehlerrate steigen kann und gleichzeitig weniger Zeit für Qualitätskontrollen zur Verfügung steht. Letzterem kann durch die industrielle Bildverarbeitung entgegengewirkt werden. Der Trend hin zur automatisierten Qualitätskontrolle wird deutlich durch die in Abbildung 1.1 dargestellte Umsatzentwicklung der Bildverarbeitungsbranche. Allein in Deutschland hat sich der Umsatz dieser Branche seit 2007 mehr als verdoppelt.

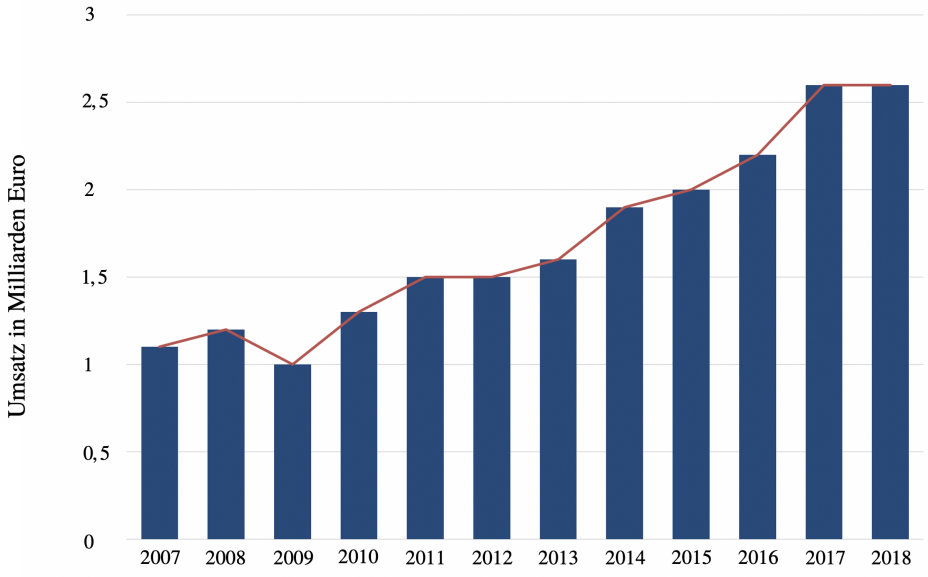



*Abbildung 1.1: Umsatzentwicklung der Industriellen Bildverarbeitung in Deutschland bis 2018; in Anlehnung an (*Machine Vision Germany: record level maintained *2018)*

Häufige Anwendungen der Bildverarbeitung in der Industrie sind beispielsweise Vollständigkeitskontrollen, Lageerkennungen, Maßkontrollen, Oberflächeninspektionen oder die Objekterkennung von etwa Barcodes. Am einfachsten umzusetzen beziehungsweise generell zu implementieren ist dies sicherlich in einer Fließbandproduktion, da die Qualitätskontrolle bei laufendem Band vorgenommen werden kann und dadurch die Umgebungsbedingungen größtenteils konstant gehalten werden können. Meist sind die Systeme lediglich für einen einzigen Anwendungsfall ausgelegt und zudem sehr empfindlich auf Änderungen der Umgebung. So können

etwa kleine Änderungen der Lichtverhältnisse oder der Orientierung des zu inspizierenden Bauteils das System bereits an seine Grenzen bringen. (Christian, 2011; Steger et al., 2018)

Bei vielen dieser Anwendungen reicht eine zweidimensionale Überprüfung bereits aus, um etwa die Silhouette oder die Oberfläche eines Bauteils zu vermessen beziehungsweise zu inspizieren. Jedoch existieren auch diverse Anwendungen bei denen eine dreidimensionale Überprüfung oder auch Röntgentechnik und Wärmefluss-Thermographie notwendig sind. (Steger et al., 2018; Norbert, 2008)

# 2 Stand der Technik

## 2.1 Optische Vermessung in der Industrie

Ein modernes Bildbearbeitungssystem, wie es häufig in der Industrie zum Einsatz kommt, besteht aus mehreren elementaren Komponenten. Zum einen natürlich aus der Kamera inklusive dem passenden Objektiv, aber auch die richtige Beleuchtung, ein Framegrabber sowie eine Bildbearbeitungssoftware sind wichtige Bestandteile. Zusätzlich denkbar sind Trigger zur automatischen Auslösung der Kamera sowie integrierte speicherprogrammierbare Steuereinheiten (SPS) zur Bedienung von nachgeschalteten Anlagen, wie beispielsweise ein Sortierer. (Steger et al., 2018)

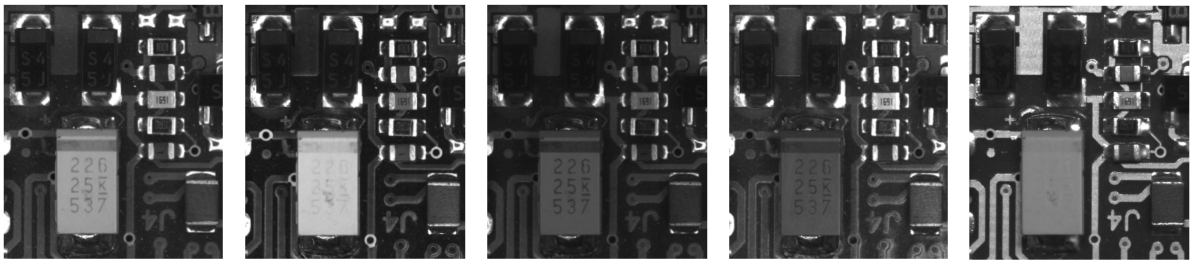
Im späteren Teil dieser Arbeit werden verschiedenen Methoden der Bildverarbeitung benötigt und daher dessen Funktionsweisen in diesem Kapitel erläutert. Da aber auch die Beleuchtung sowie die verwendete Kamera, im speziellen deren Kalibrierung, einen großen Einfluss auf die Funktionsweise sowie die Auswahl der zur Verfügung stehenden Bildverarbeitungsmethoden spielt, werden auch diese im Folgenden kurz erläutert.

## 2.2 Beleuchtung

Es ist naheliegend, dass die Genauigkeit einer optischen Vermessung größtenteils von der Qualität des jeweiligen Bildes abhängt. Mit den größten Einfluss auf diese, hat neben der verwendeten Kamera, die bei der Aufnahme vorherrschende Beleuchtung. So können beispielsweise Reflektionen oder entstehende Schatten eine Vermessung stark erschweren oder gar verhindern. Um möglichst optimale Voraussetzungen für eine Vermessung zu schaffen, muss daher je nach Anwendungsfall die optimale Kombination aus Leuchtmittel, Richtung des Lichtes sowie der Position der Lichtquelle ausgewählt werden.

### 2.2.1 Leuchtmittel

Die in der Industrie am häufigsten verwendeten Leuchtmittel zur Bewältigung von Aufgaben der Bildverarbeitung sind Hochfrequenz-Leuchtstoffröhren, Halogenlampen und Leuchtdioden. Grund hierfür ist die gute Steuerbarkeit und Helligkeitskonstanz des Lichtes. Zusätzlich hat man die Möglichkeit durch die gezielte Wahl der Farbe des Lichtes bestimmte Objekte sichtbar zu machen, wie in Abbildung 2.1 demonstriert. (Steger et al., 2018)



*(a)* *(b)* *(c)* *(d)* *(e)*

*Abbildung 2.1: PCB Leiterplatte beleuchtet mit (a) weißem, (b) rotem, (c) grünem, (d) blauen und (e) infrarotem Licht (Steger et al., 2018)*

Durch die richtige Wahl der Farbe des Lichtes kann man, bereits vor Anwendung der eigentlichen Bildbearbeitungsmethoden, die Objekte von Interesse deutlich von dem Hintergrund oder von benachbarten Objekten abgrenzen.

### 2.2.2 Richtung des Lichtes

Ein weiterer entscheidender Faktor bei der Wahl der passenden Beleuchtung ist die Richtung des einfallenden Lichtes. Hier wird zwischen diffusem und gerichtetem Licht unterschieden. Bei der diffusen Beleuchtung strahlt das Licht in alle Richtungen mehr oder weniger gleich aus. Bei der gerichteten Beleuchtung hingegen strahlt das Licht nur in eine bestimmte Richtung. (Steger et al., 2018) Ein Spezialfall der gerichteten Beleuchtung, welche für bestimmte Anwendungen von Vorteil sein kann, ist die telezentrische Beleuchtung Bei dieser Art der Beleuchtung verlaufen alle Lichtstrahlen parallel zueinander. Dies ist vor allem hilfreich wenn man an der Silhouette des Objektes interessiert ist, da durch die parallel verlaufenden Lichtstrahlen sehr scharfe Kanten entstehen (Norbert, 2008).

### 2.2.3 Position der Lichtquelle

Eine weitere Möglichkeit die Beleuchtung bestmöglich für den jeweiligen Anwendungsfall einzustellen, ist die Wahl der Position der Lichtquelle. Grundsätzlich wird hier zwischen der Auflicht-, und der Durchlichtbeleuchtung unterschieden. Bei der Auflichtbeleuchtung befindet sich die Lichtquelle auf derselben Seite des aufzunehmenden Objekts wie die Kamera. Bei der Durchlichtbeleuchtung hingegen befinden sich Lichtquelle und Kamera auf unterschiedlichen Seiten des Objektes. Zuletzt kann noch entschieden werden, ob der Großteil des Lichtes hin zum Objekt oder weg vom Objekt strahlen soll.

## 2.3 Methoden der Bildbearbeitung

### 2.3.1 Bildglättung

Üblicherweise beschäftigt sich der erste Schritt bei jeder Bildbearbeitung mit dem Beseitigen von Rauschen im Bild, der sogenannten Bildglättung. Mit Rauschen bezeichnet man die zufälligen Änderungen der im Bild enthaltenen Grauwerte. Die Ursachen für dieses Phänomen sind unter anderem die Zufälligkeit des Photoneneinflusses (poisson noise), Fehler bei der Datenübertragung (salt and pepper noise), oder elektrische Störimpulse (periodic noise) (Thakur et al., 2019). Um diese Fehler in den Grauwerten zu unterdrücken, wendet man sogenannte Glättungsfilter an. Anhand dieser Filter werden alle Pixel mit ihren Nachbarpixeln verglichen und je nach Filtertyp der zu erwartende Grauwert geschätzt. Wünschenswert wäre ein Glättungsfilter, welcher das Rauschen im Bild komplett beseitigt und zugleich die Kantenschärfe erhält. Leider muss in der Realität hierbei immer ein Kompromiss eingegangen werden, denn bei steigender Größe der Filtermaske wird zwar das Rauschen stärker unterdrückt, jedoch verschwimmen dadurch auch die im Bild enthaltenen Kanten. Die Glättungsfilter welche das Bildrauschen derzeit am stärksten unterdrücken und gleichzeitig die Kantenschärfe nahezu unbeeinflusst lassen, sind Glättungsfilter welche auf Neuronale Netze zurückgreifen, wie etwa der PDNN Algorithmus (Thakur et al., 2019). Häufig werden diese Filter genutzt um sehr stark verrauschte Bilder, aufgrund von starker Vergrößerung oder schnellen Bewegungen der aufgenommenen Objekte zu glätten.

Die in der Bildbearbeitung am häufigsten verwendeten Glättungsfilter sind der Gaußfilter, der Mittelwertfilter sowie der Medianfilter. (Steger et al., 2018) Die unterschiedlichen Filterergebnisse sind in Abbildung 2.3 anhand eines Beispiels mit künstlich hinzugefügtem Rauschen dargestellt. Hinsichtlich der Kantenschärfe liefert der Medianfilter die besten Ergebnisse, jedoch

auf Kosten von Informationsverlust, wie beispielsweise an der dünnen Linie um die vier Sterne in den Ecken der Spielkarte in Abbildung 2.3 deutlich zu sehen ist. Bei einer minimalen Vergrößerung der Filtermaske werden diese komplett verschwinden.


*(a)* Original *(b)* salt & pepper noise

*Abbildung 2.2: Originalbild und künstlich hinzugefügtes Rauschen*


*(a)* Mittelwertfilter *(b)* Gaußfilter *(c)* Medianfilter

*Abbildung 2.3: Gegenüberstellung der Glättungsfilter bei einer 35px Filtermaske*

### 2.3.2 Kantenextraktion

Für die meisten Anwendungen in der digitalen Bildbearbeitung ist es notwendig, die im Bild enthaltenen Grauwertübergänge, sprich die Kanten, zu extrahieren, da diese meist die gewünschten Informationen enthalten. Um dies mittels sogenannter Kantenfilter zu ermöglichen, müssen diese Grauwertübergänge mathematisch formuliert werden. Hierbei wird zwischen einer eindimensionalen und einer zweidimensionalen Kante unterschieden. Eine eindimensionale Kante kann als das Maximum des Betrages der 1. Ableitung angesehen werden. Da, aufgrund von Rauschen einzelne Grauwerte fälschlicherweise zu maximalen oder minimalen Werten abgeändert werden, würden all diese Grauwerte ein lokales Maximum der 1. Ableitung darstellen. Um dies zu verhindern, ist eine Bildglättung vorab meist unerlässlich. Für die Berechnung einer

zweidimensionalen Kante wählt man eine zur eindimensionalen analogen Definition. Der Betrag der Richtungsableitung in Richtung senkrecht zur Kante muss maximal sein. Allerdings wird für diese Definition die Kante bereits als bekannt vorausgesetzt. Um dies zu umgehen wird der Gradient $\nabla f$ des Bildes, welcher in die Richtung des maximalen Anstiegs im Bildkoordinatensystem $(r, c)$ zeigt, gebildet.

$$\nabla f = \nabla(f_r, f_c) = \left(\frac{\partial f(r,c)}{\partial r}, \frac{\partial f(r,c)}{\partial c}\right) = (f_r, f_c) \qquad \text{(Gleichung 2.1)}$$

Die Euklidische Norm des Gradientenvektors berechnet sich demnach aus:

$$\|\nabla f\|_2 = \sqrt{f_r^2 + f_c^2} \qquad \text{(Gleichung 2.2)}$$

Das Maximum des Gradientenbetrags in Richtung des Gradienten entspricht der Nullstelle der zweiten Richtungsableitung in Richtung des Gradienten.

$$\frac{\partial^2 f}{\partial n^2} = \frac{f_r^2 f_{rr} + 2 f_r f_c f_{rc} + f_c^2 f_{cc}}{f_r^2 + f_c^2} \qquad \text{(Gleichung 2.3)}$$

Eine alternative Definition für eine zweidimensionalen Kante ist die Nullstelle des Laplace-Operators.

$$\Delta f(r,c) = \frac{\partial^2 f(r,c)}{\partial r^2} + \frac{\partial^2 f(r,c)}{\partial c^2} = f_{rr} + f_{cc} \qquad \text{(Gleichung 2.4)}$$

Die unterschiedlichen Ergebnisse der zwei Kantendefinitionen sind in Abbildung 2.4 illustriert. Es wird deutlich, dass bei Verwendung des Laplace-Operators die tatsächliche Position der Kante berechnet wird. Die Verwendung der unterschiedlichen Kantendefinitionen, sollte demnach an den Anwendungsfall angepasst werden. (Steger et al., 2018)

Eine ideale Kante mit der Amplitude $a$ an der Position $x = 0$ kann folgendermaßen beschrieben werden:

$$f(x) = \begin{cases} a & x \geq 0, \\ 0 & x < 0 \end{cases} \qquad \text{(Gleichung 2.5)}$$

Durch Berücksichtigen des Rauschanteils $n(x)$ ergibt sich die verrauschte Kante

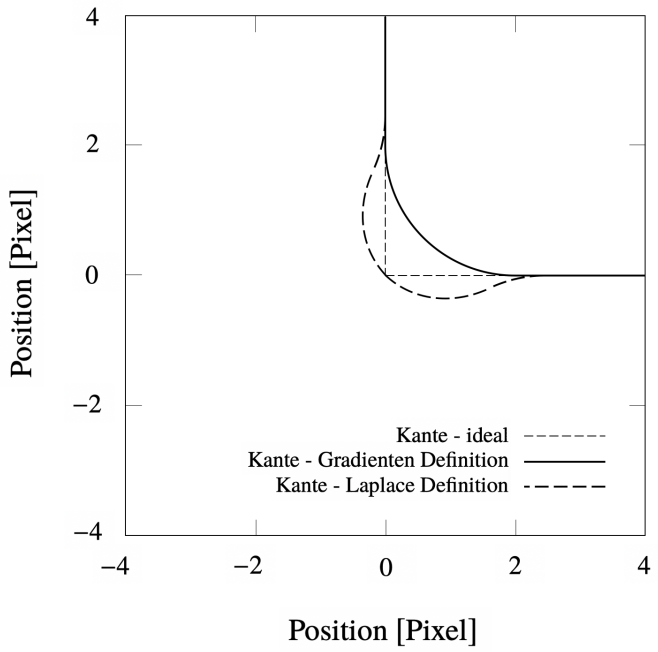



*Abbildung 2.4: Vergleich der Kantendefinitionen; in Anlehnung an (Steger et al., 2018)*

$$\hat{f}(x) = f(x) + n(x) \qquad \text{(Gleichung 2.6)}$$

Die Antwort $r$ des Kantenfilters $h'$ auf die verrauschte Kante, kann durch eine Faltung des Kantenfilters mit der Kante beschrieben werden. (Steger et al., 2018)

$$r = \hat{f} * h' \qquad \text{(Gleichung 2.7)}$$

Bei einer Varianz des Rauschens von $\sigma_n^2$ und einem Mittelwert von 0, ergibt sich nach Anwendung der linearen Filterung die abgeänderte Varianz

$$\tilde{\sigma}_n^2 = \sigma_n^2 \int_{-\infty}^{\infty} h'(x)^2 \, dx \qquad \text{(Gleichung 2.8)}$$

