

Liste der noch zu erledigenden Punkte

dieses kapitel sollte aussagekräftig sein, da projektive geometrie im titel vorkommt	7
möglicherweise Reihenfolge hier ändern, so wie sie auch später benutzt werden . . .	16
nicht ohlbergers skript zitieren	18
warum gerade dieses muster? warum ist das so gut? wie berechnet man das muster?	
welche eigenschaften hat es?	23
beschreiben, warum die Hauptachsentransformation gebraucht wird	26
hier schon erläutern oder erst in späteren kapiteln?	30
gehören Probleme schon hier her? oder erst bei analyse?	31
soll ich hier auf den ausblick verweisen?	32
Literatur checken, ins besondere seitenangaben	43



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

In der vorliegenden Arbeit

Inhaltsverzeichnis

1 Einleitung	1
2 Theoretische Grundlagen	3
2.1 Kegel	3
2.2 Kamerakalibrierung und projektive Geometrie	7
2.3 Ellipsen	10
2.3.1 allgemein	10
2.3.2 Abstand: Punkt zu Ellipse	13
2.4 RANSAC	16
2.5 Canny	16
2.6 Hough-Transformation	17
2.7 Blob-Detektor	17
2.8 Ausgleichsproblem	18
3 Methodik	21
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	33
4 Implementierung	35
5 Analyse	37
5.1 Vergleich Vorwärtsentfaltung und Rückwärtsentfaltung	37
5.2 Einfluss der intrinsischen Kalibrierung	38
5.3 Einfluss der Rotation der Kamera	38
5.4 Laufzeit der Entfaltung	39
5.5 Evaluierung des RANSAC zur Ellipsendetektion	39
6 Fazit und Ausblick	43

INHALTSVERZEICHNIS

Abbildungsverzeichnis	45
Tabellenverzeichnis	47

1 Einleitung

einleitung

hier setup mit bild

2 Theoretische Grundlagen

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

Wir benutzen ein linkshändiges Koordinatensystem.

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 bezeichne hierbei die Seitenhöhe und ist definiert durch $S = \sqrt{H^2 + R^2}$ (siehe Abbildung 2.1).

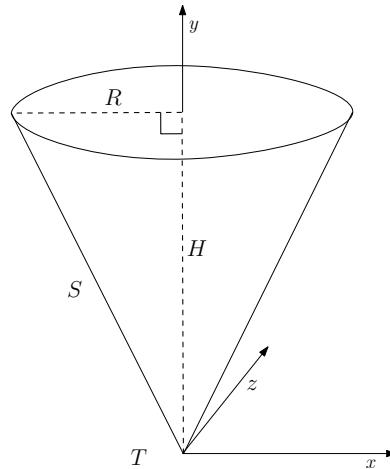


Abbildung 2.1: Gerader Kreiskegel

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).

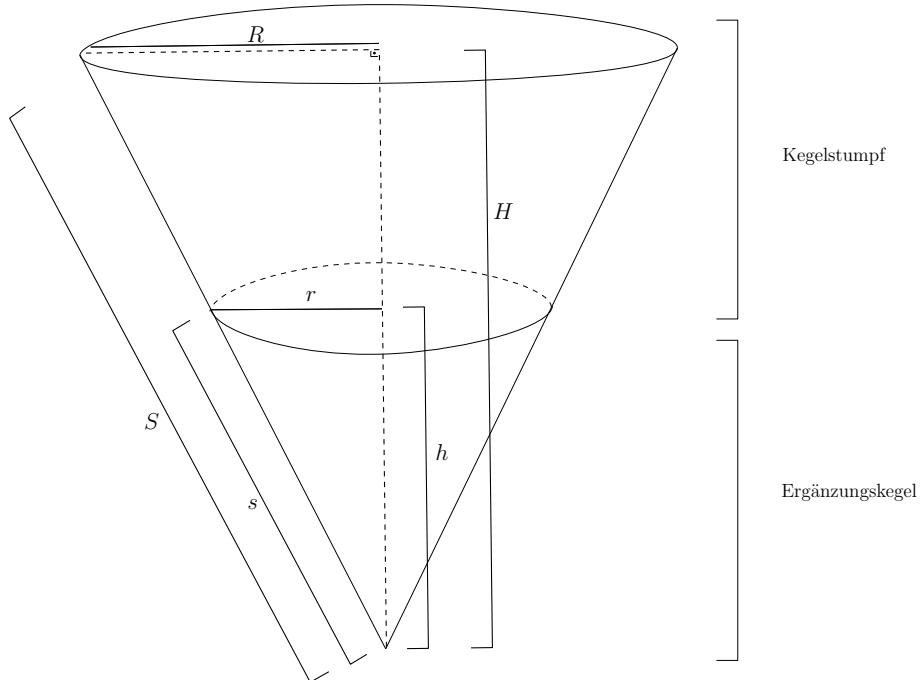


Abbildung 2.2: Kegelstumpf und Ergänzungskegel

Analog zum Kreiskegel definieren wir einen Kegelstumpf 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-

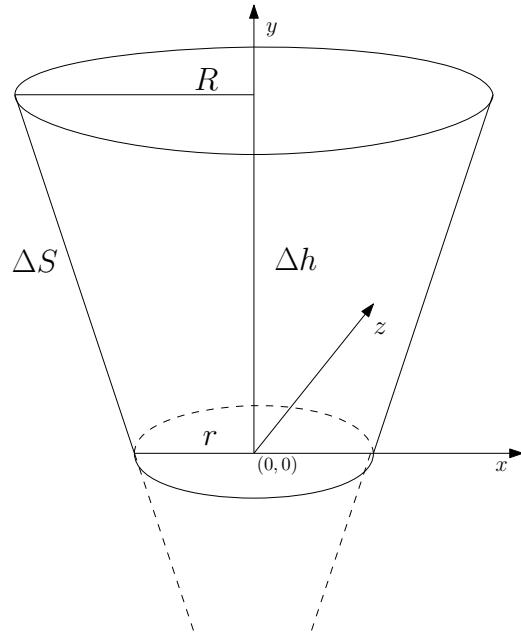


Abbildung 2.3: Kegelstumpf

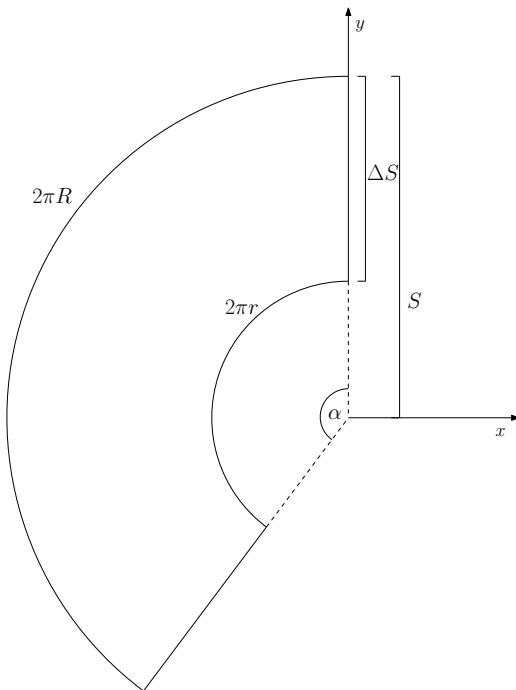


Abbildung 2.4: Kegelmantelfläche

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

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

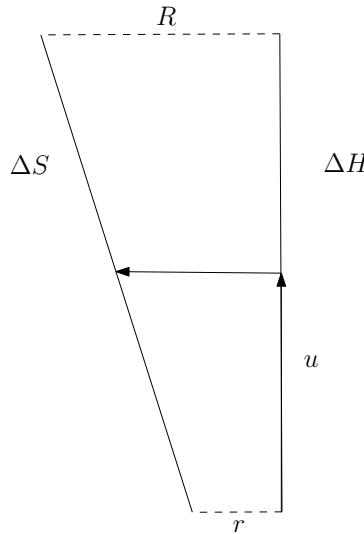


Abbildung 2.5: Abbildung der Kegelstumpfhöhe auf die Seitenhöhe

Da $R, r, \Delta H$ und nun auch die Höhe bekannt sind, kann man den Winkel θ 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 ϕ mit inverser Skalierung also analog 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 Kamerakalibrierung und projektive Geometrie

2.2.1 Definition

Kamerakalibrierung ist ein Verfahren, bei dem die interne Kamerageometrie und optischen Eigenschaften (intrinsische Parameter) und/oder die 3D-Position und Orientierung von Bildpunkten bestimmt werden.

dieses Kapitel sollte aussagekräftig sein, da projektive Geometrie im Titel vorkommt

tierung 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).

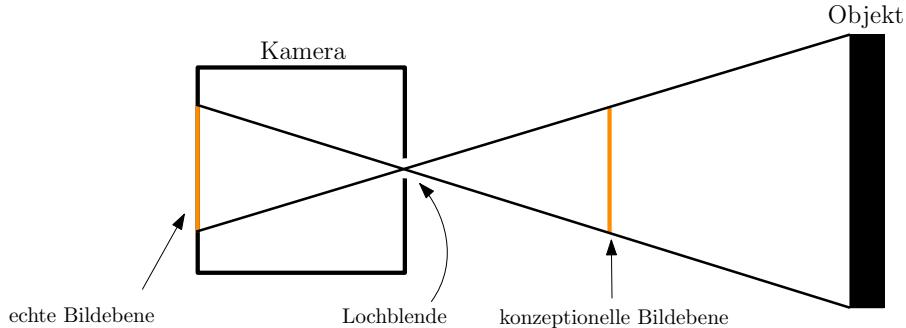


Abbildung 2.6: Lochkameramodell

Das Bild das an der Rückseite der Kamera entsteht ist dabei um 180° 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. Genauer suchen wir eine projektive Abbildung:

$$\begin{pmatrix} wu \\ wv \\ w \end{pmatrix} = \underbrace{\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{pmatrix}}_{=:A} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \quad (2.7)$$

$$\begin{aligned} u &= \frac{a_{11}x + a_{12}y + a_{13}z + a_{14}}{a_{31}x + a_{32}y + a_{33}z + a_{34}} \\ v &= \frac{a_{21}x + a_{22}y + a_{23}z + a_{24}}{a_{31}x + a_{32}y + a_{33}z + a_{34}} \end{aligned} \quad (2.8)$$

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 $a_{i,j}$ von den intrinsischen und extrinsischen Parametern der Kamera abhängen [HS97].

