

# Liste der noch zu erledigenden Punkte

abstract schreiben . . . . .	V
einleitung schreiben . . . . .	1
theorie einleitung schreiben . . . . .	3
ist noch nicht fertig . . . . .	9
hier muss noch was getan werden . . . . .	10
ist noch etwas unklar . . . . .	12
überarbeiten . . . . .	21
das ist zu knapp . . . . .	32
analyse einleitung schreiben . . . . .	37
wie kommt das hier rein? . . . . .	46
ergänzen . . . . .	47
Literatur checken, ins besondere seitenangaben . . . . .	47
muss das hier überhaupt rein? . . . . .	49





# ENTZERRUNG VON KEGELOBERFLÄCHEN AUS EINER EINKAMERAANSICHT BASIEREND AUF PROJEKTIVER GEOMETRIE

BACHELORARBEIT  
zur Erlangung des akademischen Grades  
BACHELOR OF SCIENCE

Westfälische Wilhelms-Universität Münster  
Fachbereich Mathematik und Informatik  
Institut für Informatik

Betreuung:  
*Dimitri Berh*

Erstgutachten:  
*Prof. Dr. Xiaoyi Jiang*

Zweitgutachten:  
*Prof. Dr. Klaus Hinrichs*

Eingereicht von:  
*Lars Haalck*

Münster, September 2016



## Zusammenfassung

---

abstract  
schreiben



# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>1</b>
<b>2. Theoretische Grundlagen</b>	<b>3</b>
2.1. Kegel . . . . .	3
2.2. Lineares Ausgleichsproblem . . . . .	7
2.3. Kamerakalibrierung und projektive Geometrie . . . . .	9
2.4. Blob-Detektor . . . . .	11
2.5. Canny . . . . .	12
2.6. Hough-Transformation . . . . .	12
2.7. RANSAC . . . . .	13
2.8. Ellipsen . . . . .	13
2.8.1. allgemein . . . . .	13
2.8.2. Abstand: Punkt zu Ellipse . . . . .	17
<b>3. Methodik</b>	<b>21</b>
3.1. Kalibrierungsmuster . . . . .	21
3.1.1. Anzahl der Samples . . . . .	23
3.2. Intrinsische Kamerakalibrierung . . . . .	23
3.3. Detektion der charakteristischen Punkte . . . . .	23
3.4. Ellipsen-Detektion . . . . .	24
3.5. Zuordnung der Punkte . . . . .	28
3.6. Weltkoordinaten bestimmen . . . . .	29
3.7. Entfaltung . . . . .	30
3.7.1. Vorwärtsentfaltung . . . . .	30
3.7.2. Rückwärtsentfaltung . . . . .	32
<b>4. Implementierung</b>	<b>35</b>
<b>5. Analyse</b>	<b>37</b>
5.1. Vergleich Vorwärtsentfaltung und Rückwärtsentfaltung . . . . .	37
5.2. Einfluss der intrinsischen Kalibrierung . . . . .	39
5.3. Einfluss der Rotation der Kamera . . . . .	39
5.4. Laufzeit der Entfaltung . . . . .	40
5.5. Evaluierung des RANSAC zur Ellipsendetektion . . . . .	41

## **INHALTSVERZEICHNIS**

---

<b>6. Fazit und Ausblick</b>	<b>45</b>
6.1. Parallelisierung . . . . .	45
6.2. Verbesserung der Linien-Detektion . . . . .	45
6.3. Verbesserung der Vorwärtsentfaltung . . . . .	45
6.4. Verbesserung der Rückwärtsentfaltung . . . . .	45
6.5. andere Ansätze . . . . .	46
<b>A. Ergebnisse</b>	<b>49</b>
<b>Abbildungsverzeichnis</b>	<b>51</b>

# 1. Einleitung

hier setup mit bild?

einleitung  
schreiben



## 2. Theoretische Grundlagen

In diesem Kapitel werden einige Grundlagen für die folgende Kapitel eingeführt.

Wir benutzen ein linkshändiges Koordinatensystem.

theorie  
einleitung  
schreiben

### 2.1. Kegel

#### 2.1.1 Definition (Kegel)

Ein Kegel ist ein geometrischer Körper, der durch eine beliebige Grundfläche, sowie einen Punkt definiert wird. Ein Kegel mit Kreis als Grundfläche wird als Kreiskegel bezeichnet. Liegt die Kegelspitze auf einer Geraden durch die Normale der Grundebene, bezeichnen wir den Kegel als gerade.

In der weiteren Arbeit betrachten wir nur gerade Kreiskegel.  $S$  bezeichnet hierbei die Seitenhöhe und ist definiert durch  $S = \sqrt{H^2 + R^2}$  (siehe Abbildung 2.1).

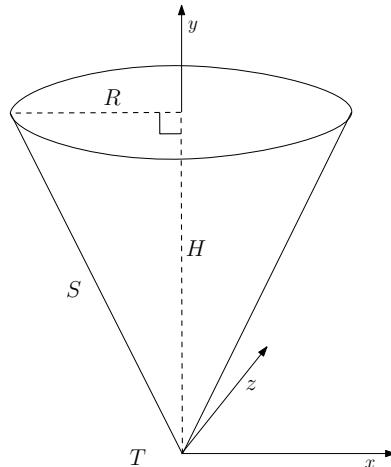


Abbildung 2.1.: Gerader Kreiskegel

Ein Kegel mit Spitze  $T(0, 0, 0)$ , Radius  $R$  und Höhe  $H$  kann parametrisch beschrieben werden als:

$$\begin{aligned}x &= \frac{u}{h}R \cos\theta \\y &= u \\z &= \frac{u}{h}R \sin\theta\end{aligned}\tag{2.1}$$

mit  $u \in [0, H]$  und  $\theta \in [0, 2\pi)$

### 2.1.2 Definition (Kegelstumpf und Ergänzungskegel)

Ein Kegelstumpf entsteht als Schnitt eines geraden Kreiskegels mit einer zur Grundfläche parallelen Ebene (siehe Abbildung 2.2). Das Stück von Grundfläche zur Schnittfläche nennen wir Kegelstumpf. Die Differenz zum eigentlichen Kegel wird als Ergänzungskegel bezeichnet.

$H, R, S$  bleiben die Angaben des gesamten Kegels. Hinzu kommen  $h, r, s$  als Angaben des Ergänzungskegels. Die Höhe, sowie die Seitenhöhe des Kegelstumpfs werden durch die Differenzen  $\Delta S = S - s$ ,  $\Delta H = H - h$  charakterisiert (siehe Abbildung 2.3).

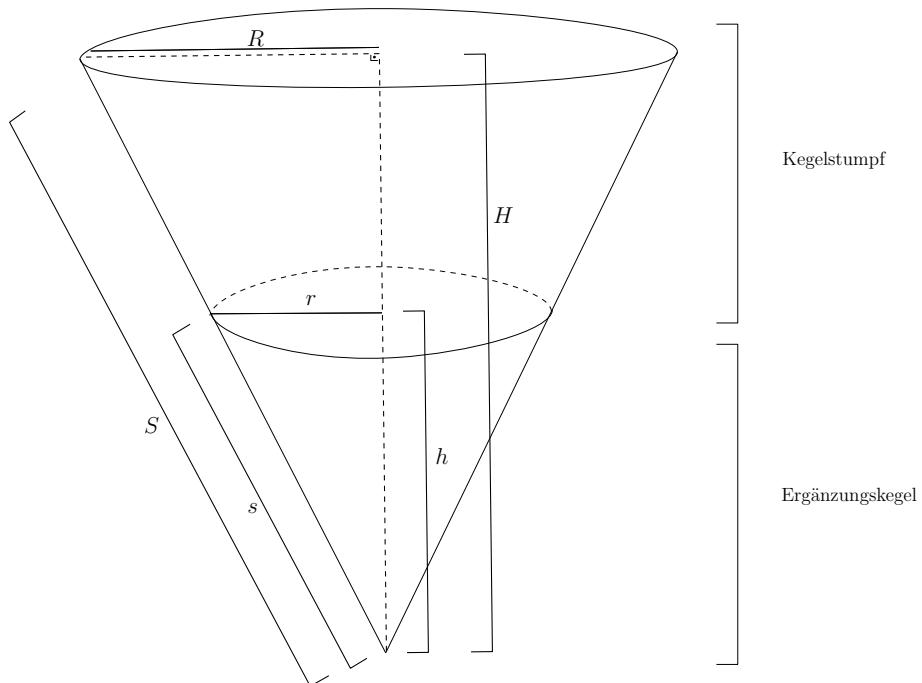


Abbildung 2.2.: Kegelstumpf und Ergänzungskegel

Analog zum Kreiskegel definieren wir einen Kegelstumpf mit Zentrum  $(0, 0)$  des kleineren Kreises durch folgende Parametrisierung:

$$\begin{aligned} x &= \left(r + \frac{u}{\Delta H}(R - r)\right) \cos\theta \\ y &= u \\ z &= \left(r + \frac{u}{\Delta H}(R - r)\right) \sin\theta \end{aligned} \tag{2.2}$$

mit  $u \in [0, \Delta H]$  und  $\theta \in [0, 2\pi)$

Die Mantelfläche des Kegelstumpfes aus Abbildung 2.4 kann dann parametrisch beschrie-

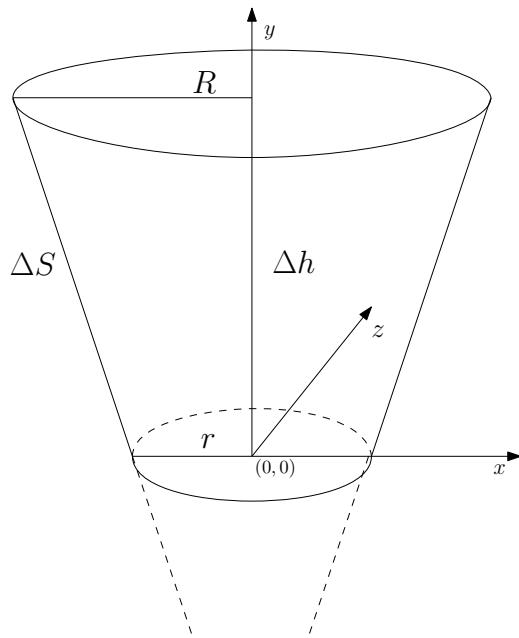


Abbildung 2.3.: Kegelstumpf

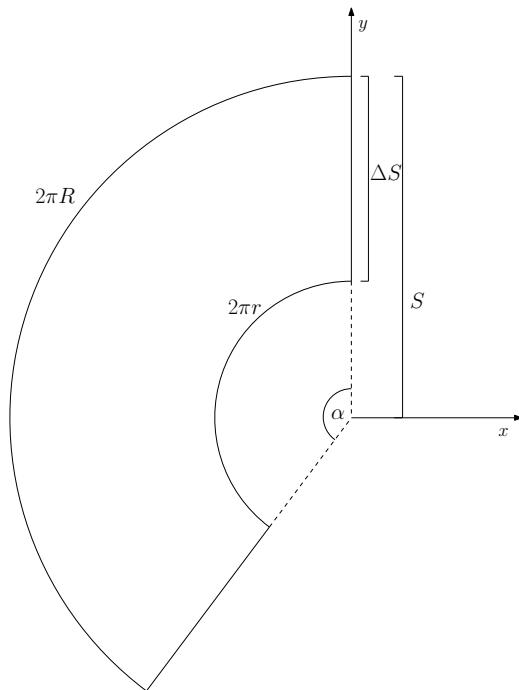


Abbildung 2.4.: Kegelmantelfläche

ben werden als:

$$\begin{aligned} x &= -(s + \frac{u}{\Delta H}(S - s)) \sin\phi \\ y &= (s + \frac{u}{\Delta H}(S - s)) \cos\phi \end{aligned} \tag{2.3}$$

mit  $u \in [0, \Delta H]$  und  $\phi \in [0, \alpha] \subseteq [0, 2\pi]$  mit  $\alpha S = 2\pi R \implies \alpha = 2\pi \frac{R}{S}$

Ein Punkt auf der Oberfläche des Kegelstumpfs kann eindeutig einem Punkt auf der Mantelfläche (und umgekehrt) zugeordnet werden. Dazu konstruieren wir folgende Abbildung und ihr Inverses:

Sei ein Punkt  $C(x, y, z)$  auf der Oberfläche des Kegelstumpfs gegeben. Wir wissen aus der parametrischen Form 2.2, dass  $C$  die Form

$$C(x, y, z) = (r + \frac{u}{\Delta H}(R - r)) \cos\theta, u, (r + \frac{u}{\Delta H}(R - r)) \sin\theta$$

für ein  $u \in [0, \Delta H]$  und  $\theta \in [0, 2\pi]$  besitzt.

Aus der  $y$ -Koordinate kann man als die Höhe ablesen und somit den Radius in der Mantelfläche als lineare Interpolation zwischen  $s$  und  $S$  bestimmen (siehe Abbildung 2.5). Wir definieren uns hierfür eine Hilfsfunktion

$$\Sigma(y) := s + \frac{y}{\Delta H}(S - s) \quad (2.4)$$

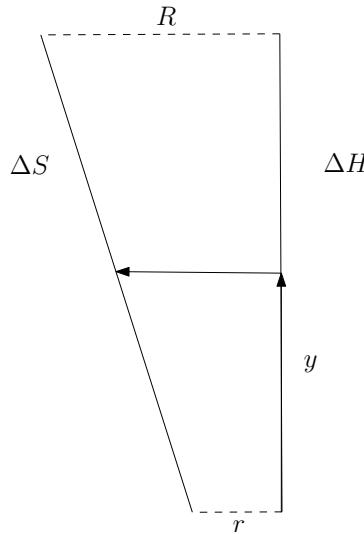


Abbildung 2.5.: Abbildung der Kegelstumpfhöhe auf die Seitenhöhe

Da  $R, r, \Delta H$  und nun auch die Höhe bekannt sind, kann man den Winkel  $\theta$  im Kegelstumpf einfach ausrechnen. Anschließend muss dieser noch mit  $\frac{R}{S}$  multipliziert werden um ihn auf  $[0, \alpha]$  zu skalieren (siehe 2.3). Auch hierfür definieren wir eine Hilfsfunktion:

$$\Phi(x, y, z) := \frac{R}{S} \text{atan2} \left( \frac{z}{r + \frac{y}{\Delta H}(R - r)}, \frac{x}{r + \frac{y}{\Delta H}(R - r)} \right),$$

wobei wir `atan2` benutzen um den Winkel im richtigen Quadranten, also in  $[0, 2\pi)$ , bestimmen zu können.

Mit diesen beiden Hilfsfunktionen und 2.3 ergibt sich insgesamt:

$$\begin{aligned} \Psi: [r, R] \times [0, \Delta H] \times [r, R] &\rightarrow [s, S] \times [s, S] \\ \begin{pmatrix} x \\ y \\ z \end{pmatrix} &\mapsto \begin{pmatrix} -\Sigma(y) \sin \Phi(x, y, z) \\ \Sigma(y) \cos \Phi(x, y, z) \end{pmatrix} \end{aligned} \quad (2.5)$$

Analog lässt die sich Umkehrabbildung konstruieren:

Sei ein Punkt  $L(x, y)$  auf der Mantelfläche des Kegelstumpfs gegeben. Aus der parametrischen Form 2.3 ergibt sich

$$L(x, y) = \left( -\left(s + \frac{u}{\Delta H}(S - s)\right) \sin \phi, \left(s + \frac{u}{\Delta H}(S - s)\right) \cos \phi \right)$$

für ein passendes  $u \in [0, \Delta H]$  und  $\phi \in [0, \alpha] \subseteq [0, 2\pi]$ .

Da  $L(x, y)$  in Polarkoordinaten gegeben ist, lässt sich der Radius durch  $\sqrt{x^2 + y^2}$  bestimmen. Wir können den Winkel  $\phi$  mit inverser Skalierung also analog zu oben durch folgende Hilfsfunktion bestimmen:

$$\Theta(x, y) := \frac{S}{R} \operatorname{atan2}\left(\frac{x}{-\sqrt{x^2 + y^2}}, \frac{z}{\sqrt{x^2 + y^2}}\right)$$

Die Höhe im Kegel und somit der Radius lässt sich nun gewissermaßen als Umkehrabbildung zu 2.4 bestimmen:

$$H(x, y) := \frac{\left(\sqrt{x^2 + y^2}\right) - s}{S - s} \Delta H$$

Insgesamt ergibt sich:

$$\begin{aligned} \Psi^{-1}: [s, S] \times [s, S] &\rightarrow [r, R] \times [0, \Delta H] \times [r, R] \\ \begin{pmatrix} x \\ y \end{pmatrix} &\mapsto \begin{pmatrix} \left(r + \frac{H(x, y)}{\Delta H}(R - r)\right) \cos(\Theta(x, y)) \\ H(x, y) \\ \left(r + \frac{H(x, y)}{\Delta H}(R - r)\right) \sin(\Theta(x, y)) \end{pmatrix} \end{aligned} \quad (2.6)$$

## 2.2. Lineares Ausgleichsproblem

Gegeben seien  $m$  Messdaten  $(x_1, y_1), \dots, (x_m, y_m)$ , sowie eine lineare Funktion  $\varphi(x) = \sum_{i=1}^n c_i \varphi_i(x)$  in  $x$  mit  $c_1, \dots, c_n \in \mathbb{R}$ ,  $n \leq m$  und  $n, m \in \mathbb{N}$ .

Gesucht ist eine Lösung der  $c_i$  welche die mittlere Abweichung, also

$$\Delta_2 = \min_{c_1, \dots, c_n \in \mathbb{R}} \left( \sum_{j=1}^m (y_j - \varphi(x_j))^2 \right) = \min_{c_1, \dots, c_n \in \mathbb{R}} \left( \sum_{j=1}^m \left( \sum_{i=1}^n y_j - (c_i \varphi_i(x_j)) \right)^2 \right)$$

minimiert [Sto07].

Diese Problem wird als lineares Ausgleichsproblem oder aber auch als Methode der kleinsten Quadrate bezeichnet und lässt sich auch wie folgt formulieren:

$$\min_{c \in \mathbb{R}^n} \|y - Ac\|_2,$$

mit

$$\begin{aligned} c &= (c_1, \dots, c_n)^T \in \mathbb{R}^n \\ x &= (x_1, \dots, x_m)^T \in \mathbb{R}^m \\ y &= (y_1, \dots, y_m)^T \in \mathbb{R}^m \\ A &= (a_{ij}) \in \mathbb{R}^{m \times n} \quad \text{mit} \quad a_{ij} = \varphi_i(x_j) \end{aligned}$$

Das Problem lässt sich auch mit Hilfe der äquivalenten sogenannten Normalengleichung

$$(A^T A)c = A^T y$$

lösen, wobei mindestens eine Lösung existiert und bei mehreren Lösungen, die Lösung mit kleinster 2-Norm ausgewählt wird.