John F. Canny formulierte 1986 drei Gütekriterien, welche ein optimaler Kantenfilter möglichst erfüllen sollte. (Canny, 1986)

**Detektionsgüte** Echte Kanten sollen mit hoher Wahrscheinlichkeit als echte Kanten erkannt werden und falsche Kanten aufgrund von Rauschen sollen mit hoher Wahrscheinlichkeit

als falsche Kanten erkannt werden. Folglich soll das sogenannte Signal-Rauschverhältnis (signal to noise ratio, SNR) möglichst groß sein.

$$\mathrm{SNR}(h') := \frac{\left| \int_{-\infty}^{\infty} f(x)h'(-x)\,dx \right|}{\sigma_n \sqrt{\int_{-\infty}^{\infty} h'(x)^2 dx}} \qquad \text{(Gleichung 2.9)}$$

**Lokalisationsgüte** Die extrahierte Kante soll möglichst genau an der echten Kante liegen, oder in anderen Worten, die Varianz der extrahierten Kantenpositionen soll minimal sein. Die Kantenposition ist gegeben durch das Maximum der Filterantwort $r'(x_e) = 0$ und die Varianz durch $\mathrm{Var}(x_e) = \mathrm{E}(x_e^2)$ . Mittels einer Taylor-Entwicklung um $x = 0$ und der Vernachlässigung der Antwort der ersten Ableitung des Kantenfilters auf das Rauschen erhält man für die Kantenposition

$$x_e \approx -\frac{r_n'(0)}{r_f''(0)}. \qquad \text{(Gleichung 2.10)}$$

Der Zähler mit $r_n'(0)$ spiegelt die Antwort der ersten Ableitung des Kantenfilters auf das Rauschen und der Nenner mit $r_f''(0)$ die erste Ableitung des Kantenantwort wieder. Mit Hilfe der Gleichung 2.8 berechnet sich die Varianz der Kantenposition durch:

$$\mathrm{E}(x_e^2) = \frac{\sigma_n^2 \int_{-\infty}^{\infty} h''(x)^2\,dx}{\left( \int_{-\infty}^{\infty} f(x)h'''(-x)\,dx \right)^2} \qquad \text{(Gleichung 2.11)}$$

Die Lokalisationsgüte (LOC) ergibt sich aus dem Kehrwert der Standardabweichung der Kantenposition

$$\mathrm{LOC}(h') = \sqrt{\frac{1}{E(x_e^2)}} \qquad \text{(Gleichung 2.12)}$$

**Keine Mehrfachantworten** Jede wahre Kante soll genau einmal extrahiert werden. Um dies zu erreichen, gilt es die Maxima der Filterantwort auf Rauschen zu reduzieren. Hierfür

wird das zu maximierende Maß ZC als der mittlere Abstand zwischen zwei Nullstellen eingeführt.

$$\mathrm{ZC}(h') = \pi \sqrt{\frac{\int_{-\infty}^{\infty} h''(x)^2\,dx}{\int_{-\infty}^{\infty} h'''(x)^2\,dx}} \qquad \text{(Gleichung 2.13)}$$

Die Kombination dieser drei Kriterien und damit die Berechnung des optimalen Kantenfilters, führt zu einem komplexen Optimierungsproblem. Aufgrund dieser Komplexität veröffentlichte Canny mit seinem Kantenfilter eine gute Näherung der analytischen Lösung. Demnach ist der optimale Kantenfilter in guter Näherung, die erste Ableitung des Gauß-Glättungsfilters $g'_\sigma$.

$$h'(x) = g'_\sigma = \frac{-x}{\sqrt{2\pi\sigma^3}} e^{-\frac{x^2}{2\sigma^2}} \qquad \text{(Gleichung 2.14)}$$

Um die Filterantwort auf die Amplitude $a$, der in Gleichung 2.5 beschriebene Kante, zu normieren, wird der Filter noch mit $\sqrt{2\pi}\sigma$ multipliziert.

$$h'(x) = g'_\sigma = \frac{-x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}} \qquad \text{(Gleichung 2.15)}$$

Mit $d'_\alpha$ und $e'_\alpha$ veröffentlichte R. Deriche zwei weitere Approximationen des optimalen Glättungsfilters auf Basis der gleichen Kriterien. Diese Kantenfilter haben den Vorteil, dass sie im Gegensatz zu Canny's Kantenfilter rekursiv implementierbar sind (Deriche, 1987).

$$h'(x) = d'_\alpha = kxe^{-\alpha|x|} \qquad \text{(Gleichung 2.16)}$$

$$h'(x) = e'_\alpha = ke^{-\alpha|x|} * \sin(\omega x) \qquad \text{(Gleichung 2.17)}$$

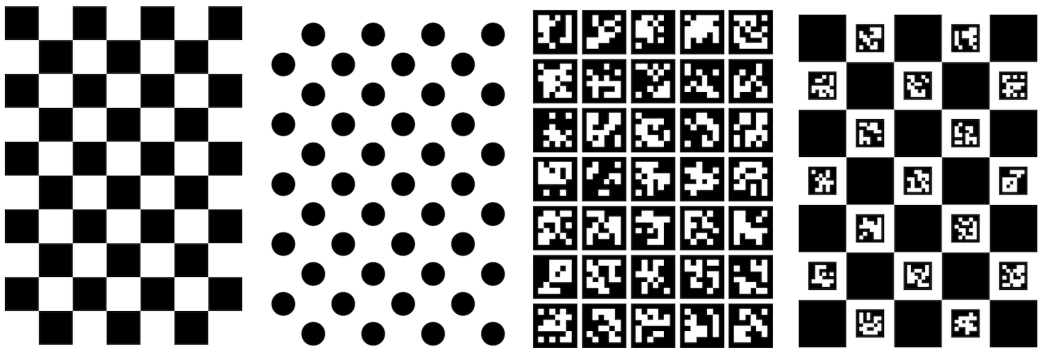
## 2.4 Grundlagen der Kamerakalibrierung

### 2.4.1 Motivation

Um eventuelle Verzeichnungen, verursacht durch die Kamera, korrigieren zu können, muss vor der optischen Vermessung die verwendete Kamera kalibriert werden. Weiterhin ist die Kalibrierung der Kamera unabdinglich für die Umrechnung vom Pixel-Koordinatensystem in das Welt-Koordinatensystem. Für letzteres ist eine Stereo Rekonstruktion notwendig.

### 2.4.2 Kalibrierkörper

Für eine erfolgreiche und präzise Ermittlung aller unbekannter Kameraparameter und damit eine erfolgreiche Kalibrierung der Kamera, wird ein sinnvoller Kalibrierkörper benötigt. Theorethisch könnte hierfür jeglicher Körper bekannter Abmessungen verwendet werden. Um die Kalibrierung jedoch so effizient und genau wie möglich zu gestalten, haben sich die in Abbildung 2.5 dargestellten Kalibrierkörper als besonders nützlich erwiesen. (Jakob, 2018b)



*(a)* Schachbrett *(b)* Kreise *(c)* Aruco *(d)* Charuco

*Abbildung 2.5: Am häufigsten verwendete Kalibrierkörper (*Pattern Generator *2020)*

Das auf dem Kalibrierkörper enthaltene Kalibrierungsmuster dient als Referenzobjekt um möglichst einfach Punkte in 3D Weltkoordinaten ermitteln zu können, damit diese dann im nächsten Schritt mit den Bildpunkten verglichen werden können. Um möglichst viele unterschiedliche Punkte mit einer hohen Genauigkeit zu erhalten, werden leicht zu extrahierende Formen, in hoher Dichte verwendet. (Steger et al., 2018)

### 2.4.3 Vorgehen bei der Kalibrierung

Die Abbildung eines Punktes aus dem Weltkoordinatensystem in das Bildkoordinatensystem ist abhängig von der inneren sowie der äußeren Orientierung der Kamera.

In Abbildung 2.6 ist der Zusammenhang der unterschiedlichen Koordinatensysteme anhand des Prinzips der Lochkamera dargestellt. Zur Vereinfachung der im Folgenden verwendeten Gleichungen, wurde das Bild- sowie das Bildebenenkoordinatensystem um die Bildweite $f$ vor den Ursprung der Kamera platziert, obwohl diese in Realität dahinter liegen.

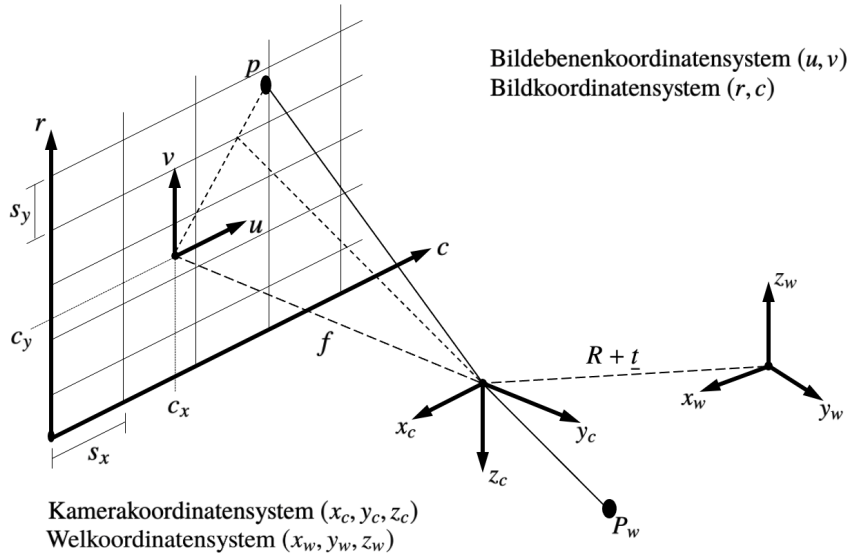



*Abbildung 2.6: Model der Lochkamera; in Anlehnung an (Steger et al., 2018)*

Die Abbildung eines Punktes $P_w$ aus dem Weltkoordinatensystem in das Kamerakoordinatensystem lässt sich durch eine Rotation mit der Rotationsmatrix $R$ und eine Translation mit dem Translationsvektor $\underline{t}$ erreichen.

$$P_c = [x_c, y_c, z_c]^\intercal = RP_w + \underline{t} \qquad \text{(Gleichung 2.18)}$$

$$\underline{t} = [t_x, t_y, t_z]^\intercal \qquad \text{(Gleichung 2.19)}$$

$$R = R(\alpha)R(\beta)R(\gamma) \qquad \text{(Gleichung 2.20)}$$

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & -\sin\alpha \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{(Gleichung 2.21)}$$

Die in Gleichung 2.18 dargestellte Abbildung spiegelt die äußere Orientierung der Kamera wieder und die drei Rotationswinkel $\alpha, \beta, \gamma$, sowie die drei Elemente $t_x, t_y, t_z$ des Translationsvektors $\underline{t}$ sind die zu bestimmenden sechs Freiheitsgrade.

Die theoretische Transformation eines Punktes $(x_c, y_c)$ aus dem Kamerakoordinatensystem in das Bildebenenkoordinatensystems $(u, v)$, ist in Gleichung 2.22 durch eine simple geometrische Umrechnung dargestellt.

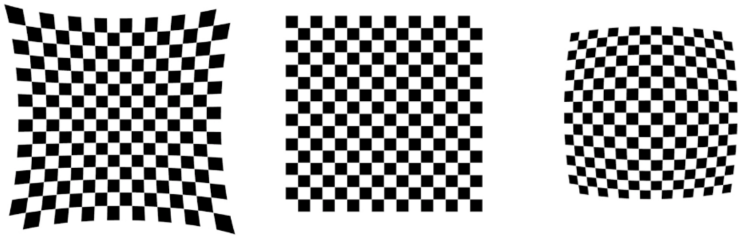
$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{f}{z_c} \begin{bmatrix} x_c \\ y_c \end{bmatrix} \quad \text{(Gleichung 2.22)}$$

In der Realität weicht die Position des Punktes $P$ von seiner theoretischen Lage leicht ab, dargestellt durch die gestrichelte Linie in Abbildung 2.6. Grund hierfür sind sogenannte radiale und tangentiale Verzeichnungen, welche in Gleichung 2.22 zusätzlich berücksichtigt werden müssen. Für den Großteil der Kameras ist die Betrachtung der radialen Verzeichnungen ausreichend, um hinreichend genaue Ergebnisse zu erzielen. Tangentiale Verzeichnungen entstehen gewöhnlicherweise durch dezentriert angebrachte Linsen.

Das in Gleichung 2.23 dargestelle sogenannte Divisionsmodell modelliert die radialen Verzeichnungen abhängig von dem Verzeichnungsfaktor $\kappa$. Übliche radiale Verzeichnungen sind die sogenannten Kissenverzeichnung (Abb. 2.7 links) und Tonnenverzeichnungen (Abb. 2.7 rechts).

$$\begin{bmatrix} \hat{u} \\ \hat{v} \end{bmatrix} = \frac{2}{1 + \sqrt{1 - 4\kappa(u^2 + v^2)}} \begin{bmatrix} u \\ v \end{bmatrix} = \frac{2}{1 + \sqrt{1 - 4\kappa r^2}} \begin{bmatrix} u \\ v \end{bmatrix} \quad \text{(Gleichung 2.23)}$$

Für die finale Transformation von dem Bildebenenkoordinatensystem in das Bildkoordinatensystem müssen noch der Hauptpunkt $(c_x, c_y)$, der Punkt an dem die optische Achse die Bildebene schneidet sowie die horizontalen und vertikalen Abstände $(s_x, s_y)$ der Sensorelemente berücksichtigt werden.

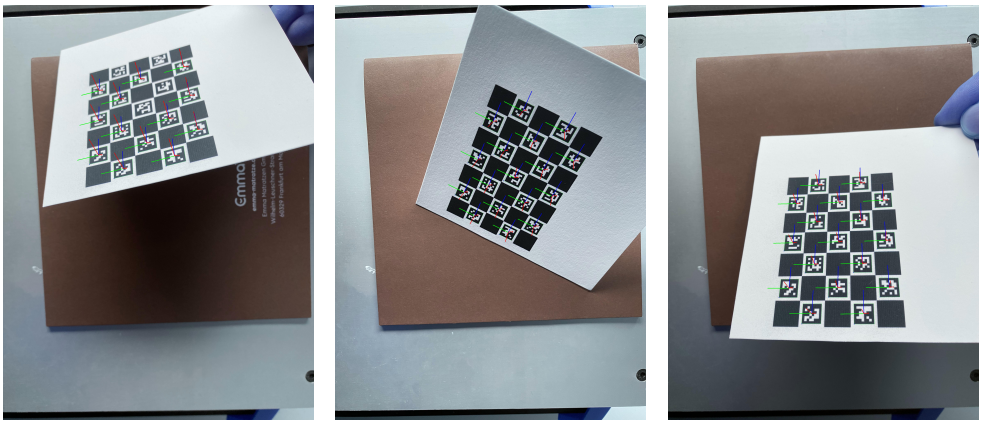


(a) $\kappa > 0$ (b) $\kappa = 0$ (c) $\kappa < 0$

*Abbildung 2.7: Radiale Verzeichnungen in Abhängigkeit von κ (Strathearn, 2012)*

$$\begin{bmatrix} c \\ r \end{bmatrix} = \begin{bmatrix} \hat{u}/s_x + c_x \\ \hat{v}/s_y + c_y \end{bmatrix} \quad \text{(Gleichung 2.24)}$$

Zusätzlich zu den sechs Parametern der äußeren Orientierung ergeben sich dadurch mit $f$, $\kappa$, $s_x$, $s_y$, $c_x$, $c_y$, sechs weitere zu bestimmende Parametern aus der inneren Orientierung. Das Ermitteln der unbekannten Kameraparameter ist nun ein nichtlineares Optimierungsproblem, indem anhand mehrerer Fotos des Kalibrierköpers in verschiedenen Lagen, die Positionen und Abmessungen der enthaltenen Muster ermittelt und durch Variation der Kamaraparameter an die tatsächlichen Abmessungen und Orientierungen angenähert werden. Abbildung 2.8 zeigt Beispiele der benötigten Fotos.



(a) (b) (c)

*Abbildung 2.8: Beispielbilder einer Kamerakalibrierung*

Mittels dieser Methode der Kamerakalibrierung und Koordinatentransformation ist jedoch lediglich die Ausdehnung eines Punktes in einer 2D Ebene bestimmbar. Um die dreidimensionale Position eines Punktes in Weltkoordinaten zu bestimmen wird mindestens ein weiteres Bild des selben Objekts benötigt. Diese Art der Kalibrierung wird Stereo Rekonstruktion genannt.

