

Fakultät für Elektrotechnik und Informationstechnik



Lehrstuhl für
Kommunikations-
technik

Bachelorarbeit B 14-2015

**Tasty Kanalmodell
für die drahtlose Kommunikation
zwischen Gebäuden und
Außeninstallationen**

von

Käpt'n Kevin Blaubär

Abgabedatum: 23. Januar 2018

Prof. Dr.-Ing Rüdiger Kays • Lehrstuhl für Kommunikationstechnik • TU Dortmund

Inhaltsverzeichnis

1 Einleitung	1
1.1 Section	1
2 Data transmission using Video devices (DaVid)	3
2.1 Einführung des DaVids	3
2.2 Systemmodell	4
2.2.1 Modulationsverfahren	5
2.2.2 DatenBlock	8
2.3 Anwendungsgebiete	8
3 Erste Methode	10
3.1 Allgemeine Struktur	10
3.2 Bildregistrierung	13
3.2.1 Speeded Up Robust Features (SURF)	13
3.2.2 RANdom SAmples Consensus (RANSAC)	18
3.2.3 Bilder Umwandlung	22
3.3 Differenzbild Optimierung	33
3.4 Bildverarbeitung	36
3.5 QR Musters Detektion	37
4 Zweite Methode	42
4.1 Allgemeine Struktur	42
4.2 Binarisierung	45
4.3 Morphologie	50
4.4 Canny detection	54
4.5 Cross Dilatation	58
4.6 Hough Transformation	59
4.7 Radon Transformation	61
5 Implementierung	64
5.1 Experiment Umgebung	64
5.2 Implementierung der ersten Verfahren	64

5.3 Implementierung der zweiten Verfahren	67
5.4 Implementierung auf einer Smartphone-GPU	69
6 Evaluierung	71
6.1 Evaluierung der 1.Verfahren	71
6.2 Evaluierung der 2.Verfahren	75
7 Zusammenfassung	77
7.1 Section	77
A Erster Anhang	78
A.1 Section	78
Abbildungsverzeichnis	80
Tabellenverzeichnis	83
Quellcodeverzeichnis	84
Literatur	85

1 Einleitung

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1.1 Section

Jetzt nur noch schreiben! :)

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacinia tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacinia congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacinia commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacinia. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacinia vel est. Curabitur consectetur.

2 DaVid

In diesem Kapitel werden das DaVid System besprochen. Zuerst stellen wir das DaVid Systems vor. Das Systemmodell und Arbeitsprinzip des Systems werden im anschließenden Abschnitt erläutert. Schließlich flogen die mögliche Anwendungsgebiete des Systems. [1]–[4]

2.1 Einführung des DaVids

DaVid ist ein neuartiges Verfahren zur optischen Freiraum-Datenübertragung zwischen einem Display als Sender und digitalen Kamera als Empfänger. Ein grundlegendes Übertragungskonzept von DaVid wird in Abbildung 2.1 gezeigt. Ein flaches Display, wie ein OLED- oder LCD-Bildschirm, zeigt ein Live-Video. Gleichzeitig werden die Daten hinter dem Bild auf die Pixel moduliert, während der Pixel auf dem Bild für menschliche Betrachter nahezu unsichtbar ist. Der Benutzer richtet eine hochauflösende Kamera beziehungsweise Smartphonekamera auf den Bildschirm, um die Szene aufzunehmen. Durch den Smartphone Prozessor können die Signale in Daten decodiert werden.

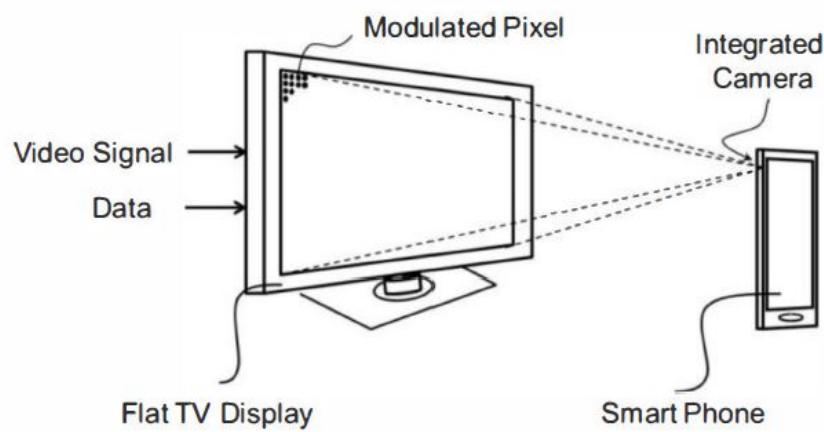


Abbildung 2.1: Eine beispielhafte Implementierung des DaVid-Systems.

2.2 Systemmodell

Das Display enthält eine große Anzahl von Pixeln, die jeweils aus einer spezifischen Anordnung von Subpixeln zur Darstellung des RGB-Farbraums bestehen. Jeder einzelne Frame des Videos wird nämlich durch eine Matrix von Subpixelwerten dargestellt. Das DaVid-System verwendet eine differentielle Modulationsmethode d.h. ein Teil der Videoinformationen wird wiederholt, sodass Daten als ein symmetrischer Manchester-Code moduliert und zu den Videosignalen hinzugefügt werden. Durch eine zeitliche Synchronisation der Empfängerseite kann die inter symbol interference (ISI) vermieden werden. Nach einer örtlichen Synchronisation mit dem modulierten Display, enthält der Empfänger ein Differenzbild. Da der Randbereich des Differenzbilds ungültig ist, wird das Modulationsgebiet durch das Verfahren dieser Arbeit beschränkt. Danach werden die überlagerten Datensequenz durch eine Reihe von Behandlungen, vom Videoinhalt getrennt. Abbildung 2.2 zeigt die schematische Darstellung des DaVid-Systems.

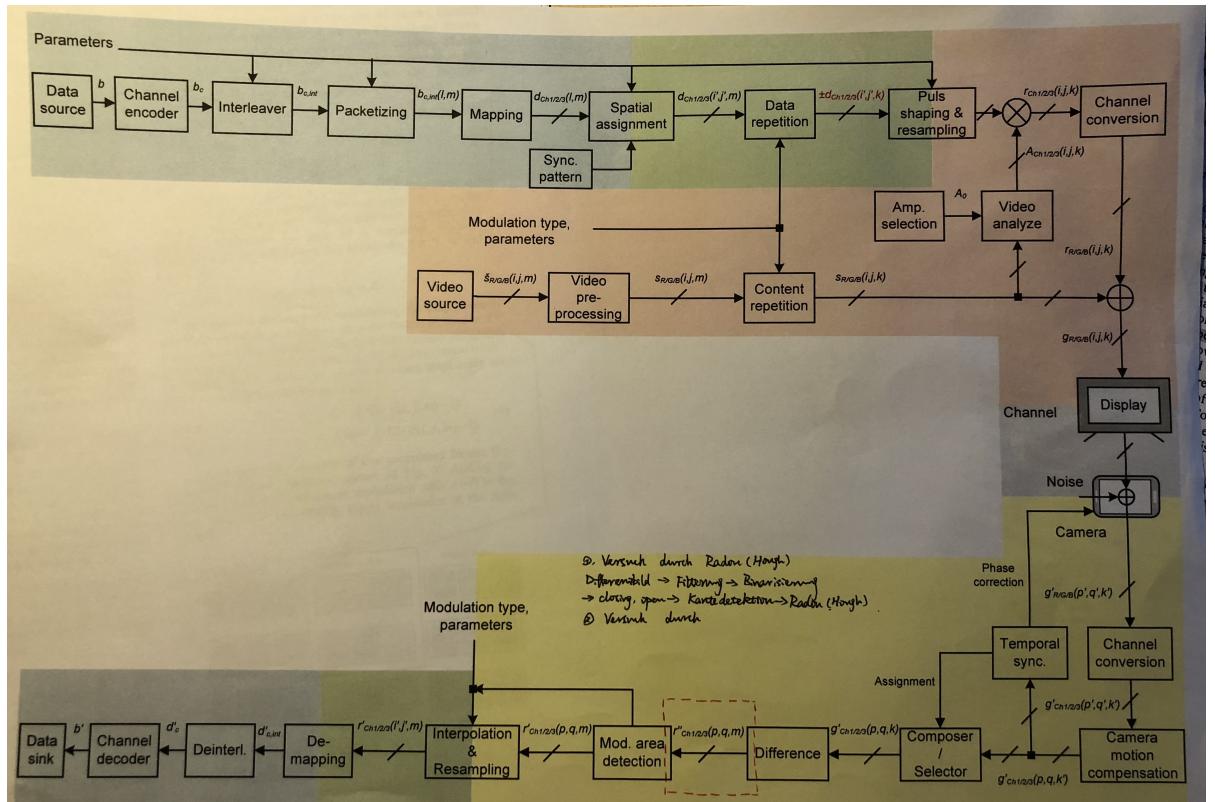


Abbildung 2.2: Schematische Darstellung vom DaVid-System.

2.2.1 Modulationsverfahren

Ein Modulationsverfahren, welches die Videoqualität nicht beträchtlich reduziert, ist sehr kritisch für ein auf Videogeräte basierendes Datenübertragungssystem. Die möglichen Modulationsverfahren im DaVid-System sind:

- Zeitlich-differentielle Modulation der Luminanz
- Zeitlich-differentielle Modulation der Chrominanz
- Örtlich-differentielle Modulation der Luminanz
- Örtlich-differentielle Modulation der Chrominanz

Zeitliche differentielle Modulationsverfahren lassen kontinuierliches eins Paar Frames den gleichen Luminanz- bzw. Chrominanz-Videoinhalt anzeigen, d.h. durch Subtrahieren dieser Frames die mit Datensquenze Differenzbild erhalten lassen können. Dagegen haben die benachbarte Pixel bei Örtliche-differentielle Modulation die gleichen Videoamplituden. In dieser Arbeit wird nur zeitlich-differentielle Modulation der Chrominanz verwendet. Abbildung 2.3 zeigt ein Blockschaltbild einer typischen Senderimplementierung durch zeitlich-differentielle Modulation.

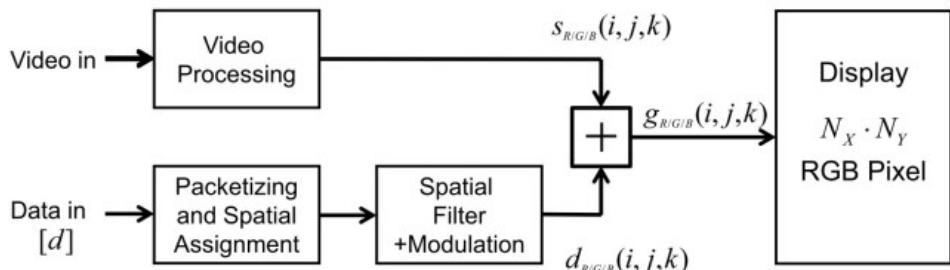


Abbildung 2.3: Blockschaltbild der Signalverarbeitung in zeitlicher differentieller Modulation.