Gilt darüber hinaus  $\text{rang}(A) = n$  so ist  $A^T A$  invertierbar und es gilt

$$c = \underbrace{(A^T A)^{-1} A^T}_{{A}^+} y \tag{2.7}$$

und  $c$  ist die eindeutige Lösung mit kleinster 2-Norm.

$A^+$  bezeichnet man auch als Pseudoinverse von  $A$  und lässt sich durch eine Singulärwertzerlegung (SVD) bestimmen [Sto11]. Die Matrizen  $A \in \mathbb{R}^{m \times n}$  und  $A^+ \in \mathbb{R}^{n \times m}$  können geschrieben werden als:

$$\begin{aligned} A &= U \Sigma V^T \\ A^+ &= V \Sigma^+ U^T \end{aligned}$$

mit orthogonalen Matrizen  $U \in \mathbb{R}^{m \times m}$  und  $V \in \mathbb{R}^{n \times n}$  und einer Diagonalmatrix  $\Sigma \in \mathbb{R}^{m \times n}$  mit den absteigend sortierten Singulärwerten auf der Diagonale (aufgefüllt mit Nullen), sowie  $\Sigma^+ \in \mathbb{R}^{n \times m}$  mit den jeweiligen reziproken Singulärwerten.

Die Lösung des Ausgleichsproblems kann also mittels SVD angeben werden als:

$$c = V \Sigma^+ U^T y$$

## 2.3. Kamerakalibrierung und projektive Geometrie

ist noch  
nicht fertig

### 2.3.1 Definition

Kamerakalibrierung ist ein Verfahren, bei dem die interne Kamerageometrie und optischen Eigenschaften (intrinsische Parameter) und/oder die 3D-Position und Orientierung der Bildebene relativ zu einem Weltkoordinatensystem (extrinsische Parameter) bestimmt werden [Tsa87].

Um die Parameter einer Kamera korrekt beschreiben zu können, ist ein Kameramodell notwendig. Das wohl bekannteste Kameramodell ist das Lochkamera-Modell. Die Lichtstrahlen der Szene gelangen dabei durch eine kleine Öffnung in die Kamera (siehe dazu Abbildung 2.6).

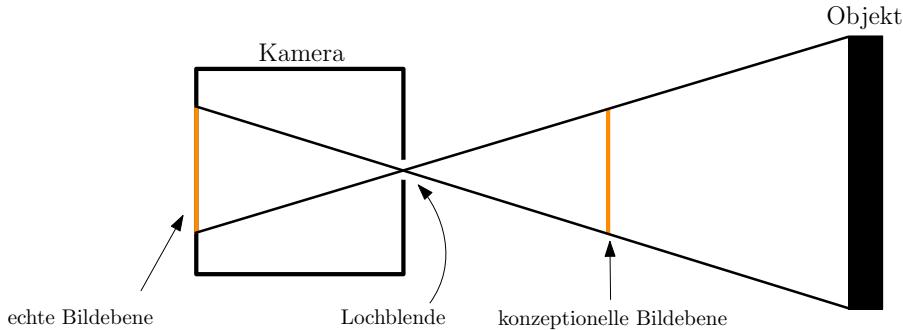


Abbildung 2.6.: Lochkameramodell

Das Bild das an der Rückseite der Kamera entsteht ist dabei um  $180^\circ$  gedreht. Damit man diese Rotation nicht betrachten muss, kann man die Bildebene virtuell vor die Lochblende setzen. Da sich der Abstand zur Blende nicht ändert, ändern sich, außer der ersparten Rotation, auch die optischen Eigenschaften nicht.

Kamerakalibrierung wird benötigt, um eine Beziehung zwischen Punkten in 3D im Weltkoordinatensystem und den Punkten auf der Bildebene herzustellen. Konkret suchen wir eine projektive Abbildung

$$\begin{pmatrix} wu \\ wv \\ w \end{pmatrix} = \underbrace{\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{pmatrix}}_{=:A} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \quad (2.8)$$

die einen Punkt  $P = (x, y, z)$  in homogenen Koordinaten  $\tilde{P} = (x, y, z, 1)$  auf einen Punkt  $\tilde{C} = (wu, wv, w)$  beziehungsweise nach der perspektivischen Division  $C = (u, v)$  auf die Bildebene abbildet, wobei die  $a_{ij}$  von den intrinsischen und extrinsischen Parametern der Kamera abhängen [HS97].

Man kann leicht zeigen, dass für  $A$  gilt

$$A = \lambda \cdot \begin{pmatrix} 1 & 0 & u_0 \\ 0 & 1 & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix} = \lambda \underbrace{\begin{pmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{pmatrix}}_{=:V} \begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix},$$

wobei  $\lambda$  ein beliebiger Skalierungsfaktor ist.  $V$  beschreibt die intrinsischen und  $R$  und  $T$  die extrinsischen Parameter (siehe [HS97]).  $f$  ist hierbei die Brennweite;  $u_0$  und  $v_0$  die Bildmitte.

Zu gegebenen Punktenpaaren  $(x_k, y_k)$ ,  $k = 1, \dots, m$  kann diese Matrix wie folgt gelöst werden:

$$\begin{aligned} u_k &= \frac{a_{11}x_k + a_{12}y_k + a_{13}z_k + a_{14}}{a_{31}x_k + a_{32}y_k + a_{33}z_k + a_{34}} \\ v_k &= \frac{a_{21}x_k + a_{22}y_k + a_{23}z_k + a_{24}}{a_{31}x_k + a_{32}y_k + a_{33}z_k + a_{34}} \end{aligned}$$

$$\underbrace{\begin{pmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1z_1 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1z_1 \\ \vdots & \vdots \\ x_m & y_m & z_m & 1 & 0 & 0 & 0 & 0 & -u_mx_m & -u_my_m & -u_mz_m \\ 0 & 0 & 0 & 0 & x_m & y_m & z_m & 1 & -v_mx_m & -v_my_m & -v_mz_m \end{pmatrix}}_{=:M} = \begin{pmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{14} \\ a_{21} \\ a_{22} \\ a_{23} \\ a_{24} \\ a_{31} \\ a_{32} \\ a_{33} \end{pmatrix} = \begin{pmatrix} u_1 \\ v_1 \\ \vdots \\ u_m \\ v_m \end{pmatrix}, \quad (2.9)$$

wobei  $a_{34}$  auf eins skaliert wird. Es handelt sich hierbei um ein überbestimmtes lineares Gleichungssystem, was mittels der Methode der kleinsten Quadrate (siehe Kapitel 2.2) gelöst werden kann. Diese Verfahren wird auch als *Direct Linear Transformation*(DLT) bezeichnet.

Das beschriebene Verfahren vernachlässigt dabei Linsenverzerrungen. Solche Verzerrungen entstehen entweder als Produktionsfehler günstiger Linsen, oder bewusst beispielsweise bei Weitwinkelkameras. Verzerrungen können in der Regel nicht linear modelliert werden. Der Ansatz über DLT funktioniert hier dementsprechend nicht.

DLT vernachlässigt alle Verzerrungen. Um diese zu kompensieren benutze die 4 schritte von tsai

Die vier Schritte der Kamerakalibrierung sind nach Tsai[Tsa87]:

1. Rotation und Translation des Weltkoordinatensystems, um es in das Bildkoordinaten-

tensystem zu überführen.

2. Perspektivische Division und Skalierung mit der Brennweite

3. Bestimmen der Verzerrungskoeffizienten  $\kappa_1$  und  $\kappa_2$

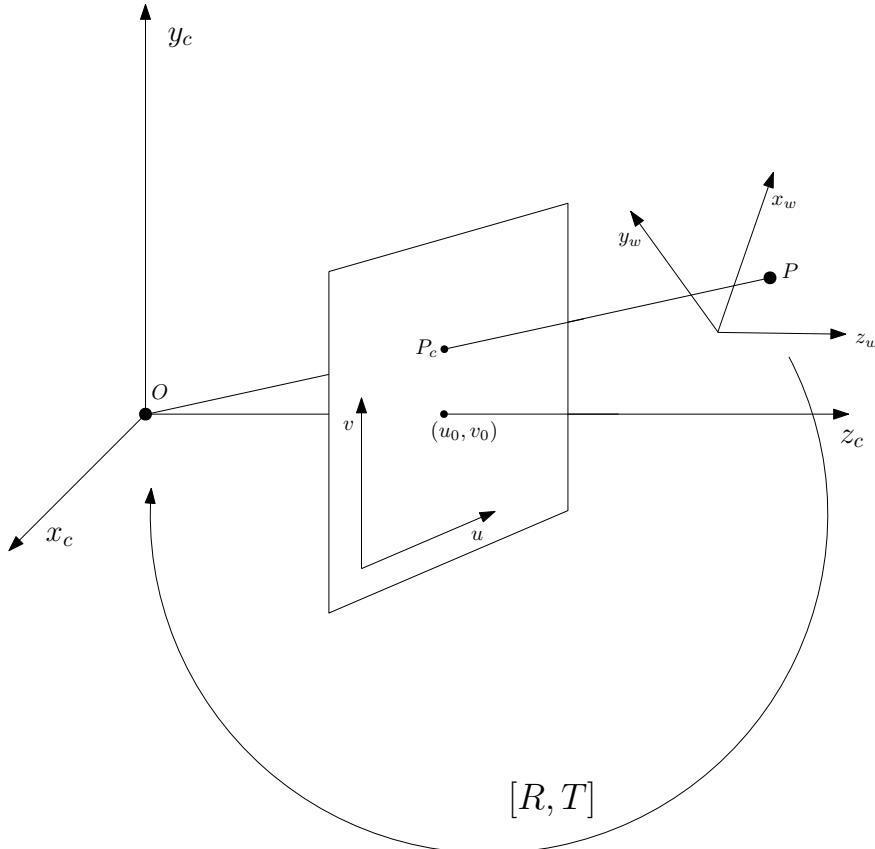


Abbildung 2.7.: Projektion eines Punktes  $P$  im Weltkoordinatensystem  $(x_w, y_w, z_w)$  auf die Bildebene im Kamerakoordinatensystem  $(x_c, y_c, z_c)$

Selfcalibrating vs bla bla

## 2.4. Blob-Detektor

### 2.4.1 Definition (Blob)

Ein Blob ist eine glatte zusammenhängende Region in einem Bild, die sich farblich von ihrer Umgebung abhebt [Lin93].

Ein Blob-Detektor ist also ein Verfahren zum Detektieren solcher Blobs. Die gefundenen Blobs können anschließend nach verschiedenen Kriterien, wie beispielsweise Größe, Farbe, Konvexität oder Rundheit gefiltert werden.

## 2.5. Canny

Der Canny-Algorithmus[Can86] ist ein Verfahren zur Kantendetektion. Im Gegensatz zu anderen Verfahren wie Sobel, oder Prewitt, versucht Canny die Fehlerrate der Kantendetektion minimal zu halten. Es sollten nicht mehr Kanten zurückgegeben werden, als auf dem Bild zu sehen und es sollten auch keine Kanten übersehen werden. Darüber hinaus markiert Canny die Kanten möglichst exakt, minimiert also die Distanz eines markierten Punktes zur eigentlichen Kante. Zuletzt gewährleistet Canny außerdem die Eindeutigkeit einer Kante. Das bedeutet, dass eine Kante nicht mehrmals markiert wird.

## 2.6. Hough-Transformation

ist noch et-  
was unklar Hough-Transformation ist ein Verfahren zur Detektion von beliebigen parametrisierbaren Konturen. In dieser Arbeit werden Hough-Transforamtionen benutzt um Geraden zu detektieren.

Eine beliebige zweidimensionale Gerade kann in Polarkoordinaten folgendermaßen implizit ausgedrückt werden:

$$x \cos \phi + y \sin \phi - d = 0, \quad (2.10)$$

wobei  $\phi \in [0, 2\pi)$  der Winkel der Geraden mit der X-Achse und  $d \geq 0$  der Radius, also der euklidische Abstand der Geraden zum Ursprung des Koordinatensystems ist.

Eine Gerade wird somit als ein Punkt  $(d, \phi)$  in den Parameterraum (auch Hough-Raum) abgebildet.

Um eine Gerade eindeutig zu definieren benötigt man, wie auch bei der klassischen Defintionen  $y = mx + b$ , zwei Punkte. Nimmt man nur einen, so lässt sich jedoch die Auswahl von  $\phi$  und  $d$  einschränken. Hat man beispielsweise einen Punkt  $(x_k, y_k)$  gegeben so lässt sich die Gleichung 2.10 nach  $d$  umstellen und man erhält eine sinusförmige Funktion in Abhängigkeit von  $\phi$ .

Um beliebige Geraden detektieren zu können, werden  $\phi$  und  $d$  passen diskretisiert:

$$\begin{aligned} \phi_i &= \phi_{min} + \frac{i}{n} \cdot (\phi_{max} - \phi_{min}) \quad \forall i \in [0, n] \\ d_j &= d_{min} + \frac{j}{m} \cdot (d_{max} - d_{min}) \quad \forall j \in [0, m] \end{aligned}$$

Es wird nun ein Akkumulator  $\mathcal{H}(\phi, d)$  für alle  $\phi_i$  und  $d_j$  auf null gesetzt.

Als Nächstes wird ein Kantentbild mittels Canny erzeugt und jene Pixel  $(x_k, y_k)$  betrachtet, die nicht null sind. Zu einem gegebenen Pixel wir nun für alle diskreten Winkel  $\phi_i$  ein  $d_i$  ausgerechnet und entsprechend im Akkumulator  $\mathcal{H}$  der Wert an der Stelle  $(\phi_i, d_i)$  erhöht. Am Ende des Verfahrens such man im Akkumulator nach Häufungspunkten. Jeder Häufungspunkt steht dort für eine Gerade.

## 2.7. RANSAC

*Random Sample Consensus* (RANSAC) [FB81] ist ein nicht-deterministisches robustes Verfahren zur Parameterschätzung eines Modells bei einer, möglicherweise durch starke Ausreißer, gestörten Messreihe. Im Gegensatz zu Verfahren, wie der Methode der kleinsten Quadrate, die versuchen eine optimale Lösung für alle Messdaten zu bestimmen, nutzt RANSAC nur eine Teilmenge der Messreihe.

RANSAC wählt aus der Menge der Messdaten wiederholt zufällig die minimale Anzahl Messdaten aus, die nötig sind um das Modell eindeutig zu beschreiben und prüft dann, wie gut das geschätzte Modell die restlichen Messdaten beschreibt. Die Güte des Modells wird im Allgemeinen durch ein Distanzmaß, wie zum Beispiel der euklidische Abstand, berechnet. Hat ein Messdatum eine vorher definierte Maximaldistanz zum geschätzten Modell nicht überschritten, wird es ins sogenannte Consensus Set des Modells aufgenommen. Das Modell mit dem größten Consensus Set wird schließlich ausgewählt.

Die Anzahl der Iterationen, die mindestens notwendig sind, um mit einer Wahrscheinlichkeit von  $p \in [0, 1)$ , bei einem relativen Ausreißeranteil von  $\epsilon \in [0, 1)$  und einer Anzahl von  $k$  Daten, um das Modell eindeutig zu beschreiben, mindestens einmal eine ausreißerfreie Teilmenge der Messreihe zu erhalten, lässt sich berechnen mit:

$$n_{min} = \frac{\log(1-p)}{\log\left(1 - (1-\epsilon)^k\right)} \quad (2.11)$$

## 2.8. Ellipsen

### 2.8.1. allgemein

#### 2.8.1 Definition (Ellipse)

Eine Ellipse wird beschrieben durch diejenigen Punkte, dessen Summe der Abstände zu zwei gegebenen Punkten  $f_1$  und  $f_2$  (Brennpunkte) konstant sind. Der Mittelpunkt der Verbindungslinie der beiden Brennpunkte wird als Zentrum der Ellipse bezeichnet (siehe Abbildung 2.8). Als Hauptachse  $a$  bezeichnen wir die Verbindungslinie vom Mittelpunkt durch einen der Brennpunkte bis zum Scheitel  $V_1$  (bzw.  $V_3$ ) der Ellipse. Die zu ihr rechtwinklige Verbindungslinie durch den Mittelpunkt bis zu einem der anderen Scheitelpunkte ( $V_2$  oder  $V_4$ ) nennen wir Nebenachse  $b$ .

In ihrer einfachsten Form liegt die Ellipse im Zentrum des Koordinatensystems und ihre Haupt- und Nebenachse  $a$  und  $b$  sind achsenausgerichtet. Das heißt ihre Hauptachse liegt auf der X-Achse und ihre Nebenachse auf der Y-Achse. Sie kann dann in der impliziten Form

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad (2.12)$$

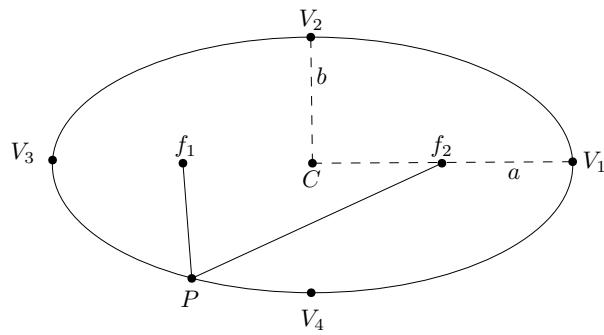


Abbildung 2.8.: Ellipse mit Brennpunkten  $f_1, f_2$ , Zentrum  $C$ , Hauptachse  $a$ , Nebenachse  $b$  und Scheitelpunkten  $V_1$  und  $V_2$

beschrieben werden.

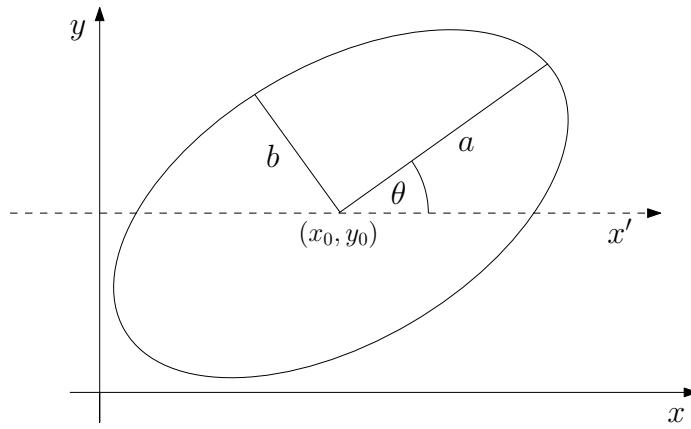


Abbildung 2.9.: Ellipse mit Zentrum  $(x_0, y_0)$ , Hauptachse  $a$ , Nebenachse  $b$ , sowie Drehwinkel  $\theta$

Befindet sich die Ellipse nicht im Ursprung so muss eine Verschiebung beziehungsweise bei einer Rotation ein Drehwinkel (siehe Abbildung 2.9) ergänzt werden.