### 2.4.4 Grundlagen der Stereo Rekonstruktion

Wie in Abbildung 2.9 demonstriert ist es anhand eines einzelnen Bildes nicht möglich die Entfernung der eingezeichneten schwarzen Punkte zu ermitteln, da jeder von ihnen auf den selben Punkt in der Bildebene von Kamera $O_L$ projiziert wird. Nimmt man jedoch eine zweites Bild hinzu, hier aufgenommen durch $O_R$, so wird jeder der schwarzen Punkte auf einen anderen Punkt in der Bildebene von $O_R$ projiziert.

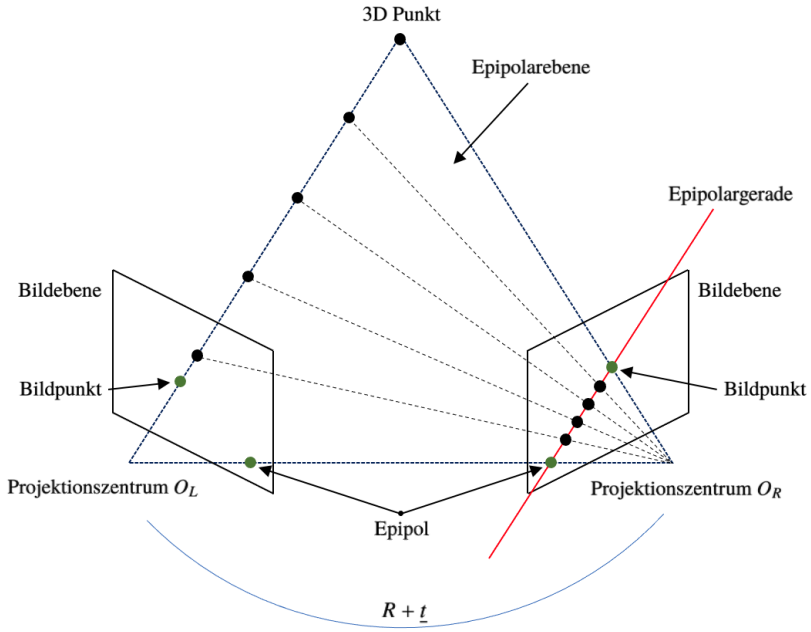



*Abbildung 2.9: Epipolar Geometrie; in Anlehnung an (Yousif et al., 2015)*

Um die Stereo Rekonstruktion durchführen zu können, werden zusätzlich zu der jeweiligen Kamerakalibrierung der einzelnen Kameras, weitere geometrische Informationen benötigt. Zum einen wird die Orientierung der zweiten Kamera relativ zur ersten Kamera benötigt. Diese Kombination aus Translation und Rotation wird in der sogenannten Essentiellen Matrix $E$ beschrieben.

Jeder Punkt der in beide Bildebenen projiziert wird spannt zusammen mit den zwei Ursprüngen der Kameras eine Ebene auf, die sogenannte Epipolarebene. Der Schnitt dieser Ebene mit den Bildebenen der Kameras erzeugt je eine Gerade, die Epipolargerade. Dieser geometrische Zusammenhang, also die Zuordnung eines Punktes zu seiner Epipolargeraden wird in der Fundamental-Matrix $F$ abgebildet.

Um nun den Abstand eines Punktes von der Kamera beziehungsweise den Kameras zu berechnen, ist es vorteilhaft die in Abbildung 2.9 dargestellte Konfiguration zunächst in die sogenannte epipolare Standardkonfiguration zu transformieren. Diese Konfiguration zeichnet sich dadurch aus, dass beide Bildebenen parallel zueinander liegen und die gleiche Brennweite $f$ besitzen. Jede Epipolarkonfiguration kann in eine Standardkonfiguration transformiert werden. Hierfür müssen lediglich die beiden Bilder auf neue Bildebenen in Standardkonfiguration projiziert werden (Steger et al., 2018). Abbildung 2.10 zeigt eine solche Standardkonfiguration.

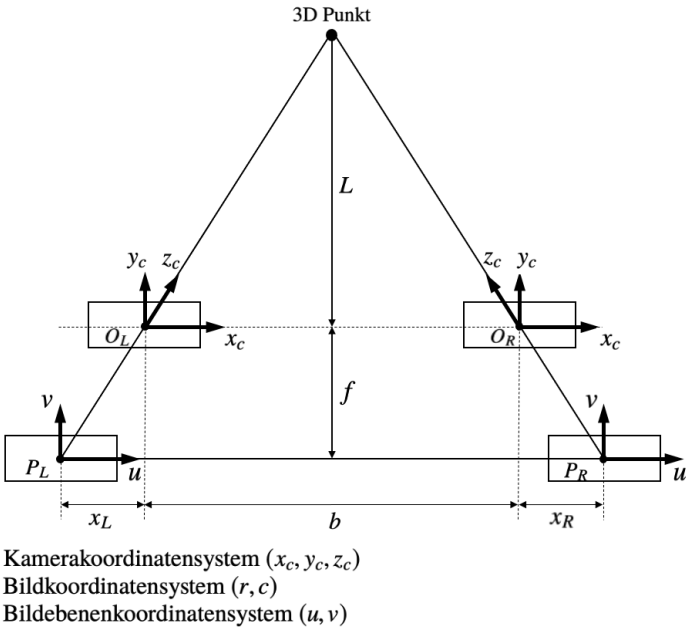



*Abbildung 2.10: Epipolare Standardkonfiguration; in Anlehnung an (Ningthoujam et al., 2019)*

Mittels ähnlicher Dreiecke kann nun durch Anwendung von Gleichung 2.25 der Abstand des Punktes $A$ von den Kameras bestimmt werden.

$$L = \frac{b * f}{x_L - x_R} \qquad \text{(Gleichung 2.25)}$$

# 3 Ziel der Arbeit

Wie bei jeder Anschaffung eines Unternehmens wird auch der Einsatz eines Bildverarbeitungssystems auf Wirtschaftlichkeit geprüft. Hierbei werden den Investitions- und Betriebskosten dieses Systems die aktuellen Kosten der Qualitätsprüfung gegenübergestellt und dadurch die möglichen Einsparungen ermittelt (Norbert, 2008). Neben den Kosteneinsparungen spielen auch humanitäre Ziele eine entscheidende Rolle bei der Überlegung, die aktuelle Qualitätsprüfung durch ein Bildverarbeitungssystem zu ergänzen oder gar zu ersetzen. Neben Spezialaufgaben welche ein Mensch nicht oder nur schwer erfüllen könnte, wird ein solches System nicht zu selten zur Bewältigung von Routineaufgaben verwendet (Bernd, 1996). Durch die Abnahme von monotonen Aufgaben, wie beispielsweise das Zählen oder Sortieren von Objekten, kann ein Mehrwert für die Mitarbeiter geschaffen werden. (Christian, 2011)

Wie im vorherigen Kapitel beschrieben, muss ein Bildverarbeitungssystem auf den jeweiligen Anwendungsfall ausgelegt werden. Hierunter fällt nicht nur das Einrichten der Bildverarbeitungssoftware, sondern auch das Anpassen der verwendeten Hardware und die damit verbundene Integration in die Fertigungskette (Grupp, 2016). Daher führen Änderungen des zu inspizierenden Objekts meist zu einer erneuten Anpassung der Software und eventuell der Hardware (Bernd, 1996). Hohe Stückzahlen und nur seltene Änderungen der Produktspezifikationen führen folglich zu einer geringeren Amortationszeit der Investition. Im Umkehrschluss bedeutet dies jedoch, dass bei Unternehmen mit geringen Produktionsmengen, vielen Einzelfertigungen oder häufig wechselnden Produkten die Wirtschaftlichkeit eines solchen Systems womöglich nicht gegeben ist.

Viele optische Bildverarbeitungssysteme sind je nach Anwendungsfall und Umgebungsbedingungen ausgelegt auf Genauigkeiten von Bruchteilen eines Pixels. Jedoch werden diese Genauigkeiten nicht immer verlangt, da etwa die verwendeten Produktionsmaschinen gar nicht innerhalb dieser Toleranzen arbeiten oder nur eine erste "Grobkontrolle"von geometrisch stark unterschiedlichen Teilen vorgenommen werden soll. Wünschenswert wäre daher ein Tool, mit welchem man mittels Handyaufnahmen Bauteile vermessen kann und dies möglichst unabhängig von den Umgebungseinflüssen. Ansprüche an ein solches Tool wären eine ausreichende Genauigkeit, Flexibilität sowie Robustheit gegenüber Umwelteinflüssen, wie etwa veränderte Licht- und Hintergrundverhältnisse.

# 4 Lösungsansatz

Im Zuge dieser Arbeit soll ein Softwaretool zur optischen Bauteilvermessung implementiert und im Anschluss dessen Einsatzfähigkeit evaluiert werden. Dieses Tool soll es dem Benutzer ermöglichen, mit geringem Aufwand, verschiedenste Bauteile zu vermessen. Für die Implementierung des Vermessungssystems soll neben dem Softwaretool lediglich eine handelsübliche Handykamera und eventuell eine passende Halterung benötigt werden. Als Programmiersprache wurde Python 3 gewählt. In dieser Programmierumgebung steht eine Vielzahl an importierbaren Bibliotheken mit den wichtigsten Methoden der modernen Bildverarbeitung zur Verfügung. Weiterhin existiert aufgrund der ständig steigenden Popularität dieser Programmiersprache eine immense Community, welche sehr stark in entsprechenden Foren vertreten ist. Um einen möglichst großen Nutzerkreis anzusprechen soll das Softwaretool sowohl für MacOSX als auch für Microsoft Windows kompatibel sein. Ein großer Fokus der Arbeit liegt auf der Gestaltung des Interfaces sowie die Erstellung einer entsprechender Bedienungsanleitung. Aufgrund der oben erwähnten gewünschten Flexibilität ist es notwendig, entscheidende Parameter der einzelnen Funktionen, auf den jeweiligen Einsatzfall anzupassen. Hierunter fallen beispielsweise entsprechende Grenzwerte für die Kantenextraktion oder die Bildglättung. Da der Benutzer hierfür nicht in den Quellcode eingreifen soll, muss es möglich sein, diese Einstellung im Interface vornehmen zu können. Um die sinnvollen Einsatzmöglichkeiten des implementierten Programms zu bestimmen, ist es zudem nötig dieses im Anschluss zu evaluieren. Die wichtigsten Faktoren sind die Messgenauigkeit sowie die Robustheit bei veränderten Umwelteinflüssen. Hierfür sollen entsprechende Versuche durchgeführt werden.

# 5 Aufbau des Programmcodes

## 5.1 Verwendete Bibliotheken

Wie zuvor erwähnt, existiert bereits eine Vielzahl an importierbaren Bibliotheken zur Bildbearbeitung. Auf die wichtigsten der hier verwendeten Bibliotheken wird im Folgenden kurz eingegangen. Ein großer Teil des Programmcodes wird für die Gestaltung des benötigten Interfaces beansprucht. Die verwendeten Bibliotheken um Programmfenster, Buttons, Eingabefelder etc. zu gestalten sind die `tkinter` sowie die `tkmacosx` Bibliothek. Die in `tkinter` enthaltenen, vordefinierten Funktionen sind theoretisch ausreichend, um ein funktionsfähiges Interface zu gestalten, da diese Funktionen jedoch größtenteils für Microsoft Windows ausgelegt sind und das fertige Programm ebenfalls für MacOSX Nutzer verfügbar sein soll, wird zusätzliche die `tkmacosx` Bibliothek benötigt. Mit Hilfe der `platform` Bibliothek erfolgt bei Start des Programms eine automatische Abfrage des vorherrschenden Betriebssystems und damit eine Entscheidung über die zu verwendeten Funktionen. Die zuvor erwähnten essentiellen Methoden der Bildbearbeitung, wie etwa Bildglättung, Kantenextraktion, Kamerakalibrierung sowie weitere benötigte Funktionen, sind in der `opencv` Bibliothek enthalten. Ein Vorteil dieser Bibliothek ist das Abspeichern der Bilder als `numpy arrays`, da der Großteil der in dem Programm definierten Berechnungen mittels der `numpy` Bibliothek gelöst werden. Dadurch kann die ständige Formatierung der Ergebnisse sowie der Bilder vermieden werden. Schlussendlich ist noch die `matplotlib` Bibliothek, im speziellen das `TkAgg Backend`, zu erwähnen. Mittels dieser werden alle Miniaturbilder und Graphen in das Interface eingebunden.

## 5.2 Objektorientierte Programmierung

Aus Gründen der besseren Verständlichkeit sowie der Übersichtlichkeit wurde der gesamte Programmcode in einzelne Klassen aufgeteilt und damit objektorientiert aufgebaut. Abbildung 5.1 zeigt die hierfür verwendeten Klassen. Der vermutlich größte Vorteil dieses Programmierstils ist die intuitive Einteilung gleicher Programmbausteine in sogenannte Objekte. So ist beispielsweise jedes der in dem Programm verwendeten Bilder ein Objekt der Klasse `Images`. Alle benötigten Eigenschaften und Funktionen eines Bildes, also dessen Attribute und Methoden müssen dadurch nur einmalig bei der Erstellung der Klasse `Images` definiert werden und dann lediglich

an jedes der zu verwendeten Bilder übergeben werden. Das reduziert den Programmieraufwand enorm und steigert die Verständlichkeit des fertigen Programmcodes.

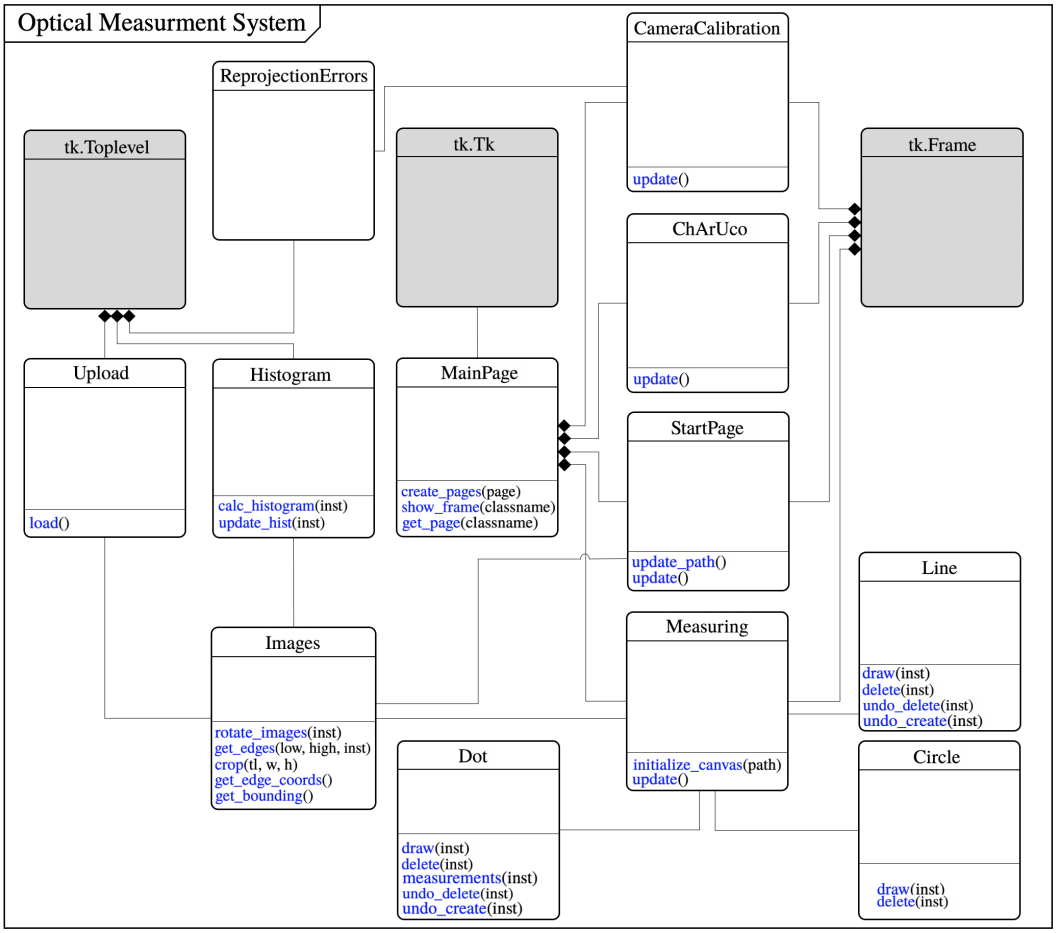



*Abbildung 5.1: Verwendete Klassen und deren Beziehungen*

Die Klasse `Mainpage`, welche die Elternklasse der Hauptfenster des Interfaces darstellt, erbt die Methoden der `tkinter` Klasse `tk.Tk`, die Hauptfenster erben die Methoden der Klasse `tk.Frame` und die zwei Popup Fenster die der Klasse `tk.Toplevel`. Dadurch besitzt jedes Fenster des Interfaces die benötigten Methoden der `tkinter` Bibliothek, welche alle direkt ohne die jeweiligen Präfixes aufgerufen werden können. Das Programm startet durch die Inititalisierung der Klasse `Mainpage`. Mittels der Methode `create_pages()`, welche bei der Initialisierung von `MainPage` aufgerufen wird, werden die Klassen der Hauptfenster initialisiert und somit die benötigten Programmfenster erstellt. Um die Kommunikation zwischen den einzelnen Programmfenstern zu gewährleisten, wird ein `controller` eingeführt. Wie in Abbildung 5.2 dargestellt, übergibt die Klasse `Mainpage` bei der Initialisierung der Hauptfenster die eigene Instanz an diese weiter, welche sie als `controller` abspeichern. Mit Hilfe dieses `controllers` kann jedes Programmfenster nun auf die Methoden von `MainPage` zugreifen.