Wir konstruieren nun schrittweise die Matrix A :

Zunächst wird das Weltkoordinatensystem wie in Abbildung 2.7 mit einer Rotation und einer Translation in das Kamerakoordinatensystem überführt.

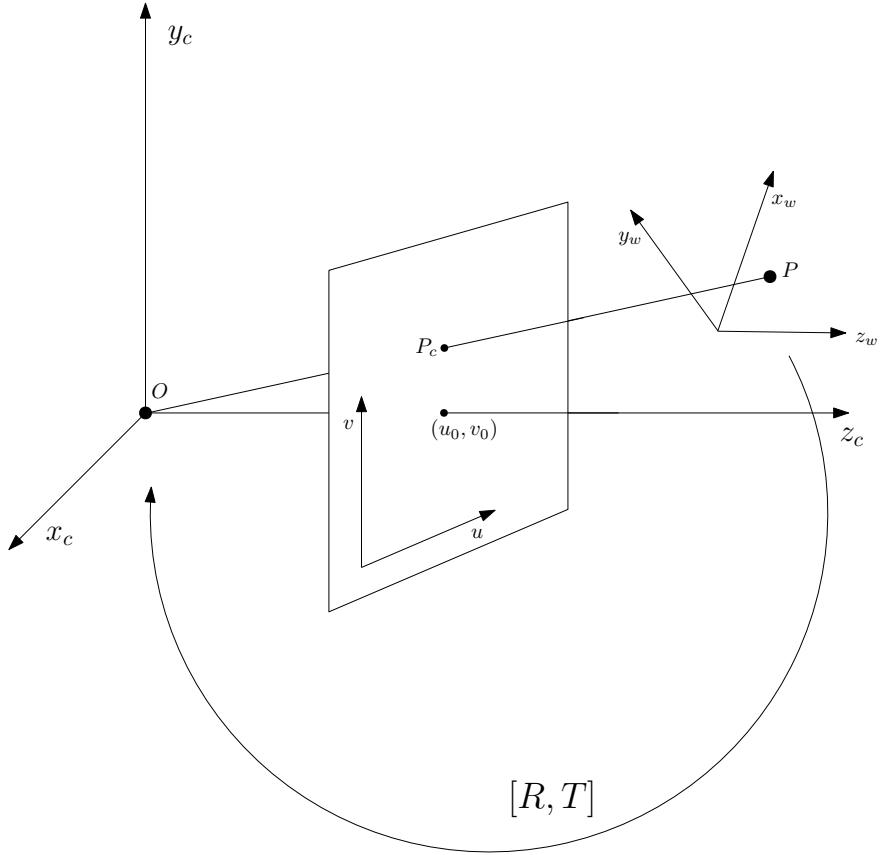


Abbildung 2.7: Projektion eines Punktes P im Weltkoordinatensystem (x_w, y_w, z_w) auf die Bildebene im Kamerakoordinatensystem (x_c, y_c, z_c)

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

1. Rotation und Translation des Weltkoordinatensystems, um es in das Bildkoordinatensystem zu überführen.
2. Perspektivische Division und Skalierung mit der Brennweite
3. Bestimmen der Verzerrungskoeffizienten κ_1 und κ_2

Selfcalibrating vs bla bla

erst mal außen vor. erst nach dem rest Zu den intrinsischen Parametern gehören neben Brennweite f der Kamera, die Bildmitte (auch Bildmittelpunkt) (u_0, v_0) , metrische Pixelgrößen s_x und s_y , sowie Verzerrungskoeffizienten. Mit Hilfe der intrinsischen und extrinsischen Parameter lässt sich eine

wofür braucht man das? billige linsen, verzerrungen, 3d rekonstruktionen. abbildung von 3d auf 2d, lochkamera, intrinisch, extrinisch Lochkamera

Linsenverzerrungen

projektionsmatrix (homogene Koordinaten????) SVD, QR, LSQ?

2.3 Ellipsen

2.3.1 allgemein

2.3.1 Definition (Ellipse)

Ellipsen entstehen durch Kegelschnitt und so bla bla. im weiteren bla bla meinen wir mit Hauptachsen immer die Semihauptachsen

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

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad (2.9)$$

beschrieben werden.

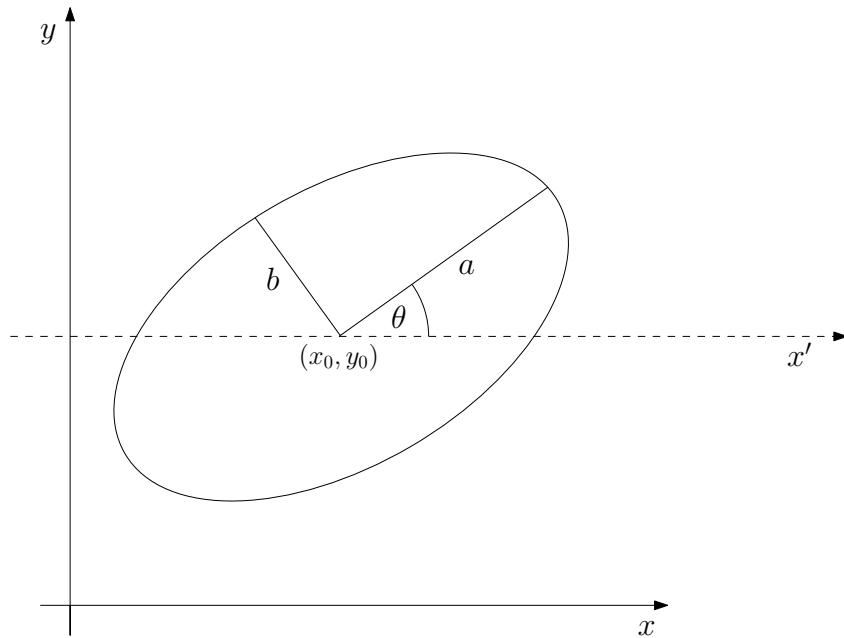


Abbildung 2.8: Ellipse mit Zentrum (x_0, y_0) , Hauptachse a , Nebenachse b , sowie Drehwinkel θ

Befindet sich die Ellipse nicht im Ursprung so muss eine Verschiebung beziehungsweise bei einer Rotation ein Drehwinkel (siehe Abbildung 2.8) ergänzt werden.

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1 \quad (2.10)$$

$$\frac{((x - x_0) \cos \theta + (y - y_0) \sin \theta)^2}{a^2} + \frac{((x - x_0) \sin \theta - (y - y_0) \cos \theta)^2}{b^2} = 1 \quad (2.11)$$

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

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_0 + a \cos \phi \cos \theta - b \sin \phi \sin \theta \\ y_0 + a \cos \phi \sin \theta + b \sin \phi \cos \theta \end{pmatrix} \quad (2.12)$$

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

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

$$ax^2 + by^2 + cxy + dx + ey + f = 0 \quad \text{mit} \quad c^2 - 4ab < 0 \quad (2.13)$$

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 2.11 und 2.13 sind äquivalent, falls die Ellipse nicht degeneriert ist (ohne Beweis). Da wir die Umformung von 2.11 nach 2.13 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 2.13 kann umgeformt werden zu:

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

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

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 2.13 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 $M = S^T DS$, 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:

$$\begin{aligned} u^T(S^T DS)u + (d \quad e) \underbrace{(S^T S)u}_{=1\mathbb{I}} + f &= 0 \\ \Leftrightarrow (Su)^T D(Su) + (d \quad e) S^T(Su) + f &= 0 \\ \Leftrightarrow v^T Dv + (d \quad e) S^T v + f &= 0 \end{aligned} \tag{2.14}$$

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 θ ist definiert durch:

$$R = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \tag{2.15}$$

Es gilt offenbar $S = R$ für ein geeignetes θ , da die Eigenvektoren normiert und orthogonal zueinander sind. θ 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 2.14 aus ergibt sich:

$$\begin{aligned} \lambda_1 v_1^2 + \lambda_2 v_2^2 + (d \quad e) S^T v + f &= 0 \\ \underbrace{\lambda_1 v_1^2 + \lambda_2 v_2^2}_{=: (d', e')} + (d' v_1 + e' v_2) + f &= 0 \\ (\lambda_1 v_1^2 + d' v_1) + (\lambda_2 v_2^2 + e' v_2) + f &= 0 \\ (\lambda_1 v_1^2 + d' v_1) + \left(\frac{d'^2}{4\lambda_1} - \frac{d'^2}{4\lambda_1} \right) + (\lambda_2 v_2^2 + e' v_2) + \left(\frac{e'^2}{4\lambda_2} - \frac{e'^2}{4\lambda_2} \right) + f &= 0 \\ \left[\lambda_1 \left(v_1^2 + \frac{2d'}{2\lambda_1} v_1 + \frac{d'^2}{4\lambda_1^2} \right) - \frac{d'^2}{4\lambda_1} \right] + \left[\lambda_2 \left(v_2^2 + \frac{2e'}{2\lambda_2} v_2 + \frac{e'^2}{4\lambda_2^2} \right) - \frac{e'^2}{4\lambda_2} \right] + f &= 0 \\ \underbrace{\lambda_1 (v_1 + \frac{d'}{2\lambda_1} v_1)^2}_{=-x'_0} + \underbrace{\lambda_2 (v_2 + \frac{e'}{2\lambda_2} v_2)^2}_{=-y'_0} - \underbrace{(\frac{d'^2}{4\lambda_1} + \frac{e'^2}{4\lambda_2} - f)}_{=: \sigma} &= 0, \end{aligned} \tag{2.16}$$

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

Das Zentrum der transformierten Ellipse kann nun aus 2.16 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:

$$\begin{aligned} \lambda_1(v_1 - x'_0)^2 + \lambda_2(v_2 - y'_0)^2 &= \sigma \\ \Leftrightarrow \frac{\lambda_1}{\sigma}(v_1 - x'_0)^2 + \frac{\lambda_2}{\sigma}(v_2 - y'_0)^2 &= 1 \end{aligned} \quad (2.17)$$

wobei $\sigma \neq 0$, wenn die Ellipse nicht zum Punkt entartet ist [Law72]. Vergleicht man nun 2.17 mit 2.10 so sieht man das

$$\begin{aligned} \frac{\lambda_1}{\sigma} = \frac{1}{a^2} \quad \text{und} \quad \frac{\lambda_2}{\sigma} = \frac{1}{b^2} \\ \Leftrightarrow \sqrt{\frac{\sigma}{\lambda_1}} = a \quad \text{und} \quad \sqrt{\frac{\sigma}{\lambda_2}} = b \end{aligned} \quad (2.18)$$

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

2.3.2 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 vom Punkt Q hat.