Wir nehmen eine Display an, welche in horizontaler Richtung N_x Pixel und in vertikaler Richtung N_y Pixel enthält. Das Videoeingangssignal wird verarbeitet, um eine Anzeigeeingabe $s(i, j, k)$ auszusenden. Mit zeitlich-differenziell Modulation gleicht der Inhalt dieses Frames des darauf folgend. Die Indizes i und j definieren die horizontale und vertikale Pixelposition auf dem Display, während k die Nummer des reproduzierten Frames ist. Indiz m symbolisiert den Zähler der Frames in einer Videosequenz. Der Amplitudenbereich des Videosignals sollte begrenzt sein, um die Addition kleiner Datenamplituden ohne Übersteuern zu ermöglichen.

$$s_{R/G/B}(i, j, k + 1) = s(i, j, k) \quad (2.1)$$

for $0 \leq i < N_x, 0 \leq j < N_y, k = 2m, m \in \mathbb{Z}$

Vor der Datenübertragung muss der Datenstrom in Schichten der Länge L aufgeteilt werden. Indiz L bedeutet die Menge der Daten, die in einem Framepaar übertragen werden können. Ein direkter Ansatz ist eine direkte Zuordnung von Datenbits zu Pixeltripeln Zeile für Zeile.

$$\begin{aligned} d(l) \rightarrow d(i, j, m) \quad d(l) \in \{-1, 1\} \\ 0 \leq l < L, L = N_x \cdot N_y \quad (2.2) \\ i = l \bmod N_x \quad j = \lfloor l/N_x \rfloor \end{aligned}$$

Die Modulationsamplitude A ist ein wichtiger Parameter für die Datenübertragung. Im Prinzip kann die Amplitude in verschiedenen Kanälen unabhängig gewählt werden um die Systemleistung zu optimieren. In dieser Arbeit werden alle Amplituden auf den gleichen Wert gesetzt.

$$A_R = A_G = A_B = A \quad (2.3)$$

Das differentielle Modulationsverfahren ordnet jede Sequenz von $\{-A, A\}$ zu $d = 0$ bzw. $\{A, -A\}$ zu $d = 1$ zu. Modulierte Datensymbole und verarbeitete Videoamplituden werden addiert, um die Anzeigeeingabe $g(i, j, k)$ zu liefern:

$$\begin{aligned} s_{R/G/B}(i, j, k) &= s_{R/G/B}(i, j, m) + A_{R/G/B} \cdot (2 \cdot d(i, j, m) - 1) \\ s_{R/G/B}(i, j, k + 1) &= s_{R/G/B}(i, j, m) - A_{R/G/B} \cdot (2 \cdot d(i, j, m) - 1) \quad (2.4) \end{aligned}$$

Ein Beispiel einer modulierten Bildfolge ist in Abbildung 2.4 gezeigt. Das Hinzufügen der modulierten Daten (hier mit $A = 4$) zu dem Videoeingang ergibt die Anzeigearamplituden in der rechten Spalte.



Abbildung 2.4: Ein Beispiel einer modulierten Bildfolge.

Im Vergleich zu Luminanzteil Y ist die Anzeigequalität in der U und V Komponente signifikant besser, wenn Informationen in Chrominanz wiederholt und moduliert werden. Auf diese Weise wird die Gesamtleuchtdichte eines Pixel-Triple durch die Datenmodulation nicht beeinflusst. Die Umwandlungsmatrix, angegeben von test (ITU-R BT.709) für **high definition television** (HDTV) Display des Standards (R, G, B) zum Standard (Y, U, V) läuft:

$$T = \begin{pmatrix} 0,213 & 0,715 & 0,072 \\ -0,115 & -0,385 & 0,5 \\ 0,5 & -0,454 & -0,0458 \end{pmatrix} \quad (2.5)$$

Das Videosignal $s(i, j, k)$ muss vor dem Anwenden der Modulation in Y-, U- und V-Komponenten umgewandelt werden. Die nachfolgende inverse Konvertierung erklärt das Display-Eingangssignal in Abbildung 2.3:

$$\begin{aligned} \begin{pmatrix} g_R(i, j, k) \\ g_G(i, j, k) \\ g_B(i, j, k) \end{pmatrix} &= T^{-1} \cdot \left(T \cdot \begin{pmatrix} S_R(i, j, m) \\ S_G(i, j, m) \\ S_B(i, j, m) \end{pmatrix} + \begin{pmatrix} 0 \\ A_U \cdot d(i, j, m) \\ A_V \cdot d(i, j, m) \end{pmatrix} \right) \\ \begin{pmatrix} g_R(i, j, k+1) \\ g_G(i, j, k+1) \\ g_B(i, j, k+1) \end{pmatrix} &= T^{-1} \cdot \left(T \cdot \begin{pmatrix} S_R(i, j, m) \\ S_G(i, j, m) \\ S_B(i, j, m) \end{pmatrix} - \begin{pmatrix} 0 \\ A_U \cdot d(i, j, m) \\ A_V \cdot d(i, j, m) \end{pmatrix} \right) \end{aligned} \quad (2.6)$$

Diese Art der Modulation kann als eine Modulation des roten und des blauen Subpixels betrachtet werden, während der grüne Subpixel verwendet wird, um die Änderung der Luminanz des Pixel-Tripels zu kompensieren. Durch Definition korreliert A_U mit A_B und A_V mit A_R .

$$\begin{pmatrix} A_R \\ A_G \\ A_B \end{pmatrix} = T^{-1} \cdot \begin{pmatrix} A_Y \\ A_U \\ A_V \end{pmatrix} \quad (2.7)$$

2.2.2 DatenBlock

Ein einfaches und unkompliziertes Verfahren um die Anforderungen an die Kameraauflösung zu lockern, ist die Zuordnung jedes Datenbits zu einem Block von $B_X \times B_Y$ Pixeln.

$$\begin{aligned} d(l) &\rightarrow d(x, y, k) \quad 0 \leq l < L \\ L &= \lfloor N_X/B_X \rfloor \cdot \lfloor N_Y/B_Y \rfloor \\ x &= (l \cdot B_X) \bmod N_X + r_X, \quad r_X = 0 \dots (B_X - 1) \\ y &= \lfloor l/\lfloor N_X/B_X \rfloor \rfloor \cdot B_Y + r_Y, \quad r_Y = 0 \dots (B_Y - 1) \end{aligned} \quad (2.8)$$

Wenn die Anzahl der Pixel Pro Zeile bzw. Spalte kein Vielfaches von B_X bzw. B_Y ist, muss die Anzahl der Pixel Pro Zeile bzw. Spalten, die für die Modulation in Gleichung (2.8) verwendet werden, ersetzt werden durch:

$$\begin{aligned} N_X &= \lfloor N_X/B_X \rfloor \cdot B_X \\ N_Y &= \lfloor N_Y/B_Y \rfloor \cdot B_Y \end{aligned} \quad (2.9)$$

In dieser Arbeit ist der Datenblock quadratisch.

$$B_X = B_Y = B. \quad (2.10)$$

2.3 Anwendungsbereiche

Die Datenübertragungsrate des DaVid-Systems beträgt voraussichtlich bis zu 100 Mbit/s. Es gehört zu einer Sichtlinienübertragung für kurze Strecken. Geeignete Abdeckungsbereiche hängen von der Größe des Displays und der Kameraoptik ab. Im Vergleich zu den letzten WLAN-Versionen Institute of Electrical and Electronics Engineers (IEEE) 802.11, erscheint die Leistung zunächst unattraktiv. Dieses spezielle Verfahren der Datenübertragung macht es jedoch zu einem Kandidaten für viele praktische Anwendungen. Außer den Hauptvorteilen von VLC, der Vorteil liegt auch in die Option zur Wiederverwendung der bestehenden Hardware,

die zum Zeigen des Videos eingebaut wurde. Ein empfohlene praktische Anwendungsbereich des DaVid-Systems ist öffentlicher Ort, wie U-Bahn-Station, großes Stadion und so weiter. Angenommen in eine Situation, wo Passanten auf eine U-Bahn warten und ihre Software mit Hilfe von Digitalen Werbetafeln aktualisieren können, indem sie ihre Kamera auf die Werbung richten.

Durch Berücksichtigung der Eigenschaften des DaVids, z.B. Synchronisation von Anwendungen und Datenübertragung, können viele attraktive Anwendungszenarien in Betracht gezogen werden. Drei Hauptzenarien sind:

- Indoor-individuelle Kommunikation: Kurzstreckenverbindungen basieren auf relativ kleinen (Tablet-Größe) Bildschirmen, Anwendungen wie z.B. die Übertragung von Hintergrundinformationen an Besucher im Museum oder Kiosk.
- Indoor-Multicast-Kommunikation: Streckenabstand ist länger als im ersten Fall auf relativ großem (40-100") Bildschirm, Anwendungen wie z.B. Herunterladen von Anwendungsdateien oder Mediendateien im Kiosk oder Restaurant.
- Freie Kommunikation: Riesige Bildschirme wie in Einkaufszentren oder Sport-Arenen. Anwendungen können denen des zweiten Szenarios ähneln.

Sobald die Dienste auf öffentlichen Bildschirmen implementiert werden, können Benutzer mit Hilfe eines modernen Smartphones mit einer geeigneten Kamera nach der Installation einer neuen App Innovation wahrnehmen.

3 Erste Methode

In diesem Kapitel wird auf die Realisierung des ersten Verfahrens eingegangen. Zuerst wird die Bildregistrierung im Detail besprochen. Anschließend läuft die Differenzbild Optimierung und die benötigte Bildverarbeitung. Schließlich wird die QR Mustererkennung erläutert. Zur Implementierung dieses Verfahrens wird Matlab unter der Lizenz der TU Dortmund verwendet.

3.1 Allgemeine Struktur

Dieses Verfahren verwendet die Charakteristiken der Datenmodulation des DaVid Systems, d.h. an jeder Ecke der Datenebene ein QR Muster hinzufügen und dann mit den Daten zusammen hinter dem Bild modulieren. Am Empfänger nach einige Operationen wird QR Muster mit die in diese Kapitel vorgestellte Verfahren detektiert und schließlich den Modulationsbereich bestimmt. Diese Methode löst effektiv den unansehnlich Effekt, das QR-Modul direkt zum Bild hinzugefügt wird, welche das Problem günstiger und effektiver durch die QR-Mustereigenschaft löst. Abbildung 3.1 zeigt das Strukturdiagramm dieses Verfahrens.

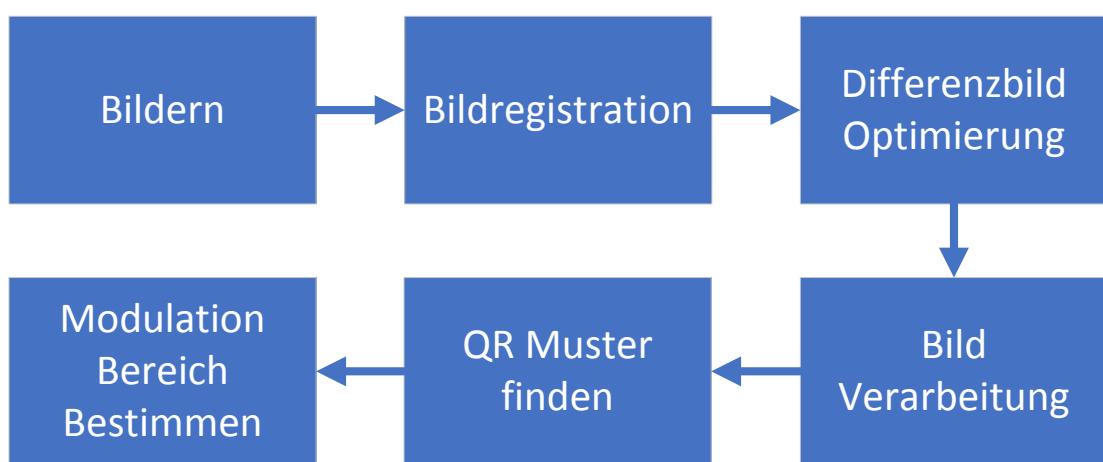


Abbildung 3.1: Strukturdiagramm.