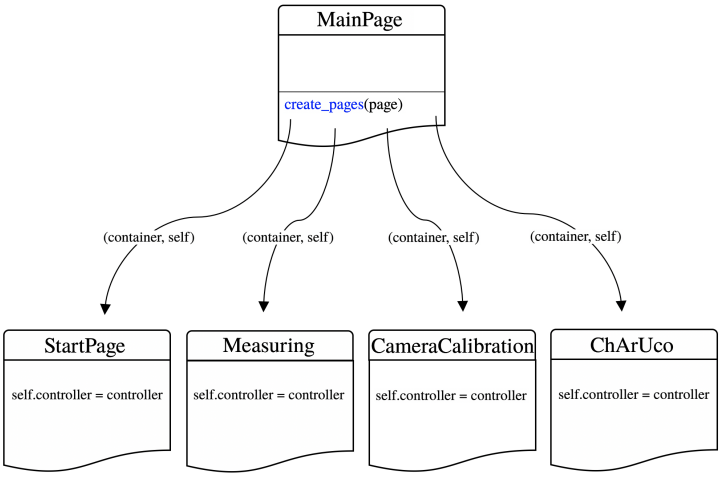



*Abbildung 5.2: Initialisierung der Programmfenster: GUI.py Zeile 58 ff.*

Die Methode `get_pages()` ermöglicht es den Programmfenstern die Instanz anderer Fenster zu erhalten, um auf deren Attribute und Methoden zugreifen zu können. Mittels der Methode `show_frame()`, wird durch Übergabe des gewünschten Fenster- beziehungsweise Klassennamens, dieses in den Vordergrund gebracht und damit sichtbar gemacht. In Abbildung 5.3 ist der beispielhafte Wechsel von `StartPage` zu `Measuring` demonstriert.

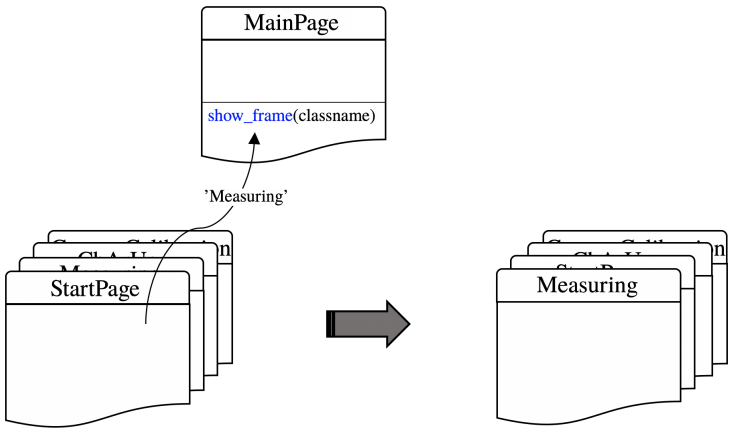



*Abbildung 5.3: Wechseln zwischen den Programmfenstern: GUI.py Zeile 77 ff.*

## 5.3 Allgemeiner Prozessfluss

Der Benutzer hat die Möglichkeit die zu vermessenden Bilder direkt im 'Start Fenster' hochzuladen oder zuerst zum Schritt der Kamerakalibrierung überzugehen. Das springen von einer

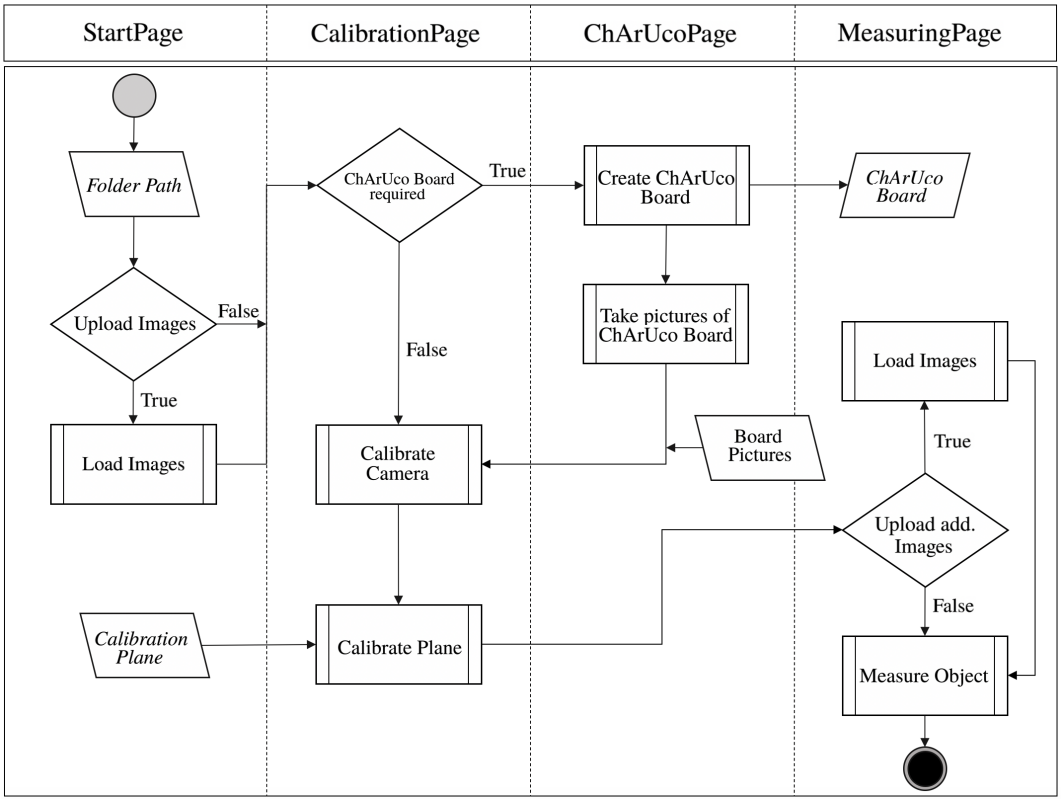



*Abbildung 5.4: Allgemeiner Prozessfluss eines Vermessungsvorgangs*

Seite zur nächsten wird durch entsprechende Menü-Button gesteuert, welche den in Abbildung 5.3 dargestellten Prozess ablaufen lassen. Im Schritt der Kalibrierung kann der Benutzer falls nötig ein individuelles ChArUco Kalibrierungsmuster erstellen und lokal abspeichern. Falls die benötigten Bilder des Kalibrierkörper bereits im entsprechenden Ordner abgelegt sind, kann der Benutzer den Schritt überspringen und direkt den Kalibrierungsprozess starten. Nach der erfolgreichen Kalibrierung der Kamera muss noch die Ebene in welcher gemessen werden soll kalibriert werden. Dies geschieht durch ein weiteres Bild des Kalibrierkörpers, welches in jener Ebene liegt. Nach diesem Schritt ist die Kalibrierung abgeschlossen und der Benutzer kann mit dem Vermessen beginnen oder falls erwünscht zusätzliche Bilder des Objekts hochladen.

## 5.4 Teilprozesse

Im Folgenden werden nun die in Abbildung 5.4 enthaltenen Teilprozessen und wie diese im Programmcode umgesetzt wurden betrachtet. Aus Gründen der Übersichtlichkeit soll auf die wortwörtliche Analyse des Programmcodes verzichtet werden und dieser stattdessen in Flussdiagrammen dargestellt werden. Für jeden dieser Teilprozesse werden die entsprechenden Verweise

zu den Codezeilen sowie zu dem Kapitel im Manual angegeben.

Beim Start des Programms wird der Benutzer aufgefordert den Link zu dem Projekt Ordner einzugeben. Hierbei sollte einer bestimmten Ordnerstruktur gefolgt werden, damit die späteren Vorgänge möglichst reibungsfrei ablaufen können. Die gewünschte Ordnerstruktur ist in Abbildung 5.5 illustriert. So sollte je ein Unterordner für die Bilder des Kalibrierkörpers, für das Bild der zu kalibrierenden Ebene sowie für die zu vermessenden Bilder angelegt und die in Abbildung 5.5 dargestellten Namen verwendet werden.

Das entsprechende Kapitel im Manual ist: *Let's get started - Seite 2*

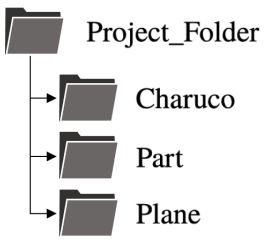



*Abbildung 5.5: Ordnerstruktur des Projekt Ordners*

### 5.4.1 Load Images

Der Prozess 'Load Images' ist zuständig für das Hochladen aller zu vermessenden Bilder welche sich in dem Unterordner 'Part' befinden. Dies ist gleichzeitig verbunden mit der Initialisierung der benötigten `tkinter canvas`. Alle Bilder des zu vermessenden Objekts müssen in ein solches zuvor erstelltes `canvas` geladen werden. Der 'Load Images' Prozess ist demnach zuständig für das Hochladen der Bilder, das Erstellen der `canvas`, das Verknüpfen der `canvas` mit den Bildern, den Button sowie den entsprechenden Scrollbars und für das Aktivieren des ersten `canvas`. Explizit dargestellt ist dieser Teilprozess in den Abbildungen 5.6 und 5.7.

Das entsprechende Kapitel im Manual ist: *Let's get started - Seite 2*

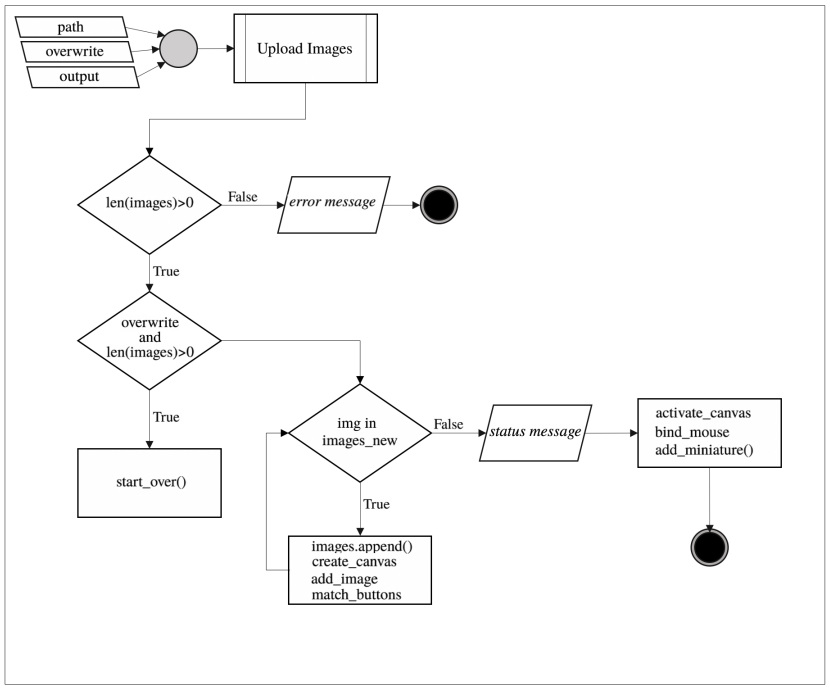



*Abbildung 5.6: Initialisierung der benötigten* `canvas`*: GUI.py Zeile 418 ff.*

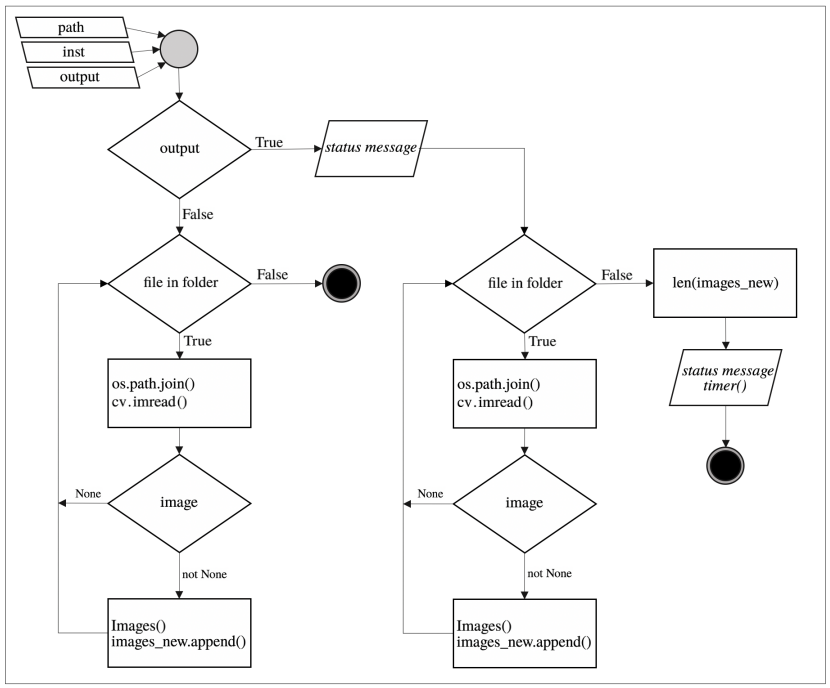



*Abbildung 5.7: Teilprozess 'Upload Images': custom_functions.py Zeile 603 ff.*

### 5.4.2 Create ChArUco Board

Für eine erfolgreiche Kamerakalibrierung ist es unerlässlich den Kalibrierkörper an das zu vermessende Objekt anzupassen. Bevor der Benutzer daher die Fotos dieses Kalibrierkörpers aufnehmen kann, muss er im 'ChArUco Fenster' ein entsprechendes Muster erstellen und ausdrucken.

Das entsprechende Kapitel im Manual ist: *Calibration - Seite 3*

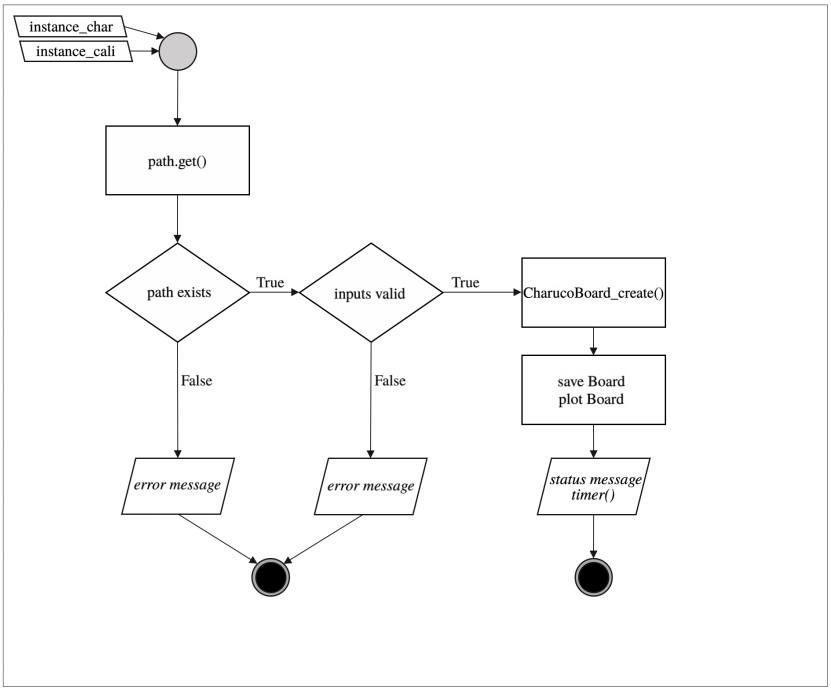



*Abbildung 5.8: Erstellen des Kalibrierungskörpers: custom_functions.py Zeile 1392 ff.*

### 5.4.3 Calibrate Camera

Nachdem der Benutzer die, für die Kalibrierung benötigten, Fotos aufgenommen und diese in dem Unterordner 'Charuco' abgespeichert hat, müssen die eingegebenen Maße des verwendeten Kalibrierkörpers mit dessen tatsächlichen Maßen abgeglichen werden. Oftmals können diese aufgrund des Druckvorgangs leicht abweichen, daher hat der Benutzer die Möglichkeit die Parameter des ChArUco Boards vor Beginn des Kalibrierungsvorgangs nochmals abzuändern. Entscheidend sind hier vor allem die Längen der Quadrate sowie der ChArUco Marker, da diese in der Einheit eingetragen werden, in welcher auch gemessen werden soll. Gleichzeitig

dienen sie als Referenzwert für die darauffolgende Kalibrierung der Messebene. Das zugehörige Flussdiagramm ist in Abbildung 5.9 zu sehen.