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

- Die Ellipse kann stets durch die implizite Gleichung

$$\frac{x_0^2}{a^2} + \frac{x_1^2}{b^2} = 1 \quad (2.19)$$

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

$$F(\theta) = |\mathcal{E}(\theta) - Q|^2. \quad (2.20)$$

Die Ableitung von F

$$F'(\theta) = 2(\mathcal{E}(\theta) - Q) \cdot \mathcal{E}'(\theta). \quad (2.21)$$

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 2.9).

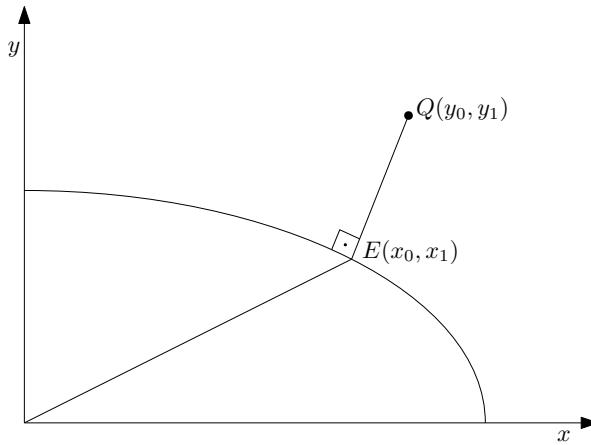


Abbildung 2.9: Ellipsenausschnitt im ersten Quadranten mit Abfragepunkt Q und eingezeichnetem kürzestem Abstand zur Ellipse

Betrachten wir also die Funktion:

$$G(x_0, x_1) = \frac{x_0^2}{a^2} + \frac{x_1^2}{b^2} - 1. \quad (2.22)$$

(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, 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:

$$(y_0, y_1) - (x_0, x_1) = t \frac{\nabla G(x_0, x_1)}{2} = t \left(\frac{x_0}{a^2}, \frac{x_1}{b^2} \right) \quad (2.23)$$

für ein $t \in \mathbb{R}$.

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

$$y_0 = x_0 \left(1 + \frac{t}{a^2} \right), \quad y_1 = x_1 \left(1 + \frac{t}{b^2} \right) \quad (2.24)$$

$$x_0 = \frac{a^2 y_0}{t + a^2}, \quad x_1 = \frac{b^2 y_1}{t + b^2} \quad (2.25)$$

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 2.24

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

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 2.25 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 2.19 $x_1 = b \cdot \sqrt{1 - \left(\frac{x_0}{a}\right)^2}$ 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 2.24 lässt sich folgende Einschränkung für t herleiten:

$$\begin{aligned} 0 < y_0 &= x_0 \left(1 + \frac{t}{a^2}\right) \\ \Leftrightarrow -1 \cdot a^2 &< t. \end{aligned} \tag{2.26}$$

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 2.25 in 2.22 ein, erhält man:

$$F(t) = \left(\frac{ay_0}{t + a^2}\right)^2 + \left(\frac{by_1}{t + b^2}\right)^2 - 1 \tag{2.27}$$

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

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

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.

2.4 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:

$$\frac{\log(1-p)}{\log\left(1-(1-\epsilon)^k\right)} \quad (2.28)$$

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

Hough-Transformation ist ein Verfahren zur Detektion von beliebigen parametrisierbaren Konturen. In dieser Arbeit werden Hough-Transformamtionen benutzt um Geraden zu detektieren.

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

$$x \cos \phi + y \sin \phi - d = 0, \quad (2.29)$$

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, ϕ) 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 ϕ und d einschränken. Hat man beispielsweise einen Punkt (x_k, y_k) gegeben so lässt sich die Gleichung 2.29 nach d umstellen und man erhält eine sinusförmige Funktion in Abhängigkeit von ϕ .

Um beliebige Geraden detektieren zu können, werden ϕ 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 ϕ_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 ϕ_i ein d_i ausgerechnet und entsprechend im Akkumulator \mathcal{H} der Wert an der Stelle (ϕ_i, d_i) erhöht. Am Ende des Verfahrens such man im Akkumulator nach Häufungspunkten. Jeder Häufungspunkt steht dort für eine Gerade.

2.7 Blob-Detektor

2.7.1 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 Größe, Farbe, Konvexität oder Rundheit gefiltert werden.

2.8 Ausgleichsproblem

nicht ohl-
bergers
skript zi-
tieren

Gegeben seien m Messdaten $(x_1, y_1), \dots, (x_m, y_m)$, sowie n Funktionen u_1, \dots, u_n mit $n \leq m$ und $n, m \in \mathbb{N}$. Hier Zitat einfügen. Skript von Ohlberger

Gesucht ist eine Linearkombination $u(x) = \sum_{i=1}^m c_i u_i(x)$ welche die mittlere Abweichung minimiert, also

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

Diese Problem wird als Ausgleichsproblem bezeichnet und ist äquivalent zur Minimierung des Funktionals:

$$F(c) = \|Ac - y\|_2,$$

mit

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

und äquivalent zur Normalengleichung

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

wobei mindestens eine Lösung existiert und die Lösung mit kleinster 2-Norm eindeutig ist.

Gilt darüber hinaus $\text{rang}(A) = n$ so gilt

$$c = \underbrace{(A^T A)^{-1} A^T}_{{A}^+} y \tag{2.30}$$

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

Die Lösung des Ausgleichproblems kann dann mit einer Singulärwertzerlegung (SVD) gelöst werden. 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:

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

3 Methodik

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

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 Position $(0, 0, 0)$ (siehe Abbildung 2.3) 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 ΔR , 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 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 diese Abbildung 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).

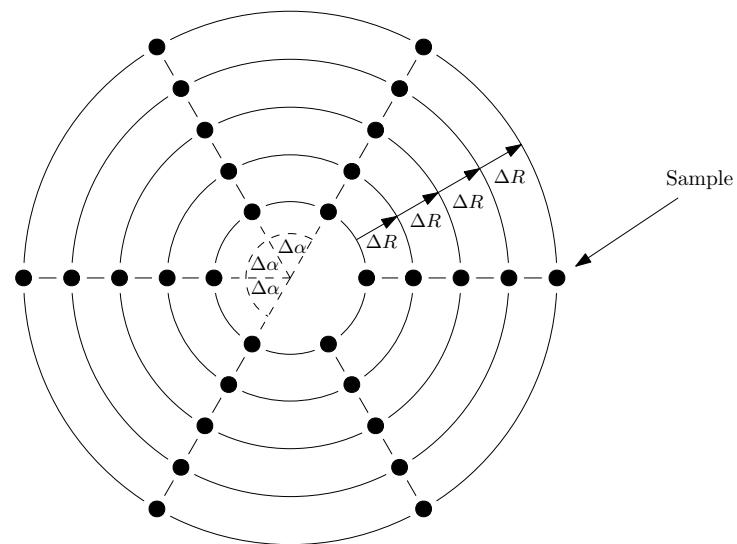


Abbildung 3.1: Kalibrierungsmuster von oben mit $n = 5, m = 6$

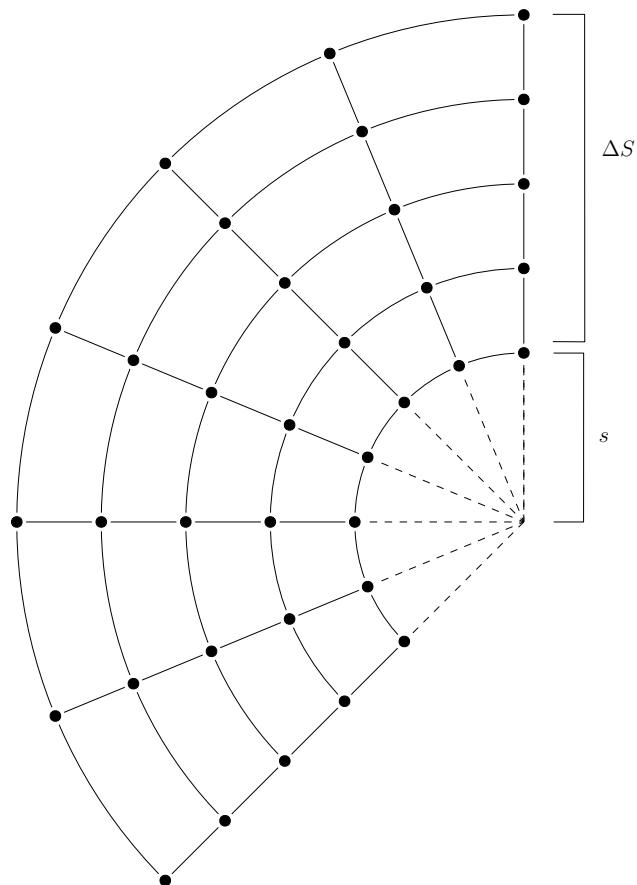


Abbildung 3.2: Kalibrierungsmuster entfaltet mit $n = 5, m = 6$

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.

Bilder von kegel mit muster drin?

3.2 Intrinsische Kamerakalibrierung

Bedingt durch die Wahl einer Weitwinkelkamera, enthält die Linse der Kamera eine starke tonnenförmige (nach außen gewölbte) Verzerrung. Ohne eine intrinsische Kamerakalibrierung würde. Es werden nur die intrinsischen Parameter benötigt.