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1 \quad (2.13)$$

$$\frac{((x - x_0) \cos \theta + (y - y_0) \sin \theta)^2}{a^2} + \frac{((x - x_0) \sin \theta - (y - y_0) \cos \theta)^2}{b^2} = 1 \quad (2.14)$$

mit Ellipsenzentrum  $(x_0, y_0) \in \mathbb{R}^2$ , Hauptachsen  $a, b \in \mathbb{R}^+$ , sowie Drehwinkel  $\theta \in [0, 2\pi)$  oder parametrisiert

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_0 + a \cos \phi \cos \theta - b \sin \phi \sin \theta \\ y_0 + a \cos \phi \sin \theta + b \sin \phi \cos \theta \end{pmatrix} \quad (2.15)$$

mit  $\phi \in [0, 2\pi)$  und  $x_0, y_0, a, b, \theta$  wie oben.

In ihrer allgemeinsten Form lässt sich eine Ellipse durch ein implizites Polynom zweiten Grades charakterisieren

$$ax^2 + by^2 + cxy + dx + ey + f = 0 \quad \text{mit} \quad c^2 - 4ab < 0 \quad (2.16)$$

mit  $a, b, c, d, e, f \in \mathbb{R}$ . Eine Ellipse lässt sich also durch sechs Punkte eindeutig beschrieben (fünf, wenn man  $f$  auf eins skaliert).

Die beiden Formen 2.14 und 2.16 sind äquivalent, falls die Ellipse nicht degeneriert ist (ohne Beweis) [Law72]. Die Umformung von 2.14 nach 2.16 ist mit Hilfe einer Hauptachsentransformation möglich. Da wir diese später brauchen, wird sie hier einmal exemplarisch vorgeführt.

Zunächst einmal fällt auf, dass der gemischte Term  $cxy$  genau dann null ist, wenn die Ellipse nicht rotiert wurde. Im ersten Schritt versuchen wir also die Rotation der Ellipse rückgängig zu machen, um den Rotationswinkel bestimmen zu können.

Die Gleichung 2.16 kann umgeformt werden zu:

$$\underbrace{\begin{pmatrix} x & y \end{pmatrix}}_{=:u^T} \underbrace{\begin{pmatrix} a & \frac{c}{2} \\ \frac{c}{2} & b \end{pmatrix}}_{=:M} \underbrace{\begin{pmatrix} x \\ y \end{pmatrix}}_{=:u} + (d \quad e) \underbrace{\begin{pmatrix} x \\ y \end{pmatrix}}_{=:u} + f = 0$$

$$\Leftrightarrow u^T M u + (d \quad e) u + f = 0$$

Der gemischte Term wird alleine durch  $M = \begin{pmatrix} a & \frac{c}{2} \\ \frac{c}{2} & b \end{pmatrix}$  bestimmt. Da die Matrix  $M$  symmetrisch ist, ist sie orthogonal diagonalisierbar. Des Weiteren hat  $M$  zwei von null verschiedene Eigenwerte, denn

$$\det M = ab - \frac{c^2}{4}$$

ist nur dann gleich null, wenn  $c^2 - 4ab = 0$ , was ein Widerspruch zur Annahme in 2.16 ist.  $M$  hat somit eine von null verschiedene Determinante und somit vollen Rang, hat also zwei von null verschiedene Eigenwerte. Insbesondere gibt es also zwei Eigenvektoren von  $M$ , die zueinander orthogonal sind.

Es gilt  $M = S^T D S$ , wobei  $S \in \mathbb{R}^{2 \times 2}$  eine orthogonale Matrix mit den normierten Eigenvektoren als Zeilen und  $D = \text{diag}(\lambda_1, \lambda_2) \in \mathbb{R}^{2 \times 2}$  eine Diagonalmatrix mit den beiden Eigenwerten von  $M$  auf der Diagonalen ist. Ohne Beschränkung der Allgemeinheit gelte  $\lambda_1 \leq \lambda_2$ , andernfalls vertausche die Eigenvektoren in  $S$ .

Sei nun  $v := Su$ . So gilt:

$$\begin{aligned} & u^T (S^T D S) u + (d \quad e) \underbrace{(S^T S) u}_{=1} + f = 0 \\ \Leftrightarrow & (Su)^T D (Su) + (d \quad e) S^T (Su) + f = 0 \\ \Leftrightarrow & v^T D v + (d \quad e) S^T v + f = 0 \end{aligned} \quad (2.17)$$

Man kann leicht nachrechnen, dass der gemischte Teil somit eliminiert wurde. Durch Anwenden der Transformation  $S$  wurde  $u$  also in das Koordinatensystem, in dem die Ellipse achsenausgerichtet ist, transformiert.

Eine Rotationsmatrix mit Rotationswinkel  $\theta$  ist definiert durch:

$$R = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \quad (2.18)$$

Es gilt offenbar  $S = R$  für ein geeignetes  $\theta$ , da die Eigenvektoren normiert und orthogonal zueinander sind.  $\theta$  kann also einfach ausgerechnet werden, denn es gilt:

$$\theta = \text{atan2}(\sin \theta, \cos \theta) = \text{atan2}(S_{2,1}, S_{1,1})$$

Multipliziert man nun 2.17 aus, ergibt sich:

$$\begin{aligned} & \lambda_1 v_1^2 + \lambda_2 v_2^2 + \underbrace{(d \quad e) S^T v + f}_{=: (d', e')} = 0 \\ \Leftrightarrow & \lambda_1 v_1^2 + \lambda_2 v_2^2 + d' v_1 + e' v_2 + f = 0 \\ \Leftrightarrow & (\lambda_1 v_1^2 + d' v_1) + (\lambda_2 v_2^2 + e' v_2) + f = 0 \\ \Leftrightarrow & (\lambda_1 v_1^2 + d' v_1) + \left( \frac{d'^2}{4\lambda_1} - \frac{d'^2}{4\lambda_1} \right) + (\lambda_2 v_2^2 + e' v_2) + \left( \frac{e'^2}{4\lambda_2} - \frac{e'^2}{4\lambda_2} \right) + f = 0 \quad (2.19) \\ \Leftrightarrow & \left[ \lambda_1 \left( v_1^2 + \frac{2d'}{2\lambda_1} v_1 + \frac{d'^2}{4\lambda_1^2} \right) - \frac{d'^2}{4\lambda_1} \right] + \left[ \lambda_2 \left( v_2^2 + \frac{2e'}{2\lambda_2} v_2 + \frac{e'^2}{4\lambda_2^2} \right) - \frac{e'^2}{4\lambda_2} \right] + f = 0 \\ \Leftrightarrow & \underbrace{\lambda_1 (v_1 + \frac{d'}{2\lambda_1} v_1)^2}_{=-x'_0} + \underbrace{\lambda_2 (v_2 + \frac{e'}{2\lambda_2} v_2)^2}_{=-y'_0} - \underbrace{(\frac{d'^2}{4\lambda_1} + \frac{e'^2}{4\lambda_2} - f)}_{=:\sigma} = 0, \end{aligned}$$

da  $\lambda_1, \lambda_2 \neq 0$ .

Das Zentrum der transformierten Ellipse kann nun aus 2.19 einfach abgelesen werden. Um das Zentrum der eigentlichen Ellipse zu bestimmen, muss mit der inversen Rotation  $S^T$  multipliziert werden:

$$(x_0, y_0)^T = S^T(x'_0, y'_0)^T$$

Obige Gleichung lässt sich anschließend weiter vereinfachen:

$$\begin{aligned} & \lambda_1 (v_1 - x'_0)^2 + \lambda_2 (v_2 - y'_0)^2 = \sigma \\ \Leftrightarrow & \frac{\lambda_1}{\sigma} (v_1 - x'_0)^2 + \frac{\lambda_2}{\sigma} (v_2 - y'_0)^2 = 1 \quad (2.20) \end{aligned}$$

wobei  $\sigma \neq 0$ , wenn die Ellipse nicht zum Punkt entartet [Law72]. Vergleicht man nun 2.20 mit 2.13 so sieht man das folgendes gelten muss:

$$\begin{aligned} \frac{\lambda_1}{\sigma} &= \frac{1}{a^2} \quad \text{und} \quad \frac{\lambda_2}{\sigma} = \frac{1}{b^2} \\ \Leftrightarrow \sqrt{\frac{\sigma}{\lambda_1}} &= a \quad \text{und} \quad \sqrt{\frac{\sigma}{\lambda_2}} = b \end{aligned} \tag{2.21}$$

Es gilt wie erwartet  $a \geq b$ , da  $\lambda_1 \leq \lambda_2$ .

### 2.8.2. Abstand: Punkt zu Ellipse

Das hier beschriebene Verfahren zur Bestimmung der kürzesten euklidischen Distanz eines Punktes zu einer Ellipse stammt aus der Arbeit von David Eberly [Ebe13]. Wir betrachten nur Ellipsen im Ursprung, die achsenausgerichtet sind und darüber hinaus nur Punkte im ersten Quadranten. Ansonsten wir die Ellipse in den Ursprung verschoben und um ihren entgegengesetzten Drehwinkel rotiert. Da die Ellipse dann bezüglich der X- und Y- Achse symmetrisch ist, kann der Punkt einfach durch Spiegelung in den richtigen Quadranten transformiert werden. Der Abstand ändert sich dadurch nicht.

Wir bezeichnen von nun an  $Q = (y_0, y_1)$  als eine Punkt, dessen Distanz zur Ellipse wir berechnen wollen und  $E = (x_0, x_1)$  als denjenigen eindeutigen Punkt, welcher auf der Ellipse liegt und die kürzeste euklidische Distanz zum Punkt  $Q$  hat.

Aufgrund dieser Forderungen können wir ohne Beschränkung der Allgemeinheit folgende Aussagen treffen:

- Die Ellipse kann stets durch die implizite Gleichung

$$\frac{x_0^2}{a^2} + \frac{x_1^2}{b^2} = 1 \tag{2.22}$$

mit  $a \geq b \geq 0$  beziehungsweise in der parametrischen Form

$$\mathcal{E}(\theta) = (a \cos \phi, b \sin \phi) \quad \phi \in [0, 2\pi)$$

beschrieben werden.

- Es gilt  $y_0, y_1, x_0, x_1 \geq 0$ .

Für die quadrierte Distanz von einem beliebigen Punkt  $Q$  zu einem Punkt  $\mathcal{E}(\theta)$  auf der Ellipse gilt dann

$$F(\theta) = |\mathcal{E}(\theta) - Q|^2. \tag{2.23}$$

Wir wollen  $F$  minimieren und betrachten die Ableitung:

$$F'(\theta) = 2(\mathcal{E}(\theta) - Q) \cdot \mathcal{E}'(\theta) \tag{2.24}$$

$F'$  wird null, wenn  $(\mathcal{E}(\theta) - Q)$  und  $\mathcal{E}'(\theta)$  zu einander orthogonal sind. Daraus folgt, dass der Vektor von  $Q$  zu  $E$  senkrecht zur Ellipse stehen muss. (siehe Abbildung 2.10).

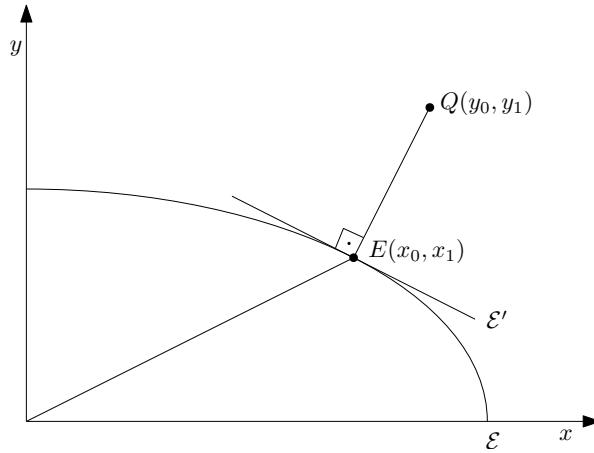


Abbildung 2.10.: Ellipsenausschnitt im ersten Quadranten mit Abfragepunkt  $Q$  und eingezeichneter kürzester Distanz zur Ellipse

Betrachten wir also die Funktion:

$$G(x_0, x_1) = \frac{x_0^2}{a^2} + \frac{x_1^2}{b^2} - 1. \quad (2.25)$$

$(x_0, x_1)$  ist genau dann ein Punkt auf der Ellipse, wenn  $G(x_0, x_1) = 0$ . Der Gradient von  $G$  in  $(x_0, x_1)$  ist ein Normalenvektor auf der Ellipse und somit auch der halbe Gradient  $\nabla G(x_0, x_1)/2$ . Der Vektor von  $E$  zu  $Q$  muss dieselbe Richtung haben. Es gilt somit:

$$(y_0, y_1) - (x_0, x_1) = t \frac{\nabla G(x_0, x_1)}{2} = t \left( \frac{x_0}{a^2}, \frac{x_1}{b^2} \right) \quad (2.26)$$

für ein  $t \in \mathbb{R}$ .

Umgestellt nach  $y_0$  und  $y_1$ , beziehungsweise nach  $x_0$  und  $x_1$  ergibt sich:

$$y_0 = x_0 \left( 1 + \frac{t}{a^2} \right), \quad y_1 = x_1 \left( 1 + \frac{t}{b^2} \right) \quad (2.27)$$

$$x_0 = \frac{a^2 y_0}{t + a^2}, \quad x_1 = \frac{b^2 y_1}{t + b^2} \quad (2.28)$$

Man macht nun eine Fallunterscheidung:

1. Der einfachste Fall ist, wenn sich der Punkt  $Q$  auf der  $Y$ -Achse (außer  $(0, 0)$ ) befindet, wenn also gilt  $y_0 = 0, y_1 > 0$ . Da die Hauptachse nach der  $X$ -Achse ausgerichtet ist und  $a \geq b$  gilt, ist der Punkt auf der Ellipse mit der kürzesten Distanz zu  $Q$  offenbar  $E = (0, b)$  und für die Distanz gilt  $d = |y_1 - b|$ .
2. Als nächstes betrachten wir ein  $Q$  auf der  $X$ -Achse (einschließlich  $(0, 0)$ ), wenn also

gilt  $y_0 \geq 0, y_1 = 0$ . Es gilt also mit 2.27

$$y_0 = x_0 \left(1 + \frac{t}{a^2}\right), \quad 0 = x_1 \left(1 + \frac{t}{b^2}\right)$$

Wenn  $x_1 = 0$  gilt, muss  $x_0 = a$  gelten, damit  $E(x_0, x_1)$  auf der Ellipse ist. Es gilt analog zum ersten Fall  $E = (a, 0)$  mit  $d = |y_0 - a|$

Gilt  $x_1 \neq 0$ , so können wir in der zweiten Gleichung durch  $x_1$  teilen und es folgt  $t = -b^2$  und somit  $y_0 = x_0 \left(1 - \frac{b^2}{a^2}\right)$ . Auf Grund der Krümmung der Ellipse gilt außerdem  $x_0 < a$  und somit ergibt sich zusammen mit 2.28 die Ungleichung:

$$\begin{aligned} x_0 &= \frac{a^2 y_0}{a^2 - b^2} < a \\ \Leftrightarrow y_0 &< \frac{a^2 - b^2}{a} < a \end{aligned}$$

Für Punkte  $Q(y_0, 0)$  mit  $y_0 \geq \frac{a^2 - b^2}{a}$ , ist der kürzeste Punkt also wieder  $E(a, 0)$

Für Punkte  $Q(y_0, 0)$  mit  $y_0 < \frac{a^2 - b^2}{a}$ , muss für  $E(x_0, x_1)$  nach Umstellen der Ellipsengleichung 2.22  $x_1 = b \cdot \sqrt{1 - \left(\frac{x_0}{a}\right)}$  gelten. Die Distanz beträgt dann:

$$d^2 = (x_0 - y_0)^2 + x_1^2 = b^2 \left(1 - \frac{y_0^2}{a^2 - b^2}\right)$$

3. Der letzte Fall, den wir betrachten müssen, ist der allgemeinste Fall. Es gilt  $y_0 > 0$ , sowie  $y_1 > 0$ . Da wir uns nur im ersten Quadranten bewegen, gilt darüber hinaus  $x_0, x_1 \geq 0$ . Mit diesen Eigenschaften und 2.27 lässt sich folgende Einschränkung für  $t$  herleiten:

$$\begin{aligned} 0 &< y_0 = x_0 \left(1 + \frac{t}{a^2}\right) \\ \Leftrightarrow -1 \cdot a^2 &< t. \end{aligned} \tag{2.29}$$

Analog ergibt sich mit  $y_1$ :  $-b^2 < t$ . Da  $a \geq b$  gilt, reicht es, sich nur die zweite Ungleichung anzuschauen, da sie die erste impliziert. Setzt man nun 2.28 in 2.25 ein, erhält man:

$$F(t) = \left(\frac{ay_0}{t + a^2}\right)^2 + \left(\frac{by_1}{t + b^2}\right)^2 - 1 \tag{2.30}$$

Man kann nun zeigen, dass diese Funktion auf dem gesamten Intervall  $[-b^2, \infty)$  monoton fällt und links gekrümmmt ist. Darüber hinaus gilt:

$$\lim_{t \searrow -b^2} F(t) = +\infty, \quad \lim_{t \rightarrow \infty} F(t) = 1.$$

## 2. THEORETISCHE GRUNDLAGEN

---

Da  $F$  stetig ist, muss es also eine Nullstelle geben, die aufgrund des Monotonie- und Krümmungsverhaltens sogar eindeutig ist. Die Nullstelle lässt sich beispielsweise durch Intervallschachtelung oder Newton-Verfahren bestimmen.

# 3. Methodik

In diesem Kapitel stellen wir ein Verfahren vor, dass ermöglicht anhand eines Bildes (siehe Einleitung).

überarbeiten

Zunächst gehen wir auf das verwendete Kalibrierungsmuster ein, woraufhin die genaue Vorgehensweise zur Entfaltung erläutert wird.

Die geometrischen Eigenschaften des Kegelstumpfs ( $r, R, \Delta H$ ) können gemessen und somit als bekannt angenommen werden. Darüber hinaus nehmen wir an, dass sich das Zentrum des kleineren Kreises im links-händigen Weltkoordinatensystem an der Positon  $(0, 0, 0)$  (siehe Abbildung 2.3 in Kapitel 2.1) befindet. Durch diese Einschränkung gehen jegliche absolute Größenverhältnisse verloren. Die Larven können jedoch weiterhin relativ zu einander verglichen werden.

## 3.1. Kalibrierungsmuster

Um eine Beziehung zwischen Bildpunkten und Kegelpunkten herstellen zu können, ist ein Kalibrierungsmuster notwendig.

Die Wahl des Kalibrierungsmusters spielt dabei eine entscheidende Rolle bei der Robustheit und Präzision der Entfaltung. Es muss gewährleistet sein, dass die charakteristischen Merkmale des Musters auch bei leichten Abweichungen der Kamera vom Lot und schlechteren Beleuchtungssituationen zuverlässig erkannt werden. Das Muster muss darüber hinaus so entworfen sein, dass beim Zusammenlegen im Kegel, dessen geometrische Eigenschaften nicht verfälscht, sondern realitätsgerecht wiedergeben werden.