Das entsprechende Kapitel im Manual ist: *Calibration - Seite 6*

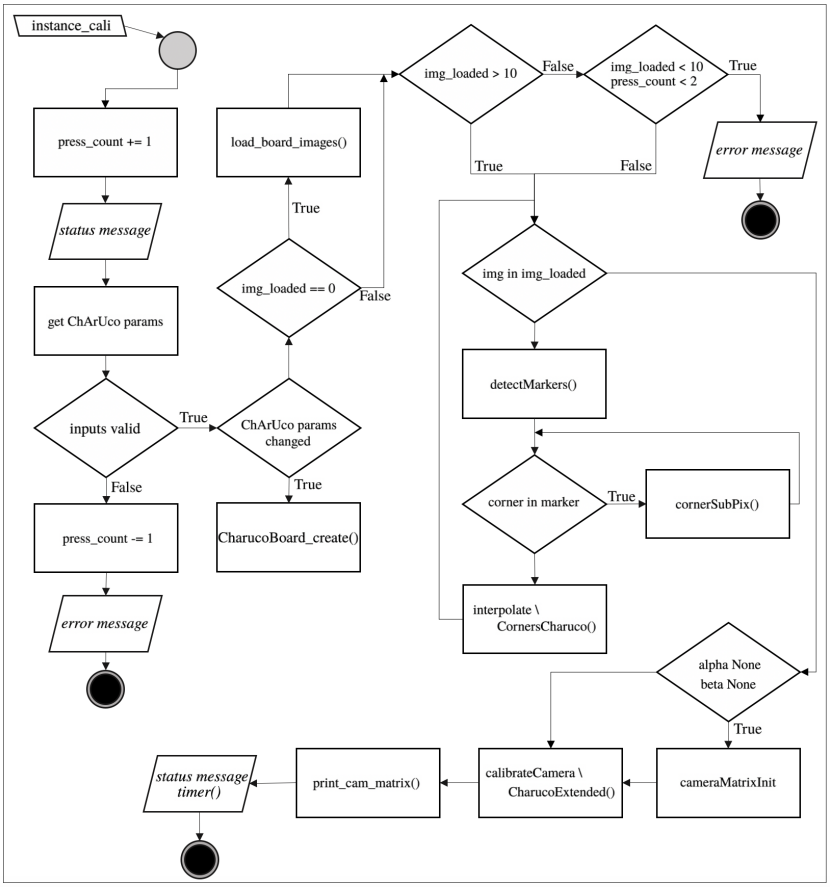



*Abbildung 5.9: Prozess der Kamerakalibrierung: custom.py Zeile 1360 ff.*

### 5.4.4 Calibrate Plane

Die Kalibrierung der Messebene und damit der letzte Schritt vor Beginn der Messungen, ist die Transformation der Bildpunkte in die Messebene. Ohne diesen Schritt könnte man die Abstände lediglich in Pixelkoordinaten messen. Um die Kalibrierung einer Ebene vornehmen zu können wird die Kameramatrix $C$ benötigt. Diese Matrix enthält Informationen über die innere Orientierung in Form der intrinsichen Matrix $K$ und gleichzeitig Informationen über die äußere Orientierung in Form der extrinsischen Matrix $E$. Für die Berechnung dieser Matrizen ist es von Vorteil die kartesischen Koordinaten aus Gleichung 2.23 vorerst in homogene Koordinaten zu transformieren.

$$\begin{aligned} \tilde{\hat{u}} &= x_c f \\ \tilde{\hat{v}} &= y_c f \\ \tilde{z} &= z_c \end{aligned} \qquad \text{(Gleichung 5.1)}$$

Die Gleichung 5.1 kann in eine Matrix Multiplikation umgeformt werden.

$$\begin{bmatrix} \tilde{\hat{u}} \\ \tilde{\hat{v}} \\ \tilde{z} \end{bmatrix} = \begin{bmatrix} f_x & 0 & 0 & 0 \\ 0 & f_y & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \qquad \text{(Gleichung 5.2)}$$

Um diese Gleichung von dem Bildebenenkoordinatensystem $(u, v)$ in das Bildkoordinatensystem $(r, c)$ umzurechnen, muss zusätzlich zu den Brennweiten $(f_x, f_y)$ der Versatz des Ursprungs sowie die Breite und Höhe der einzelnen Pixel berücksichtigt werden.

$$\begin{bmatrix} \tilde{r} \\ \tilde{c} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} f/s_x & 0 & c_x & 0 \\ 0 & f/s_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \qquad \text{(Gleichung 5.3)}$$

Die daraus resultierende intrinsische Kamera Matrix $K$ spiegelt die innere Orientierung der Kamera wieder. Diese ist individuell für jede Kamera und ortsunabhängig.

$$K = \begin{bmatrix} f_x/s_x & 0 & c_x & 0 \\ 0 & f_y/s_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad \text{(Gleichung 5.4)}$$

Um letztendlich noch die Transformation der Kamerakoordinaten in Weltkoordinaten zu berücksichtigen, muss die äußere Orientierung und damit die Rotation durch $R$ und Translation durch $\underline{t}$, mit in die Gleichung aufgenommen werden.

$$E = \begin{bmatrix} R & \underline{t} \\ 0_{1x3} & 1 \end{bmatrix}^{-1} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & t_x \\ R_{21} & R_{22} & R_{23} & t_y \\ R_{31} & R_{32} & R_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(Gleichung 5.5)}$$

Die Multiplikation der intrinsischen Matrix $K$ und der extrinsischen Matrix $E$ ergibt die gesuchte Kameramatrix $C$.

$$C = K \cdot E = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & C_{34} \end{bmatrix} \quad \text{(Gleichung 5.6)}$$

Der mathematische Zusammenhang eines Punkt im Ebenen- beziehungsweise im Weltkoordinatensytem und dessen Transformation ins Bildkoordinatensystem, ergibt sich demnach durch die Multiplikation des Punktes mit der Kamera Matrix $C$. Eine umgekehrte Transformation, von Pixel- zu Weltkoordinaten entspricht einer Multiplikation mit der inversen Kameramatrix $C^{-1}$.

$$\begin{bmatrix} \tilde{r} \\ \tilde{c} \\ \tilde{w} \end{bmatrix} = C \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} \quad \text{(Gleichung 5.7)}$$

Aufgrund der Skalierungsinvarianz der homogenen Koordinaten, dargestellt in Gleichung 5.8 kann die Kameramatrix $C$ mit einem beliebigen Skalierungsfaktor $\lambda$ multipliziert werden.

$$r = \frac{\tilde{r}}{\tilde{w}} \quad \text{and} \quad c = \frac{\tilde{c}}{\tilde{w}} \quad \text{(Gleichung 5.8)}$$

$$\begin{bmatrix} \tilde{r} \\ \tilde{c} \\ \tilde{w} \end{bmatrix} = \lambda C \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} \quad \text{(Gleichung 5.9)}$$

Üblicherweise wird der Skalierungsfaktor so gewählt dass $C_{34}$ zu 1 wird.

$$\begin{bmatrix} \tilde{r} \\ \tilde{c} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} \quad \text{(Gleichung 5.10)}$$

Für die Kalibrierung einer bestimmten Ebene wird ein Koordinatensystem eingeführt dessen x- und y-Achse in der entsprechenden Ebene liegen, hier Ebenenkoordinatensystem genannt.

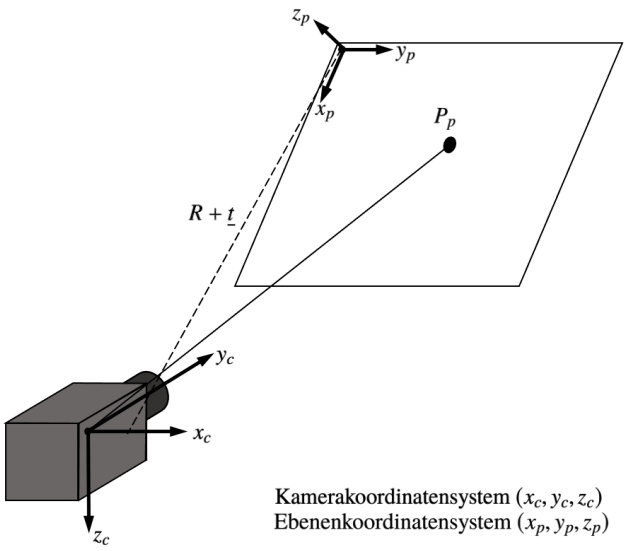



*Abbildung 5.10: Einführung des Ebenenkoordinatensystems; in Anlehnung an (Corke, 2017)*

Wie in Abbildung 5.10 ersichtlich, gilt für jeden Punkt $P_p$ in dieser Ebene $z_p = 0$. Gleichung 5.10 vereinfacht sich damit zu

$$\begin{bmatrix} \tilde{r} \\ \tilde{c} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{14} \\ C_{21} & C_{22} & C_{24} \\ C_{31} & C_{32} & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ 0 \\ 1 \end{bmatrix} \qquad \text{(Gleichung 5.11)}$$

Die daraus entstehende Homografiematrix $H$ mit acht Unbekannten lässt sich bereits, durch 4 Punktkorrespondenzen zwischen der zu kalibrierenden Ebene und dem Bildkoordinatensystem, bestimmen. Diese Korrespondenzen werden durch den Kalibrierkörper, liegend in der gewünschten Messebene, festgelegt.

$$\begin{bmatrix} \tilde{r} \\ \tilde{c} \\ \tilde{w} \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} \qquad \text{(Gleichung 5.12)}$$

Abbildung 5.11, Abbildung 5.12 und Abbildung 5.13 demonstrieren wie dies in dem Programm-

code umgesetzt wurde.

Das entsprechende Kapitel im Manual ist: *Calibration - Seite 10*

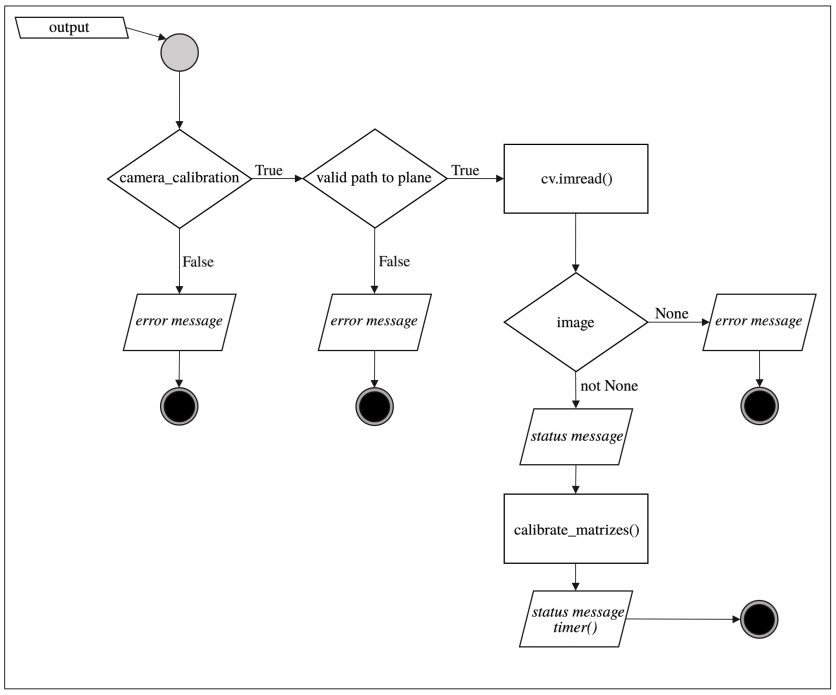



*Abbildung 5.11: Kalibrierung der Messebene: custom_functions.py Zeile 1507 ff.*

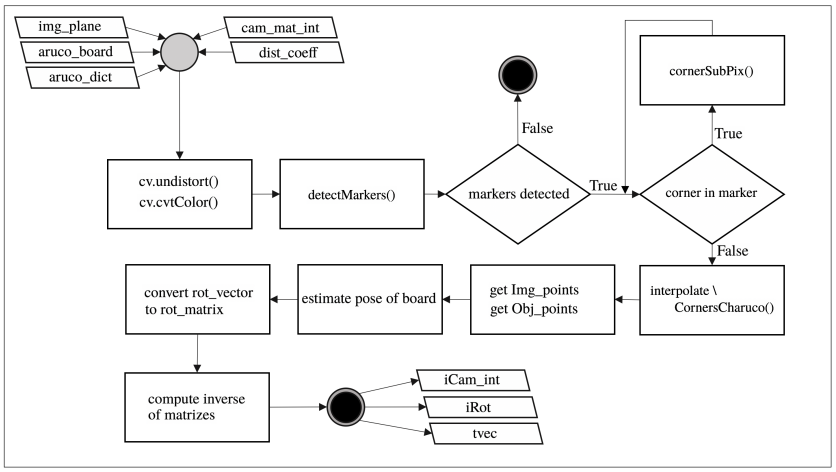



*Abbildung 5.12: Berechnung der benötigten Matrizen: custom_functions.py Zeile 1444 ff.*

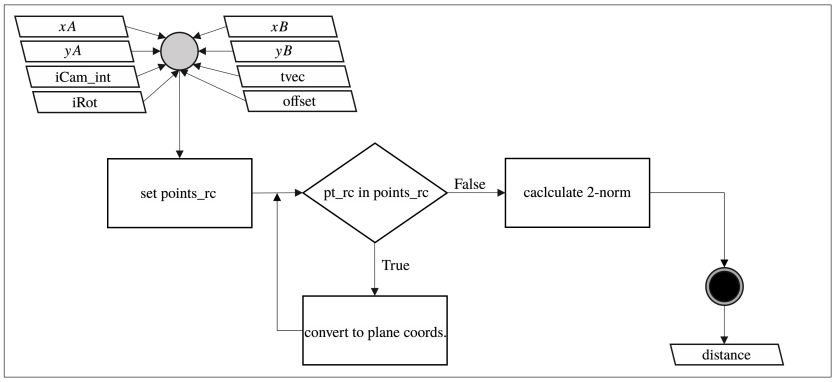



*Abbildung 5.13: Berechnung des Abstandes zweier Punkte: custom.py Zeile 2203 ff.*

## 5.5 Hilfreiche OpenSource Quellen

Wie in Abbildung 5.14 ersichtlich ist Pyhton aktuell die mit Abstand beliebteste Programmiersprache weltweit. Die Statistik beruht darauf, wie häufig eine Programmiersprache in Google-Suchanfragen auftaucht. Aufgrund dessen ist der Großteil, der online zu findenden Code Beispiele und Tutorials, in Python umgesetzt. Besonders hilfreich für diese Arbeit waren, neben den eigentlichen Dokumentationen der importierbaren Bibliotheken wie beispielsweise docs.opencv.org, die unzähligen zusätzlichen Anwendungsbeispiele für diese Bibliotheken, veröffentlicht in diversen Foren. Viele programmierbezogene Probleme, die bei Erstellung dieser Arbeit auftauchten, konnten meist durch das Durchforsten von Foren wie beispielsweise stackoverflow.com gelößt werden.

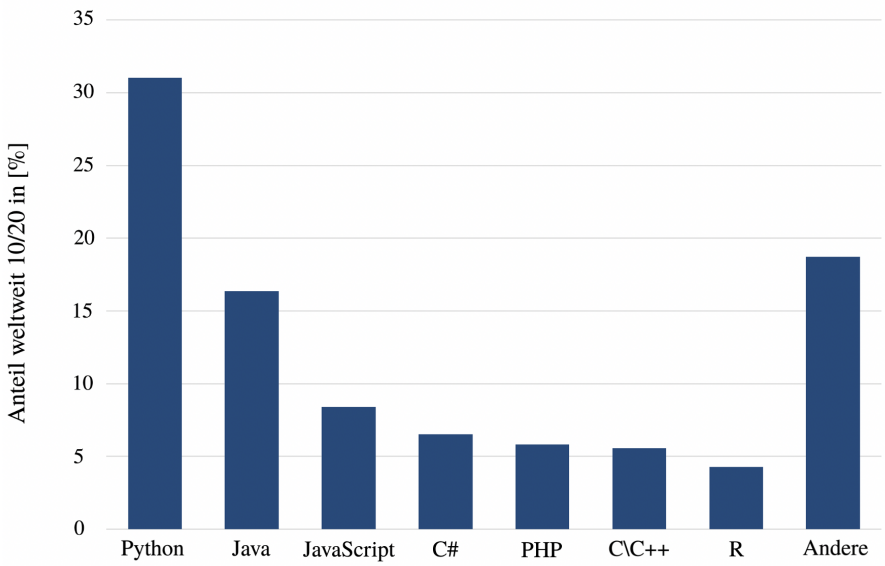



*Abbildung 5.14: Beliebtheit der Programmierspachen; in Anlehnung an (Carbonelle, 2020)*

# 6 Evaluierung der Software

Das hier implementierte Softwaretool ist in zwei große Bereiche aufgeteilt. Die Kalibrierung der Kamera und der eigentliche Messvorgang. Es ist naheliegend, dass der Messvorgang beziehungsweise die Messergebnisse sehr stark von der Kamerakalibrierung abhängen, daher sollten auch beide Vorgänge einzeln evaluiert werden.