Das Objekt, das mit dieser Methode bearbeitet wird, ist ein Reihe aus einer spezifischen Smartphone App “VLCReceiver” sammelt Bild. Diese App wurde speziell für das DaVid-System entwickelt und erstellt eine Reihe von Bildern bei jeder Aufnahme. Im Allgemeinen wird die Kamera bei der Aufnahme in der Hand gehalten. Aufgrund von Handschütteln während der Aufnahme entsteht eine leichte Verschiebung zwischen den Bildern. Um dieses Problem zu beheben, wird eine Bildregistrierung Operation verwendet, die zwei Bilder einer Aufnahme in dasselbe Koordinatensystem konvertiert. Zuerst werden durch SURF Merkmalserkennung Merkmale der Bilder verglichen. Anschließend erhalten wir durch Merkmalsextraktion und Merkmalsanpassung die entsprechenden Punkte zwischen den verglichenen Bildern. Dazu wird ein RANSAC Algorithmus benötigt, um die irrelevanten Korrespondentenpunkte zu beseitigen und die Genauigkeit zu verbessern. Aus dem Kameramodell wird anschließend ein mathematisches Transformationsmodell zwischen den entsprechenden Punkten in den beiden Bildern erstellt. Es ist zu beachten, dass der Prozess zur Lösung des Transformationsmodells als ein nichtlineares Optimierungsproblem angesehen werden kann. Durch die Lösung dieses Problems kann die Transformationsmatrix erhalten werden. In dieser Arbeit wird immer das erste Bild als Referenzbild gewählt und nehmen anschließend eine Bildregistrierung mit folgenden Bildern vor. Danach bekommen wir eine Reihe Bilder, die in dasselbe Koordinatensystem umgewandelt werden. Es werden je zwei Bilder subtrahiert, wodurch eine Reihe Differenzbilder entstehen. Um die folgende Detektion zu vereinfachen, wird eine Optimierungsoperation mit der Hilfe der geometrischen Eigenschaft des QR Musters unternommen und erstellen dann ein erkennendes Bild. Anschließend eine Reihe von Bildverarbeitungen bzw. Binarisierung, Medianfilter, Morphologie Operation wird behandelt. Dadurch können vereinzelt Punkte und Lücken, die durch Rauschen und Fehler verursacht wurden, entfernt werden. Schließlich wird es durch die Charakteristik des QR Musters, dessen Breiteverhältnis $1 : 1 : 3 : 1 : 1$ beträgt um QR Muster zu detektieren, also den Modulationsbereich zu bestimmen. Das Flussdiagramm wird in Abbildung 3.2 gezeigt. Die Details jeden Teils werden in den folgenden Abschnitten beschrieben.

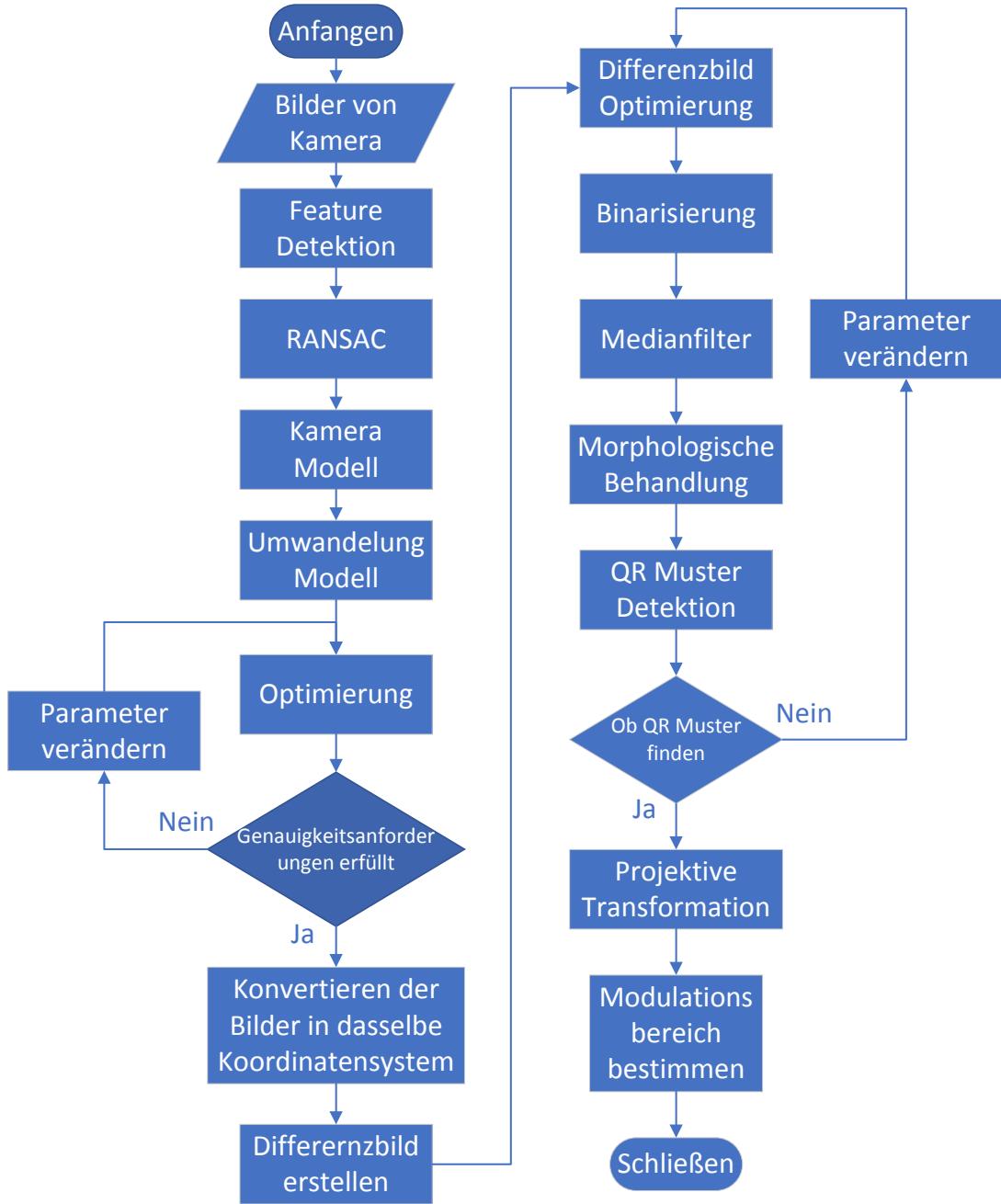


Abbildung 3.2: Flussdiagramm der ersten Methode.

3.2 Bildregistrierung

Es wird angenommen, eine Reihe Fotos wird mit einer Handheld-Kamera aufgenommen. Es kommt aufgrund von Handbewegungen zu einer leichten Verschiebung zwischen den beiden benachbarten Fotos. Wenn Sie diese Fotos subtrahieren, um die Differenzbilder zu erhalten, wird das Ergebnis sehr schlecht sein. Um dieses Problem zu lösen, wird hier Bildregistrierung eingeführt. Ein Flussdiagramm der Bildregistrierung wird in Abbildung 4.2 gezeigt.

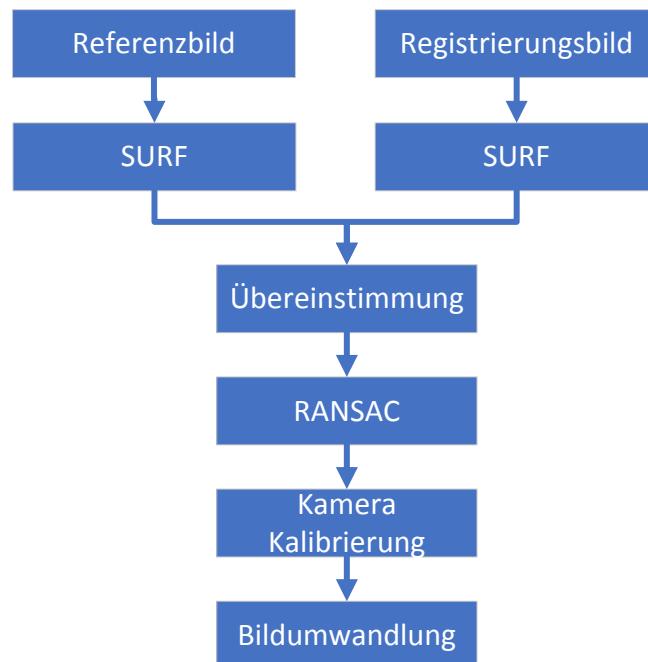


Abbildung 3.3: Flussdiagramm der Bildregistrierung.

3.2.1 SURF

In jedem Bild gibt es eindeutige Pixelwertpunkte, d.h. Merkmalspunkte des Bildes. Um die Bilder in dasselbe Koordinatensystem zu transformieren, müssen diese Merkmalspunkte erkannt und analysiert werden. Deswegen ist es in diesem Verfahren besonders wichtig, die Merkmalspunkte eines Bilds zu definieren und zu finden. Um dieses Problem zu lösen, wird das Konzept der Merkmalserkennung eingeführt. Die wird oft in Computer Vision und Bildverarbeitungs Bereichen verwendet. Die Merkmalserkennung beinhaltet Verfahren zum Berechnen von Abstraktionen von Bildinformationen und zum Treffen lokaler Entscheidungen an jedem

Bildpunkt, ob es an einem Punkt ein Bildmerkmal eines bestimmten Typs gibt oder nicht. Einige typische Merkmalserkennungen sind wie z.b. Kantenerkennung, Eckenerkennung, Tropfenerkennung und so weiter.

In dieser Arbeit wird das SURF [5] genutzt. Dieser ist ein patentierter lokaler Merkmal-Detektor und Deskriptor und kann für Aufgaben wie Objekterkennung, Bildregistrierung, Klassifizierung oder 3D-Rekonstruktion angewandt werden. Merkmalserkennungen sind eine verbesserte Version von **Scale Invariant Feature Transform** (SIFT) Merkmalserkennungen die Haar-Wavelet verwendet um die Gradientenoperation in der SIFT-Methode anzunähern, und gleichzeitig eine Integralgraph-Technik für schnelle Berechnungen verwendet. Die Faltung bezieht sich nur auf das vorherige Bild. Mit Vergrößerung des Bildkerns kann das Heruntertaktungs-Verfahren realisiert werden. Die Geschwindigkeit von SURF ist 3-7 mal schneller als die von SIFT welche in den meisten Fällen der Leistung von SIFT entspricht. Daher wird es in vielen Anwendungen eingesetzt, insbesondere in Anwendungen in denen die Laufzeitanforderungen hoch sind.

Der Verlauf einer SURF Merkmalserkennung ist wie folgend:

- **Aufbau einer hessischen Matrix.**

Die Hesse-Matrix stellt den Kern des SURF Algorithmus dar. Zur Vereinfachung der Operation wird für die Funktion $f(x, y)$ angenommen, dass sich die Hesse-Matrix H aus partiellen Ableitungen und Funktionen zusammensetzt.

$$H(f(x, y)) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \cdot \partial y} \\ \frac{\partial^2 f}{\partial x \cdot \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \quad (3.1)$$

Determinante der H-Matrix wie folgt:

$$\det(H) = \frac{\partial^2 f}{\partial x^2} \cdot \frac{\partial^2 f}{\partial y^2} - \left(\frac{\partial^2 f}{\partial x \cdot \partial y} \right)^2 \quad (3.2)$$

Der Wert der Determinante ist der Eigenwert der H-Matrix. Durch dessen positiven und negativen Wert wird bestimmt, ob der Punkt ein Extrempunkt ist oder nicht. Im SURF Algorithmus wird das Bildpixel $I(x, y)$ anstelle des Funktionswertes $f(x, y)$ verwendet. Es wird eine Gauß-Funktion zweiter Ordnung als Filter verwendet. Die zweiten partiellen Ableitungen können durch Faltung zwischen bestimmten Kernen berechnet werden. Dadurch können die Werte der drei Matrixelemente der H-Matrix ebenfalls berechnet werden.

$$\begin{aligned}\mathcal{H}(\mathbf{x}, \sigma) &= \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix} \\ L(\mathbf{x}, \sigma) &= G(\sigma) * I(\mathbf{x}) \\ G(\sigma) &= \frac{\partial^2 g(\sigma)}{\partial x^2}\end{aligned}\tag{3.3}$$