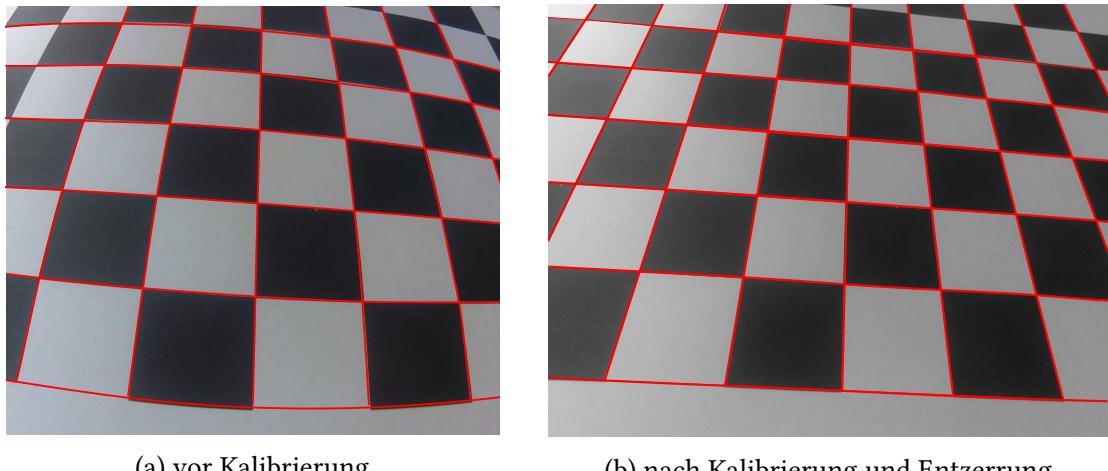


Abbildung 3.3: Kamerakalibrierung

warum gerade dieses muster?
warum ist das so gut?
wie berechnet man das muster?
welche eigenschaften hat es?

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 2.7) benutzt. Um ein Sample korrekt detektieren zu können, muss sich der Punkt farblich stark von seiner Umgebung abheben (siehe Definition: Blob 2.7.1). 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 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 auf dem gleichen Bild nach der Kameraentzerrung zu sehen.

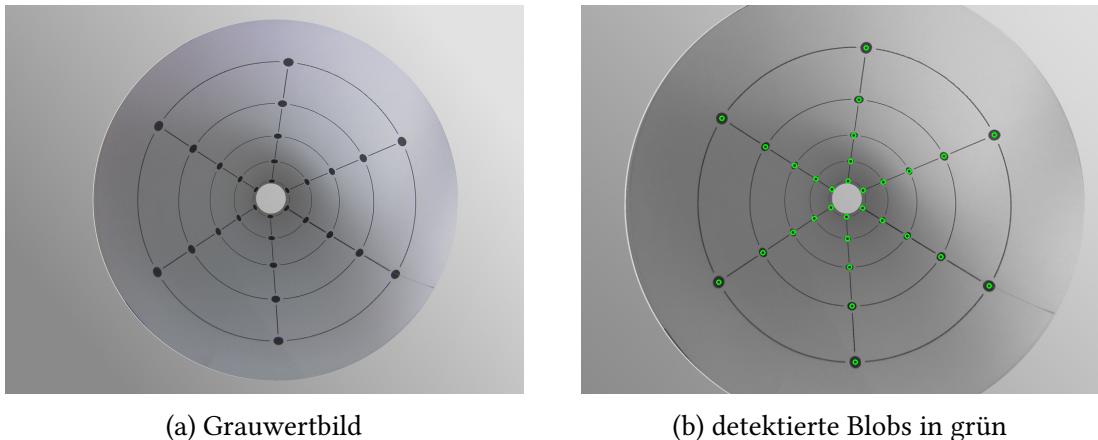


Abbildung 3.4: Detektion der Samples

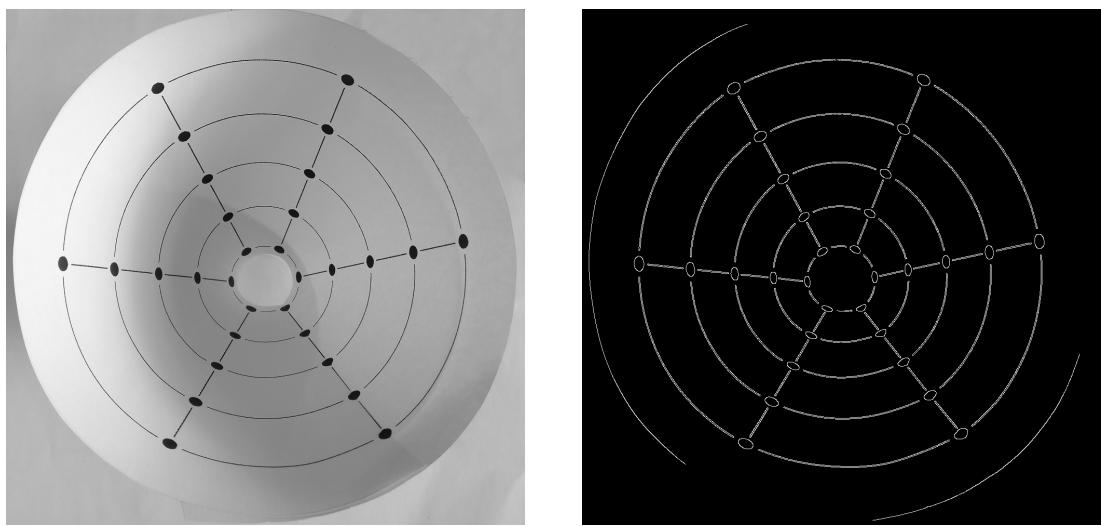
3.4 Ellipsen-Detektion

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 (2.5) detektiert (siehe Abbildung 3.5).

Anschließend versuchen wir möglichst genau das Zentrum der innersten Ellipsen zu bestimmen. Wir benutzen dafür Hough-Transformationen, um Linien im Canny-Bild 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 Punkt. Darüber hinaus werden, auf Grund der Liniendicke auf dem Kalibrierungsmuster, durch Canny viele Linien doppelt erkannt¹. Auch ein inhomogener Hintergrund, erschwert die Schnittpunktsbestimmung. Um also möglichst robust einen Kandidaten auszuwählen,

¹Dies ist kein Widerspruch zur Eindeutigkeit von Canny-Kanten (siehe 2.5), da dickere Linien zwei Kanten besitzen. 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



(a) Grauwertbild

(b) Canny-Kanten

Abbildung 3.5: Canny-Kantendetektion auf Grauwertbild

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).

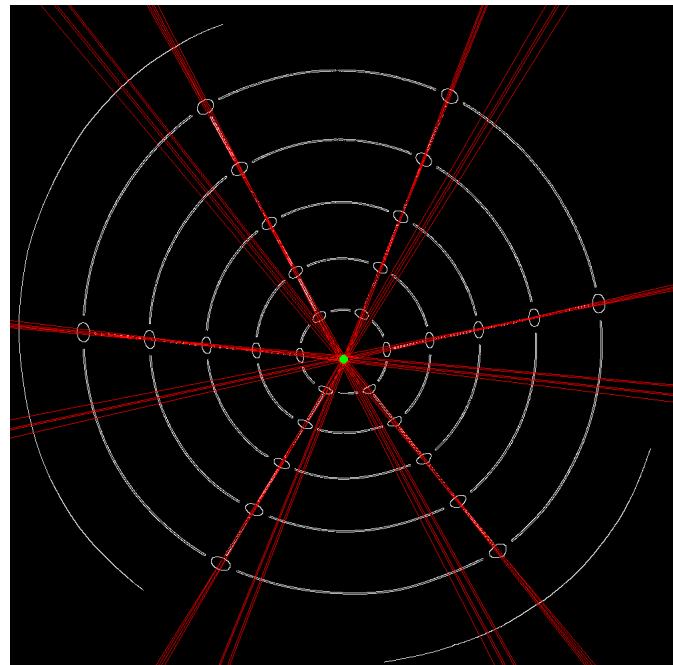
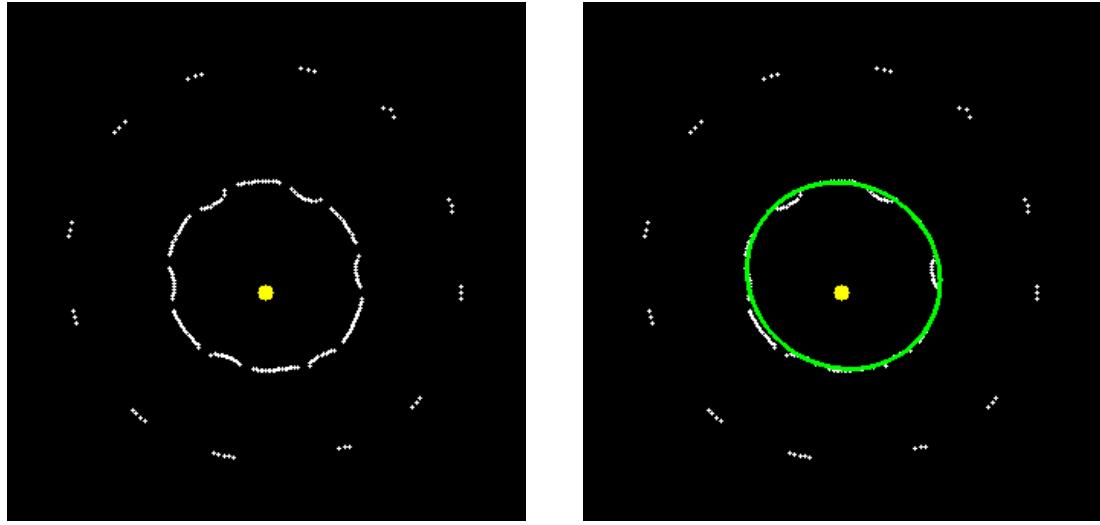


Abbildung 3.6: Hough-Transformation zur Linien-Detektion (in rot gekennzeichnet) und bestimmter Schnittpunkt (in grün)

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 eingezeichnet.