## 6.1 Kamerakalibrierung

Ein Maß für die Evaluierung der Kamerakalibrierung ist der Reprojektionsfehler RMS (*Camera Calibration* 2019). Dieser Fehler spiegelt den Abstand eines, auf dem Kalibrierkörper erkanntem, Punktes und dem dazugehörigen projiziertem 3D Punkt wieder. Abbildung 2.6 im Kapitel zur Kamerakalibrierung visualisiert den Reprojektionsfehler als den Unterschied des perfekten Lichtstrahls, dargestellt durch die gestrichelte Linie und dem tatsächlichen Versatz des Bildpunktes $p$. Ziel der Kalibrierung ist es nun, wie bereits erwähnt, die unbekannten Kameraparameter iterativ anzupassen, sodass der Reprojektionsfehler minimal wird. Die verwendeten Bildaufnahmen des Kalibrierkörpers können daher als Trainingsbilder angesehen werden. Der Erfolg der Kalibrierung hängt demnach ausschließlich von der Qualität dieser Trainingsbilder ab. Im Folgenden soll nun der Erfolg der Kamerakalibrierung, gemessen an dem Reprojektionsfehler RMS, bei Variation der Eingabebilder untersucht werden. Hierfür werden zwei unterschiedliche ChArUco Kalibrierkörper verwendet.

*Tabelle 6.1: Vergleich der zur Evaluierung der Kamerakalibrierung verwendeten ChArUco Kalibrierkörper*

| Bezeichnung | Zeilen | Spalten | Länge Quadrat | Länge Marker | Aruco Bibliothek |
|---|---|---|---|---|---|
| Klein | 5 | 7 | 11 mm | 8.35 mm | 6x6_250 |
| Groß | 5 | 7 | 23 mm | 17.20 mm | 6x6_250 |

*(a)* 2. Versuchsreihe Versuch 02 *(b)* 1. Versuchsreihe Versuch 04

*Abbildung 6.1: Beispielbilder der verwendeten Kalibrierkörper mit eingezeichneter Orientierung*

Folgende Versuche sollen nun durchgeführt werden, um den Erfolg der Kalibrierung zu evaluieren. Der Neigungswinkel $\theta$ stellt hierbei den geschätzten Winkel zwischen Kalibrierkörper und Kamera dar. Zudem soll der Einfluss des Autofokuses auf das Kalibrierergebnis untersucht werden, da durch das Scharfstellen die Bildebene verschoben wird und demnach angenommen wird, dass dieser Vorgang die Kalibrierung verschlechtert. Um eine größere Anzahl an Messwerten zu erhalten, wird jeder dieser Versuche dreimal durchgeführt und die Einzelergebnisse sowie das arithmetische Mittel untersucht.

**Versuch 01** wenige Bilder mit kleinem Kalibrierkörper bei geringem Neigungswinkel $\theta$

**Versuch 02** wenige Bilder mit kleinem Kalibrierkörper bei starkem Neigungswinkel $\theta$

**Versuch 03** wenige Bilder mit großem Kalibrierkörper bei geringem Neigungswinkel $\theta$

**Versuch 04** wenige Bilder mit großem Kalibrierkörper bei starkem Neigungswinkel $\theta$

**Versuch 05** wenige Bilder mit kleinem Kalibrierkörper bei geringem Neigungswinkel $\theta$ und aktiviertem Autofokus

**Versuch 06** wenige Bilder mit kleinem Kalibrierkörper bei starkem Neigungswinkel $\theta$ und aktiviertem Autofokus

**Versuch 07** wenige Bilder, großer Kalibrierkörper, geringer Neigungswinkel $\theta$ und aktiviertem Autofokus

**Versuch 08** wenige Bilder mit großem Kalibrierkörper bei starkem Neigungswinkel $\theta$ und aktiviertem Autofokus

**Versuch 09** viele Bilder mit kleinem Kalibrierkörper bei geringem Neigungswinkel $\theta$

**Versuch 10** viele Bilder mit großem Kalibrierkörper bei geringem Neigungswinkel $\theta$

Im Folgendem sind die Ergebnisse der drei Versuchsreihen sowie die arithmetische Mittelung dieser Versuche in tabellarischer Form dargestellt.

*Tabelle 6.2: Versuchsergebnisse der ersten Versuchsreihe mit* $\mathrm{RMS}_{ges}$ *als Reprojektionsfehler der Kalibrierung und* $\mathrm{RMS}_{img,min/max}$ *als Reprojektionsfehler einzelner Bilder.*

| **Versuch** | $\mathrm{RMS}_{ges}$ | $\mathrm{RMS}_{img,max}$ | $\mathrm{RMS}_{img,min}$ | **Anzahl Bilder** | **Anteil** $\mathrm{RMS}_{img} < 1$ |
|---|---|---|---|---|---|
| 01 | 0,41 | 0,49 | 0,24 | 5 | 1,00 |
| 02 | 0,44 | 0,54 | 0,37 | 5 | 1,00 |
| 03 | 3,07 | 5,64 | 0,76 | 5 | 0,20 |
| 04 | 3,05 | 4,46 | 1,20 | 5 | 0 |
| 05 | 1,04 | 1,44 | 0,82 | 5 | 1,00 |
| 06 | 0,45 | 0,60 | 0,39 | 5 | 1,00 |
| 07 | 2,73 | 3,97 | 1,09 | 5 | 0 |
| 08 | 2,97 | 6,25 | 0,84 | 5 | 0,20 |
| 09 | 0,46 | 2,44 | 0,23 | 40 | 0,98 |
| 10 | 3,65 | 5,78 | 0,72 | 40 | 0,13 |

Aus den Ergebnissen der ersten Versuchsreihe wird ersichtlich, dass der gesamte Reprojektionsfehler bei der Verwendung des kleineren Kalibrierkörpers deutlich geringer ist. Weiterhin scheint es, dass eine deutliche Vermehrung der Bilder von 5 auf 40 das Ergebnis nicht verbessert. Der Grund hierfür ist der sinkende Anteil an Bildern mit einem Reprojektionsfehler $\mathrm{RMS}_{img}$ kleiner 1. Interessant ist außerdem, dass der Autofokus nur einen geringen bis keinen Einfluss auf den erzielten Reprojektionsfehler dieser Versuchsreihen hat.

Die Ergebnisse der zweiten Versuchsreihe bestätigen die Annahmen aufgrund der vorherigen Ergebnisse bezüglich des Unterschiedes der Kalibrierkörper. Der gesamte Reprojektionsfehler bei Verwendung des kleineren Kalibrierkörpers ist weiterhin deutlich geringer. Jedoch ist bei diesen Versuchen der Einfluss des Autofokus auf das Ergebnis deutlich zu sehen. Versuch 05 weist ein deutlich schlechteres Ergebnis auf als bei der ersten Versuchsreihe. Zudem stieg der Anteil an Bildern mit einem Reprojektionsfehlern $\mathrm{RMS}_{img}$ größer 1. Bei Verwendung des großen Kalibrierkörpers scheint der Autofokus nur einen geringen Einfluss zu haben.

*Tabelle 6.3: Versuchsergebnisse der zweiten Versuchsreihe mit* $\mathrm{RMS}_{ges}$ *als Reprojektionsfehler der Kalibrierung und* $\mathrm{RMS}_{img,min/max}$ *als die minimalen und maximalen Reprojektionsfehler einzelner Bilder.*

| **Versuch** | $\mathrm{RMS}_{ges}$ | $\mathrm{RMS}_{img,max}$ | $\mathrm{RMS}_{img,min}$ | **Anzahl Bilder** | **Anteil** $\mathrm{RMS}_{img} < 1$ |
|---|---|---|---|---|---|
| 01 | 0,42 | 0,47 | 0,36 | 5 | 1,00 |
| 02 | 0,44 | 0,51 | 0,32 | 5 | 1,00 |
| 03 | 2,79 | 4,05 | 1,11 | 5 | 0 |
| 04 | 4,59 | 7,59 | 1,52 | 5 | 0 |
| 05 | 3,38 | 5,86 | 1,70 | 5 | 0 |
| 06 | 0,51 | 0,66 | 0,36 | 5 | 1,00 |
| 07 | 2,81 | 4,43 | 0,76 | 5 | 0,20 |
| 08 | 5,27 | 8,31 | 1,72 | 5 | 0 |
| 09 | 1,88 | 8,63 | 0,36 | 40 | 0,60 |
| 10 | 4,23 | 6,95 | 0,70 | 40 | 0,03 |

*Tabelle 6.4: Versuchsergebnisse der dritten Versuchsreihe mit* $\mathrm{RMS}_{ges}$ *als Reprojektionsfehler der Kalibrierung und* $\mathrm{RMS}_{img,min/max}$ *als die minimalen und maximalen Reprojektionsfehler einzelner Bilder.*

| **Versuch** | $\mathrm{RMS}_{ges}$ | $\mathrm{RMS}_{img,max}$ | $\mathrm{RMS}_{img,min}$ | **Anzahl Bilder** | **Anteil** $\mathrm{RMS}_{img} < 1$ |
|---|---|---|---|---|---|
| 01 | 0,38 | 0,44 | 0,35 | 5 | 1,00 |
| 02 | 0,40 | 0,45 | 0,32 | 5 | 1,00 |
| 03 | 1,94 | 3,30 | 0,83 | 5 | 0,20 |
| 04 | 2,99 | 4,08 | 0,98 | 5 | 0 ,20 |
| 05 | 2,17 | 3,33 | 1,41 | 5 | 0 |
| 06 | 0,60 | 0,87 | 0,33 | 5 | 1,00 |
| 07 | 2,44 | 3,74 | 0,76 | 5 | 0,20 |
| 08 | 4,09 | 6,11 | 1,13 | 5 | 0 |
| 09 | 0,39 | 0,49 | 0,29 | 40 | 1,00 |
| 10 | 3,76 | 5,82 | 0,80 | 40 | 0,08 |

Die Ergebnisse der dritten Versuchsreihe bestätigen die Annahmen der vorherigen Versuche. Die Verwendung des kleineren Kalibrierkörpers führt zu signifikant besseren Kalibrierergebnissen. Der Autofokus zeigt einen klaren Einfluss auf das Gesamtergebnis, besonders anfällig hierfür ist der kleinere Kalibrierkörper. Weiterhin führt eine Vergrößerung des Neigungswinkels zwischen Kamera und Kalibrierkörper unabhängig von der Größe des verwendeten Kalibrierkörpers, zu einem höheren Reprojektionsfehler. Das beste erzielte Ergebnis bei Verwendung des kleineren Kalibrierkörpers war der Versuch 01 der dritten Versuchsreihe mit einem Reprojektionsfehler $\mathrm{RMS}_{ges}$ von $0,38$. Bei Verwendung des größeren Kalibrierkörpers wurde ein minimaler Reprojektionsfehler $\mathrm{RMS}_{ges}$ von $1,94$ im Versuch 03 der dritten Versuchsreihe erzielt.

Tabelle 6.6 zeigt die über die drei Versuchsreihen gemittelten Ergebnisse.

*Tabelle 6.5: Versuchsergebnisse gemittelt über die drei Versuchsreihen.*

| **Versuch** | $\overline{\text{RMS}}_{ges}$ | $\overline{\text{RMS}}_{img,max}$ | $\overline{\text{RMS}}_{img,min}$ | **Anzahl Bilder** | **Anteil** $\overline{\text{RMS}}_{img} < 1$ |
|---|---|---|---|---|---|
| 01 | 0,40 | 0,46 | 0,31 | 5 | 1,00 |
| 02 | 0,43 | 0,50 | 0,33 | 5 | 1,00 |
| 03 | 2,60 | 4,33 | 0,90 | 5 | 0,13 |
| 04 | 3,54 | 5,38 | 1,23 | 5 | 0 ,07 |
| 05 | 2,20 | 3,54 | 1,31 | 5 | 0,2 |
| 06 | 0,52 | 0,71 | 0,35 | 5 | 1,00 |
| 07 | 2,66 | 4,05 | 0,87 | 5 | 0,13 |
| 08 | 4,11 | 6,89 | 1,23 | 5 | 0,07 |
| 09 | 0,91 | 3,85 | 0,29 | 40 | 0,86 |
| 10 | 3,88 | 6,18 | 0,74 | 40 | 0,08 |

Das Softwaretool erzeugt nach erfolgreicher Kalibrierung eine Auflistung aller Bilder mit deren jeweiligen Reprojektionsfehlern, aufgeteilt in $\text{RMS}_{img} < 1$ und $\text{RMS}_{img} > 1$. Dies soll als Stütze dienen, um sich ein Bild von dem Kalibrierergebnis machen zu können. Auch wenn die einzelnen Reprojektionsfehler nicht unabhänging voneinander sind, können extreme Ausreißer erkannt werden. Manchmal ist es hilfreich, die Bilder mit einem Reprojektionsfehler deutlich höher als der Durchschnitt der Bilder, zu entfernen und den Kalibrierprozess erneut zu starten (*Stereo Camera Calibrator App* 2020). In folgender Tabelle ist der Reprojektionsfehler der Versuche 09, und 10 der drei Versuchsreihen aufgelistet, nachdem die zwei Bilder mit den höchsten Fehlern entfernt wurden. Es ist deutlich zu sehen, dass die einzelnen Repro-

*Tabelle 6.6: Versuchsergebnisse der Versuche 09, und 10 der drei Versuchsreihen, nach Elimination der zwei stärksten Ausreißer*

| **Versuch** | $\text{RMS}_{ges,vor}$ | $\text{RMS}_{img,max,1}$ | $\text{RMS}_{img,max,2}$ | $\text{RMS}_{ges,nach}$ | $\text{RMS}_{img,max,1}$ | $\text{RMS}_{img,max,2}$ |
|---|---|---|---|---|---|---|
| 09/01 | 0,46 | 2,44 | 0,54 | 1,70 | 6,88 | 4,33 |
| 09/02 | 1,88 | 8,63 | 4,71 | 0,52 | 1,01 | 0,78 |
| 09/03 | 0,39 | 0,49 | 0,49 | 0,38 | 0,49 | 0,47 |
| 10/01 | 3,65 | 5,78 | 5,63 | 3,52 | 5,29 | 5,08 |
| 10/02 | 4,23 | 6,95 | 6,12 | 4,10 | 5,97 | 5,74 |
| 10/03 | 3,76 | 5,82 | 5,27 | 3,75 | 5,40 | 5,29 |

jektionsfehler nicht unabhängig voneinander sind. So wurde beispielsweise das gute Ergebnis von $\text{RMS}_{ges,vor} = 0,46$ von Versuch 09 der ersten Versuchsreihe nach Entfernen der Bilder mit den zwei größten Einzelfehlern deutlich verschlechtert. Der Reprojektionsfehler von Versuch 09 der zweiten Versuchsreihe hingegen konnte durch das Entfernen von zwei Bildern enorm verringert werden. Der Einzelfehler $\text{RMS}_{img,max,1}$ dieses Versuchs lag mit $8,63$ auch deutlich über dem gesamten Reprojektionsfehler von $1,88$. Bei den restlichen Versuchen konnte der gesamte Reprojektionsfehler zwar verringert werden, jedoch nur unwesentlich.

Die in den Versuchsreihen ermittelten Ergebnisse decken sich mit den in der Literatur vorgeschlagenen Methoden für eine erfolgreiche Kamerakalibrierung (Jakob, 2018a).

**Kalibrierungskörper** Die Größe des Kalibrierkörpers sollte an die Größe des zu messenden Objekts angepasst werden und maximal die Hälfte des Bildes einnehmen.

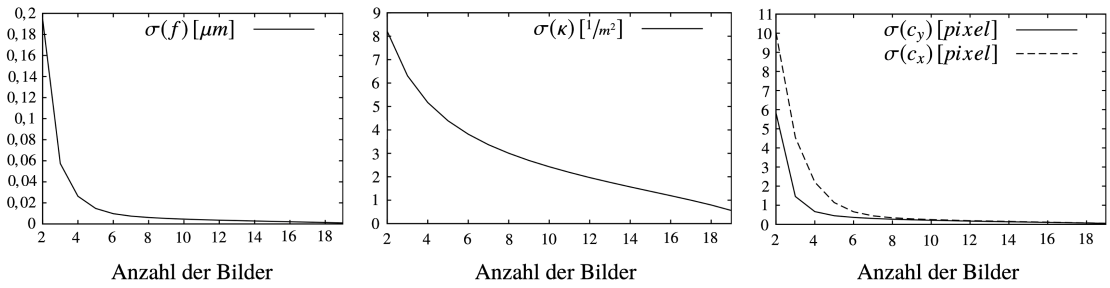
**Autofokus** Da jedes Scharfstellen eine Verschiebung der Bildebene bedeutet sollte der Autofokus deaktiviert werden.

**Neigungswinkel** Um neben der Linsenverzerrung auch die Brennweite einschätzen zu können, sollte der Kalibrierkörper in verschiedene Richtungen geneigt werden. Von Neigungswinkeln größer als 45 Grad ist jedoch abzusehen.

**Anzahl der Bilder** Um jede mögliche Orientierung mit hoher Genauigkeit abdecken zu können, sollten mindestens 10 Bilder verwendet werden. Bei starken radialen Verzeichnungen noch deutlich mehr, siehe Abbildung 6.2

**Analyse der Ergebnisse** Der gesamte Reprojektionsfehler ist zwar ein gutes Maß, um den Erfolg der Kalibrierung einschätzen zu können, jedoch sollten die einzelnen Fehler untersucht werden um eventuelle Ausreißer zu entfernen. Weiterhin ist es vorteilhaft, sich die eingezeichneten Orientierungen der einzelnen Bilder anzusehen. Falls diese stark abweichen, sollte das jeweilige Bild entfernt werden und eventuell weitere hinzugefügt werden.