Wir haben uns für ein Muster entschieden, dass in äquidistanten Abständen  $\Delta R$ , beginnend mit dem kleinen Radius  $r$  des Kegelstumpfs (siehe Abbildung 2.3) Kreislinien und in gleichen Winkelabständen  $\Delta\alpha$  auf der Seitenhöhe Liniensegmente besitzt. Das zusammengelegte Muster ist in Abbildung 3.1 zu sehen, beziehungsweise das entfaltete in 3.2. Die Anzahl der Kreislinien wird mit  $n$  gekennzeichnet, die Anzahl sichtbarer Liniensegmente im Kegel mit  $m$ . Zu beachten ist, dass bedingt durch das Entfalten, in Abbildung 3.2 ein Liniensegment doppelt zu sehen ist. Die schwarzen Kreise bezeichnen wir als Samples.

Dadurch dass die Geometrie des Kegels bekannt ist, kann jedem Sample nun ein Punkt auf dem Kegel im Weltkoordinatensystem zugeordnet werden. Da ein Kegel beliebig um die Y-Achse rotiert werden kann, ist diese Zuordnung zunächst nicht eindeutig. Dazu nehmen wir an, dass das Liniensegment mit dem kleinsten Winkel zur X-Achse mit dem Kegelwinkel  $\theta = 0$  korrespondiert (siehe 2.2).

Zur Konstruktion des Musters benötigt man  $\Delta S$  und  $s$ , die man aus der Geometrie des Kegels errechnen kann und außerdem den Öffnungswinkel, der gegeben ist als  $\alpha = 2\pi \frac{R}{S}$

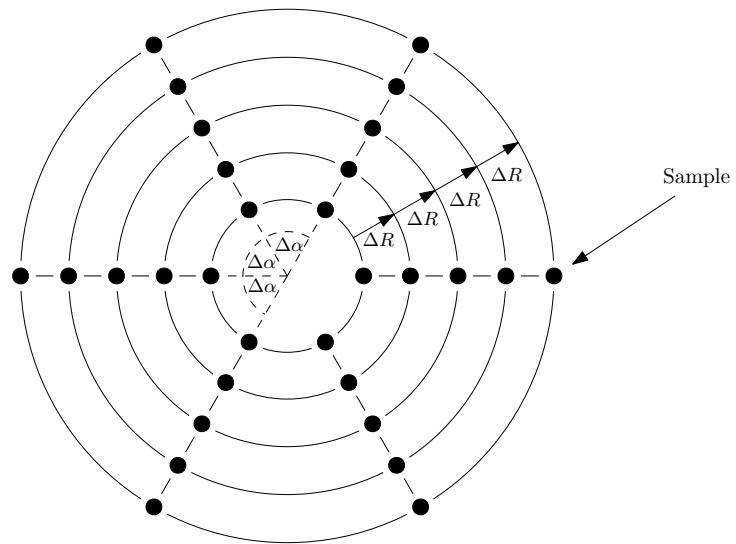


Abbildung 3.1.: Kalibrierungsmuster von oben mit  $n = 5, m = 6$

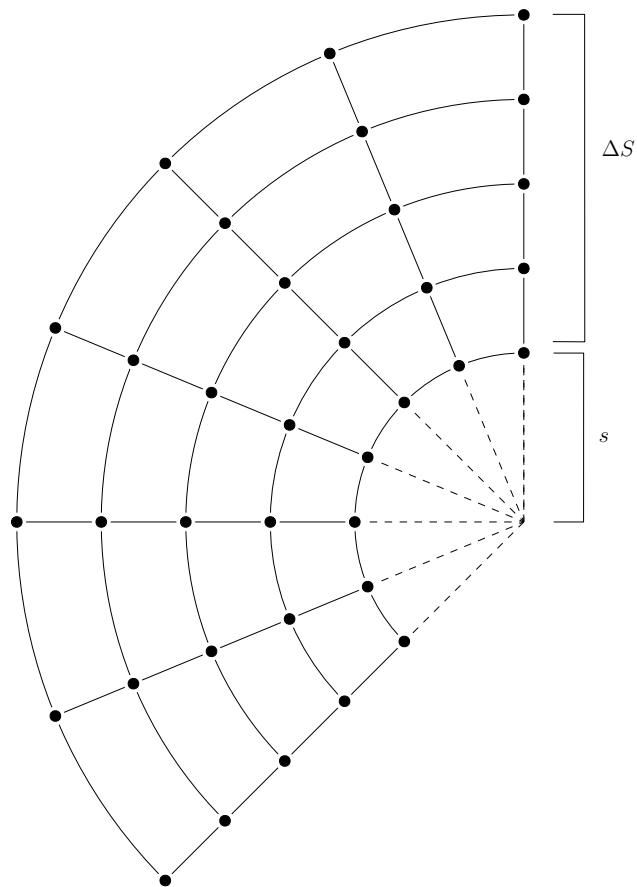


Abbildung 3.2.: Kalibrierungsmuster entfaltet mit  $n = 5, m = 6$

(siehe 2.3 in Kapitel 2.1).

### 3.1.1. Anzahl der Samples

Die Anzahl der Samples sollte groß genug sein, um möglichst viel geometrische Informationen des Kegels zu erhalten, aber klein genug, dass eine Detektion der Samples problemlos möglich ist. Insbesondere auf dem innersten Kreis, macht sich eine zu hohe Sampleanzahl negativ bemerkbar, da der Abstand zueinander sehr klein wird, was eine Detektion erschwert. Des Weiteren sollte noch ein möglichst großer Teil der Kreislinien zu sehen bleiben, da diese für die Ellipsendetektion benötigt werden.

## 3.2. Intrinsische Kamerakalibrierung

Bedingt durch die Wahl einer Weitwinkelkamera, enthält die Linse der Kamera eine starke tonnenförmige (nach außen gewölbte) Verzerrung (siehe Abbildung 3.3). Diese muss herausgerechnet werden, da sonst Abstände im Bild nicht mehr der Realität entsprechen und dadurch die Präzision der Entfaltung stark abnimmt (siehe Kapitel 5). Es wird also zunächst eine Kamerakalibrierung durchgeführt, wobei wir nur an den intrinsischen Parametern, inklusive Verzerrungskoeffizienten, interessiert sind, um das Bild anschließend entzerren zu können.

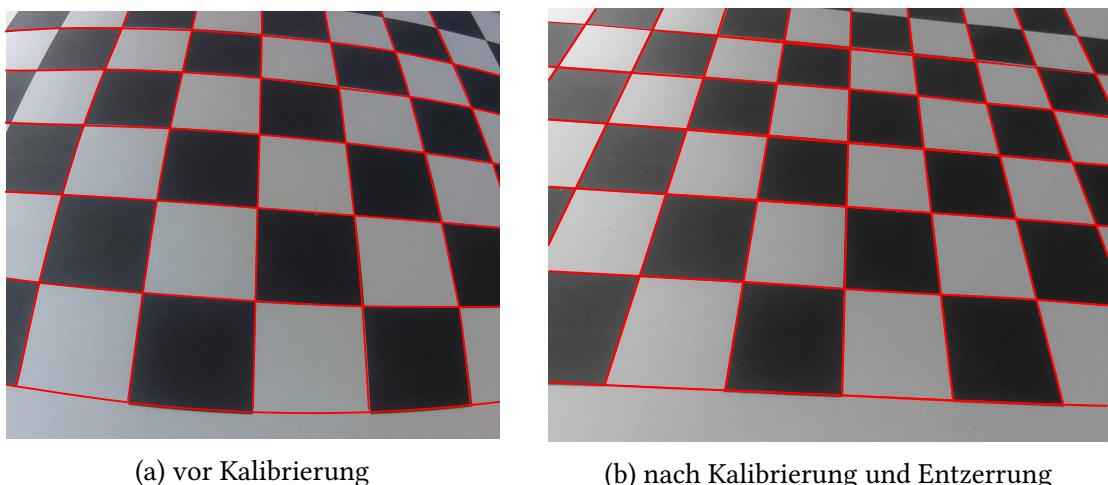


Abbildung 3.3.: Kamerakalibrierung

## 3.3. Detektion der charakteristischen Punkte

Nach der Kamerakalibrierung und entsprechender Entzerrung werden die Bildkoordinaten der Samples bestimmt. Dazu wird ein Blob-Detektor (siehe Kapitel 2.4) benutzt. Um ein

Sample korrekt detektieren zu können, muss sich der Punkt farblich stark von seiner Umgebung abheben (siehe Definition: Blob 2.4.1). Insbesondere dürfen die Kreislinien und Liniensegmente des Kalibrierungsmusters also nicht durchgezogen sein. Nach der Detektion werden die Blobs nach folgenden Kriterien gefiltert:

- **Fläche:** zu kleine Blobs werden verworfen
- **Rundheit:** zu unrude Blobs werden verworfen. Rundheit ist hier definiert als  $\text{circ} = \frac{4\pi \cdot \text{Fläche}}{(\text{Umfang})^2} \in [0, 1]$ , wobei also ein Kreis mit  $\text{circ} = 1$  maximal rund ist.
- **Konvexität:** zu unkonvexe Blobs werden verworfen. Konvexität ist hier definiert als  $\text{conv} = \frac{\text{Fläche Blob}}{\text{Fläche konvexe Hülle}}$

In Abbildung 3.4 ist beispielhaft links ein Grauwertbild und rechts die detektierten Blobs (grün) auf dem gleichen Bild nach der Kameraentzerrung zu sehen.

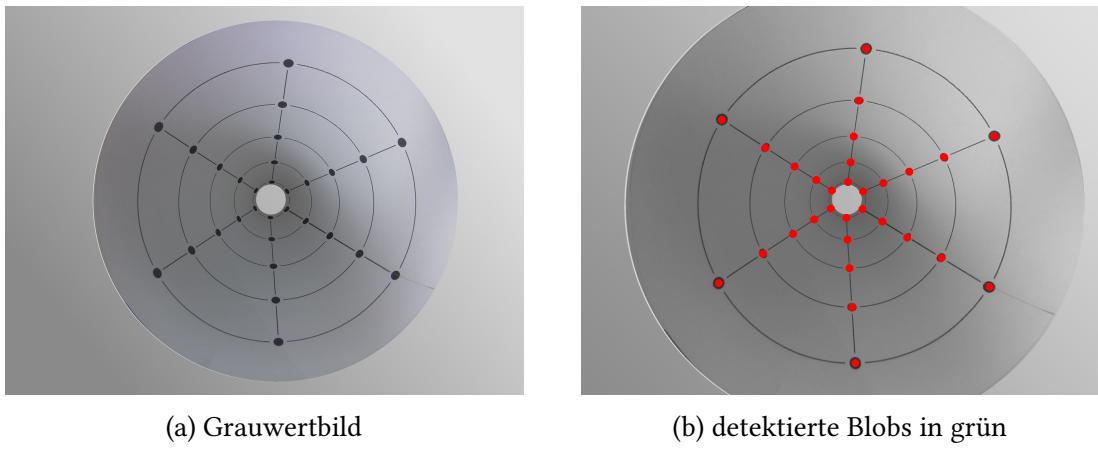


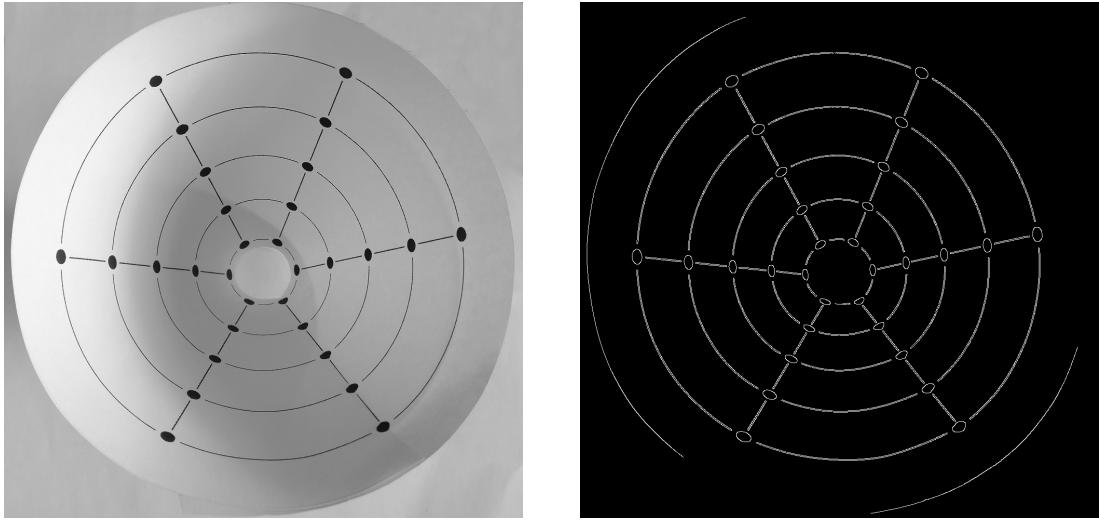
Abbildung 3.4.: Detektion der Samples

### 3.4. Ellipsen-Detektion

Nachdem die Sample-Positionen bestimmt wurden, muss für jeden Sample entschieden werden, auf welcher der Kreislinien er liegt. Da die Kreise, bedingt durch perspektivische Verzerrung, zu Ellipsen werden, wird eine Verfahren benötigt, dass Ellipsen erkennt.

Zunächst werden die Kanten mit Hilfe von Canny (siehe Kapitel 2.5) detektiert (siehe Abbildung 3.5).

Anschließend versuchen wir möglichst genau das Zentrum der innersten Ellipsen zu schätzen. Wir benutzen dafür Hough-Transformationen (siehe Kapitel 2.6), um Linien im Kantenbild zu detektieren. Es werden anschließend die Schnittpunkte aller Liniensegmente bestimmt. Bedingt durch Ungenauigkeiten beim Ausschneiden und Zusammenlegen im Kegel und perspektivischer Verzerrung, schneiden sich nicht alle Liniensegmente in einem



(a) Grauwertbild

(b) Canny-Kanten

Abbildung 3.5.: Canny-Kantendetektion auf Grauwertbild

Punkt. Darüber hinaus werden, auf Grund der Liniendicke auf dem Kalibrierungsmuster, durch Canny viele Linien als doppelt Linien gekennzeichnet<sup>1</sup>. Auch ein inhomogener Hintergrund, erschwert die Schnittpunktsbestimmung. Um also möglichst robust einen Kandidaten auszuwählen, wird zuerst der Median der  $x$ -Koordinaten der Schnittpunkte und dann der Median der  $y$ -Koordinaten bestimmt. Die erhaltenen Koordinaten bilden den Schnittpunkt (siehe Abbildung 3.6).

Von diesem Schnittpunkt aus werden, in einer vorher definierte Anzahl, gleichmäßig, in alle Richtungen Strahlen ausgesendet. Trifft ein Strahl ein weißes Pixel, wird dessen Position gekennzeichnet, trifft er den Rand des Bildes, wird er ignoriert. In Abbildung 3.7(a) sind die getroffenen weißen Pixel und der zugehörige Aussendepunkt eingezzeichnet.

Mit Hilfe der Positionen der weißen Pixel, wird anschließend durch RANSAC (siehe Kapitel 2.7) eine Ellipse geschätzt.

Es wird konkret für sechs zufällig ausgewählte Punkte das lineare Gleichungssystem

$$\begin{pmatrix} x_1^2 & y_1^2 & x_1y_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_6^2 & y_6^2 & x_6y_6 & x_6 & y_6 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

gelöst, was auf der Gleichung 2.16 aus Kapitel 2.8 basiert. Nach dem Lösen wird geprüft,

<sup>1</sup>Dies ist kein Widerspruch zur Eindeutigkeit von Canny-Kanten (siehe 2.5). Stellt man sich ein relativ breites Liniensegment vor, so gibt es einmal den Übergang vom Hintergrund auf die Linie, sowie den Übergang von der Linie wieder auf den Hintergrund

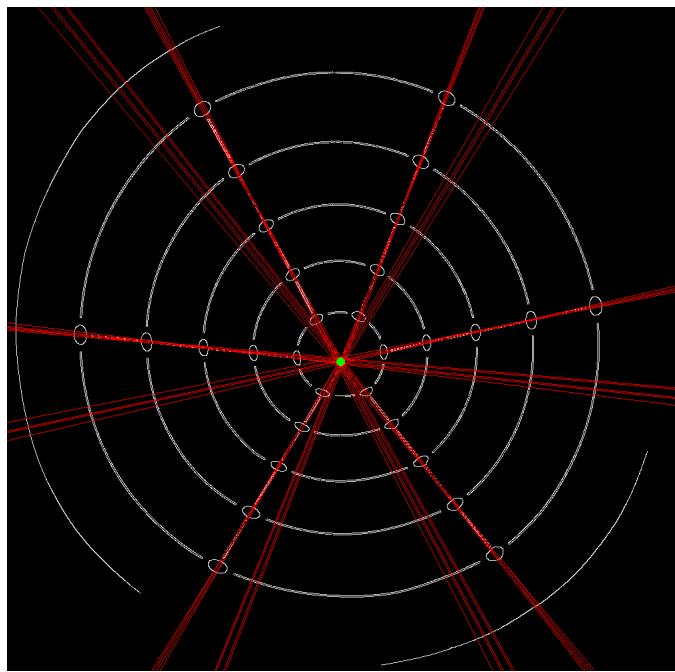


Abbildung 3.6.: Hough-Transformation zur Linien-Detektion (in rot gekennzeichnet) und bestimmter Schnittpunkt (in grün)