Hier $L_{xx}(\mathbf{x}, \sigma)$ bedeutet die Faltung der zweiten Gaußschen Ableitung $G(\sigma)$ mit dem Bild I in Punkt $\mathbf{x}(x,y)$, ähnlich für $L_{xy}(\mathbf{x}, \sigma)$ und $L_{yy}(\mathbf{x}, \sigma)$. Auf diese Weise kann der Wert der Determinante für jedes Pixel in dem Bild berechnet werden und dieser Wert kann verwendet werden um den Merkmalspunkt festzustellen. Zur einfacheren Anwendung schlägt Herbert Bay[5] vor, L mit einer Approximation zu ersetzen. Um den Fehler zwischen dem genauen Wert und der Approximation auszugleichen, kann die H-Matrix-Diskriminante wie folgt ausgedrückt werden:

$$\det(\mathcal{H}_{\text{Approx}}) = D_{xx}D_{yy} - (0.9D_{xy})^2\tag{3.4}$$

• Erstellen des Maßstabraums

Der Maßstabraum $L(\mathbf{x}, \sigma)$ des Bildes ist die Darstellung dieses Bildes bei unterschiedlichen Auflösungen(Skalierungen). Im Bereich der Computer Vision wird der Maßstabraum symbolisch als Bildpyramide ausgedrückt, wobei die Eingangsbildfunktion wiederholt mit dem Kern der Gaußschen Funktion gefaltet und wiederholt unterabgetastet wird. Diese Methode wird hauptsächlich für die Implementierung des SIFT Algorithmus verwendet. Jede Bildschicht hängt jedoch von der vorherigen Bildschicht ab, und das Bild muss in der Größe angepasst werden. Daher büßt diese Berechnungsmethode viele Leistung ein. Das Gegenstück in SURF ist die Erhöhung der Größe des Bildkerns. Dies ist ein Unterschied zwischen dem SIFT Algorithmus und dem SURF Algorithmus bei der Verwendung des Pyramidenprinzips. Der Algorithmus ermöglicht, dass mehrere Bilder des Maßstabraums gleichzeitig verarbeitet werden, ohne dass das Bild unterabgetastet wird, wodurch die Leistung des Algorithmus verbessert wird. Das linke Bild in Abbildung 3.4 ist eine Pyramidenstruktur, die auf herkömmliche Weise erstellt wird. Die Größe des Bildes wird geändert, und die Operation wird die Unterebene unter Verwendung der Gaußschen Funktion wiederholt glätten. Der SURF Algorithmus auf der rechten Seite in Abbildung 3.4 erhält das ursprüngliche Bild und ändert nur die Filtergröße.

• Präzise Lokalisierung von Feature-Punkten

Die Größe jedes Pixels die von der Hesse-Matrix verarbeitet wird, wird mit den 26 Punkten der drei Dimensionen im Raum, wie in Abbildung 3.5 verglichen. Wenn einer davon das Maxi-

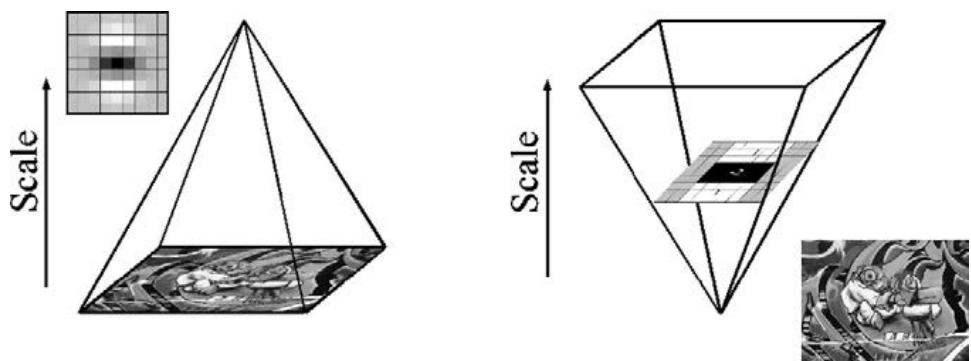


Abbildung 3.4: Scale space.

um oder Minimum dieser 26 Punkte ist, wird er als vorläufiger Merkmalspunkt beibehalten. Das dreidimensionale lineare Interpolationsverfahren wird angewandt, um die Merkmalspunkte des Subpixel-Niveaus zu erhalten, und die Punkte deren Werte kleiner als ein bestimmter Schwellenwert sind werden ebenfalls entfernt.

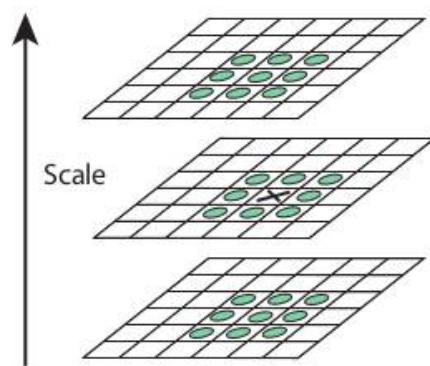


Abbildung 3.5: Extremwert Erkennung.

• Hauptrichtungsermittlung

SIFT wählt die Hauptrichtung des Merkmalspunkts unter Verwendung des Gradientenhistogramms im Merkmalspunktbereich aus. Die Richtung, in der der Bin-Wert des Histogramms der größte und oder 80% des maximalen Bin-Werts überschreitet, wird als Hauptrichtung des Merkmalspunkts genommen. Dagegen beim SURF, wird nicht das Gradientenhistogramm, sondern die Haar-Wavelet-Eigenschaft im Merkmalspunktbereich statistisch analysiert. Das heißt, im Bereich der Merkmalspunkte (zum Beispiel innerhalb eines Kreises mit einem Radius von $6s$ Klammer auf, wobei s der Maßstab ist auf dem der Punkt liegt) die Summe der Horizontal-Haar-Wavelet-Merkmale und der Vertikal-Haar-Wavelet-Merkmale aller Punkte im 60-Grad-Sektor($\pi/3$) gezählt. Die Größe des Haar-Wavelets beträgt $4s$, sodass jeder Sektor einen Wert bekommt. Anschließend wird 60-Grad-Sektor in einem bestimmten Intervall gedreht, schließlich lässt die Richtung des Sektors anhand des Maximalwerts als Hauptrichtung

vom Merkmalspunkt nehmen. Ein schematisches Diagramm des Prozesses ist wie folgt in Abbildung 3.6.

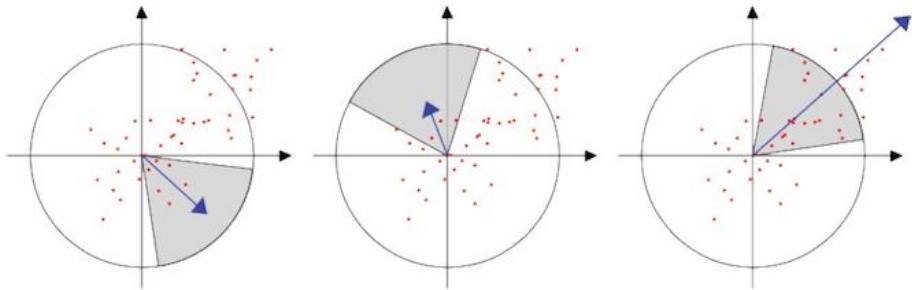


Abbildung 3.6: Dominante Orientierung feststellen.

- **Merkmalspunkt Deskriptor Generierung**

SURF generiert einen quadratischen Rahmen um den Merkmalspunkt. Die Größe der Seite des Rahmens ist $20s$ (s ist die Skala, bei der der Merkmalspunkt erkannt wird). Die Richtung des Rahmens ist die Hauptrichtung, die im vorherigen Schritt erfasst wurde. Der Rahmen wird dann in 16 Unterbereiche unterteilt, von denen jeder die Haar-Wavelet-Merkmale der horizontalen und vertikalen Richtungen von 25 Pixeln berechnet. Hier die horizontalen und vertikalen Richtungen sind relativ zur Hauptrichtung. Das Haar-Wavelet-Merkmal ist die Summe der horizontalen Richtungswerte, die Summe der absoluten Werte in der horizontalen Richtung, die Summe der vertikalen Richtungen und die Summe der absoluten Werte in der vertikalen Richtung. Das schematische Diagramm in Abbildung 3.7 zeigt dieses Prozesse.

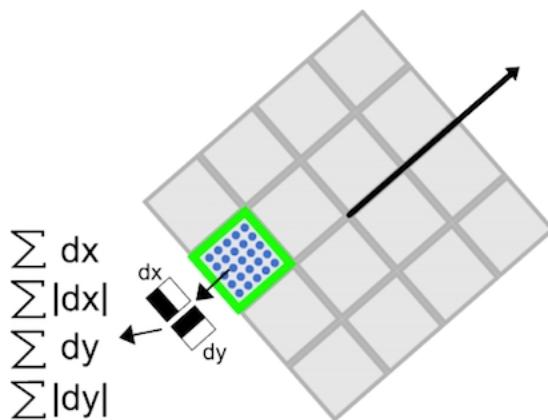


Abbildung 3.7: Merkmalspunkt Deskriptor.

Auf diese Weise hat jeder kleine Bereich 4 Werte, so dass jeder Merkmalspunkt über einen $16 * 4 = 64$ dimensionalen Vektor verfügt, der halb so klein wie SIFT(128 Dimension) ist, deshalb den Anpassungsprozess beim Merkmalanpassungsprozess stark beschleunigt. Die

folgende Abbildung 3.8 zeigt den Merkmalspunkt, den wir durch den SURF-Algorithmus erhalten haben.

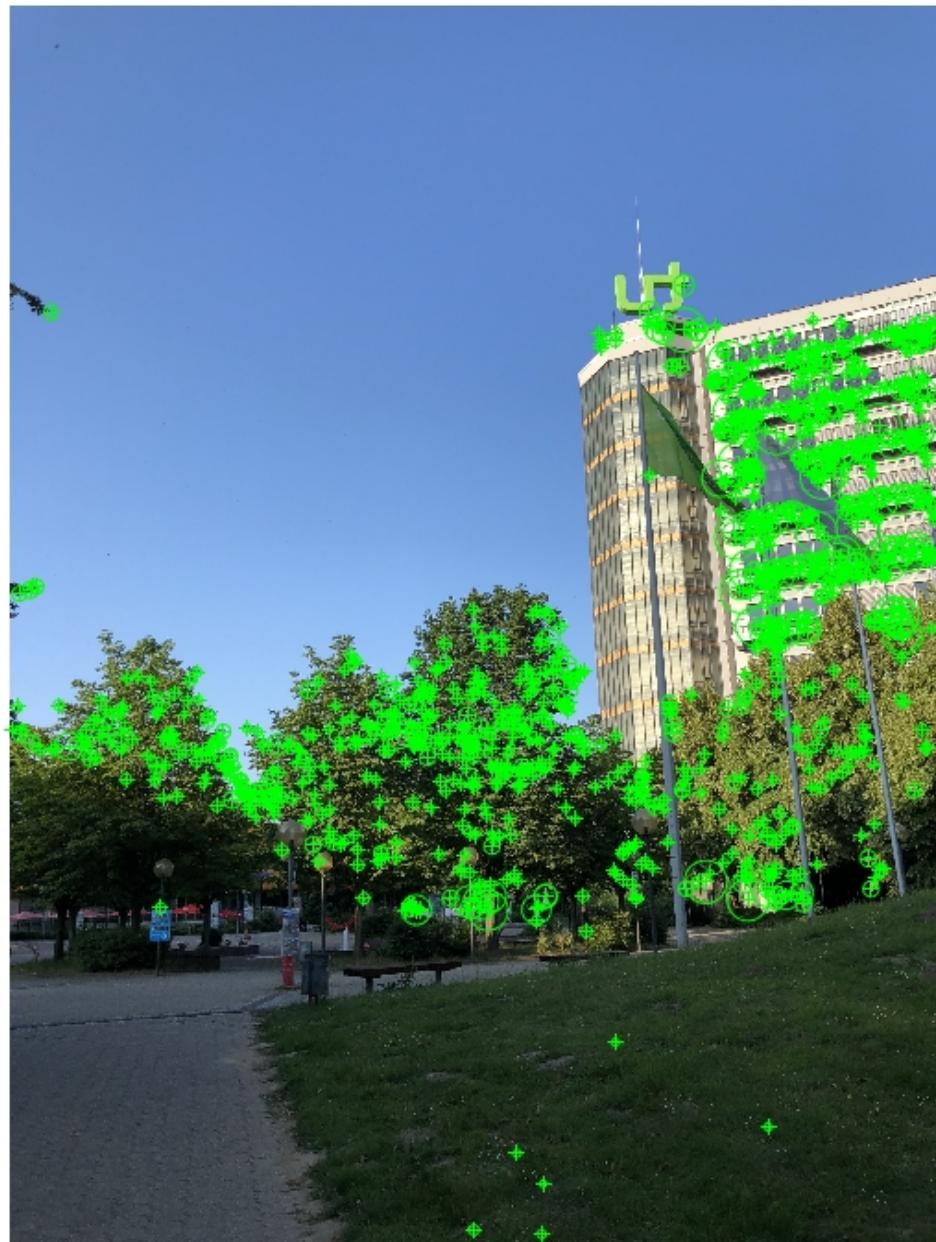


Abbildung 3.8: SURF Merkmal.