(a) bestimme Pixel-Positionen

(b) bestimme Ellipse (grün)

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

beschreiben,
warum die
Hauptachsentransformation
gebraucht
wird

Mit Hilfe der Positionen der weißen Pixel, wird anschließend durch RANSAC (siehe Kapitel 2.4) eine Ellipse geschätzt.

Es wird konkret für sechs zufällig ausgewählte Punkte das lineare Gleichungssystem:

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

gelöst, was auf der Gleichung 2.13 aus Kapitel 2.3 basiert. Nach dem Lösen wird geprüft ob, es sich tatsächlich um eine Ellipse handelt und mittels Hauptachsentransformation (siehe Kapitel 2.3) in die allgemeine Ellipsenform (x_0, y_0, a, b, ϕ) umgewandelt.

Um die, für RANSAC benötigte, Distanz zu berechnen, wird das Verfahren aus Kapitel 2.3.2 genutzt, was die exakte euklidische Distanz eines Punktes zu einer Ellipse bestimmt. Ein Verfahren wie das Verfahren der kleinsten Quadrate 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 2.4). Die letztendlich bestimmten Ellipsen sind beispielhaft in Abbildung 3.9 zu sehen.

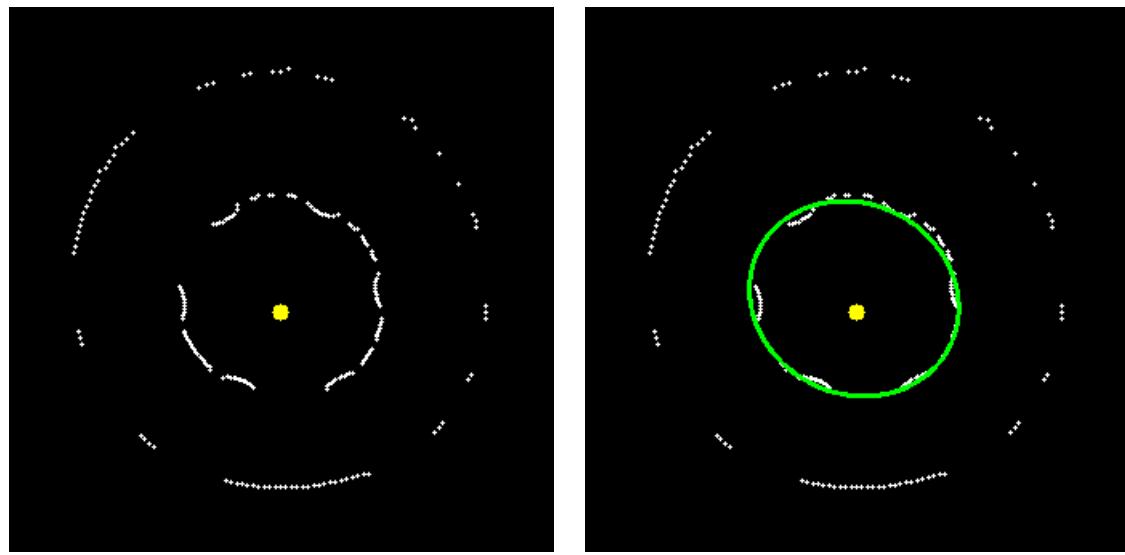


Abbildung 3.8: Ellipsendetektion bei Ausreißern

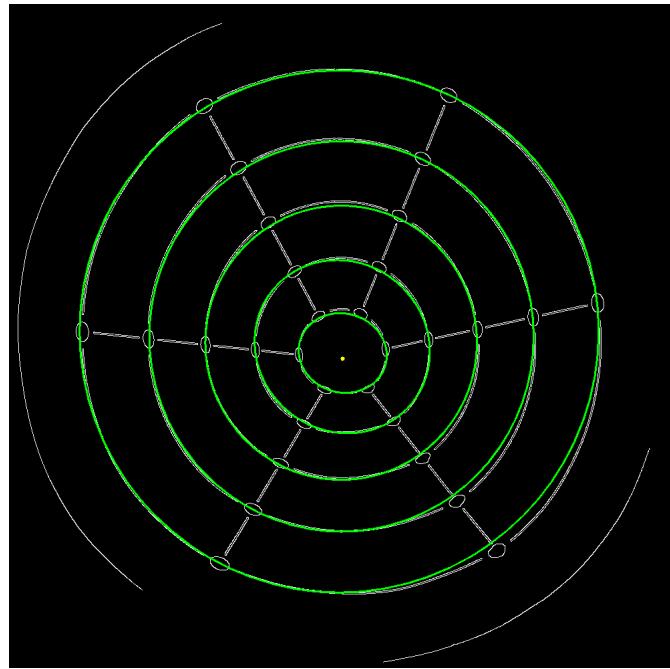


Abbildung 3.9: detektierte Ellipsen

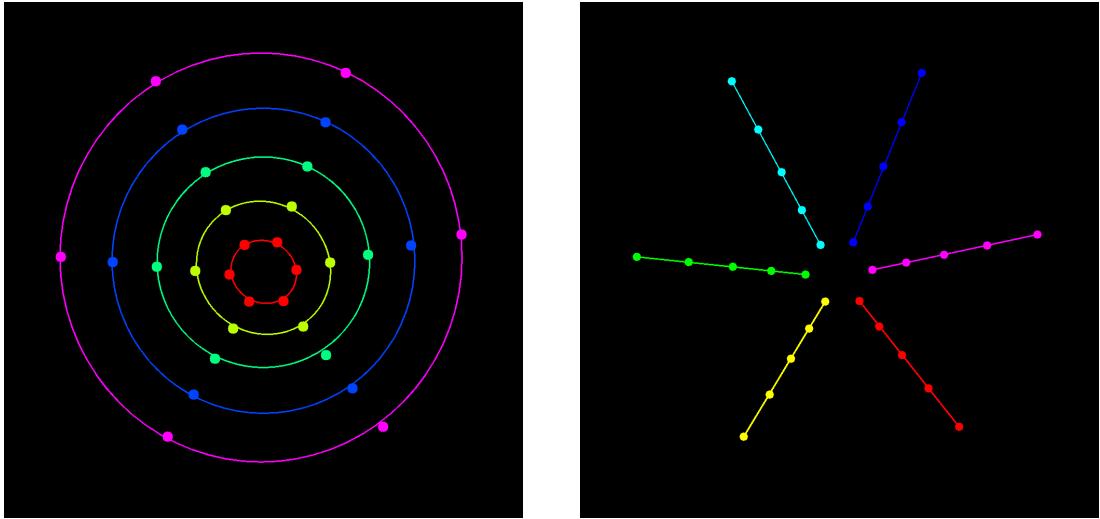
3.5 Zuordnung der Punkte

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 3.10(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 Ellipsen 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 Winkel zwischen den Samples auf einem Liniensegment und des bestimmten Mittelpunkts ist also nicht identisch. 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 3.10(b) zu sehen.



(a) Zuordnung von Punkten zu Ellipsen

(b) Zuordnung von Punkten zu Liniensegmenten

Abbildung 3.10: Zuordnung von Punkten zu Ellipsen (links) und Liniensegmenten (rechts)

3.6 Weltkoordinaten bestimmen

Aus dem vorherigen Kapitel wissen wir nun für jedes Sample den Ort auf dem Kegel. Wir können die 3D-Koordinaten also folgendermaßen angeben:

Ohne Beschränkung der Allgemeinheit, seien die Ellipsen $i = 0, \dots, n - 1$ aufsteigend nach ihrer „Größe“² 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 2.1 ergibt sich:

$$x_{ij} = r_i \cos\theta_j$$

$$y_{ij} = h_i$$

$$z_{ij} = r_i \sin\theta_j$$

²Etwas 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. $\pi \cdot a_0 \cdot b_0 \leq \pi \cdot a_1 \cdot b_1$

$\forall(i,j) \in [0,n-1] \times [0,m-1]$ 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 3.11 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_{int} 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 θ wird hierbei bei 2π 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 (ϕ_L, ϕ_P, ϕ_R) auf der Ellipse bezüglich des Ellipsenkoordinatensystems bestimmt werden.

hier schon
erläutern
oder erst
in späteren
kapiteln?

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 $\frac{\phi_L - \phi_P}{\phi_L - \phi_R}$

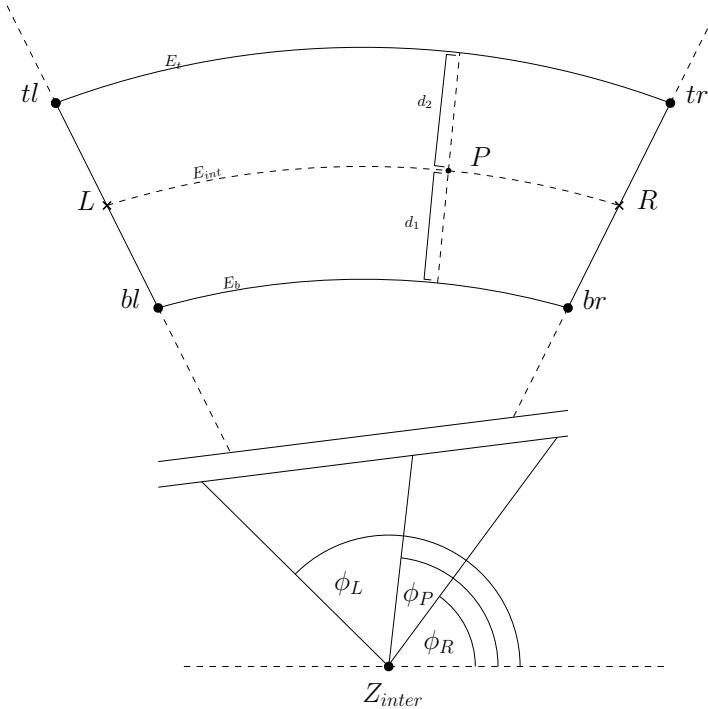


Abbildung 3.11: Interpolation

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 3.12 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.

gehören
Probleme
schon hier
her? oder
erst bei
analyse?