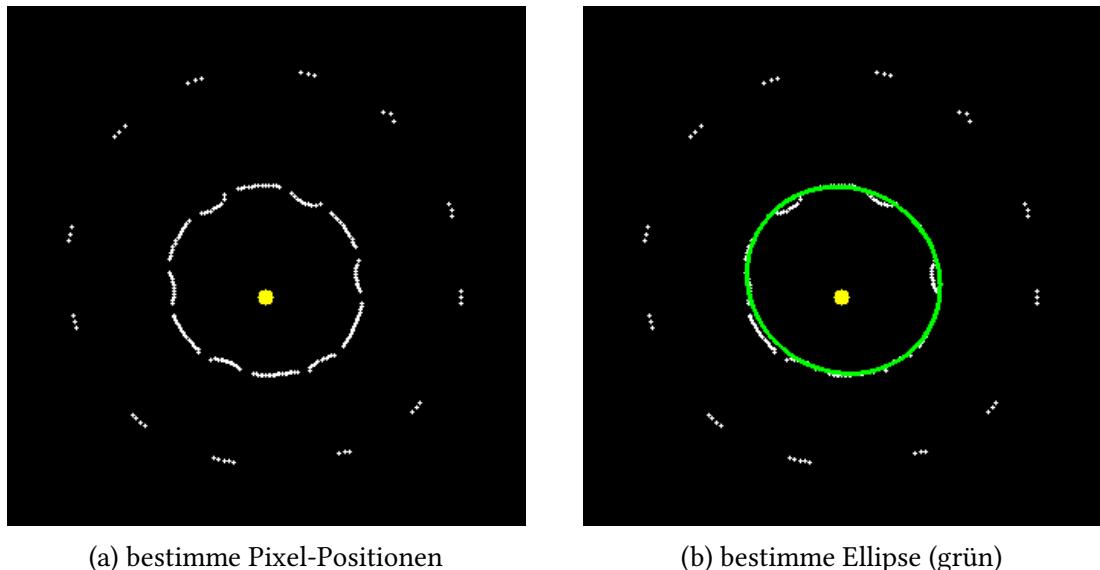
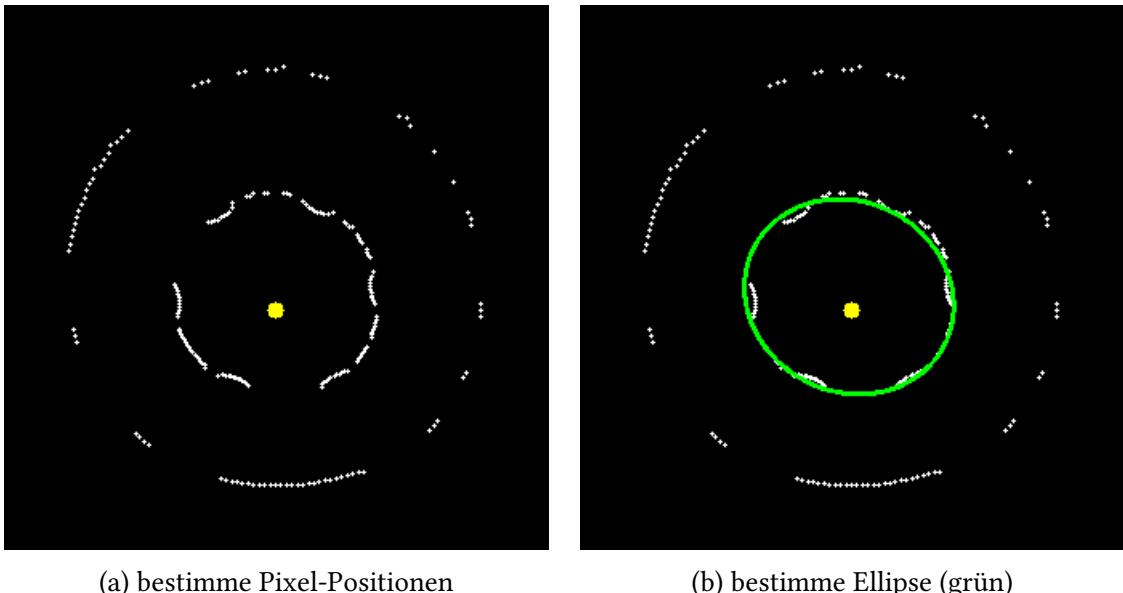


Abbildung 3.7.: Ellipsendetektion: bestimme Pixel-Positionen (weiß), Aussendepunkt (gelb)

ob es sich tatsächlich um eine Ellipse handelt und mittels Hauptachsentransformation (siehe Kapitel 2.8) in die Ellipsenform ( $x_0, y_0, a, b, \phi$ ) umgewandelt. Wir verwerfen anschließend Ellipsen mit, dessen Hauptachse wesentlich größer ist, als ihre Nebenachse, da wir eher wenig gestauchte Ellipsen erwarten.

Um die, für RANSAC benötigte, Distanz zu berechnen, wird das Verfahren aus Kapitel 2.8.2 genutzt, was die exakte euklidische Distanz eines Punktes zu einer Ellipse approximiert. Ein Verfahren wie das Verfahren der kleinsten Quadrate anstelle von RANSAC funktioniert hier nicht, da die weißen Pixel bezüglich einer zu bestimmenden Ellipse, ausreißerbehaftet sind. Wird zum Beispiel auf Grund schlechter Lichtverhältnisse eine Kreislinie nicht deutlich aufgenommen, kann es in dem Kantenbild (Abbildung 3.5) zu „Löchern“ in den Kreislinien kommen und folglich treffen die ausgesendeten Strahlen die nächst äußere Kreislinie (siehe Abbildung 3.8).

Da die Laufzeit nicht im Vordergrund steht, kann eine großzügige Schätzung des Fehleranteils von  $\epsilon = 0.4$  mit einer gewünschten Wahrscheinlichkeit  $p = 0.9999$  gewählt werden, was zu einer Mindestanzahl an Iterationen von circa 200 führt (siehe Kapitel 2.7). Die letztendlich bestimmten Ellipsen sind beispielhaft in Abbildung 3.9 zu sehen.



(a) bestimme Pixel-Positionen

(b) bestimme Ellipse (grün)

Abbildung 3.8.: Ellipsendetektion bei Ausreißern

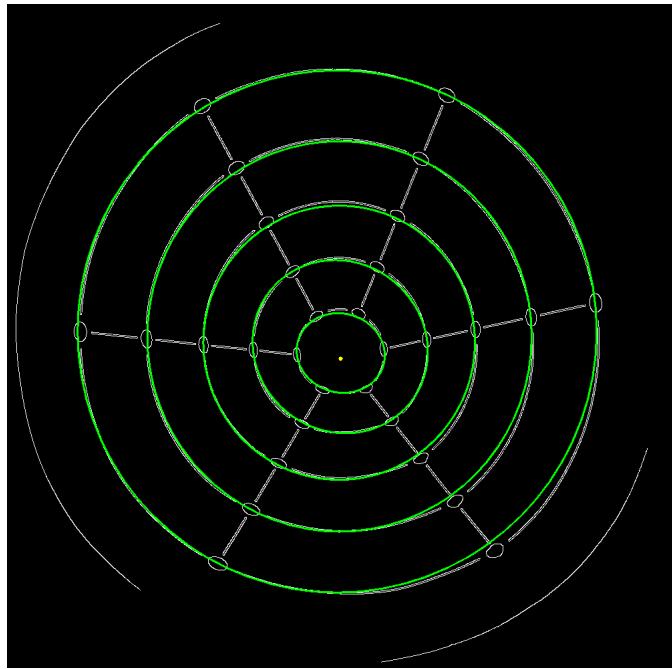


Abbildung 3.9.: detektierte Ellipsen

### 3.5. Zuordnung der Punkte

Nach der Bestimmung der Ellipsen muss jede Sample-Positionen der zugehörigen Kreislinie, sowie Liniensegment zugeordnet werden, um seine Position auf dem Kegel bestimmen zu können. Zunächst wird für jeden Punkt diejenige Kreislinie ausgewählt, dessen zugehörige Ellipse die kürzeste Distanz zu ihm hat (siehe Abbildung 3.11(a)).

Mit Hilfe dieser Zuordnung können die Ellipsen aus Kapitel 3.4 erneut geschätzt werden. Diesmal wird das Verfahren der kleinsten Quadrate genutzt, da nur die ausreißerfreien Samples als Messdaten dienen und wir eine optimale Lösung für alle Samples anstreben.

Um nun die Samples auch ihren Liniensegmenten zuzuordnen, wird zunächst der Mittelpunkt der Samples auf der innersten Ellipse bestimmt. Anschließend werden die Samples auf der innersten Ellipse nach dem Winkel der Verbindungslien zwischen Sample und Mittelpunkt mit der X-Achse sortiert. Die restlichen Samples können nicht nach dem gleichen Schema sortiert werden, da der bestimmte Mittelpunkt nicht der genaue Schnittpunkt aller Liniensegmente ist. Der Mittelpunkt liegt also nicht auf der Verlängerung eines Liniensegments. Die Winkel der Verbindungslien der Samples (auf einem gemeinsamen Liniensegment) und dem Mittelpunkt mit der X-Achse sind nicht identisch (siehe Abbildung 3.10).

Stattdessen wird für jedes Sample auf den darauffolgenden Ellipsen das Sample auf der vorherigen Ellipse mit der kürzesten Distanz bestimmt. Die Samples können nun entsprechend sortiert werden. Die zugeordneten Liniensegmente sind exemplarisch in Abbildung 3.11(b) zu sehen.

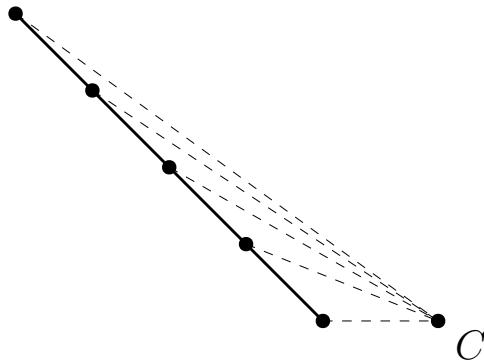


Abbildung 3.10.: Winkel zwischen Samples und Mittelpunkt  $C$  sind nicht identisch

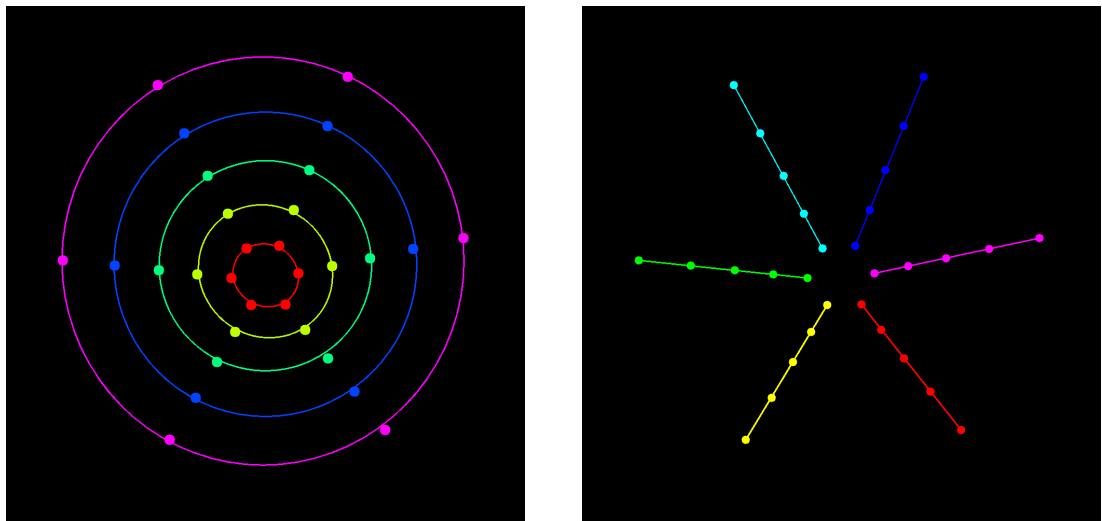


Abbildung 3.11.: Zuordnung von Punkten zu Ellipsen (links) und Liniensegmenten (rechts)

## 3.6. Weltkoordinaten bestimmen

Mit Hilfe der Parametrisierung des Kegelstumpfs aus Kapitel 2.5 können wir die 3D-Koordinaten eines Samples folgendermaßen angeben:

Ohne Beschränkung der Allgemeinheit, seien die Ellipsen  $i = 0, \dots, n - 1$  aufsteigend nach ihrer „Größe“<sup>2</sup> sortiert, so wie es das Verfahren in 3.4 beschreibt. Außerdem seien die Liniensegmente  $j = 0, \dots, m - 1$  aufsteigend nach Winkel mit der X-Achse, wie in 3.5 beschrieben, sortiert. Eine Sample kann also eindeutig durch ein Tupel  $(i, j) \in [0, n - 1] \times [0, m - 1]$  identifiziert werden und  $(x_{ij}, y_{ij}, z_{ij})$  bezeichne seine Koordinaten im Weltkoordinatensystem.

Analog zur parametrischen Darstellung von Kegelstümpfen (Gleichung 2.2) in Kapitel

<sup>2</sup>Etwas formaler, könnte man die Ellipsen hier nach ihrem Flächeninhalt sortieren. Für Ellipsen  $E_0(x_0, y_0, a_0, b_0, \theta_0)$  und  $E_1(x_1, y_1, a_1, b_1, \theta_1)$  gilt  $E_0 \leq E_1$  g.d.w.  $\pi \cdot a_0 \cdot b_0 \leq \pi \cdot a_1 \cdot b_1$

2.1 ergibt sich:

$$\begin{aligned}x_{ij} &= r_i \cos\theta_j \\y_{ij} &= h_i \\z_{ij} &= r_i \sin\theta_j\end{aligned}$$

$\forall(i, j) \in [0, n - 1] \times [0, m - 1]$  mit

$$\begin{aligned}r_i &= r + \frac{i}{n} \cdot (R - r) & \forall i \in [0, n - 1] \\h_i &= \frac{i}{n} \cdot \Delta H & \forall i \in [0, n - 1] \\\theta_j &= \frac{j}{m - 1} \cdot 2\pi & \forall j \in [0, m - 1]\end{aligned}$$

## 3.7. Entfaltung

Die eigentliche Entfaltung des Kegels kann mit zwei unterschiedlichen Ansätzen realisiert werden. Die erste Möglichkeit ist die *Vorwärtsentfaltung*. Hierbei wird für jedes Pixel auf dem Kegelbild eine 3D-Koordinate durch geeignete Interpolation bestimmt und dann auf die Mantelfläche abgebildet. Beim zweiten Ansatz, der *Rückwärtsentfaltung*, wird ein Punkt von der Mantelfläche zurück auf den Kegel abgebildet und von dort mit einer Projektionsmatrix auf die Bildebene projiziert und dann interpoliert.

Im folgenden wird genauer auf beide Verfahren eingegangen, sowie deren Probleme erläutert.

### 3.7.1. Vorwärtsentfaltung

Bei der *Vorwärtsentfaltung* muss wie oben erwähnt zu jedem Pixel die zugehörige 3D Koordinate im Weltkoordinatensystem berechnet werden. Da bisher jedoch nur die Positionen der Samples bekannt sind muss hier

Zunächst betrachten wir diejenigen Pixel, die sich weder auf einer Kreislinie, noch auf einem Liniensegment befinden. Es gibt zu einem Pixel  $P$  also immer vier Sample-Nachbarn ( $bl, br, tr, tl$ ). Diese Situation ist in Abbildung 3.12 illustriert.

Nachdem die vier Nachbarn bestimmt wurden, können im ersten Schritt die Abstände  $d_1$  und  $d_2$  zu inneren Ellipse  $E_b$ , respektive äußeren Ellipse  $E_t$  berechnet werden. Mithilfe dieser Abständen kann nun eine *Interpolationsellipse*  $E_1$  definiert werden als

$$E_{int} = \left( \frac{d_1}{d_1 + d_2} \right) \cdot E_t + \left( \frac{d_2}{d_1 + d_2} \right) E_b,$$

wobei eine Multiplikation mit einem Skalar alle Charakteristika einer Ellipse skaliert. Der Drehwinkel  $\theta$  wird hierbei bei  $2\pi$  umgebrochen. Eine Addition geschieht elementweise. Im nächsten Schritt wird der Schnittpunkt  $L$  mit dem Liniensegment  $\overline{bltl}$ , sowie der

Schnittpunkt  $R$  mit dem Liniensegment  $\overline{brtr}$  bestimmt.

Da sich  $tl, L$  und  $bl$  nun auf einem gemeinsamen Liniensegment befinden, kann bezüglich der Weltkoordinaten linear interpoliert werden. Analoges gilt für  $tr, R$  und  $br$ .

Die drei Punkte ( $L, P, R$ ) befinden sich auf der Interpolationsellipse und somit können die Winkel  $(\phi_L, \phi_P, \phi_R)$  bezüglich des gemeinsamen Ellipsenkoordinatensystems bestimmt werden.

Analog zu oben werden die 3D-Koordinaten von  $P$  als lineare Interpolation zwischen den gerade bestimmten 3D-Koordinaten von  $L$  und  $R$  bestimmt. Als Interpolationsfaktor benutzen wir  $\frac{\phi_L - \phi_P}{\phi_L - \phi_R}$

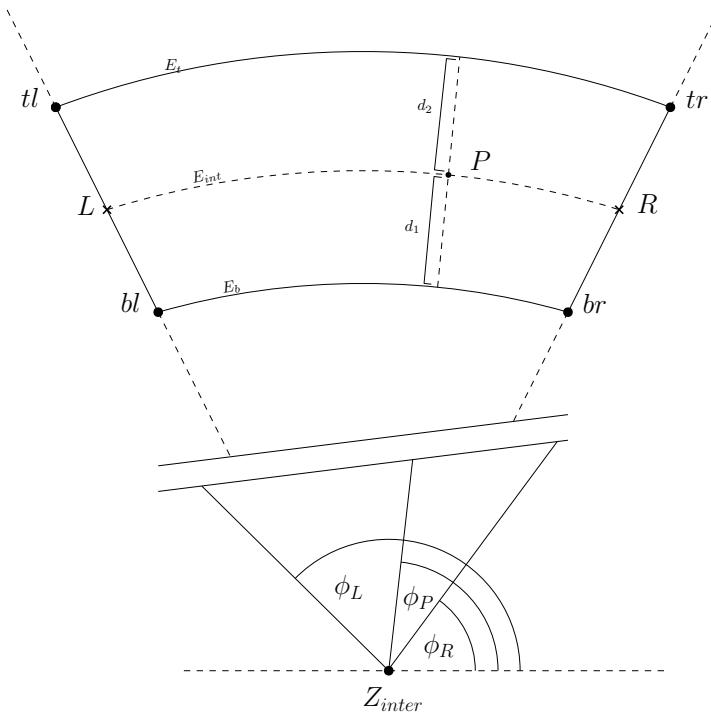


Abbildung 3.12.: Interpolation der 3D-Koordinaten

Analog werden die auf Liniensegmenten befindlichen Punkte einfach linear interpoliert. Die Punkte, die sich auf Kreislinien befinden werden über Winkel interpoliert.

Jedes Pixel hat nun 3D-Koordinaten im Kegel erhalten, die beispielhaft in Abbildung 3.13 zu sehen sind. Anhand der Abbildung lässt sich das erste Problem bei der Vorwärtsentfaltung feststellen. Zwischen den Samples auf einer Kreislinie sollten die interpolierte Positionen nach außen gewölbt sein, da ein Kegel rund ist. Da wir jedoch Interpolieren und nicht Extrapolieren, gehen die Werte nie über die zur Interpolation genutzten Werte hinaus. Konkreter kann eine interpolierte Position keine größeren oder kleineren  $X$  und  $Z$  Koordinaten erhalten, als die der genutzten Samples. Diese wäre jedoch notwendig, sodass die Rundung des Kegels erhalten bleibt. Die Oberfläche des Kegels scheint eckig.

Neben der außerdem sehr hohen Laufzeit, bedingt durch die komplexe Interpolation, ergibt sich noch ein weiteres Problem, dass sich erst bei der Entfaltung ergibt.