3.2.2 RANSAC

Nach SURF Merkmalserkennung werden die Merkmale von zwei benachbarten Bildern generiert. Diese Merkmale werden dann extrahiert und abgeglichen um entsprechende Punkte in

den benachbarten zwei Bildern zu erhalten. Leider verbleiben durch diese Operation immernoch viele fehlerhafte zusammenpassende Paare. Deswegen wird hier RANSAC verwendet, um die falschen Punkte zu beseitigen.

Der im Jahre 1981 vorgestellte RANSAC Algorithmus von Fischler und Bolles [6], ist ein allgemeiner Parameterschätzungsansatz um den großen Anteil von Ausreißern in den Eingabedaten zu bewältigen. Im Gegensatz zu vielen der üblichen robusten Schätzverfahren wie M-Schätzer und kleinste Quadrate, die von der Computer Vision Community aus der Statistik-Literatur übernommen wurden, wurde RANSAC von der Computer-Vision-Community entwickelt.

Ein einfaches Beispiel ist in der Abbildung 3.9 dargestellt. Das Ziel besteht darin, die am besten geeignete Linie unter einer Menge von Datenpunkten zu finden. Wenn es die einfache Methode der kleinsten Quadrate verwendet um diese Linie zu finden, wie auf der linken Seite gezeigt, kann es leider nicht richtig funktionieren, da die Methode der kleinsten Quadrate von allen Datenpunkten beeinflusst wird. Dagegen kann mit RANSAC das Modell nur von der inlieren Punkte berechnet werden, wie die Ergebnisse wie auf der rechten Seite zeigen.

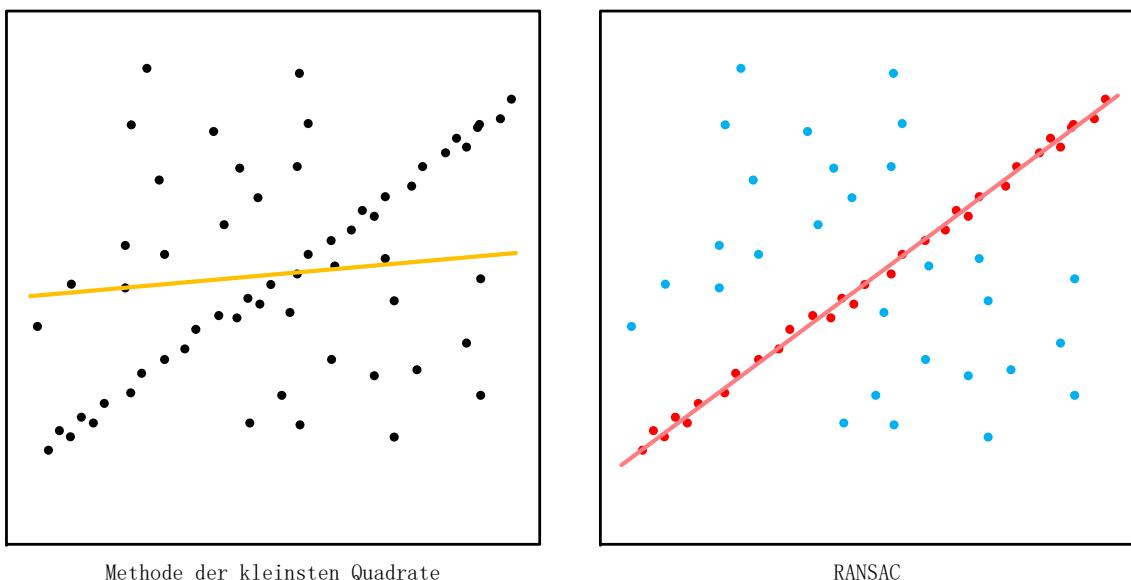


Abbildung 3.9: Linien Detektion.

RANSAC ist ein Wiederholungsprobennahme-Verfahren, welches durch die minimale Anzahl von Beobachtungspunkten (Datenpunkten) die Kandidatenlösung generiert. Diese Dateipunkte sind erforderlich, um die zugrundeliegenden Modellparameter zu schätzen woauf Fischler und Bolles [6] hingewiesen haben. Die Erhaltung einer anfänglichen Lösung und Aus-

schließung der Ausreißer mit dem RANSAC Verfahren benötigt nicht so viele Daten, sondern verwendet die kleinste mögliche Menge und fährt fort diese Menge mit konsistenten Datenpunkten zu vergrößern.

Der grundlegende Algorithmus ist wie folgt zusammengesetzt:

- Zufällige Auswahl der Mindestanzahl erforderlicher Punkte zum Bestimmen der Modellparameter.
- Lösen der Parameter des Modells.
- Bestimmen wie viele Punkte aus der Menge aller Punkte mit einer vordefinierten Toleranz ϵ übereinstimmen
- Wenn ein Bruchteil der Anzahl von Inlieren über die Gesamtzahl der Punkte in dem Satz einen vordefinierten Schwellenwert τ überschreitet, schätzen die Modellparameter mit allen identifizierten Inlieren und terminieren weiter.
- Ansonsten Wiederholung der Schritte 1 bis 4 (maximal N-mal).

N ist die Anzahl der Iterationen. Diese wird hoch genug gewählt, um die Wahrscheinlichkeit p (normalerweise 0.99) sicherzustellen, dass mindestens eine der Gruppen von Stichproben keinen Ausreißer enthält. Anschließend die Wahrscheinlichkeit, dass bei N mal iterieren mit erforderlich minimaler Anzahl von Punkten (hier m annahmen) mindestens ein Ausreißer mit ausgewählt wird, läuft folgendermaßen:

$$1 - p = (1 - u^m)^N \quad (3.5)$$

u stellt die Wahrscheinlichkeit dar, dass jeder ausgewählte Datenpunkt ein Inlier ist. Dagegen ist $v = 1 - u$ die Wahrscheinlichkeit, dass jeder ausgewählte Datenpunkt ein Ausreißer ist. Durch einige Gleichheitsumwandlungen kann die Anzahl der Iterationen ausgedrückt werden als:

$$N = \frac{\log(1 - p)}{\log(1 - (1 - v)^m)} \quad (3.6)$$

Abbildung 3.10 zeigt die passenden Punkte durch Merkmalübereinstimmung mit SURF Detektion. Es ist ersichtlich, dass es viele fehlerhafte Kombinationen gibt. Durch die Anwendung von RANSAC kann dieses Problem effektiv gelöst und die übereinstimmenden Punkte verfeinert werden, wie das Ergebnis in Abbildung 3.11 zeigt.



Abbildung 3.10: Ohne RANSAC.

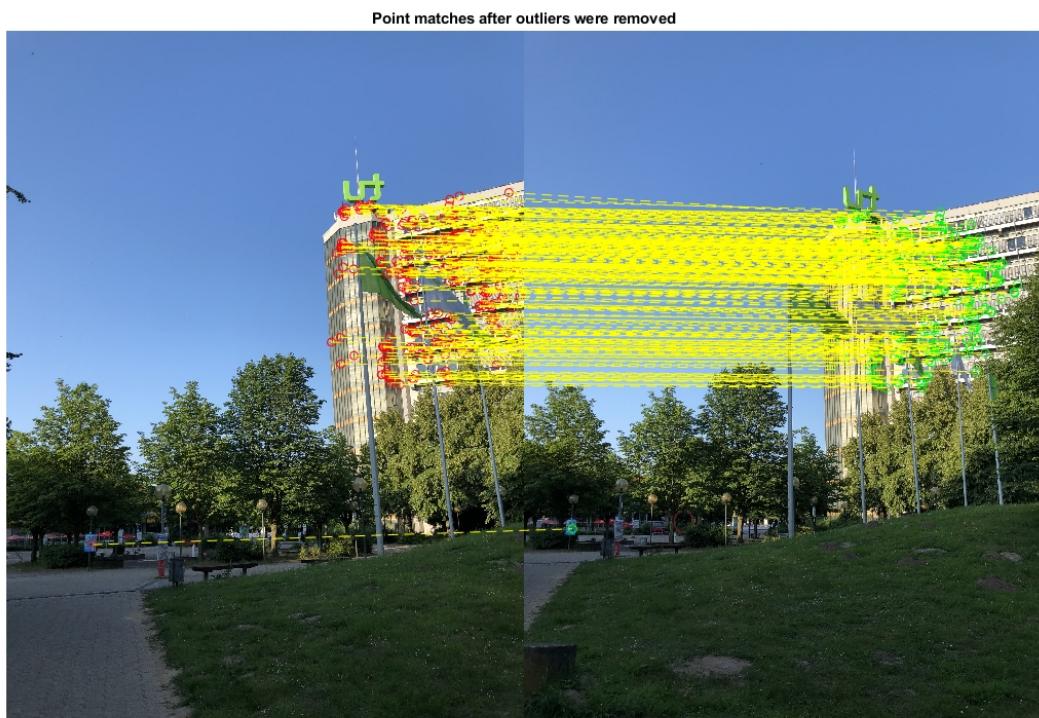


Abbildung 3.11: Mit RANSAC.

3.2.3 Bilder Umwandlung

Wie in dem vorherigen Abschnitten vorgestellt, durch Verwenden des SURF werden übereinstimmende Punkte in aufeinanderfolgenden Bildern gefunden, anschließend lassen sich durch RANSAC die Ausreißer ausschließen. Das Ziel dieses Abschnitts besteht darin, das Bild in dasselbe Koordinatensystem zu konvertieren. Der erste Schritt ist ein Kameramodell zu erstellen und anschließend die Umwandlungsbeziehung zwischen den entsprechenden Punkten in den zwei Bildern zu erhalten, um schließlich durch den Optimierungsalgorithmus die endgültige Transformationsmatrix zu erhalten. Die verschiedenen Schritte werden im Folgenden detailliert beschrieben.

Kamera Modell

Das Modell der Lochkamera ist in Abbildung 3.12 dargestellt. In dem Modell ist O_C das optische Zentrum(Fokus) und f ist die Kamerabrennweite.

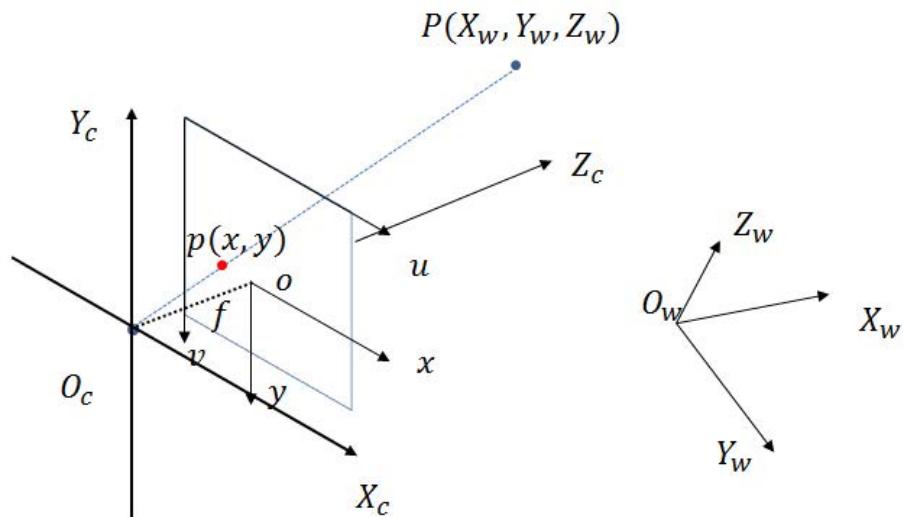


Abbildung 3.12: Modell einer Lochkamera.