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

Die eigentliche Entfaltung geschieht über die Abbildung 2.5, die in Kapitel 2.1 konstruiert wurde. Gegebenenfalls nach einer Skalierung und Translation, um den Ursprung zu verschieben ergibt sich das entfaltete Bild, wie in Abbildung 3.13. Auch bei kleinen Skalierungen entstehen schon auffällige „Löcher“ im entfalten 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

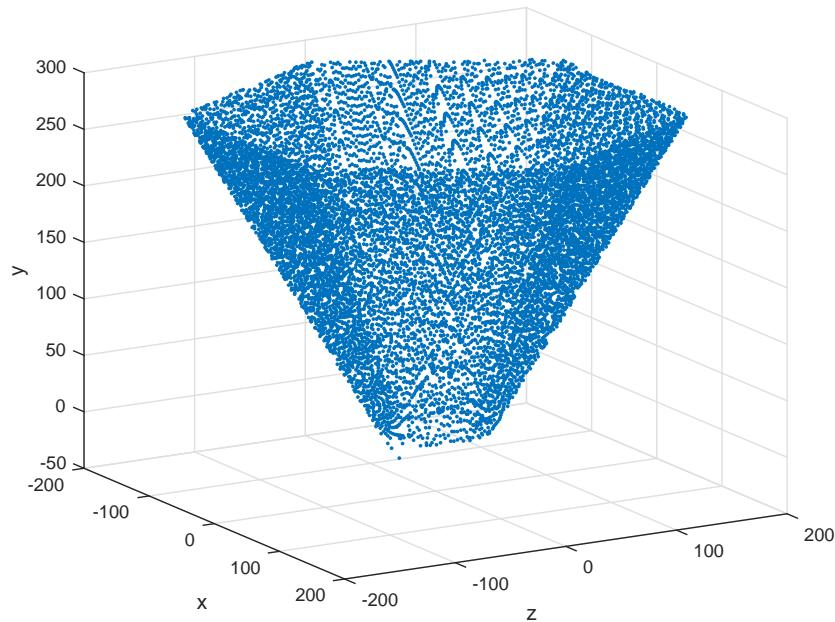
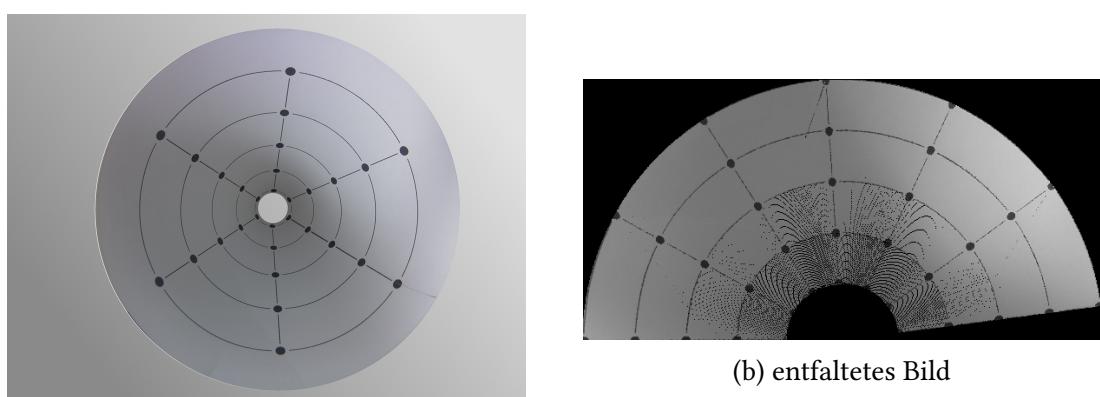


Abbildung 3.12: Interpolation

um ganzzahlige Werte. Es muss im entfalteten Bild gerundet werden. Auch wenn Rundungsfehler nicht entstanden, fehlt 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 eine Interpolation für Subpixel nicht möglich, da wir vorher nicht genau wissen, wo Löcher genau entstehen.



(a) Ursprungsbild

(b) entfaltetes Bild

Abbildung 3.13: Vorwärtsentfaltung

3.7.2 Rückwärtsentfaltung

Bei der Rückwärtsentfaltung gehen wir von dem entfalteten Bild aus und bestimmen zunächst mit der Umkehrabbildung ?? aus Kapitel 2.1, die 3D Kegelkoordinaten. Mit Hilfe einer Projektionsmatrix werden diese 3D-Koordinaten dann auf Bildkoordinaten abgebildet. Konkret lösen wir das überbestimmte lineare Gleichungssystem:

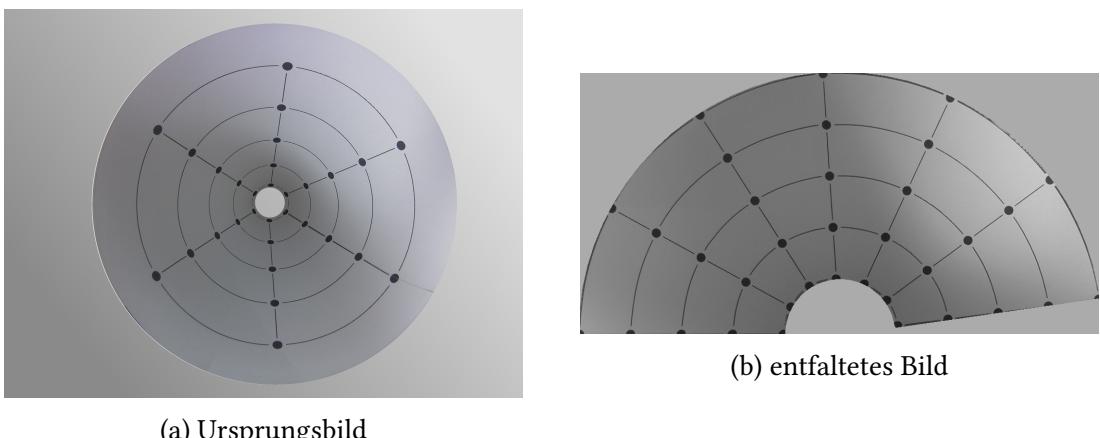


Abbildung 3.14: Rückwärtsentfaltung

4 Implementierung

GUI mit bildern

5 Analyse

In diesem Kapitel bla bla bla.

5.1 Vergleich VorwärtSENTfaltung und RückwärtSENTfaltung

Bla bla RückwärtSENTfaltung ist optisch viel besser und schneller, deshalb beziehen sich alle folgenden Analysen nur noch auf Rückwärts

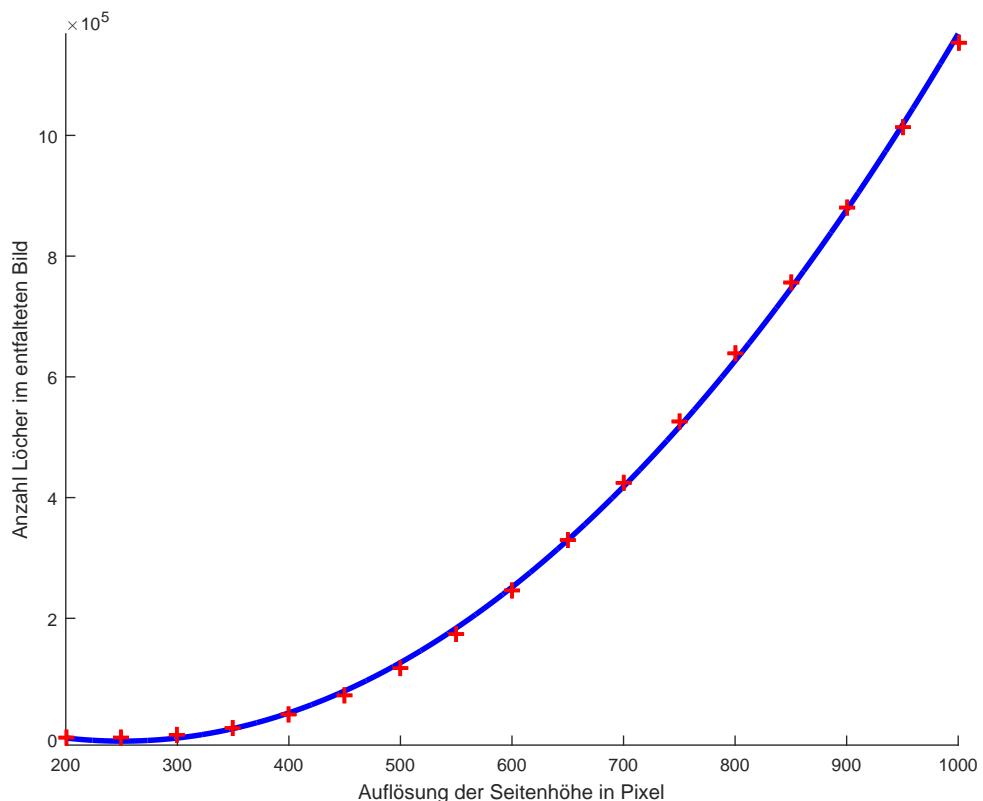


Abbildung 5.1: Einfluss der Ausgabeauflösung auf die Anzahl der Löcher

5.2 Einfluss der intrinsischen Kalibrierung

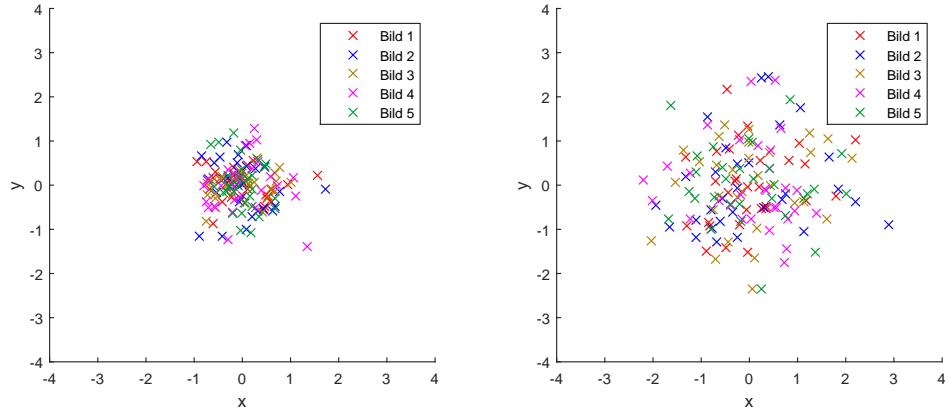


Abbildung 5.2: Einfluss der intrinsischen Kalibrierung

projektionsfehler ohne kalib ist wesentlich größer, da die implizite intrinsische kalib durch die svd keine verzerrung entfernt

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 äußere Faktoren wie Lichtverhältnisse und inhomogene Hintergründe kontrolliert werden können.