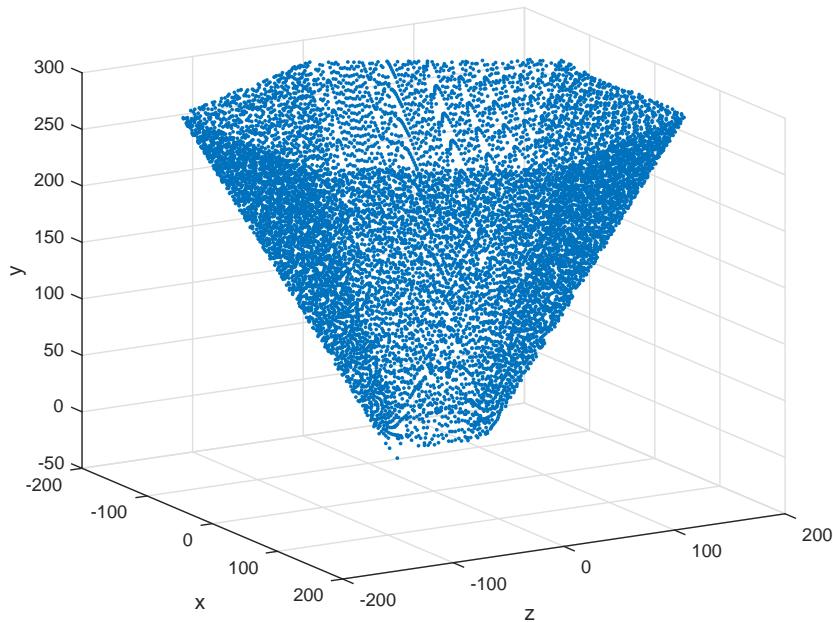


Abbildung 3.13.: interpolierter Kegel

Die eigentliche Entfaltung geschieht über die Abbildung 2.5, die in Kapitel 2.1 konstruiert wurde. Die erhaltenen Werte müssen anschließend skaliert werden, da im entfalteten Bild sonst 1 mm einem Pixel entspreche. Es ergibt sich das entfaltete Bild, wie in Abbildung 3.14 dargestellt. Auch schon bei kleinen Skalierungen entstehen auffällige „Löcher“ im entfalteten Bild. Grund dafür ist, dass jedes Pixel aus dem Ursprungsbild auf eine 2D-Koordinate der Mantelfläche abgebildet wird. Auch nach einer Skalierung handelt es sich bei diesen Werten im Allgemeinen nicht um ganzzahlige Werte. Es muss im entfalteten Bild gerundet werden. Auch wenn Rundungsfehler nicht entstehen, fehlte es einfach an genügend Informationen, gerade in den inneren, kleineren Regionen des Ursprungsbild.

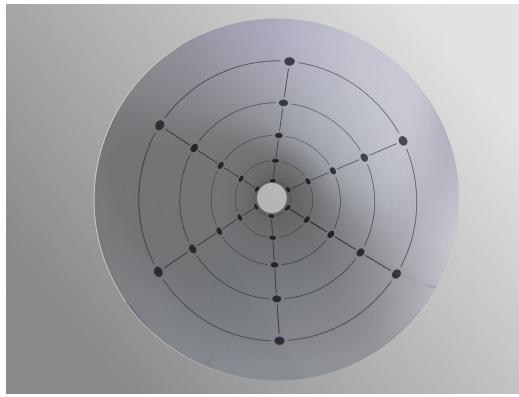
Da wir von dem Ursprungsbild aus auf das entfaltete Bild abbilden, ist außerdem eine Interpolation auf dem Ursprungsbild nicht möglich, da wir vorher nicht genau wissen, wo Löcher entstehen. Man müsste als entweder auf dem resultierendem Bild interpolieren (siehe Kapitel 6), oder zu gegebenen Löchern über die Umkehrabbildung auf dem Ursprungsbild interpolieren.

Es bietet sich dann jedoch an, direkt die Umkehrabbildung zu nutzen, was die Motivation hinter der Rückwärtsentfaltung ist.

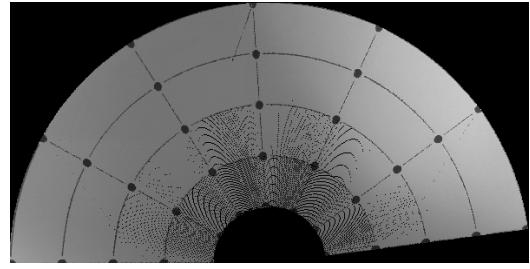
### 3.7.2. Rückwärtsentfaltung

das ist zu  
knapp

Bei der Rückwärtsentfaltung gehen wir von dem entfalteten Bild aus und bestimmen zunächst mit der Umkehrabbildung 2.6 aus Kapitel 2.1, die 3D Kegelkoordinaten. Mit Hilfe



(a) Ursprungsbild

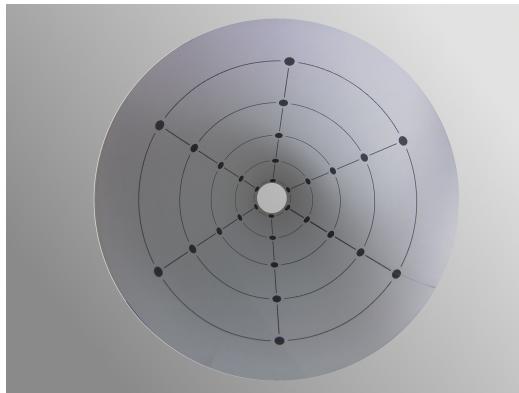


(b) entfaltetes Bild (um  $90^\circ$  im Uhrzeigersinn gedreht)

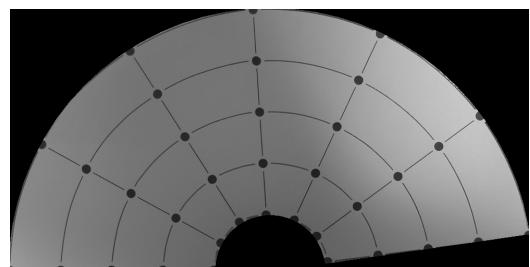
Abbildung 3.14.: Vorwärtsentfaltung

einer Projektionsmatrix werden diese 3D-Koordinaten dann auf Bildkoordinaten abgebildet.

Konkret lösen wir das überbestimmte lineare Gleichungssystem 2.9 aus Kapitel 2 mittels *Direct Linear Transformation*. Die resultierenden Bildkoordinaten sind im Allgemeinen nicht ganzzahlig. Es kann hier allerdings im Gegensatz zur Vorwärtsentfaltung einfach auf dem Ursprungsbild interpoliert werden.



(a) Ursprungsbild



(b) entfaltetes Bild (um  $90^\circ$  im Uhrzeigersinn gedreht)

Abbildung 3.15.: Rückwärtsentfaltung



# 4. Implementierung

Zwecks Vereinfachung des Kalibrierungsprozesses wurde ein Assistent in Form einer graphischen Benutzerschnittstelle programmiert.

Im ersten Schritt des Assistenten wird dabei optional eine intrinsische Kamerakalibrierung durchgeführt (siehe Abbildung 4.1).

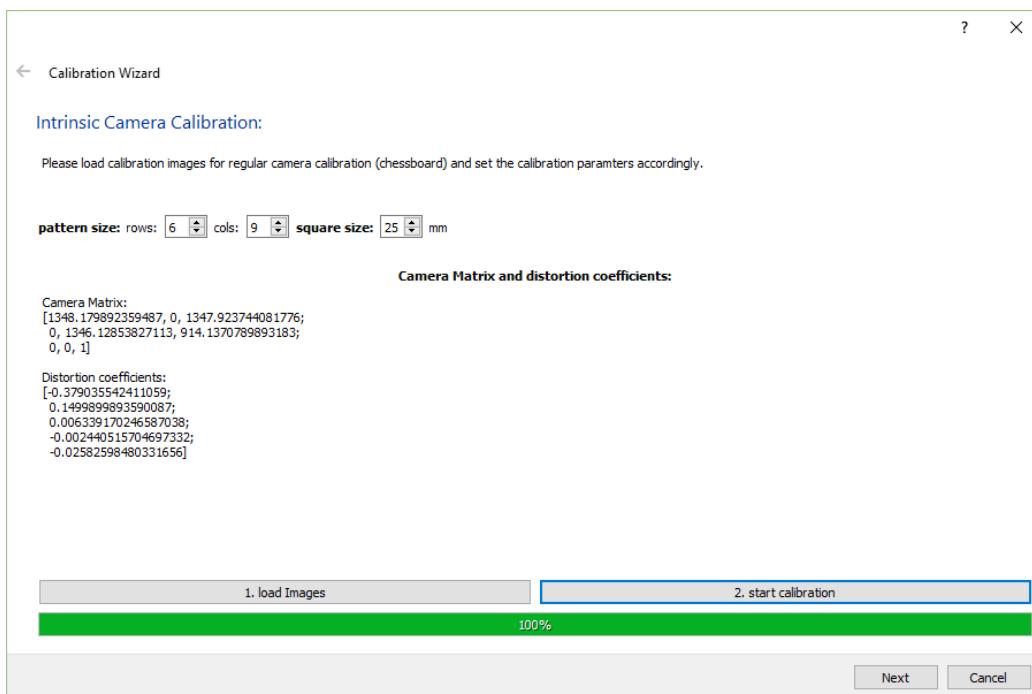


Abbildung 4.1.: Kalibrierungsassistent: intrinsische Kamerakalibrierung

Anschließend wird die eigentliche Kegelkalibrierung ausgeführt. Dabei wird zunächst das Kalibrierungsbild geladen und gegebenenfalls nach einer stattgefundenen intrinsischen Kalibrierung entzerrt. Es werden nun die Sample-Positionen detektiert und der Nutzer hat die Möglichkeit zu überprüfen, ob alle Positionen korrekt detektiert wurden und andernfalls fehlerhafte Punkte zu entfernen und / oder Punkte hinzuzufügen. Im Anschluss werden die Ellipsen und Liniensegmente bestimmt und Punktcorrespondenzen hergestellt (siehe Abbildung 4.2).

Im letzten Schritt kann zwischen beiden Entfaltungsverfahren gewählt werden. Anschließend können die Einstellungen in eine XML-Datei exportiert werden, in der, neben der Kamera-Matrix und Verzerrungskoeffizienten, auch die zwei Abbildungsmatrizen des ausgewählten Verfahrens gespeichert werden.

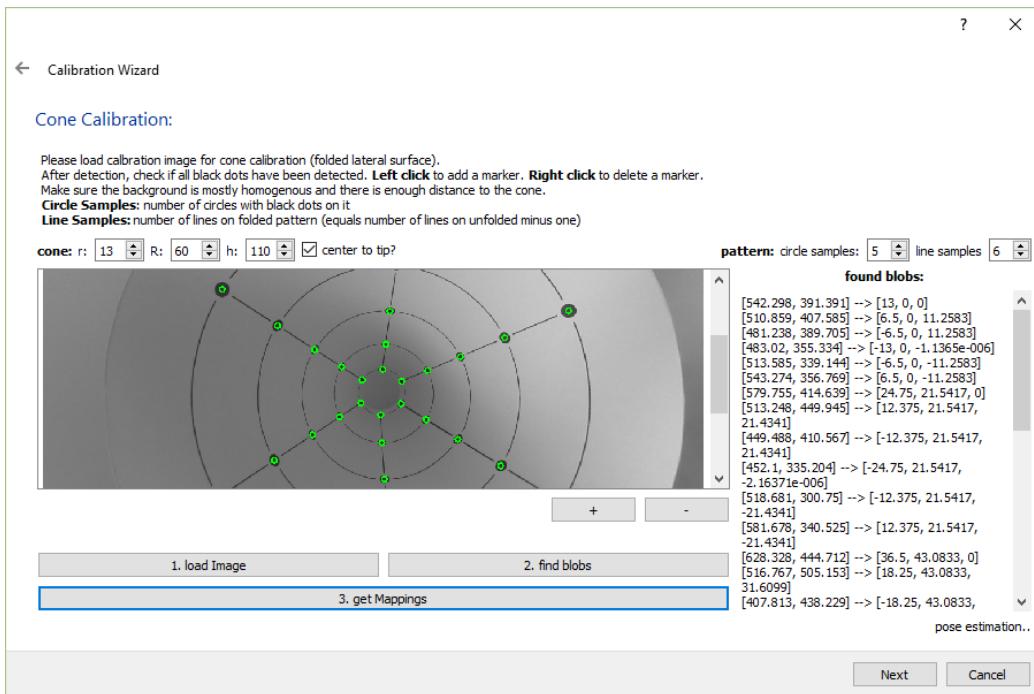


Abbildung 4.2.: Kalibrierungsassistent: Kegelkalibrierung

Die Abbildungsmatrizen sind dabei wie folgt aufgebaut. Als  $dst$  bezeichnen wir das entfaltete Bild.  $src$  ist das Ursprungsbild.

Bei der Vowärtsentfaltung gilt:

$$dst(map_x(x, y), map_y(x, y)) = src(x, y),$$

wobei  $map_x$  und  $map_y$  die Abbildungsmatrizen sind und die gleiche Größe wie das Ursprungsbild haben. Da wird bei der Entfaltung die Größe des Ergebnisbildes brauchen, und diese bei der Erstellung der Abbildungsmatrizen berechnet haben, sind Breite und Höhe, in  $map_x(0, 0)$ , respektive  $map_y(0, 0)$ , kodiert. Wir verlieren damit die Information  $dst(0, 0)$  im Ergebnisbild. Dieses Pixel ist aber ohnehin null (siehe zum Beispiel Abbildung 3.14 in Kapitel 3.7).

Bei der Rückwärtsentfaltung gilt:

$$dst(x, y) = src((map_x(x, y), map_y(x, y)),$$

wobei  $map_x$  und  $map_y$  wieder die Abbildungsmatrizen sind und diesmal gleiche Größe wie das entfaltete Bild haben. Eine Kodierung wie bei der Vowärtsentfaltung ist hier also nicht notwendig.

analyse  
einleitung  
schreiben

# 5. Analyse

In diesem Kapitel...

Die Laufzeitanalysen wurden auf dem Raspberry pi gemessen

Raspberry Pi 2 Model B:

- 900MHz ARM Cortex-A7 CPU
- 1GB RAM
- GCC-4.9.2
- OpenCV 2.4.13

## 5.1. Vergleich Vorwärtsentfaltung und Rückwärtsentfaltung

In Kapitel 3 haben wir festgestellt, dass das entfaltete Bild mittels Vorwärtsentfaltung Löcher enthält. Es ist hierbei von Interesse wie sich die Anzahl der Löcher bei einer Änderung der Auflösung verhält. Der Einfluss der Ausgabeauflösung auf die Anzahl der Löcher lässt sich in Abbildung 5.1 ablesen.

Wie erwartet verhält sich die Anzahl der Löcher quadratisch zur gewählten Ausgabeauflösung. Der Informationsgehalt (Pixelanzahl) des Ursprungsbildes bleibt natürlich trotz erhöhter Ausgabeauflösung konstant. Da sich die Auflösung der Seitenhöhe quadratisch zur Gesamtanzahl der Pixel verhält, ist auch ein quadratisches Wachstum der Anzahl der Löcher zu erwarten.

Als Reprojektionsfehler einer Abbildung wird die Distanz zwischen einem gemessenen Punkt und einem korrespondierendem projizierten Punkt bezeichnet.

Im Falle der Rückwärtsentfaltung sind die gemessenen Punkte die Bildpositionen der Samples im Ursprungsbild. Da die Geometrie des Kegels bekannt ist, wissen wir wo die Samples auf dem entfalteten Bild sein müssen (siehe Parametrisierung der Mantelfläche 2.3 in Kapitel 2). Wir können nun diese Positionen mit Hilfe der Abbildung zur Entfaltung und der Projektionsmatrix zurück auf das Ursprungsbild abbilden. Diese Punkte sind die projizierten Punkte. Da die Abbildung von der Mantelfläche zur Kegeloberfläche exakt ist, ist der Reprojektionsfehler der Rückwärtsentfaltung alleine durch die Projektionsmatrix definiert.

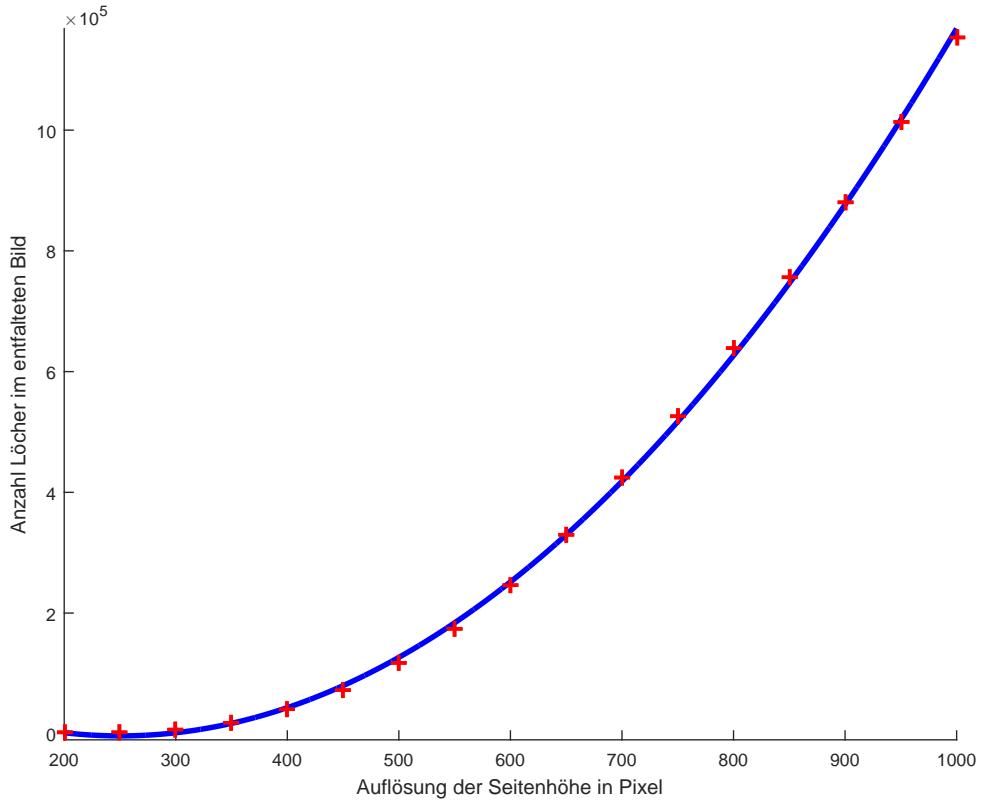


Abbildung 5.1.: Einfluss der Ausgabeauflösung auf die Anzahl der Löcher

Bei der Vorwärtsentfaltung sind die gemessenen Punkte gegeben durch die bekannten Sample-Positionen auf der Mantelfläche. Die Projizierten erhält man, nach der Abbildung der detektierten Sample-Positionen des Ursprungsbild auf die Mantelfläche. Der Reprojektionsfehler ist also alleine durch die Genaugkeit der Sample-Detektion definiert und somit bei diesem Verfahren immer nahe null.

Obwohl das Endergebnis, bedingt durch die Löcher, bei der Vorwärtsentfaltung optisch schlechter ist, ist der Reprojektionsfehler also bei der Vorwärtsentfaltung immer kleiner. Als Vergleich zwischen den beiden Verfahren eignet sich der Reprojektionsfehler also nicht.