Die vier Koordinatensysteme im Modell sind wie folgt definiert:

- 3D Weltkoordinatensystem $P(X_W, Y_W, Z_W)$
Punktkoordinaten werden durch homogene Koordinaten dargestellt: $\tilde{X}_W \sim (X_W, Y_W, Z_W, 1)^T$
- 3D Kamerakoordinatensystem $C(X_C, Y_C, Z_C)$
Punktkoordinaten werden durch homogene Koordinaten dargestellt: $\tilde{X}_C \sim (X_C, Y_C, Z_C, 1)^T$

- 2D Bildabbildung Koordinatensystem $p(x, y)$
Punktkoordinaten werden durch homogene Koordinaten dargestellt: $\tilde{x} \sim (x, y, 1)^T$
- 2D Bildpixel Koordinatensystem $I(u, v)$
Punktkoordinaten werden durch homogene Koordinaten dargestellt: $\tilde{u} \sim (u, v, 1)^T$

Unter diesem Modell wird ein 3D-Punkt im Weltkoordinatensystem durch drei Koordinaten den 2D-Bildpixelkoordinaten zugeordnet.

(1). 3D-Weltkoordinatensystem zum 3D-Kamera-Koordinatensystem.

Die Transformation vom Weltkoordinatensystem zum Kamerakoordinatensystem ist eine Starrekörpertransformation, d.h. das Objekt verformt sich nicht und sondern nur rotiert und parallel verschoben. Diese Transformation wird in Abbildung 3.13 gezeigt. R bedeutet Rotationsmatrix und T ist Translationsmatrix.

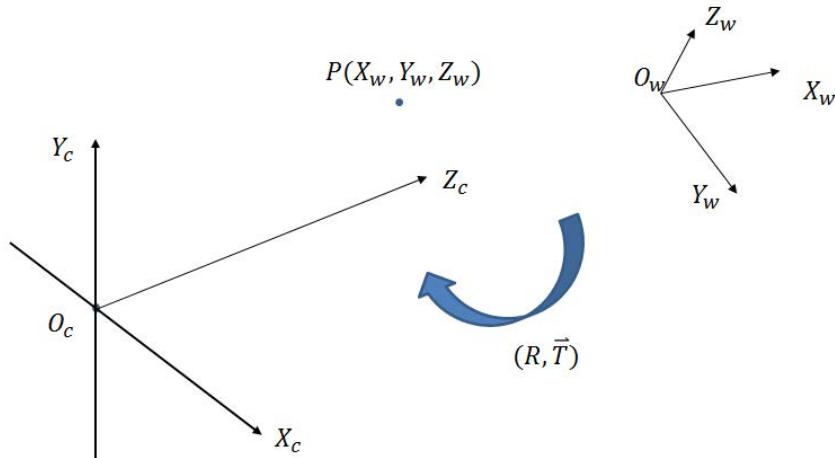


Abbildung 3.13: Transformation vom Weltkoordinatensystem zum Kamerakoordinatensystem.

Um die entsprechende Rotationsmatrix zu erhalten, wird um verschiedene Winkel die Koordinatenachsen gedreht. Ein simples Beispiel wird in Abbildung 3.14 gezeigt.

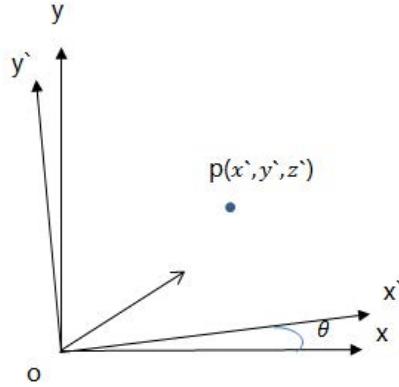


Abbildung 3.14: Rotation um Z-Achse.

Aus dem Bild können wir leicht entnehmen:

$$\begin{cases} x = x' \cos \theta - y' \sin \theta \\ y = x' \sin \theta + y' \cos \theta \\ z = z' \end{cases} \quad (3.7)$$

In Matrixform wie folgend ausgedrückt:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R_1 \cdot \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \quad (3.8)$$

In ähnlicher Weise dreht sich die x-Achse, y-Achse um φ und ω Grad wie folgt:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \varphi & -\sin \varphi \\ 0 & \sin \varphi & \cos \varphi \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R_2 \cdot \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \quad (3.9)$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos \omega & 0 & \sin \omega \\ 0 & 1 & 0 \\ -\sin \omega & 0 & \cos \omega \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R_3 \cdot \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} \quad (3.10)$$

Schließlich kann die Rotationsmatrix erhalten werden:

$$R = R_1 \cdot R_2 \cdot R_3 \quad (3.11)$$

Durch das kombinieren der obigen Ergebnisse, lassen sich die Koordinaten von Punkt P im Kamerakoordinatensystem bestimmen:

$$\begin{bmatrix} X_C \\ Y_C \\ Z_C \end{bmatrix} = R \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + T \quad (3.12)$$

Im homogenen Koordinatensystem dargestellt:

$$\begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ \vec{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (3.13)$$

(2). 3D-Kamera-Koordinatensystem zum 2D-Bildabbildung Koordinatensystem.

Die Transformation vom Kamerakoordinatensystem zum Bildkoordinatensystem gehört zur perspektivischen Projektionsbeziehung von 3D zu 2D, wie in Abbildung 3.15 gezeigt.

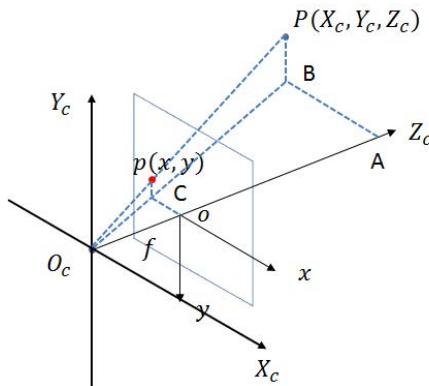


Abbildung 3.15: Transformation vom Kamerakoordinatensystem zum Bildkoordinatensystem.

Es gibt zwei Paare mit ähnlichen Dreiecken:

$$\begin{aligned} \triangle ABC &\sim \triangle OBC \\ \triangle PBC &\sim \triangle OBC \end{aligned} \quad (3.14)$$

Aus ähnlichen Dreiecksbeziehungen kann diese Gleichung erstellt werden:

$$\frac{AB}{OC} = \frac{AO_C}{OO_C} = \frac{PB}{PC} = \frac{X_C}{x} = \frac{Z_C}{f} = \frac{Y_C}{y} \quad (3.15)$$

Durch die Transformationsgleichung folgt:

$$x = f \cdot \frac{X_C}{Z_C}, y = f \cdot \frac{Y_C}{Z_C} \quad (3.16)$$

Im homogenen Koordinatensystem dargestellt:

$$Z_C \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} \quad (3.17)$$

Zu dieser Zeit ist die Einheit des Projektionspunkts p noch nicht Pixel, sondern Millimeter und muss weiter in das Pixelkoordinatensystem umgewandelt werden.

(3). 2D-Bildabbildung Koordinatensystem zum 2D-Bildpixel Koordinatensystem.

Das Pixelkoordinatensystem und das Bildkoordinatensystem befinden sich alle auf der Abbildungsebene, jedoch sind die jeweiligen Ursprünge und Maßeinheiten unterschiedlich. Der Ursprung des Bildkoordinatensystems ist der Schnittpunkt der optischen Achse der Kamera und der Abbildungsebene, üblicherweise der Mittelpunkt der Abbildungsebene oder der Hauptpunkt. Die Einheit des Bildkoordinatensystems ist Millimeter welche eine physikalische Einheit ist. Die Einheit des Pixelkoordinatensystems ist Pixel. Wie gewohnt wird beschrieben in welcher Zeile und Spalte ein Pixel ist. So ist der Übergang zwischen den beiden wie in Abbildung 3.16 folgt.

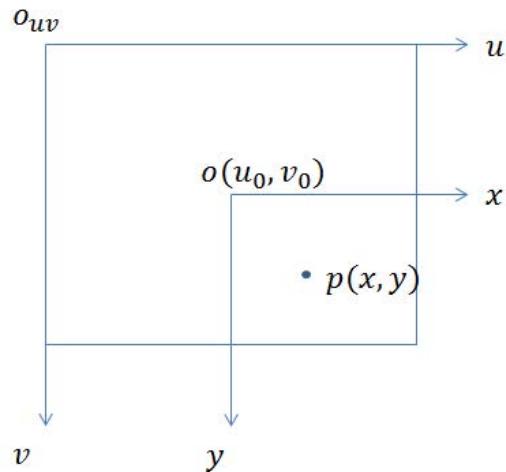


Abbildung 3.16: Konvertierung vom Bildkoordinatensystem zum Pixelkoordinatensystem.

d_x und d_y ist die Größe jedes Pixels in den X- und Y-Achsenrichtungen. Jedes Pixel des Bildes hat die folgende Beziehung zwischen den zwei Koordinatensystemen.

$$\begin{cases} u = \frac{x}{d_x} + u_0 \\ v = \frac{y}{d_y} + v_0 \end{cases} \quad (3.18)$$

Im homogenen Koordinatensystem darstellt:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{d_x} & 0 & u_0 \\ 0 & \frac{1}{d_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (3.19)$$

Es ist erwähnenswert, dass das Bildkoordinatensystem eine zweidimensionale Ebene(Bildebene), praktisch die Oberfläche des Kamera-CCD-Sensors ist. Jeder CCD-Sensor hat eine bestimmte Größe und Auflösung. Diese beiden bestimmen die Konvertierungsbeziehung. Gegeben sei in simples Beispiel: Eine Größe des CCD-Sensors ist $8 \text{ mm} \times 6 \text{ mm}$, die Auflösung dafür ist $640 \text{ pixels} \times 480 \text{ pixels}$. Die Beziehung zwischen mm und Pixel ist 80 pixel/mm . Ist die physikalische Größe jedes Pixels des CCD-Sensors $d_x \times d_y$ sein, entspricht diese $d_x = d_y = \frac{1}{80} \text{ mm}$.

Durch die Umwandlung der obigen vier Koordinatensysteme kann ein Punkt vom Weltkoordinatensystem zum Pixelkoordinatensystem extrahiert werden.

$$\begin{aligned} Z_C \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} &= \begin{bmatrix} \frac{1}{d_x} & 0 & u_0 \\ 0 & \frac{1}{d_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} R & t \\ \vec{o} & 1 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} R & t \\ \vec{o} & 1 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \end{aligned} \quad (3.20)$$

Die erste Matrix der rechten Gleichung ist die allgemein bekannte interne Referenz der Kamera. Dagegen ist die zweite Matrix die externe Referenz der Kamera. Beide Parameter der Kamera können durch Zhang Zhengyou [7] Kalibrierung erhalten werden. Einige typische Kamera Parameter vom Werk aus sind der Tabellen 3.1 zu entnehmen.

Wenn die internen und externen Parameter der Kamera bekannt sind, ist es möglich daraus die Projektionsmatrix zu erstellen um daraus die Bildkoordinaten jeden beliebigen räumlichen

Tabelle 3.1: Parameter der Kameras im Vergleich.