**Bildaufnahme** Wie zu Beginn erwähnt ist die Grundlage der Kamerakalibrierung die Kantenextraktion, welche stark von Licht- und Rauschverhältnissen abhängt. Schlechte Lichtverhältnisse und starkes Rauschen erschweren das Extrahieren der Kanten und damit die Kamerakalibrierung.




*(a)* Abstand der Bildebene *(b)* Radiale Verzeichnungen *(c)* Lage des Hauptpunktes

*Abbildung 6.2: Erfolg der Kalibrierung in Abhängigkeit von der Anzahl an verwendeten Kalibrierbildern, beschrieben durch die jeweiligen Standardabweichungen; in Anlehnung an (Steger et al., 2018)*

## 6.2 Messvorgang

Um die Einsatzmöglichkeiten des Softwaretools ermitteln zu können, sollen verschiedene Test durchgeführt werden. Wie schon im vorherigen Abschnitt zur Evaluierung der Kamerakalibrierung, sollen pro Versuch mehrere Messungen durchgeführt werden. Im Zuge dieser Evaluierung soll auf die Einflüsse einer schlechter Kamerakalibrierung sowie die Bedeutung der Kalibrierung der Ebene eingegangen werden. Als Messobjekte dienen verschiedene Objekte bekannter Abmessungen, wie beispielsweise das zur Kalibrierung verwendete Kalibrierungsmuster.

**Versuch 01** $\mathrm{RMS}_{ges} \gg 1$; Objekt in kalibrierter Ebene bei guter Beleuchtung

**Versuch 02** $\mathrm{RMS}_{ges} < 0.5$; Objekt in kalibrierter Ebene bei guter Beleuchtung

**Versuch 03** $\mathrm{RMS}_{ges} < 0.5$; Objekt in kalibrierter Ebene bei starken Reflexionen

**Versuch 04** $\mathrm{RMS}_{ges} < 0.5$; Objekt nicht in kalibrierter Ebene bei guter Beleuchtung

**Versuch 05** $\mathrm{RMS}_{ges} < 0.5$; Objekt in kalibrierter Ebene bei schräger Kamera und guter Beleuchtung

**Versuch 06** $\mathrm{RMS}_{ges} < 0.5$; Objekt in kalibrierter, schräger Ebene bei horizontaler Kamera und guter Beleuchtung

**Versuch 07** $\mathrm{RMS}_{ges} < 0.5$; Dreidimensionales Objekt teilweise in kalibrierter Ebene bei guter Beleuchtung

### 6.2.1 Versuch 01

Ziel des ersten Versuchs ist es, die Messgenaugigkeit des Softwaretools bei einer ungünstigen Kalibrierung der Kamera zu ermitteln. Hierfür war während des Kalibriervorgangs der Autofokus aktiviert und es wurden teilweise ungünstige Neigungswinkel verwendet. Dies führte zu einem Reprojektionsfehlers $\mathrm{RMS}_{ges}$ von $3,83$. Das Foto welches für den Messvorgang verwendet wurde ist von guter Qualität und wurde bei guten Lichtverhältnissen aufgenommen, wodurch sich die Kanten einfach extrahieren lassen konnten. Abbildung 6.3 zeigt die Messergebnisse des Versuchs. Für den Versuch war die automatische Kantensuche aktiviert, welche die nächstgelegene Kantenposition auswählt, falls der Benutzer an eine Nicht-Kantenposition klickt. Die hier vermessenen Längen sind die schwarzen Quadrate des Kalibriermusters, welche in Realität exakt 11 Millimeter auf 11 Millimeter sind. Es fällt auf, dass die horizontalen Messungen

deutlich stärker von den tatsächlichen Längen abweichen als die vertikalen. Es ergibt sich ein durchschnittlicher relativer Messfehler von $\overline{\text{RMS}}_{rel}$ von $11,47\%$.

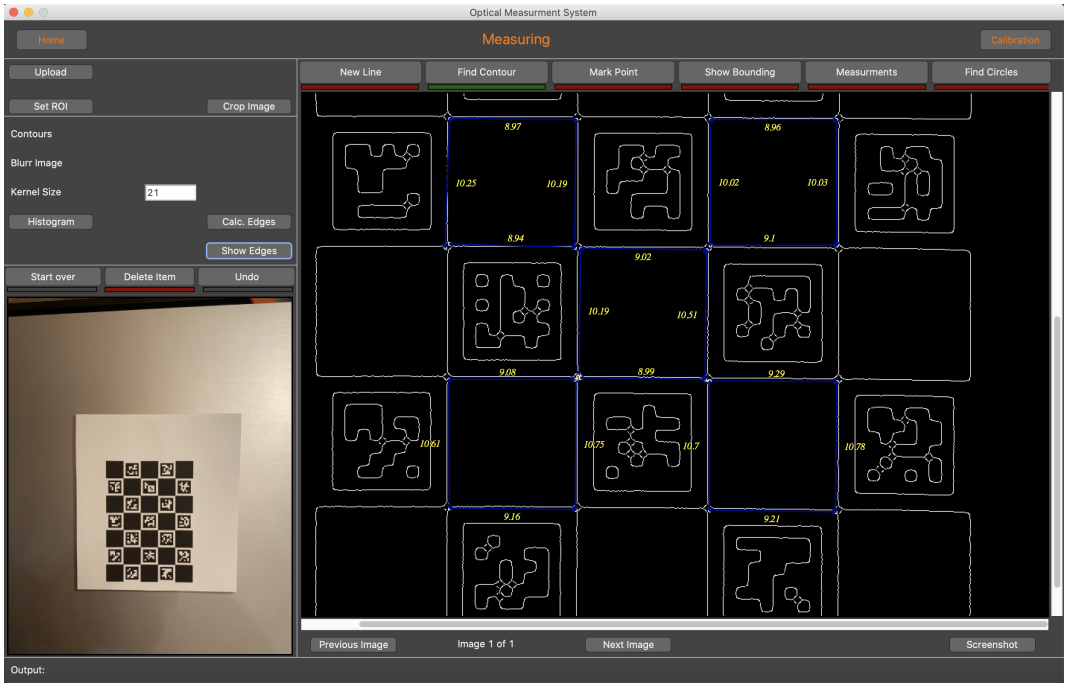



*Abbildung 6.3: Messergebnisse bei ungünstiger Kamerakalibrierung*

### 6.2.2 Versuch 02

Der zweite Versuch soll nun den Einfluss einer guten Kalibrierung auf die Messgenauigkeit verdeutlichen, indem die gleichen Bedingungen wie im ersten Versuch vorherrschen, jedoch bei einem deutlich besseren Reprojektionsfehler $\text{RMS}_{ges}$ von $0,39$. Abbildung 6.4 zeigt die Messergebnisse welche einen relativen Messfehler von $\overline{\text{RMS}}_{rel}$ von $0,7\%$ enthalten.

### 6.2.3 Versuch 03

Für den dritten Versuch wurde als Unterlage ein Spiegel verwendet, um den Einfluss von ungünstiger Beleuchtung beziehungsweise starker Reflexionen zu verdeutlichen. Auch wenn die Messergebnisse einen vergleichsweise geringen relativen Messfehler $\overline{\text{RMS}}_{rel}$ von $3,14\%$ enthalten, ist deutlich, dass die Kantenextraktion sehr erschwert wurde. Viele Kanten konnten nicht mehr korrekt dargestellt werden, da sie aufgrund der Reflexionen einen stark unterschiedlichen Grauwert besitzen.

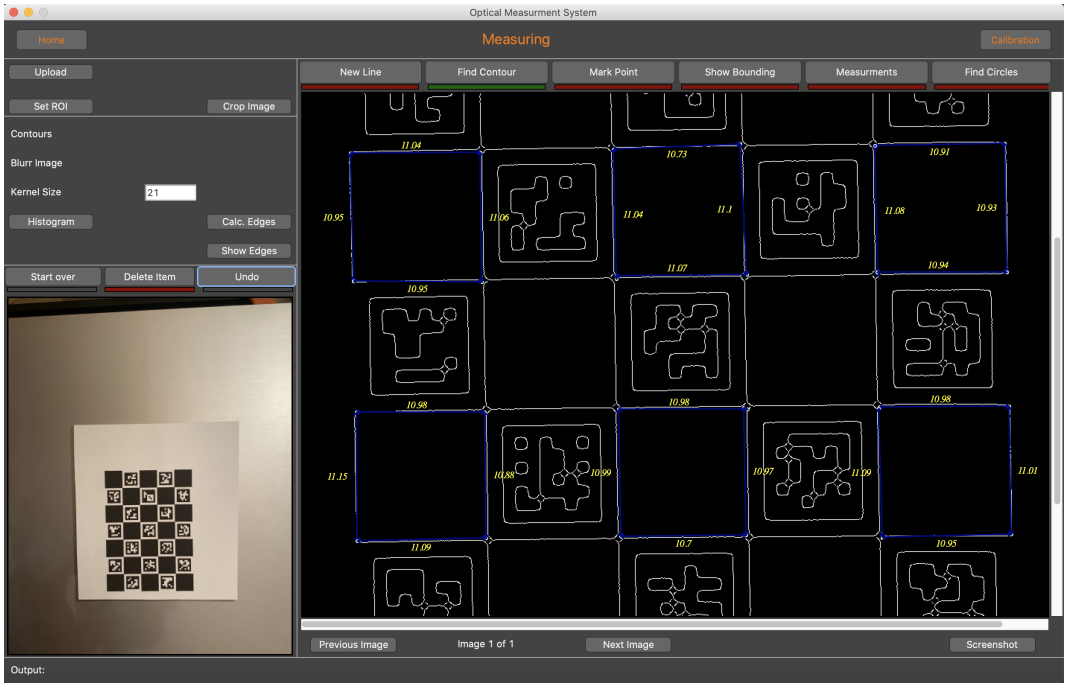



*Abbildung 6.4: Messergebnisse bei guter Kamerakalibrierung*

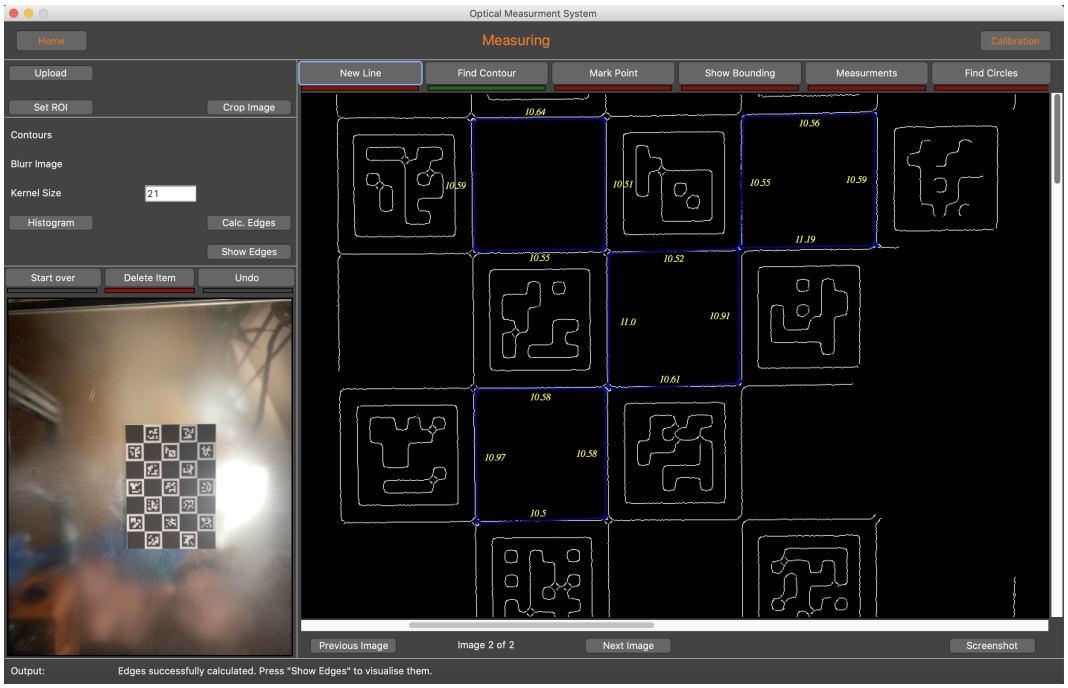
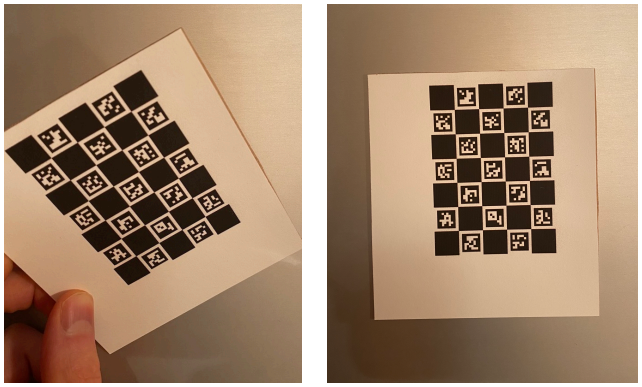



*Abbildung 6.5: Messergebnisse bei starken Reflexionen*

### 6.2.4 Versuch 04

Wie in den vorherigen Kapiteln beschrieben, wird für das Messen in einer zweidimensionalen Ebene eine Kalibrierung eben dieser Ebene benötigt. In diesem Versuch soll demonstriert werden welchen Einfluss das Kalibrieren einer anders orientierten Ebene auf die Messgenauigkeit hat. Abbildung 6.6 zeigt die hier kalibrierte Ebene sowie die Messebene. Aufgrund der unzureichenden Kalibrierung ergibt sich ein relativer Messfehler $\overline{\text{RMS}}_{rel}$ von $16,51\%$. Dieser Fehler ist allerdings nur bedingt aussagekräftig, da die ausgewählten Bildpunkte nicht in die korrekte Ebene projiziert werden und dadurch in keinem Zusammenhang mit den tatsächlichen Punkten auf dem Testobjekt stehen.



*(a)* Kalibrierte Ebene *(b)* Messebene

*Abbildung 6.6: Einfluss einer fehlerhaften Kalibrierung*

### 6.2.5 Versuch 05

Versuch 05 soll nun zeigen, dass jede Orientierung einer zweidimensionalen Ebene als Messebene verwendet werden kann, vorausgesetzt diese wird vorher kalibriert. Die Kamera ist in diesem Versuch schräg zu der horizontalen Messebene orientiert wie in Abbildung 6.7 dargestellt. Abbildung 6.8 zeigt den zweiten Teil dieses Versuches in welchem die Kamera horizontal ausgerichtet ist und die kalibrierte Ebene schräg im Raum steht.

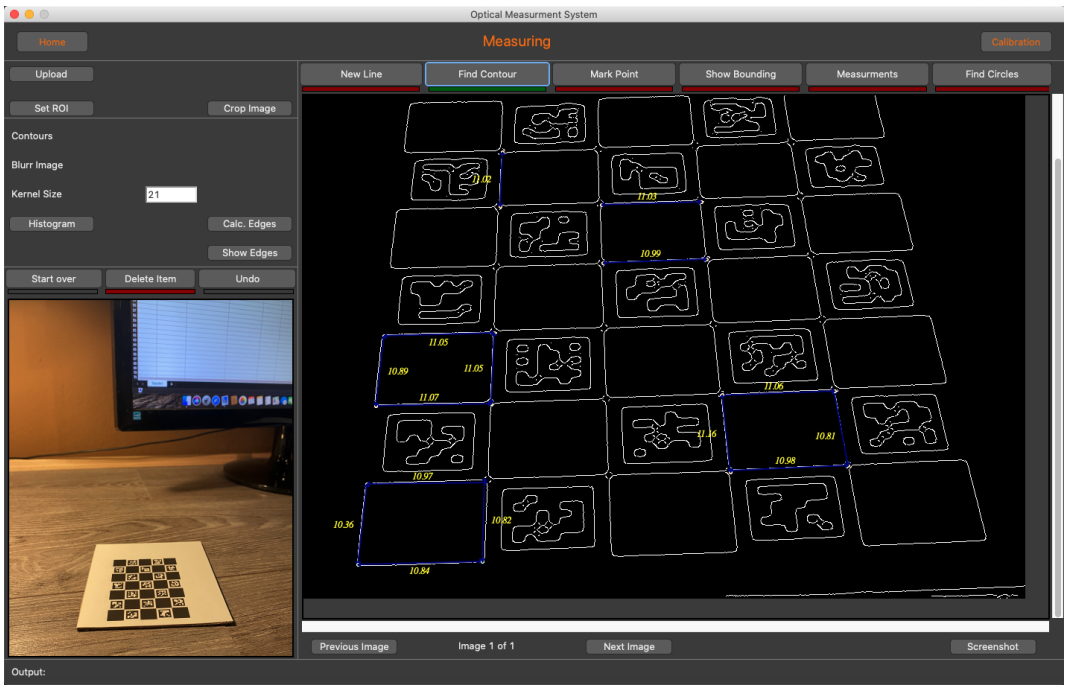



*Abbildung 6.7: Kamera schräg bei horizontaler kalibrierter Ebene. Der Messfehler* $\overline{\text{RMS}}_{rel}$ *beträgt* $1,08\%$