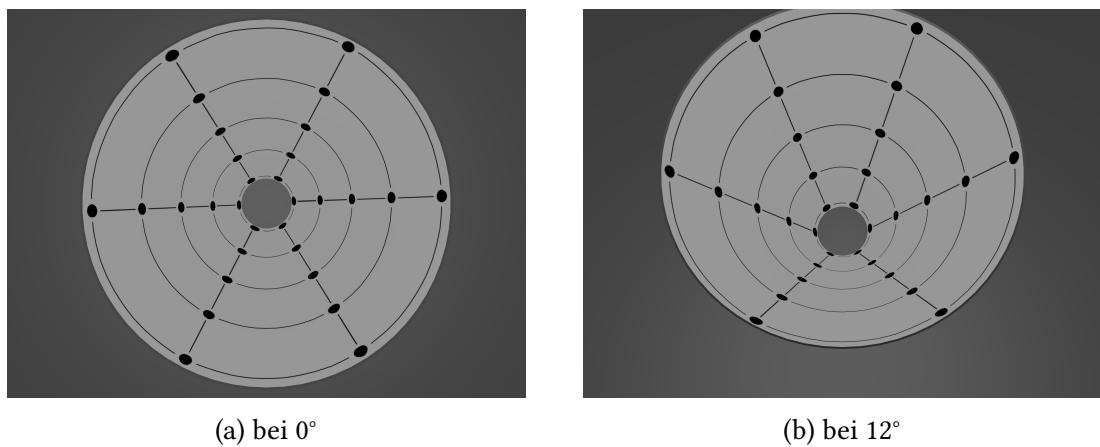


Abbildung 5.3: gerendeter Kegel mit Kalibrierungsmuster in Blender

Es werden anschließend Bilder erzeugt, in denen in 1° Schritten die Kamera von 0° bis 12° um die X-Achse rotiert wird. Für jedes dieser Bilder wird anschließend eine Rückwärts-

entfaltung durchgeführt und der Reprojection-Fehler bestimmt. Abbildung 5.4 zeigt, dass der Reprojection-Fehler rotationsinvariant ist.

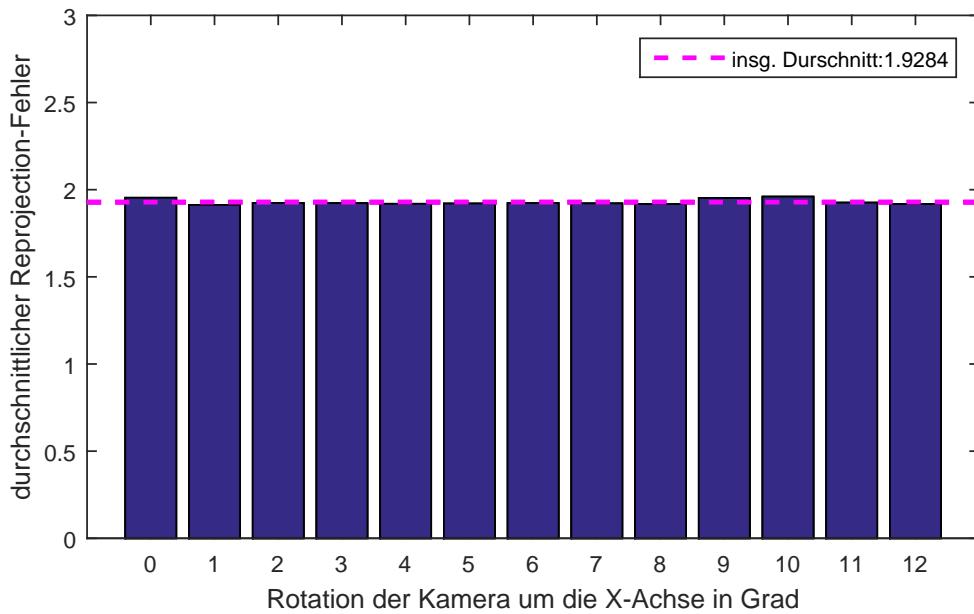


Abbildung 5.4: Einfluss der Rotation der Kamera

5.4 Laufzeit der Entfaltung

Raspberry Pi 2 Model B:

- 900MHz ARM Cortex-A7 CPU
- 1GB RAM
- GCC-4.??? Release
- OpenCV 2.4.13

projektionsfehler bei dem einen anzahl löcher bei dem anderen
robustheit reverse warping? bestimmung der keypoints?

5.5 Evaluierung des RANSAC zur Ellipsendetektion

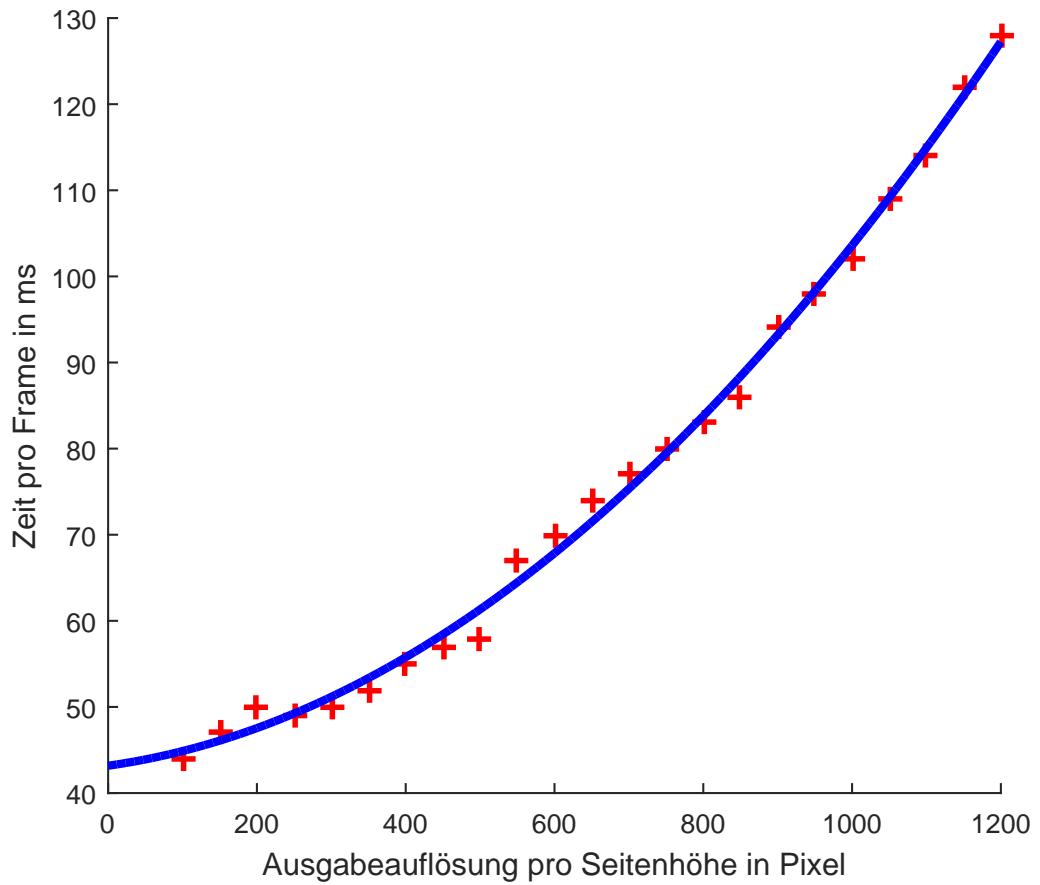


Abbildung 5.5: Einfluss der Ausgabeauflösung auf die Laufzeit

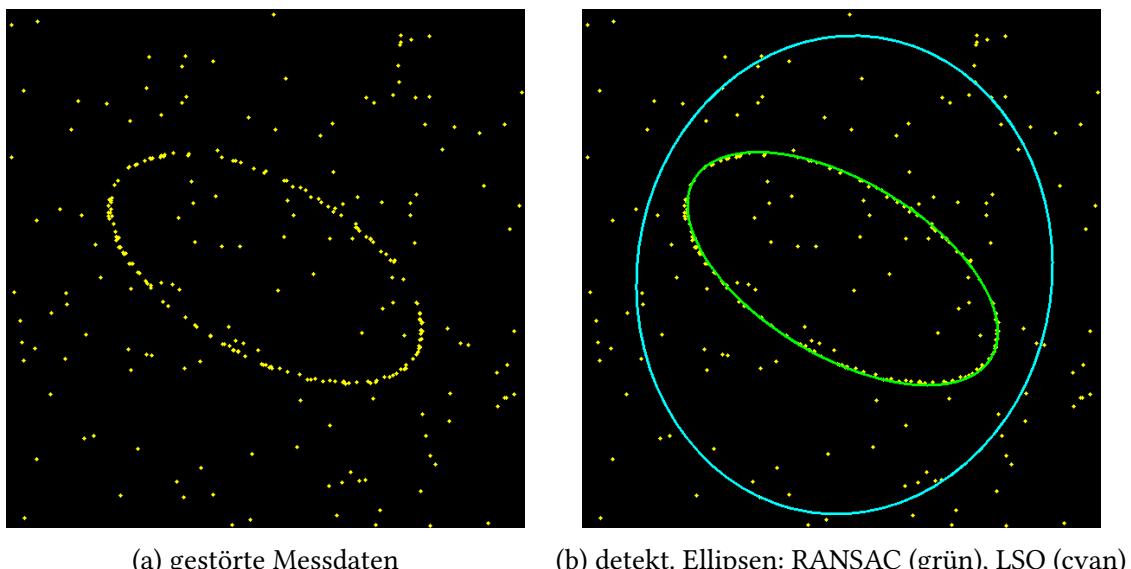


Abbildung 5.6: Vergleich RANSAC und LSQ bei gleichverteilten Ausreißern $\epsilon = 0.5, p = 0.99$