Parameter	Google Pixel	Google Pixel2	Iphone X
Sensor Größe "	1/2.3	1/2.6	1/3
Bild Auflösung <i>pixels</i>	4048×3036	4032×3024	4032×3024
Pixel Größe μm	1.544	1.4	1.22
Brennweite <i>mm</i>	4.67	4.47	3.99
Formatfaktor 35mm	5.55	6.04	7.02

Punkts zu erhalten. Dagegen wenn die Position $m(u, v)$ einer Bildkoordinate, und die Parameter innerhalb und außerhalb der Kamera bekannt sind, kann der entsprechende Punkt in Wert Koordinate nur ungefähr bestimmt werden. Der Grund dafür ist, dass die Z_c Information während des Projektionsprozesses eliminiert wird.

Umwandlungsmodelle

Als nächstes wird die Umwandlungsbeziehung zwischen den entsprechenden Punkten in den beiden Bilder vorgenommen. Zuerst wird in dieser Arbeit eine vereinfachte Situation betrachtet, d.h. nur mit Rotationseinfluss. Abbildung 3.17 zeigt das 3D-Rotationsbewegungsmodell der Kamera. Die Position des optischen Zentrums ändert sich, während der Kamerabewegung im Drehbewegungsmodell der Kamera, nicht. Unter diesem Modell ist die Abbildungsbeziehung zwischen dem Punkt X im Weltkoordinatensystem und der Bildkoordinate x im homogenen Koordinatensystem dargestellt:

$$x = KRX, X = \lambda K^{-1}x \quad (3.21)$$

Wobei K der interne Parameter der Kamera ist, wie zuvor definiert. λ ist der unbekannte Skalierungsfaktor, d.h. dass unter dem Kameramodell die Quelle jeder Bildpunktkoordinaten einem Strahl zugeordnet ist.

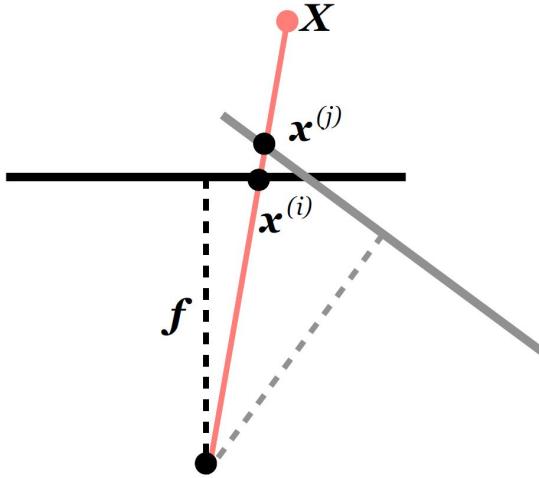


Abbildung 3.17: Rotationsbewegungsmodell.

Nun wird die Beziehung zwischen Bildpunkten in einem Framepaar zwei verschiedener Kameraausrichtungen hergeleitet (siehe Abbildung 3.17). Für einen Weltkoordinatepunkt X sind die projizierten Punkte x_i und x_j in der Bildebene von zwei Bildern i und j gegeben durch

$$x_i = KR_i X, x_j = KR_j X \quad (3.22)$$

Nach weiterem anordnen und ersetzen von X , wird eine Beziehung aller Punkte im Bildrahmen i auf alle Punkte im Rahmen j erhalten:

$$x_j = KR_j R_i^T K^{-1} x_i \quad (3.23)$$

Bisher wird nur die Beziehung zwischen zwei Bildern der selben Aufnahme betrachtet. Diese Einschränkung wird gelockert, indem Frames von einer Kamera, die sich gemäß R dreht, zu einer anderen Kamera, die sich gemäß R' dreht, abgebildet werden. Es gibt eine Hypothese, dass beide Kamerazentren sich im Ursprung befinden. Mit der Warping-Matrix können die abgebildeten Punkte der Kameras wie folgt definiert werden:

$$W = KR' R^T K^{-1} \quad (3.24)$$

Hier wird das erste Bild als Referenzbild mit einem Rotationswinkel von 0 Grad gewählt, wodurch die Gleichung vereinfacht werden kann:

$$W = KRK^{-1} \quad (3.25)$$

Kombiniert mit Formel 3.23 kann es ausgedrückt werden als:

$$x_j = Wx_i \quad (3.26)$$

Diese Formel zeigt, dass jeder Punkt in dem Bild i nach der Transformationsmatrix W in einen entsprechenden Punkt in dem Bild j umgewandelt werden kann. Weiter in einem allgemeineren Fall, d.h. derzeit nicht nur mit Rotationseinfluss, sondern auch Translationseinfluss nehmen. Der Ableitungsverlauf ist im Allgemeinen gleich. Der Hauptunterschied ist die Anzahl der Parameter, die ursprünglich nur 3 Rotationsparameter aber jetzt 6 Parameter einschließlich 3 Rotationsparameter und 3 Translationsparameter beinhalten. Die neue Warping-Matrix sieht folgendermaßen aus:

$$W = \begin{bmatrix} f & 0 & \frac{w}{2} & 0 \\ 0 & f & \frac{h}{2} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} R_{11} & R_{21} & R_{31} & 0 \\ R_{12} & R_{22} & R_{32} & 0 \\ R_{13} & R_{23} & R_{33} & 0 \\ t1 & t2 & t3 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{f} & 0 & -\frac{w}{2f} & 0 \\ 0 & \frac{1}{f} & -\frac{h}{2f} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.27)$$

Transformationsoptimierung

In den letzten beiden Abschnitten wurde die Transformationsmatrix zwischen den beiden Bildern aus dem Kameramodell abgeleitet. Hier in diesen Abschnitt werden die Parameter der Transformationsmatrix berechnet, also wird nun die Transformationsmatrix bestimmt. Hier sind die entsprechenden Punkte von zwei benachbarten Bildern x_i, x_j bekannt, und die Umwandlungsbeziehung dazwischen wird als die Formel 3.26 dargestellt. Angesichts dieser Bedingung kann die Berechnung des Transformationsmatrix als ein Optimierungsproblem formuliert werden, wobei der Fehler J bei der Summenwertbildung im Quadrat aller Punktkorrespondenzen minimiert werden soll:

$$J = \sum_{(i,j)} \|x_j - Wx_i\|^2 \quad (3.28)$$

Zu Beachten ist, dass dies ein nichtlineares Optimierungsproblem ist. Einige nichtlineare Optimierer können ebenfalls verwendet werden um diese Zielfunktion zu minimieren. Jedoch ist es bewiesen, dass der Koordinatenabstieg durch direkte objektive Funktionsbewertung schnell konvergiert. Jedes Mal, wenn ein Schritt gemacht wird bei dem die Zielfunktion J nicht abnimmt, wird die Schrittrichtung umgekehrt und verringert die Schrittweite des entsprechenden Parameters. Der Algorithmus endet, sobald die Schrittgröße für alle Parameter unter einen gewünschten Schwellenwert fällt (d.h. Wenn die Zielgenauigkeit erreicht ist).

Der detaillierte Algorithmus ist wie folgt dargestellt. Einige Definitionen werden hier eingeführt.
 P_0 ist der Vektor, der die Anfangswerte der Parameter speichert. Anschließend speichert er die Schrittgröße jedes Parameters in den Vektor d_p . $temp_{dp}$ ist der Vektor, in dem die gewünschten Schwellenwerte der Parameter gespeichert werden. D ist die Anzahl der Unbekannter Parameter und W die Transformationsmatrix.

1. Zuerst durch Anfangswert P_0 Berechnung den anfänglich Fehler J_0 .
 2. Verändern eines Parameters in eine entsprechende Schrittrichtung mit entsprechender Schrittgröße.
 3. Berechnen der neuen Fehler J_{new} .
 4. Vergleichen der beiden Fehler, ob der neue Fehler J_{new} kleiner ist. Wenn Ja wird dieser Wert beibehalten und des Objekt wird auf den nächsten Parameter geändert. Zurückkehren zum zweiten Schritt. Dort wird die Schrittrichtung umgekehrt, verringert die Schrittgröße auf ein Drittel des vorherigen Werts reduziert. Ändern des Objekt auf den nächsten Parameter und Rückkehr zum zweiten Schritt zurück.
 5. Wenn alle Schrittgrößen in d_p den zuvor festgelegten Schwellenwert erreichen, wird der Algorithmus, mit Ausgabe des minimalen Fehlers J und der entsprechenden Transformationsmatrix, terminiert.

Das Flussdiagramm des Algorithmus ist wie in Abbildung 3.19 darstellt.

Durch die Transformationsmatrix können die Koordinaten des zweiten Bildes in die Koordinaten des ersten Bildes übertragen werden. Abbildung 3.18 zeigt diesen Verlauf. Schließlich kann durch Subtraktion das Differenzbild erzeugt werden.

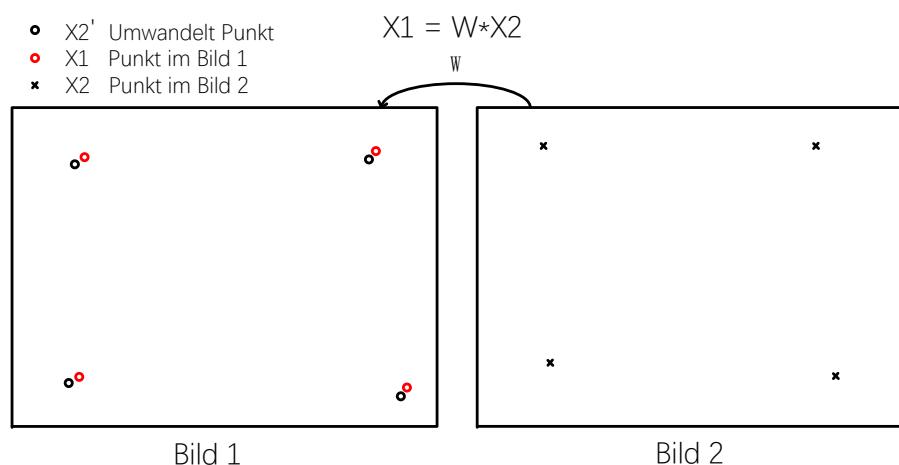


Abbildung 3.18: Transformation in die selben Koordinate.