Wir entscheiden uns rein optisch und auf Grund der schlechteren Laufzeit bei der Vorwärtsentfaltung (siehe Abbildung 5.2) für die Rückwärtsentfaltung. Alle weiteren Auswertungen beziehen sich von nun an auf die Rückwärtsentfaltung.

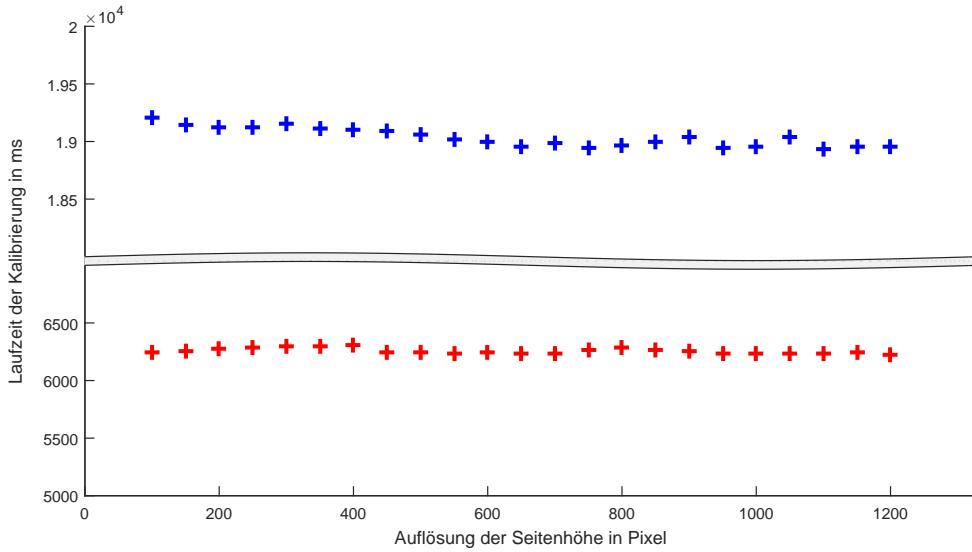


Abbildung 5.2.: Laufzeitvergleich zwischen Vorwärtsentfaltung (blau) und Rückwärtsentfaltung (rot)

## 5.2. Einfluss der intrinsischen Kalibrierung

Ein wichtiger Einflussfaktor auf die Qualität der Entfaltung ist die intrinsische Kamera-Kalibrierung, die vor der eigentlichen Kegelkalibrierung stattfindet. Ihre Hauptaufgabe besteht darin, die Linsenverzerrungen der Kamera zu kompensieren.

Wir messen den Einfluss der Kamerakalibrierung mit Hilfe des Reprojektionsfehlers. Dazu betrachten wir fünf verschiedene Bilder, die ein Mal mit und ein Mal ohne intrinsische Kalibrierung entfaltet werden. Bei beiden Gruppen wird anschließend jeweils der Reprojektionsfehler bestimmt und verglichen. Die Ergebnisse sind in 5.3 zu sehen. In der linken Abbildung sind hierbei die Fehler im kalibrierten Fall zu erkennen, im Linken die Unkalibrierten. In der Abbildung ist dabei ein Kreuz bei  $(u, v)$ , falls die Abweichung des projizierten Punktes in  $x$ -Richtung  $u$ , sowie in  $y$ -Richtung  $v$  beträgt.

Es ist klar zu erkennen, dass die Reprojektionsfehler bei den Bildern ohne intrinsische Kamerakalibrierung wesentlich größer ist. Der starke Einfluss kommt unter Anderem daher, dass wir eine Weitwinkelkamera mit starker Tonnenverzerrungen eingesetzt haben. Ohne eine Modellierung der Linsenverzerrungen weichen die Abstände zwischen den Sample-Positionen stark von der Realität ab. Die Projektionsmatrix wird mit fehlerhaften Daten bestimmt.

## 5.3. Einfluss der Rotation der Kamera

Um den Einfluss der Rotation der Kamera zu untersuchen, wurde der Kegel mit Kalibrierungsmuster in Blender gerendert, da die Kameraposition dann genau bekannt ist und

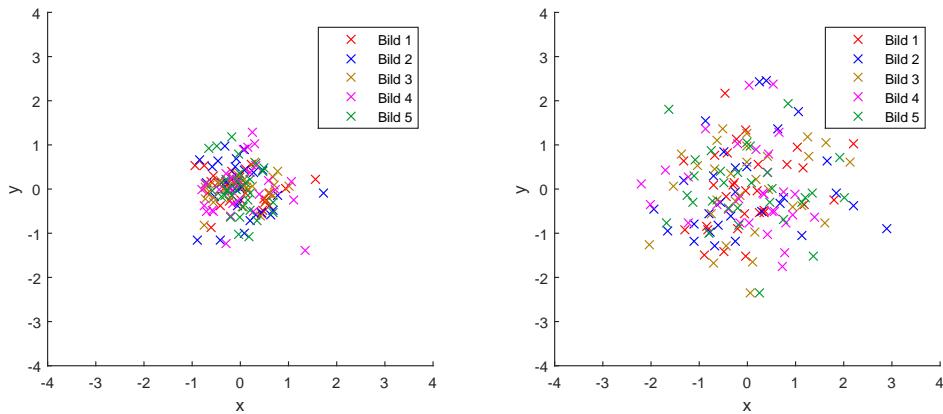


Abbildung 5.3.: Einfluss der intrinsischen Kalibrierung auf den Reprojektionsfehler

äußere Faktoren wie Lichtverhältnisse und inhomogene Hintergründe kontrolliert werden können.

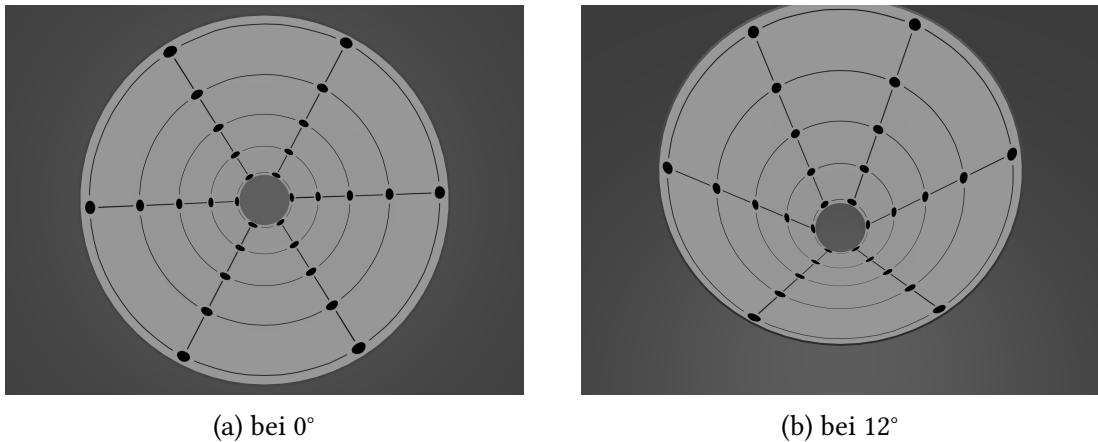


Abbildung 5.4.: gerendeter Kegel mit Kalibrierungsmuster in Blender aus verschiedenen Blickrichtungen

Es werden anschließend Bilder erzeugt, in denen in  $1^\circ$  Schritten die Kamera von  $0^\circ$  bis  $12^\circ$  um die X-Achse rotiert wird. Für jedes dieser Bilder wird anschließend eine Rückwärtsentfaltung durchgeführt und der durchschnittliche Reprojektionsfehler bestimmt. Abbildung 5.5 zeigt, dass der Reprojektionsfehler relativ rotationsinvariant ist.

## 5.4. Laufzeit der Entfaltung

robustheit reverse warping? bestimmung der keypoints?

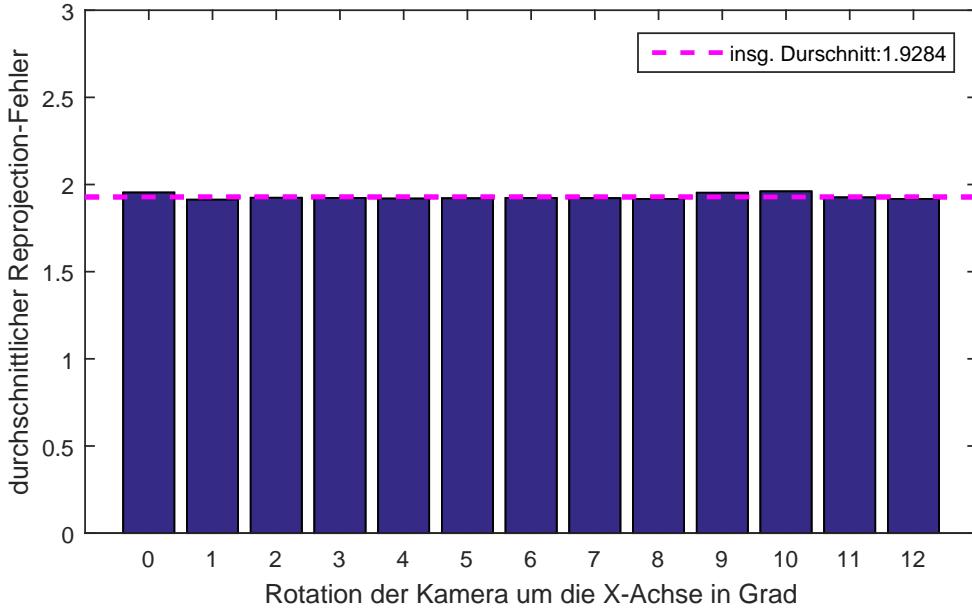


Abbildung 5.5.: Einfluss der Rotation der Kamera

## 5.5. Evaluierung des RANSAC zur Ellipsendetektion

Die robuste Ellipsendetektion ist ein wichtiger Schritt bei der Entfaltung. Bei beiden Verfahren werden die bestimmten Ellipsen genutzt um Korrespondenzen zwischen den Sample-Positionen und Punkten auf dem Kegel im Weltkoordinatensystem herzustellen. Bei der Vorwärtsentfaltung werden die Ellipsen darüber hinaus benötigt, um für die restlichen Pixel geeignete 3D-Koordinaten interpolieren zu können. Es ist also von großem Interesse wie gut die Ellipsendetektion mittels RANSAC funktioniert.

In unserem Verfahren ist eine Gleichverteilung der Ausreißer unwahrscheinlich. Viel wahrscheinlicher ist es, dass eine der nächst äußeren Ellipsen frühzeitig sichtbar wird (siehe Kapite 3.4).

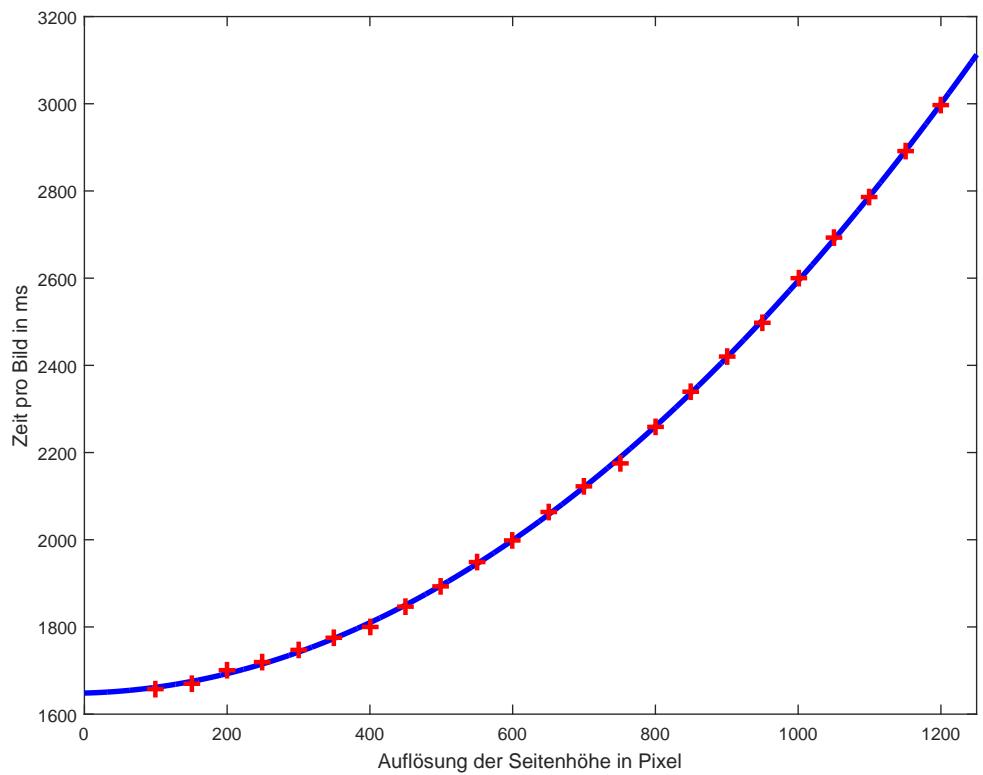


Abbildung 5.6.: Einfluss der Ausgabeauflösung auf die Laufzeit der Entfaltung

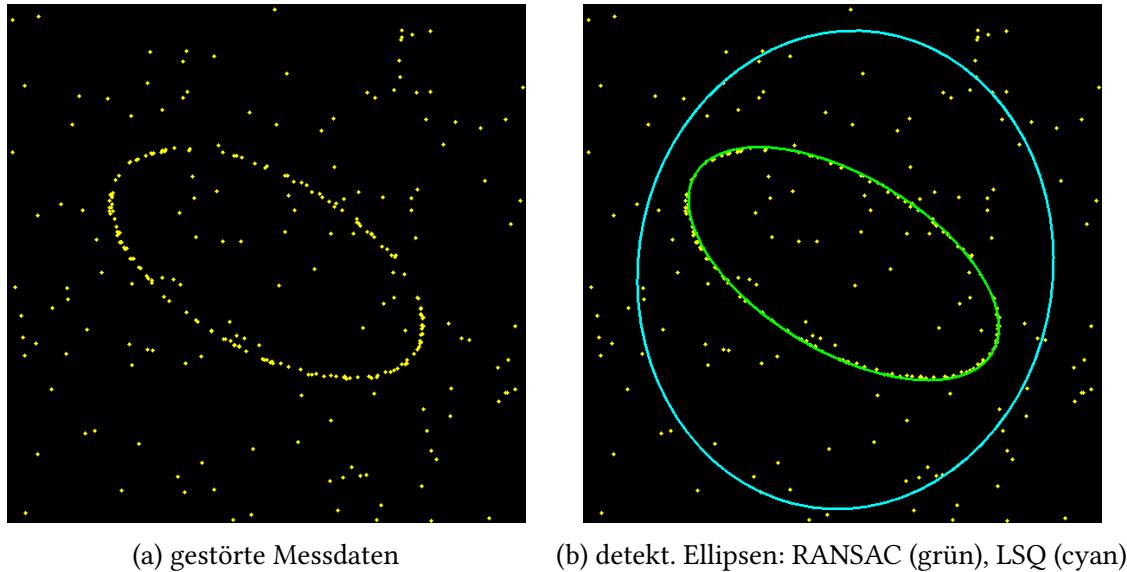


Abbildung 5.7.: Vergleich RANSAC und LSQ bei gleichverteilten Ausreißern  $\epsilon = 0.5, p = 0.99$

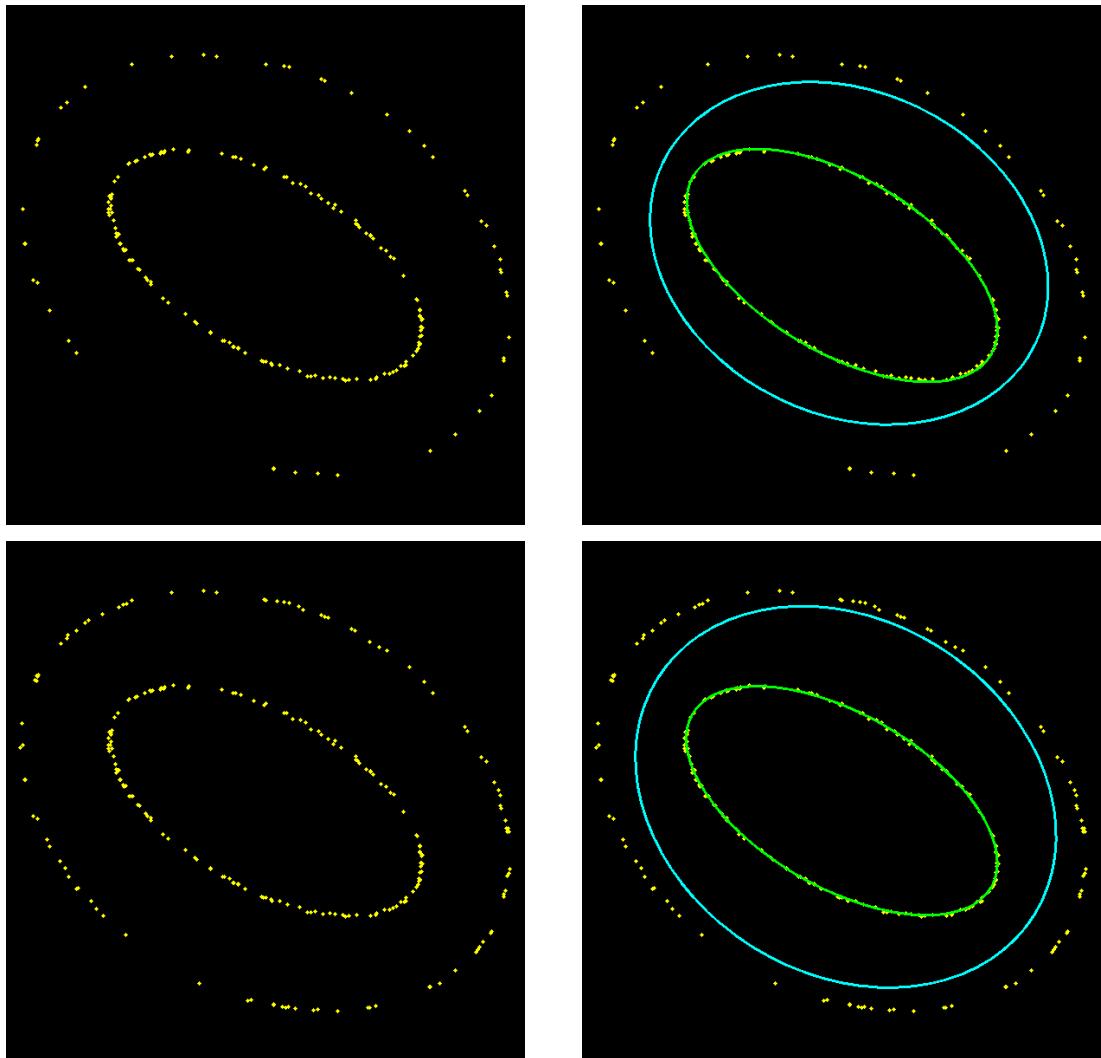


Abbildung 5.8.: Vergleich RANSAC und LSQ bei Schattenellipsen mit  $p = 0.99$  und  $\epsilon = 0.25$  (oben),  $\epsilon = 0.4$  unten, links gestörte Messdaten, rechts detektierte Ellipsen RANSAC (grün), LSQ (cyan)



# 6. Fazit und Ausblick

## 6.1. Parallelisierung

eigentlich quatsch, weil raspberr pi

## 6.2. Verbesserung der Linien-Detektion

Um die Linien-Detektion etwas genauer zu machen, könnte man die Gradientenrichtung des Kantenbilds miteinbeziehen. Dies reduziert die Anzahl falscher Votes und verbessert darüber hinaus die Laufzeit [OC76].

probabilistic hough?