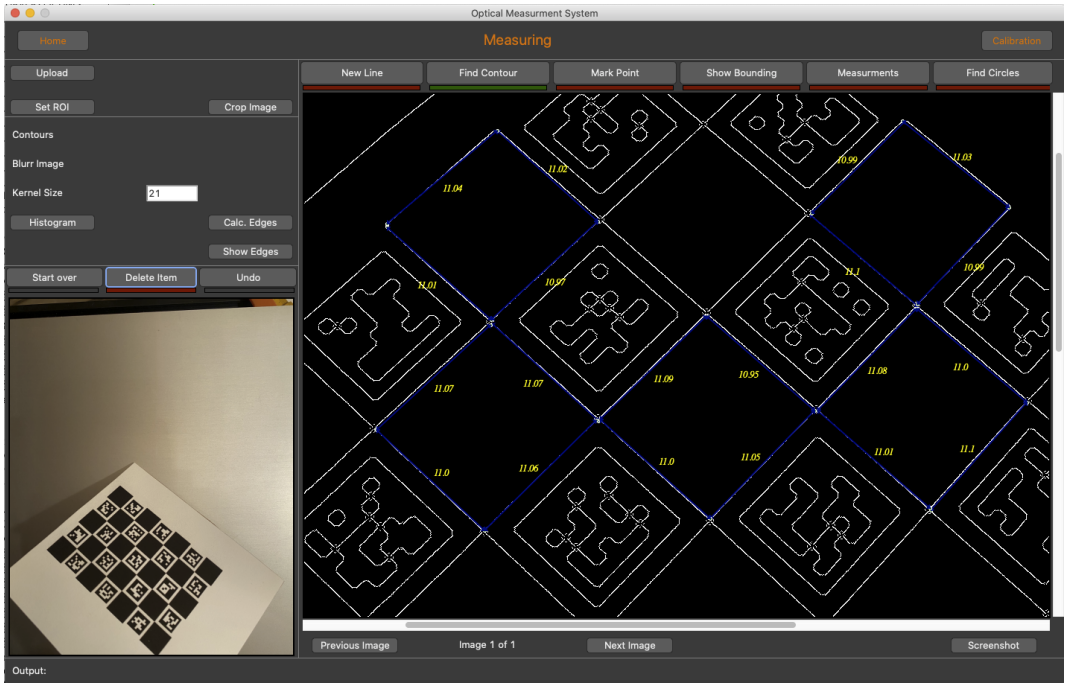
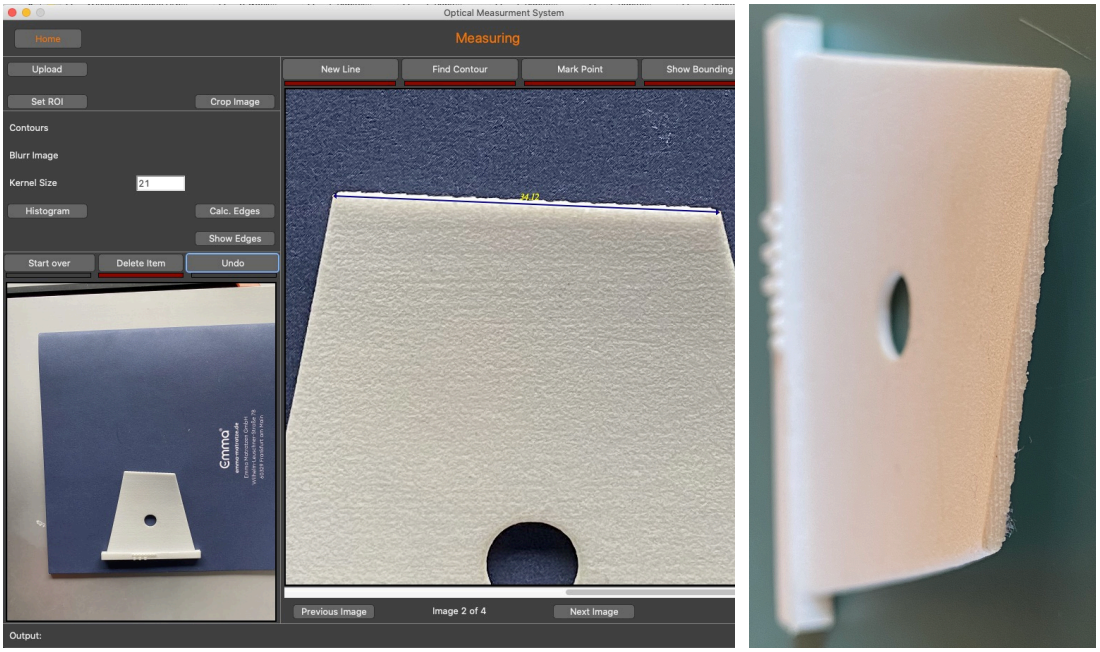



*Abbildung 6.8: Kamera horizontal bei schräger kalibrierter Ebene. Der Messfehler* $\overline{\text{RMS}}_{rel}$ *beträgt* $0,38\%$

### 6.2.6 Versuch 06

Wie Versuch 04 bereits verdeutlichte, sollte eine Messung nur innerhalb der kalibrierten Ebene vorgenommen werden, um realistische Messwerte zu erhalten. Sobald ein Punkt diese Ebene verlässt, verfälscht es die Ergebnisse. Abbildung 6.9 zeigt eine beispielhafte Vermessung eines dreidimensionalen Bauteils. Das Bauteil befindet sich zwar innerhalb der kalibrierten Ebene, doch die gesuchte Länge, in diesem Fall die obere Kante des Objektes, verlässt die kalibrierte Ebene. Die durch die Vermessung erhaltenen Ergebnisse sind dadurch stark fehlerbehaftet. Um ein solches Objekt mit den aktuell verfügbaren Funktionen des Softwaretools korrekt vermessen zu können, müsste vor jeder einzelnen Messung eine neue Ebene kalibriert werden. In diesem Beispiel, um die obere Kante des Objektes korrekt vermessen zu können, müsste das Objekt von Oben aufgenommen werden und die Ebene in welcher die obere Kante liegt kalibriert werden. Falls man an mehr als einer Messung interessiert ist und dadurch mehrere Ebenen einzeln kalibrieren müsste, führt dies schnell zu einem unverhältnismäßigen Aufwand.




*(a)* Zweidimensionale Vermessung der oberen Kante *(b)* Nahaufnahme der oberen Kante

*Abbildung 6.9: Fehlerbehaftete Vermessung eines stark dreidimensional ausgedehnten Objektes*

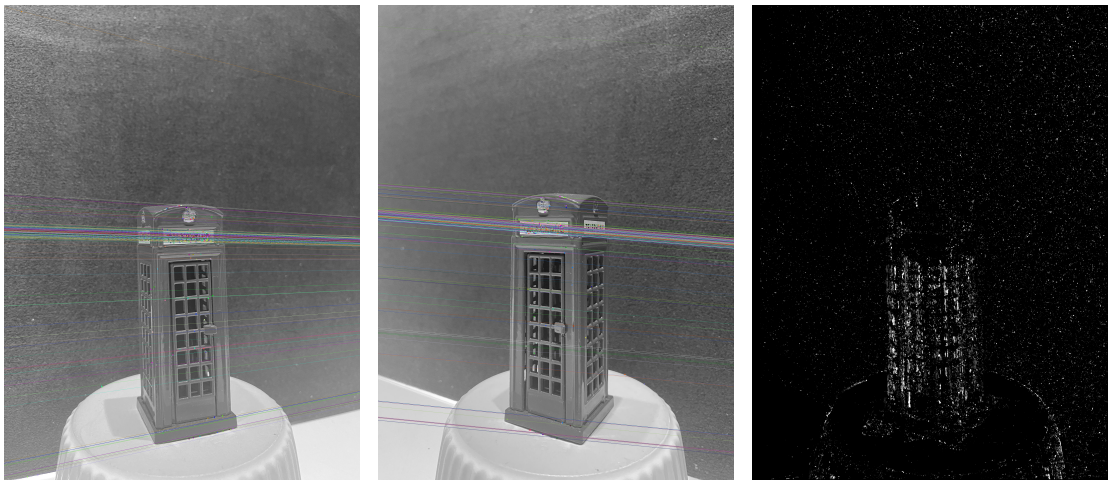
# 7 Ausblick

Die im vorherigen Kapitel durchgeführten Versuche zur Evaluierung des Softwaretools bestätigten, was das Kapitel zur Stereo Rekonstruktion bereits vermuten ließ. Das hier implementierte Vermessungstool ist ausgelegt auf eine Kamera beziehungsweise die Auswertung eines Bildes und dadurch auf Vermessungen in einer zweidimensionalen Ebene. Sobald das Objekt diese Ebene verlässt müsste eine erneute Kalibrierung der Ebene durchgeführt werden. Bei stark dreidimensional ausgedehnten Objekten, kann dies schnell zu einem nicht unerheblichen Aufwand führen. Der realistische Einsatzbereich dieses Softwaretools ist daher die Vermessung von quasi zweidimensionalen Objekten oder einzelnen Ebenen eines dreidimensionalen Objektes. Grundsätzlich spielt es keine Rolle wie komplex das zu untersuchende Objekt ausgedehnt ist oder wie es orientiert ist, solange die gesuchten Maße in einer Ebene liegen und diese vorher kalibriert wurde. Das hier erstellte Interface, welches problemlos unter MacOSX wie Microsoft Windows funktioniert, bildet die Basis eines optischen Vermessungstools. Die elementaren Funktionen der Bildverarbeitung, deren Handhabung im Manual ausführlich beschrieben sind, wurden implementiert und benutzerfreundlich in das Interface integriert. Im Folgenden soll nun auf mögliche Erweiterungen des Programms kurz eingegangen werden.

Der nächste Schritt, um dieses Vermessungstool weiter zu verbessern und damit dessen Einsatzbereich zu vergrößern, wäre sicherlich die Erweiterung von einer zweidimensionalen zu einer dreidimensionalen Vermessung. Zwei Möglichkeiten dies umzusetzen sind denkbar: Einerseits die Effizienzsteigerung der aktuellen Methode sowie die Verwendung mehrerer Bilder beziehungsweise Kameras. Um die bisherige Methode der Ebenenkalibrierung zu verbessern, wären kleine Aufkleber auf dem zu vermessenden Objekt vorstellbar. So könnte man die Ebenen von Interesse, je nach Größe mit einzelnen oder mehreren ChArUco Markern markieren und diese dann im Zuge der eigentlichen Vermessung als Referenz für die Berechnungen heranziehen.

Bei der Verwendung mehrerer Aufnahmen des Objekts von verschieden Richtungen, wird die bereits erwähnte Stereo Rekonstruktion angewendet. Die schon bei der zweidimensionalen Vermessung, hier häufig verwendete Bibliothek `opencv`, stellt auch für die Stereo Rekonstruktion diverse Funktionen zur Verfügung. In Abbildung 7.1 ist die beispielhafte Umsetzung der bereits erwähnten Epipolargeometrie sowie die daraus berechnete Disparitätskarte illustriert. Das zugehörige Flussdiagramm ist in Abbildung 7.2 dargestellt. Die Disparität berechnet sich aus der Differenz der Vektoren der Pixelkoordinaten eines Objektpunktes. Angenommen ein Objektpunkt wird auf die beiden Bildebenen projiziert. $P_L$ auf der linken Bildebene und $P_R$ auf

der rechten Bildebene, so entspricht die Disparität dieses Objektpunktes dem Betrag des Differenzvektors $\overrightarrow{P_R P_L}$. Je größer die Disparität und daher der Versatz eines Pixels desto heller wird er in der Disparitätskarte dargestellt. Helle Pixel befinden sich demnach näher an den beiden Kameras. Die darin enthaltenen Informationen der $z_w$ Koordinaten können für die Berechnungen im dreidimensionalen Raum verwendet werden



*(a)* *(b)* *(c)*

*Abbildung 7.1: Epipolarlinien (a) links, (b) rechts, (c) Disparitätskarte*

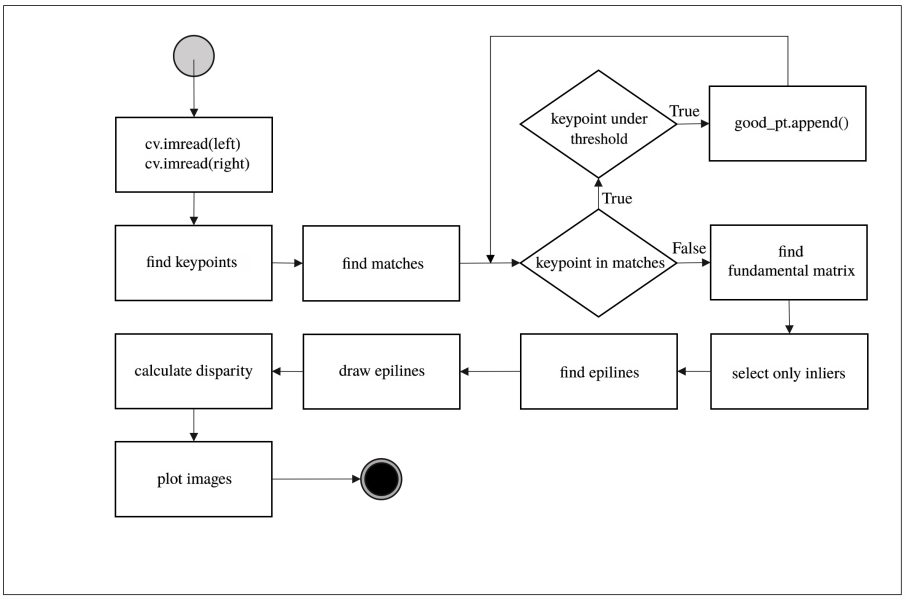



*Abbildung 7.2: Stereo Rekonstruktion und Disparitätsberechnung*

# A Abbildungsverzeichnis

# B Tabellenverzeichnis

# C Literaturverzeichnis


BERND, J. (1996). „Technische Bildverarbeitung - Maschinelles Sehen“. In:

*Camera Calibration* (2019). URL: https://docs.opencv.org/4.1.0/dc/dbb/tutorial_py_calibration.html (besucht am 26. 10. 2020).

CANNY, J. (Nov. 1986). „A Computational Approach to Edge Detection“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8.6, S. 679–698. ISSN: 1939-3539. DOI: 10.1109/TPAMI.1986.4767851.

CARBONELLE, P. (2020). *PYPL PopularitY of Programming Language*. URL: http://pypl.github.io/PYPL.html (besucht am 22. 10. 2020).

CHRISTIAN, D. (2011). *Industrielle Bildverarbeitung*. Heidelberg, GERMANY: Springer.

CORKE, P. (2017). *Summary of image geometry*. URL: https://robotacademy.net.au/lesson/summary-of-image-geometry/ (besucht am 21. 10. 2020).

DERICHE, R. (1987). „Using Canny's criteria to derive a recursively implemented optimal edge detector“. In: *International Journal of Computer Vision* 1.2, S. 167–187. DOI: 10.1007/BF00123164. URL: https://doi.org/10.1007/BF00123164.

GRUPP, M. (2016). *Industrielle Bildverarbeitung wird Schlüsseltechnologie für Industrie 4.0*. URL: https://quality-engineering.industrie.de/termine-veranstaltungen/messe-vision/der-blick-auf-die-wirtschaftlichkeit/ (besucht am 10. 10. 2020).

JAKOB (2018a). *Calibration Best Practices*. URL: https://calib.io/blogs/knowledge-base/calibration-best-practices (besucht am 26. 10. 2020).

– (2018b). *Calibration Patterns Explained*. URL: https://calib.io/blogs/knowledge-base/calibration-patterns-explained (besucht am 09. 10. 2020).

*Machine Vision Germany: record level maintained* (2018). URL: https://ibv.vdma.org/documents/256550/27036512/VDMA_MV_record_level_maintained_1541500691088.jpg/7ea4ce9b-4cba-05f6-1b31-3e8b51ee9f76 (besucht am 09. 10. 2020).

NINGTHOUJAM, J. und K. NONGMEIKAPAM (Nov. 2019). „Stereo System based Distance Calculation of an Object in Image“. In:

NORBERT, B. (2008). *Handbuch zur industriellen Bildverarbeitung*. Stuttgart, GERMANY: Fraunhofer-IRB-Verl.

*Pattern Generator* (2020). URL: https://calib.io/pages/camera-calibration-pattern-generator (besucht am 09. 10. 2020).

STEGER, C., M. ULRICH und C. WIEDEMANN (2018). *Machine Vision Algorithms and Applications*. Newark, GERMANY: John Wiley & Sons, Incorporated.

*Stereo Camera Calibrator App* (2020). URL: https://de.mathworks.com/help/vision/ug/stereo-camera-calibrator-app.html (besucht am 26. 10. 2020).

STRATHEARN, M. (2012). *Perspective Camera*. URL: https://docs.arnoldrenderer.com/x/0wtkAg (besucht am 09. 10. 2020).

THAKUR, R., R. YADAV und L. GUPTA (Aug. 2019). „State-of-Art Analysis of Image Denoising Methods using Convolutional Neural Networks“. In: *IET Image Processing* 13. DOI: 10.1049/iet-ipr.2019.0157.

YOUSIF, K., A. BAB-HADIASHAR und R. HOSEINNEZHAD (Nov. 2015). „An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics“. In: *Intelligent Industrial Systems* 1. DOI: 10.1007/s40903-015-0032-7.