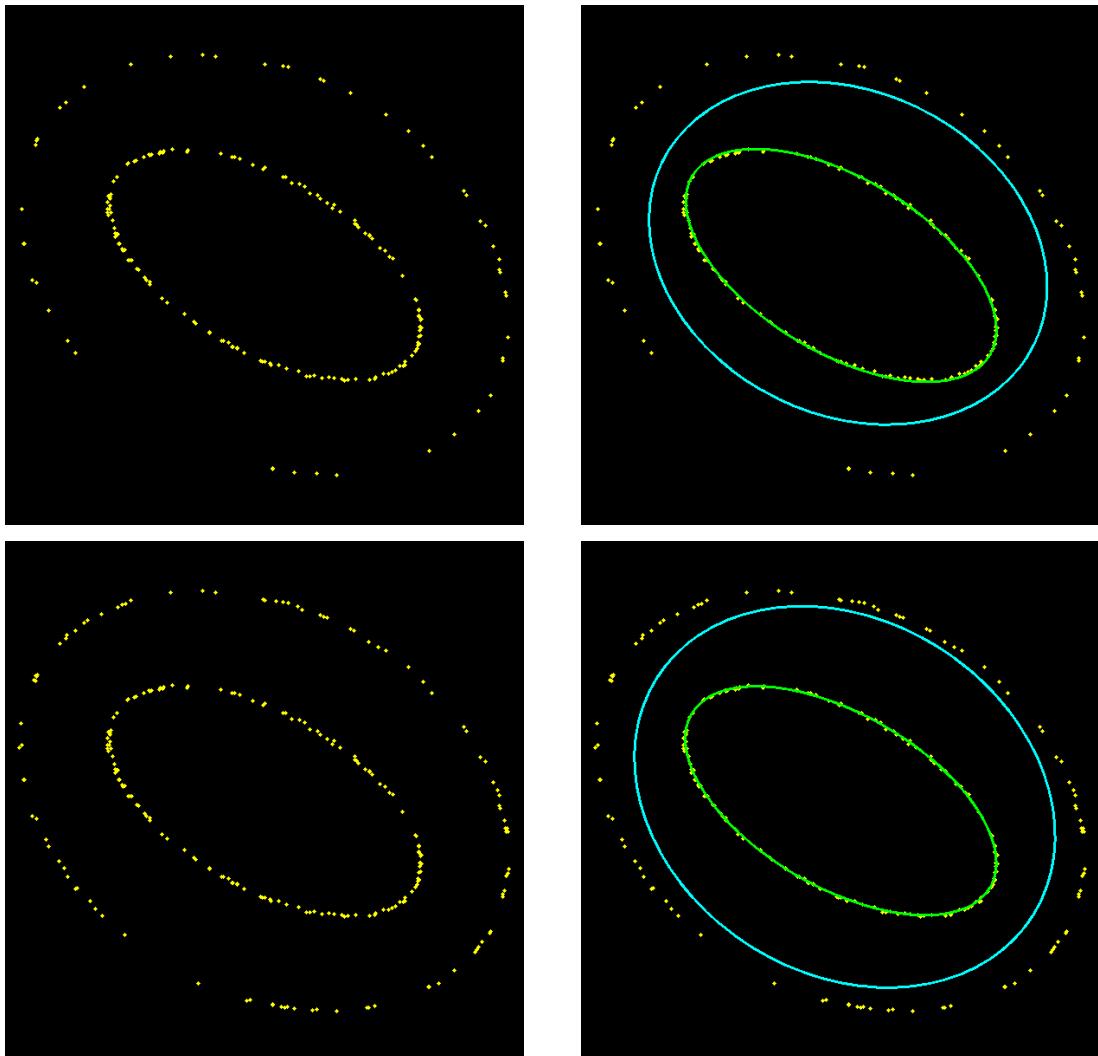


Abbildung 5.7: Vergleich RANSAC und LSQ bei Schattenellipsen mit $p = 0.99$ und $\epsilon = 0.25$ (oben), $\epsilon = 0.4$ unten, links gestörte Messdaten, rechts detektierte Ellipsen RANSAC (grün), LSQ (cyan)

6 Fazit und Ausblick

Parallelisierung? geht immer, weil Imageprocessing.

 hough evtl besser probabilistic oder richtungsinformationen mit nehmen

 splatting

 iteratives verfahren zum minimieren des reprojection fehlers ausprobieren robustes
projektionsgedöns

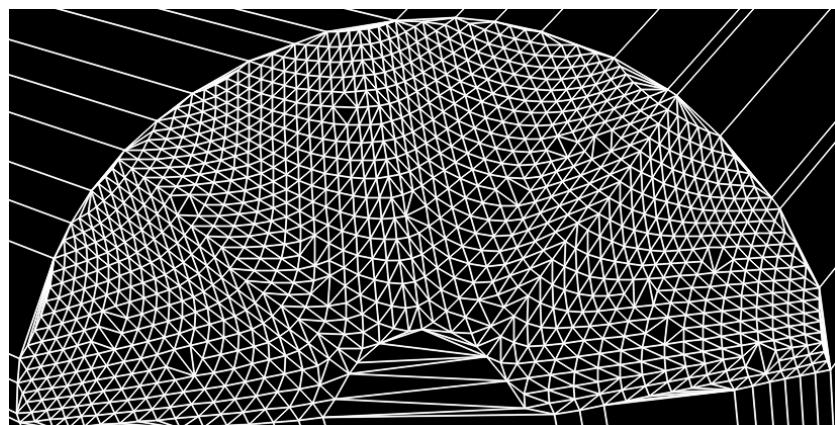
Um die Löcher bei der Vörwärtsentfalung zu schließen, könnte man eine Delaunay-Triangulation durchführen. Da das Verfahren jedoch ohne Triangulation schon relativ rechenintensiv ist, und die Rückwärtsentfaltung sehr gute Ergebnisse liefert wurde dieses Möglichkeit nicht weiter untersucht.

$$G(x, y) = \frac{((x - x_0) \cos \theta + (y - y_0) \sin \theta)^2}{a^2} + \frac{((x - x_0) \sin \theta - (y - y_0) \cos \theta)^2}{b^2} - 1 = 0 \quad (6.1)$$

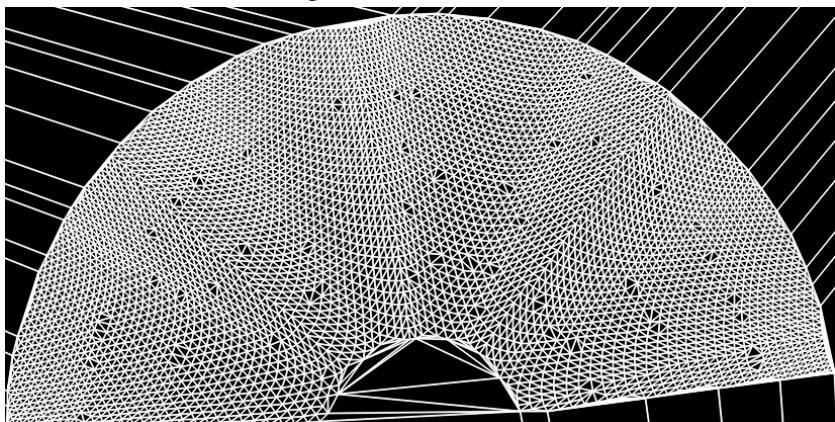
$$E = E_M + E_A + E_S \quad (6.2)$$

$$\begin{aligned} E_M &= -\alpha \frac{1}{n} \sum_{i=0}^{n-1} I_M(p_i) \\ E_A &= \beta \frac{1}{n} \sum_{i=0}^{n-1} \left(I_O(pi) - \text{atan2}\left(\frac{\partial G}{\partial y}(p_i), \frac{\partial G}{\partial x}(p_i)\right)\right)^2 \\ E_S &= \gamma \frac{1}{n} \sum_{i=0}^{n-2} |p_i - p_{i+1}| \end{aligned} \quad (6.3)$$

Man macht die Initialellipse möglichst groß 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. α, β und γ 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.



(a) Triangulation mit 10% der Punkte



(b) Triangulation mit 40% der Punkte

Abbildung 6.1: Delaunay-Triangulation

Literatur
checken,
ins besonde-
re seitenan-
gaben

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	8
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)	9
2.8	Ellipse mit Zentrum (x_0, y_0) , Hauptachse a , Nebenachse b , sowie Drehwinkel θ	10
2.9	Ellipsenausschnitt im ersten Quadranten mit Abfragepunkt Q und eingezeichneter kürzester Distanz zur Ellipse	14
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)	25
3.7	Ellipsendetektion: bestimme Pixel-Positionen (weiß), Aussendepunkt (gelb)	26
3.8	Ellipsendetektion bei Ausreißern	27
3.9	detektierte Ellipsen	28
3.10	Zuordnung von Punkten zu Ellipsen (links) und Liniensegmenten (rechts)	29
3.11	Interpolation	31
3.12	Interpolation	32
3.13	Vorwärtsentfaltung	32
3.14	Rückwärtsentfaltung	33
5.1	Einfluss der Ausgabeauflösung auf die Anzahl der Löcher	37
5.2	Einfluss der intrinsischen Kalibrierung	38
5.3	gerendeter Kegel mit Kalibrierungsmuster in Blender	38
5.4	Einfluss der Rotation der Kamera	39
5.5	Einfluss der Ausgabeauflösung auf die Laufzeit	40
5.6	Vergleich RANSAC und LSQ bei gleichverteilten Ausreißern $\epsilon = 0.5, p = 0.99$	41

ABBILDUNGSVERZEICHNIS

5.7 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)	42
6.1 Delaunay-Triangulation	44

Tabellenverzeichnis

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. 16).
- [Ebe13] David Eberly. „Distance from a Point to an Ellipse, an Ellipsoid, or a Hyperellipsoid“. In: (2013), S. 1–13 (siehe S. 13).
- [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. 16).
- [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. 8).
- [Law72] J D Lawrence. *A Catalog of Special Plane Curves*. Dover Publications, 1972, S. 62–63 (siehe S. 13).
- [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. 17).
- [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. 8, 9).

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, 7. 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, 7. September 2016