## 6.3. Verbesserung der Vorwärtsentfaltung

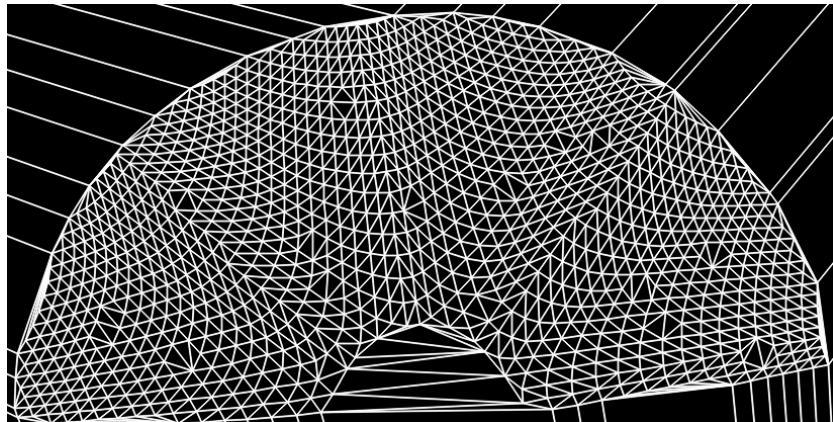
Das Hauptproblem der Vorwärtsentfaltung sind die Löcher auf dem entfalteten Bild. Um diese schließen zu können, könnte man eine Delaunay-Triangulation durchführen. Da das Verfahren jedoch ohne Triangulation schon relativ rechenintensiv ist, und die Rückwärtsentfaltung sehr gute Ergebnisse liefert wurde dieses Möglichkeit nicht weiter untersucht. Solch eine Triangulation ist beispielhaft in Abbildung 6.1 abgebildet. Zum Zwecke der Veranschaulichung wurde hierbei nur ein Teil der Daten benutzt.

## 6.4. Verbesserung der Rückwärtsentfaltung

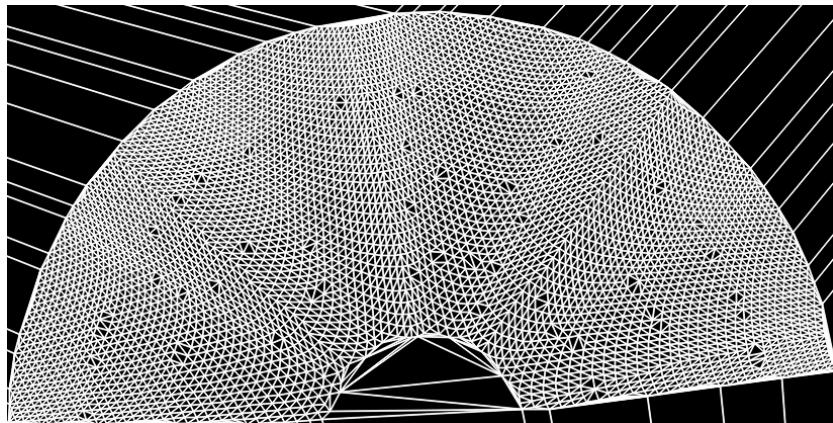
Aktuell wird die Projektionsmatrix bei der Rückwärtsentfaltung durch *Direct Linear Transformation* bestimmt. Dieses Verfahren minimiert jedoch nicht den Reprojection-Fehler. Beser wäre hier deshalb ein iteratives Verfahren, wie der Levenberg–Marquardt-Algorithmus. Im Gegensatz dazu könnte auch ein RANSAC-Ansatz verwendet werden. Statt eine optimale Lösung für alle detektierte Samples zu bestimmen, berechnet man wiederholt für sechs Punkte<sup>1</sup> eine Projektionsmatrix und untersucht dann den Reprojection-Fehler für alle anderen Punkte.

---

<sup>1</sup>6 Punkte sind mindestens notwendig um eine Projektionsmatrix bestimmen zu können, da es elf unbekannte gibt (siehe Kapitel 2.3).



(a) Triangulation mit 10% der Punkte



(b) Triangulation mit 40% der Punkte

Abbildung 6.1.: Delaunay-Triangulation

## 6.5. andere Ansätze

wie kommt  
das hier  
rein?

*Deformable Templates* ist ein Verfahren bei dem eine analytische Kurve, wie beispielsweise eine Ellipse, mittels Optimierung einer Energiefunktion,

Wir betrachten dazu folgende Funktion, deren Nullstellen wie in Kapitel 2.8, eine um  $\theta$  gedrehte Ellipse, mit Zentrum  $(x_0, y_0)$  und Haupt- und Nebenachse  $a$  und  $b$  beschreibt.

$$G(x, y) = \frac{((x - x_0) \cos \theta + (y - y_0) \sin \theta)^2}{a^2} + \frac{((x - x_0) \sin \theta - (y - y_0) \cos \theta)^2}{b^2} - 1$$

Wir konstruieren eine Energiefunktion:

$$E = E_M + E_A + E_S,$$

die sich zusammensetzt aus einem Term  $E_M$ , der die Kantenstärke auf dem Ellipsenrand maximiert, sowie einem Term  $E_A$  der die Winkelähnlichkeit zwischen Ellipsennormale

und Gradientenrichtung maximiert. Darüber hinaus fügen wir einen Term  $E_S$  hinzu, der die Ellipse schrumpfen lässt.

Genauer sehen die Terme wie folgt aus

$$E_M = -\alpha \frac{1}{n} \sum_{i=0}^{n-1} I_M(p_i)$$

$$E_A = \beta \frac{1}{n} \sum_{i=0}^{n-1} \left( I_O(p_i) - \text{atan2}\left(\frac{\partial G}{\partial y}(p_i), \frac{\partial G}{\partial x}(p_i)\right)\right)^2$$

$$E_S = \gamma \frac{1}{n} \sum_{i=0}^{n-2} |p_i - p_{i+1}|,$$

wobei  $n$  die Anzahl der Punkte auf dem Ellipsenrand sind.  $p_i$  ist definiert durch die parametrische Darstellung einer beliebigen Ellipse und ...Man wählt dabei als Startwert eine Ellipse, deren Zentrum dem Bildzentrum entspricht und deren Hauptachse und Nebenachse möglichst groß ist und bestimmt dann numerisch, beispielsweise durch Gradient Descent, ein Minimum der Funktion. Der Term  $E_M$  wird minimal, wenn entlang der Ellipse die Kantenstärke (*Magnitude*) groß ist. Der Term  $E_A$  wird minimal wenn die Winkel der Normalenvektoren ähnlich zu denen der Kanten sind (*Angle*) und  $E_S$  wird minimal, wenn die Distanz aufeinanderfolgender Ellipsepunkte klein wird, also die Ellipse als ganzes klein wird (*Size*). Der Term lässt die Ellipse also schrumpfen.  $\alpha$ ,  $\beta$  und  $\gamma$  steuern hierbei den Einfluss der einzelnen Terme. Das Problem bei diesem Ansatz, ist dass die Funktion auch nach starker Gaussglättung, schnell in kleine lokale Minima läuft, obwohl ein wesentlicher stärkeres Minimum in näherer Umgebung wäre. Darüber hinaus muss für ein robustes Optimierungsverfahren der Gradient und im besten Fall sogar die Hesse-Matrix zur Verfügung gestellt werden.

ergänzen

In Abbildung 6.2 sieht man beispielhaft einen Auschnitt der Energiefunktion, wobei in diesem Fall zwecks Veranschaulichung nur die Haupt- und Nebenachse der Ellipse variable sind. Das Zentrum bleibt das Zentrum des Bildes und der Winkel  $\theta$  ist konstant null. Trotz dieser Einschränkungen lässt sich gut erkennen, dass circa bei  $a = b = 300$  ein starkes Minimum liegt. Es gibt jedoch neben diesem Minimum in näherer Umgebung kleine irrelevante Minima, wie beispielsweise in (360, 260). Solche „Becken“ erschweren die Suche nach der nächsten Ellipse erheblich.

Literatur  
checken,  
ins besonde-  
re seitenan-  
gaben

## 6. FAZIT UND AUSBLICK

---

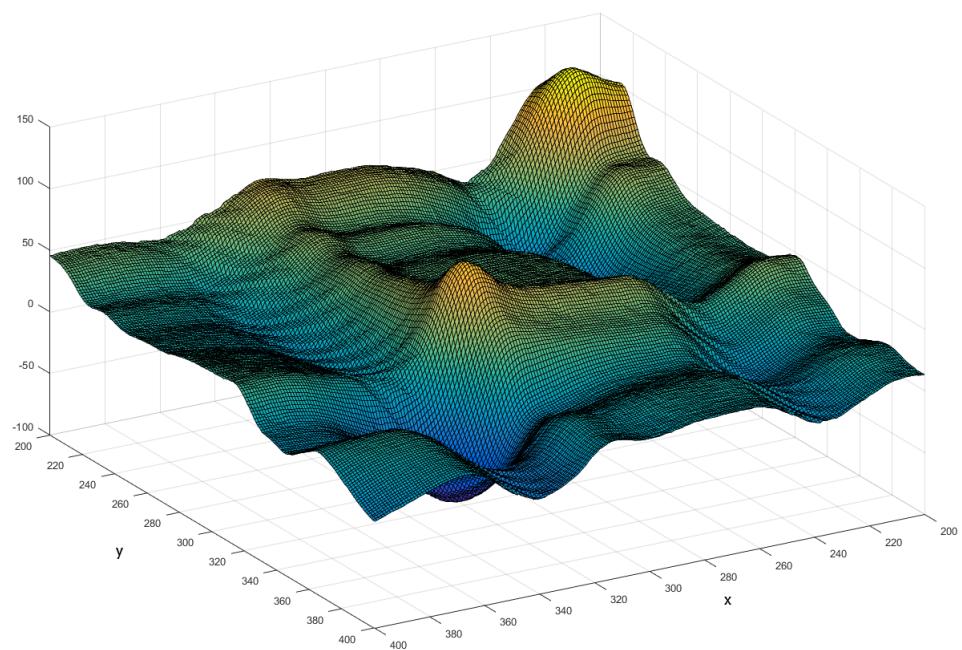


Abbildung 6.2.: Deformable Templates: Ausschnitt der Energiefunktion für variable Haupt- und Nebenachse

## A. Ergebnisse

---

muss das  
hier über-  
haupt rein?

res	200	250	300	350	400	450	500	550	600	650	700	750	800	850	900	950	1000
holes	1623	2721	7799	19337	40168	72349	116576	173543	245216	329354	423877	527094	638029	756185	881294	1013880	1153448

Tabelle A.1.: test123

Auflösung	100	150	200	250	300	350	400	450	500	550	600	650	700	750	800	850	900	950	1000				
vorwärts	6245	6256	6276	6290	6294	6300	6305	6344	6240	6239	6243	6236	6236	6270	6285	6266	6252	6238	6237	6239	6240	6224	
rückwärts	19202	19147	19126	19127	19154	19114	19102	19092	19056	19013	18993	18952	18989	18949	18964	19002	19034	18949	18956	19043	1893	18951	18954

Tabelle A.2.: test445

# Abbildungsverzeichnis

2.1.	Gerader Kreiskegel . . . . .	3
2.2.	Kegelstumpf und Ergänzungskegel . . . . .	4
2.3.	Kegelstumpf . . . . .	5
2.4.	Kegelmantelfläche . . . . .	5
2.5.	Abbildung der Kegelstumpfhöhe auf die Seitenhöhe . . . . .	6
2.6.	Lochkameramodell . . . . .	9
2.7.	Projektion eines Punktes $P$ im Weltkoordinatensystem $(x_w, y_w, z_w)$ auf die Bildebene im Kamerakoordinatensystem $(x_c, y_c, z_c)$ . . . . .	11
2.8.	Ellipse mit Brennpunkten $f_1, f_2$ , Zentrum $C$ , Hauptachse $a$ , Nebenachse $b$ und Scheitelpunkten $V_1$ und $V_2$ . . . . .	14
2.9.	Ellipse mit Zentrum $(x_0, y_0)$ , Hauptachse $a$ , Nebenachse $b$ , sowie Drehwinkel $\theta$ . . . . .	14
2.10.	Ellipsenausschnitt im ersten Quadranten mit Abfragepunkt $Q$ und eingezeichneter kürzester Distanz zur Ellipse . . . . .	18
3.1.	Kalibrierungsmuster von oben mit $n = 5, m = 6$ . . . . .	22
3.2.	Kalibrierungsmuster entfaltet mit $n = 5, m = 6$ . . . . .	22
3.3.	Kamerakalibrierung . . . . .	23
3.4.	Detektion der Samples . . . . .	24
3.5.	Canny-Kantendetektion auf Grauwertbild . . . . .	25
3.6.	Hough-Transformation zur Linien-Detektion (in rot gekennzeichnet) und bestimmter Schnittpunkt (in grün) . . . . .	26
3.7.	Ellipsendetektion: bestimme Pixel-Positionen (weiß), Aussendepunkt (gelb) . . . . .	26
3.8.	Ellipsendetektion bei Ausreißern . . . . .	27
3.9.	detektierte Ellipsen . . . . .	28
3.10.	Winkel zwischen Samples und Mittelpunkt $C$ sind nicht identisch . . . . .	29
3.11.	Zuordnung von Punkten zu Ellipsen (links) und Liniensegmenten (rechts) . . . . .	29
3.12.	Interpolation der 3D-Koordinaten . . . . .	31
3.13.	interpolierter Kegel . . . . .	32
3.14.	Vorwärtsentfaltung . . . . .	33
3.15.	Rückwärtsentfaltung . . . . .	33
4.1.	Kalibrierungsassistent: intrinsische Kamerakalibrierung . . . . .	35
4.2.	Kalibrierungsassistent: Kegelkalibrierung . . . . .	36
5.1.	Einfluss der Ausgabeauflösung auf die Anzahl der Löcher . . . . .	38

## ABBILDUNGSVERZEICHNIS

---

5.2.	Laufzeitvergleich zwischen Vorwärtsentfaltung (blau) und Rückwärtsentfaltung (rot) . . . . .	39
5.3.	Einfluss der intrinsischen Kalibrierung auf den Reprojektionsfehler . . . . .	40
5.4.	gerendeter Kegel mit Kalibrierungsmuster in Blender aus verschiedenen Blickrichtungen . . . . .	40
5.5.	Einfluss der Rotation der Kamera . . . . .	41
5.6.	Einfluss der Ausgabeauflösung auf die Laufzeit der Entfaltung . . . . .	42
5.7.	Vergleich RANSAC und LSQ bei gleichverteilten Ausreißen $\epsilon = 0.5, p = 0.99$	42
5.8.	Vergleich RANSAC und LSQ bei Schattenellipsen mit $p = 0.99$ und $\epsilon = 0.25$ (oben), $\epsilon = 0.4$ unten, links gestörte Messdaten, rechts detektierte Ellipsen RANSAC (grün), LSQ (cyan) . . . . .	43
6.1.	Delaunay-Triangulation . . . . .	46
6.2.	Deformable Templates: Ausschnitt der Energiefunktion für variable Haupt- und Nebenachse . . . . .	48

# Literatur

- [Can86] John Canny. „A Computational Approach to Edge Detection“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8.6 (1986), S. 679–698. ISSN: 01628828. DOI: 10.1109/TPAMI.1986.4767851 (siehe S. 12).
- [Ebe13] David Eberly. „Distance from a Point to an Ellipse, an Ellipsoid, or a Hyperellipsoid“. In: (2013), S. 1–13 (siehe S. 17).
- [FB81] Martin a Fischler und Robert C Bolles. „Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography“. In: *Communications of the ACM* 24.6 (1981), S. 381–395. ISSN: 0001-0782. DOI: 10.1145/358669.358692. URL: <http://dx.doi.org/10.1145/358669.358692> (siehe S. 13).
- [HS97] Janne Heikkilä und Olli Silvén. „A Four-step Camera Calibration Procedure with Implicit Image Correction.“ In: *Cvpr* (1997), S. 1106–1112. ISSN: 1063-6919. DOI: 10.1109/CVPR.1997.609468. URL: <http://dx.doi.org/10.1109/CVPR.1997.609468> (siehe S. 9, 10).
- [Law72] J D Lawrence. *A Catalog of Special Plane Curves*. Dover Publications, 1972, S. 62–63 (siehe S. 15, 16).
- [Lin93] Tony Lindeberg. „Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention“. In: *International Journal of Computer Vision* 11.3 (1993), S. 283–318. ISSN: 09205691. DOI: 10.1007/BF01469346 (siehe S. 11).
- [OC76] F. O’Gorman und M. B. Clowes. „Finding Picture Edges Through Collinearity of Feature Points“. In: *IEEE Transactions on Computers* C-25.4 (Apr. 1976), S. 449–456. ISSN: 0018-9340. DOI: 10.1109/TC.1976.1674627 (siehe S. 45).
- [Sto07] Burlisch Stoer. *Numerische Mathematik 1*. Springer, 2007, S. 249–253 (siehe S. 8).
- [Sto11] Burlisch Stoer. *Numerische Mathematik 2*. Springer, 2011, S. 21–23 (siehe S. 8).
- [Tsa87] Roger Y. Tsai. „A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses“. In: *IEEE Journal on Robotics and Automation* 3.4 (1987), S. 323–344. ISSN: 08824967. DOI: 10.1109/JRA.1987.1087109 (siehe S. 9, 10).



# Plagiatserklärung

Hiermit versichere ich, dass die vorliegende Arbeit über

*Entzerrung von Kegeloberflächen aus einer Einkameraansicht basierend auf projektiver Geometrie*

selbstständig verfasst worden ist, dass keine anderen Quellen und Hilfsmittel als die angegebenen benutzt worden sind und dass die Stellen der Arbeit, die anderen Werken – auch elektronischen Medien – dem Wortlaut oder Sinn nach entnommen wurden, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht worden sind.

---

Lars Haalck, Münster, 11. September 2016

Ich erkläre mich mit einem Abgleich der Arbeit mit anderen Texten zwecks Auffindung von Übereinstimmungen sowie mit einer zu diesem Zweck vorzunehmenden Speicherung der Arbeit in eine Datenbank einverstanden.

---

Lars Haalck, Münster, 11. September 2016