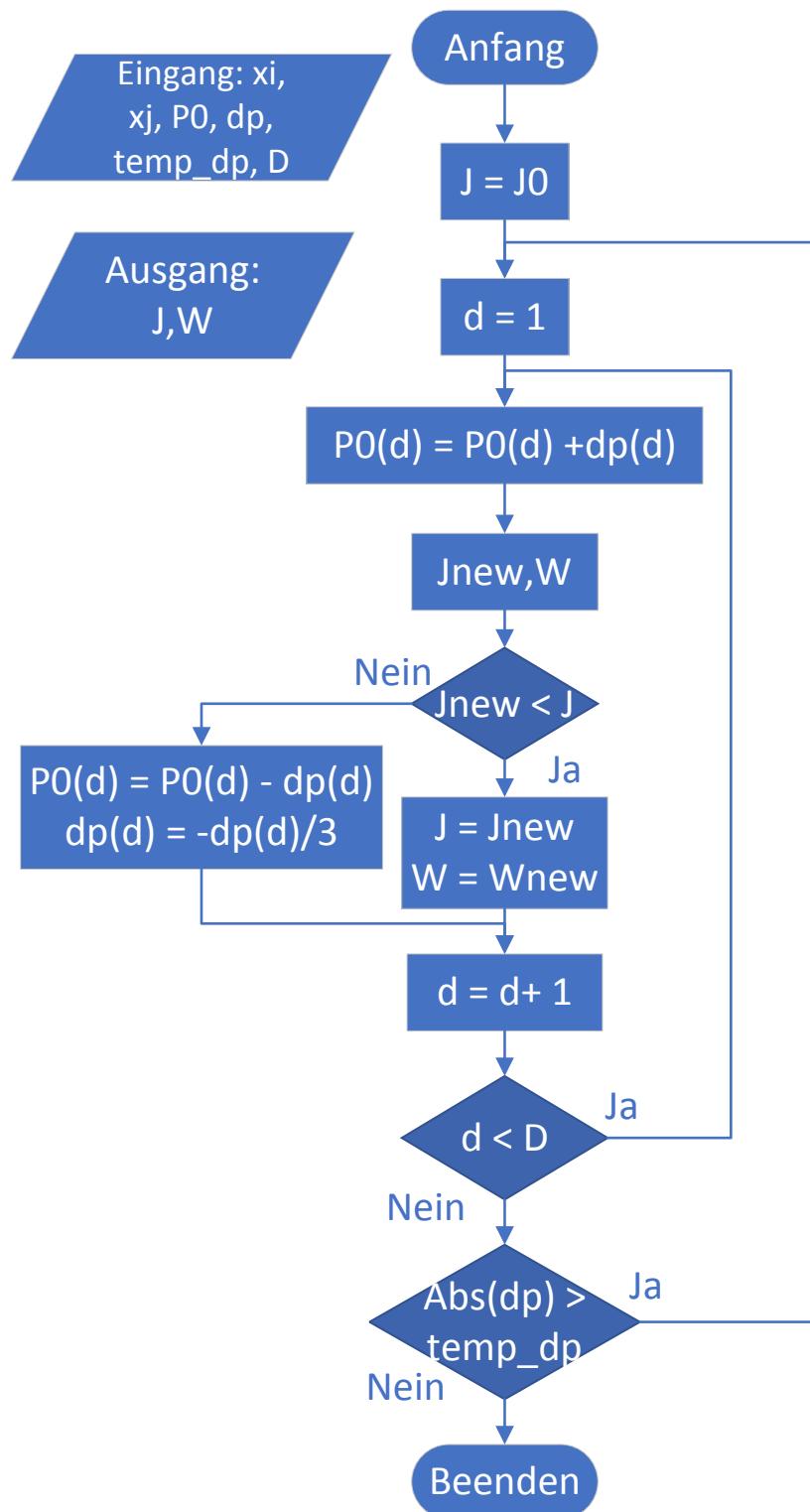


Abbildung 3.19: Flussdiagramm für Optimierung.

3.3 Differenzbild Optimierung

Die Bildregistration liefert eine Reihe Bilder von der Kamera, deren Koordinaten in dasselbe Koordinatensystem umgewandelt wurden. Es werden je zwei Bilder ausgewählt und subtrahiert, um eine Reihe Differenzbilder zu erhalten. Das Ziel in diesem Abschnitt ist, von diesen Differenzbildern ein zu detektierendes Bild herzustellen, die QR Muster Detektion vereinfacht werden kann. Es sollte hier beachtet werden, dass aufgrund der Zeitsynchronisation die QR Muster in den Differenzbildern einige unerwartete Effekte aufweisen könnten. Eine mögliche Formel der Differenzbilder wie folgt zeigt. Nehmen an, dass es in vertikaler Richtung ist.

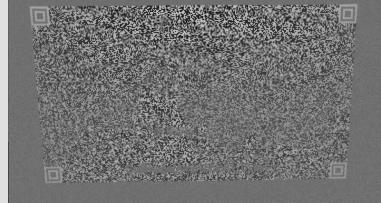
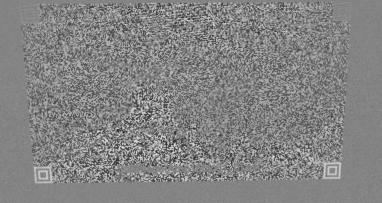
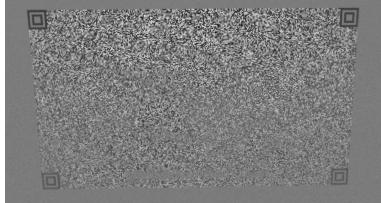
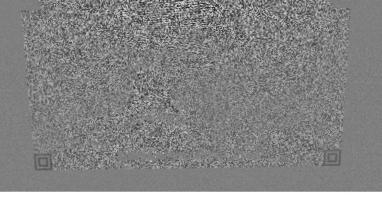
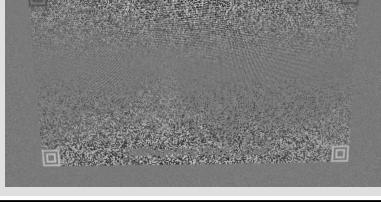
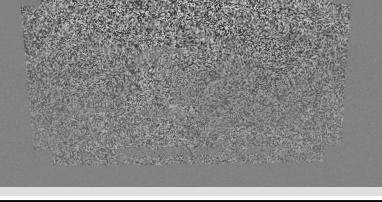
- total Schwarz-Weiß-Schwarz-Weiß-Schwarz Ordnung.
- halb Schwarz-Weiß-Schwarz-Weiß-Schwarz Ordnung, halb nicht gezeigt.
- total Weiß-Schwarz-Weiß-Schwarz-Weiß Ordnung.
- halb Weiß-Schwarz-Weiß-Schwarz-Weiß Ordnung, halb nicht gezeigt.
- halb Schwarz-Weiß-Schwarz-Weiß-Schwarz Ordnung, halb Weiß-Schwarz-Weiß-Schwarz-Weiß Ordnung.
- total nicht gezeigt.

Tabelle 3.2 zeigt eine solche Situationen und listet auf, ob sie zur folgenden Detektion angepasst sind.

Die direkte Verwendung dieser Differenzbilder ist ein kniffliges Problem für die nächste Detektion. Um dieses Problem zu lösen, wurde ein Algorithmus zur Optimierung der Differenzbilder entwickelt.

Die Struktur eines QR Musters ist in Formel 3.29 aufgeführt. Die äußerste “1” Schicht ist hier ein Trennmuster und der Zentralbereich repräsentiert das QR Muster. Aufgrund der Modulationseigenschaften des DaVid Systems wird nur die Pixelwerte der Punkte mit dem Element “1” nach der Modulation verändert. Im Vergleich dazu werden die Pixelwerte der Punkte mit dem Element “0” nur gering verändert. Deswegen wird durch eine Absolutwertoperation die QR Muster als Schwarz-Weiß-Schwarz-Weiß-Schwarz Ordnung dargestellt. Dies ist die erwartete Modellstruktur die im nächsten Schritt operiert wird. Ein detailliertes Operieren wird im Abschnitt “*QRMusterDetektion*” besprochen.

Tabelle 3.2: Die mögliche Differenzbilder.

Ob angepasst zu folgender Detektion?	Differenzbild	Ob angepasst zu folgender Detektion?	Differenzbild
Ja		Nein	
Nein		Nein	
Nein		Nein	

$$QR_{base} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad (3.29)$$

Als nächstes wird der Begriff “Energie” vorgestellt. Auf numerischer Ebene beschreibt dieser den Mittelwert des quadrierten Pixels des Bildes. Es ist bekannt, dass aufgrund der Subtraktion, die Pixelwerte in das Gebiet, welches rund um den Modulationsbereich liegt, gegeneinander aufgehoben wird. Deswegen werden die Pixelwerte in diesem Gebiet nur von Rauschen beeinflusst und sehr gering werden. Im Vergleich dazu betragen die Pixelwerte im Modulationsbereich einen relativ großen Wert $\pm 2A$, hier beutet A die Modulationsamplitude des

Systems. Der Einfluss von Zeitsynchronisation gleicht die Pixelwerte in einigen teilen des Modulationsbereichs aufgrund von Synchronisation aus, und zwar durch aufheben der "Energie". Die Größe der "Energie" hängt hauptsächlich von den Pixelwerten im Modulationsbereich ab. Je größer die Pixelwerte sind, desto höher die "Energie" des Bildes, und desto klarer kann das QR Muster werden. Gemäß dieser Regel wird die "Energie" jedes Differenzbildes berechnet und in absteigender Reihenfolge angeordnet. Um die folgende Detektion zu vereinfachen, werden die ersten paar Bilder hinzugefügt um zu detektierende Bilder zu erhalten. Aus dem Experiment hat bewiesen, dass es ausreicht die ersten drei Bilder zu nehmen.

Die Formel der "Energie" wird in Gleichung 3.30 dargestellt. Hier repräsentiert m, n die Anzahl der Zeilen und Spalten der Matrix. $diff(i, j)$ ist der Pixelwert des Punkts in der m -ten Zeile und n -ten Spalte der Matrix.

$$Energie = \frac{1}{m \times n} \sum_{i=1}^m \sum_{j=1}^n diff^2(i, j) \quad (3.30)$$

Es folgen die detaillierten Schritte dieses Algorithmus:

1. Wähle zwei beliebige Bilder aus und subtrahiere diese um das Differenzbild zu erhalten.
2. Mit einer Absolutoperation für jedes Differenzbild die Energie berechnen.
3. Sortiere in absteigender Reihenfolge und nehme die erste 3 Differenzbilder.
4. Addiere diese 3 Differenzbilder, um ein zu detektierendes Bild zu erhalten.

Dazu, das passende Flussdiagramm in Abbildung 3.20.

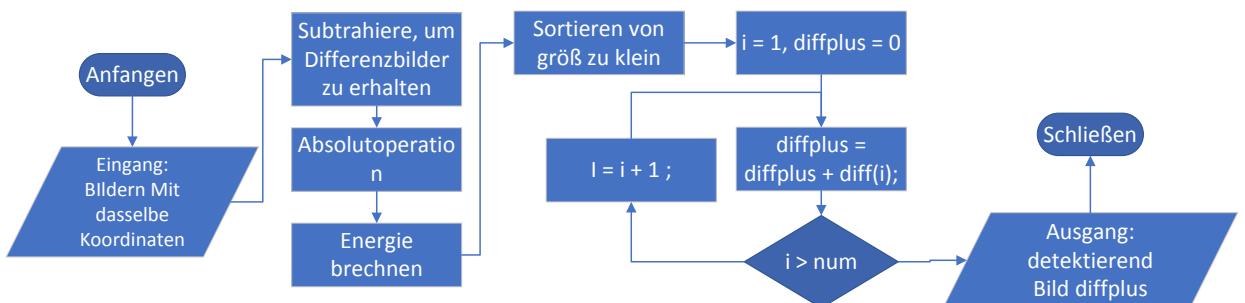


Abbildung 3.20: Differenzbild Flussdiagramm.

Ein Beispiel für ein zu detektierendes Bild wird in Abbildung 3.21 vorgeführt. Aus der Abbildung ist ersichtlich, dass der Modulationsbereich besser dargestellt ist und an den vier Ecken des

Bereichs ein eindeutiges QR Muster vorhanden ist. Als nächstes wird die Bildverarbeitung eingeführt.



Abbildung 3.21: Ein zu detektierendes Bild.

3.4 Bildverarbeitung

Durch die Differenzbild-Optimierung, wird ein zu detektierendes Bild erhalten. Abbildung 3.21 ist noch ein Graustufenbild und muss noch einiger Bildverarbeitung unterzogen werden. Dadurch können kleine Punkte und Lücken, die durch Rauschen und Fehler verursacht werden, entfernt werden um die nachfolgende Detektion zu erleichtern. Der detaillierte Inhalt der Bildverarbeitung wurde in der anderen Methode aufgeführt, hier ist eine kurze Beschreibung.

Bild Binarisierung

Für die Schwellenwertbildung und das Erstellen eines Binärbildes wurde eine Funktion namens “*imbinarize*” in Matlab verwendet. Dieser Funktion wird das Bild übergeben ,damit es einen anpassungsfähigen Schwellwert generiert um das Schwarz-Weiß-Bild zu erstellen. Das ist genug für QR Muster Detektion, weil es nur schwarze und weiße Module enthält, die binär 1 bzw. 0 sind.

Morphologie

Mit öffnenden und schließenden Filtern können die Lücken zwischen Blöcken und die Punkte vom Rauschen stark reduziert werden, was das binär Bild zu einer guten Interpretation des QR Musters macht.

3.5 QR Musters Detektion

Nach der Bildverarbeitung wird nun die QR Muster Detektion ausgeführt. Das Ziel einer QR Muster Detektion ist, das Zentrum des Musters im Bild zu lokalisieren um dadurch das Bild zu rekonstruieren. Abbildung 3.22 zeigt eine geometrische Struktur des QR Musters. An jeder Ecke des Modularisationsbereichs gibt es ein solches QR Muster.

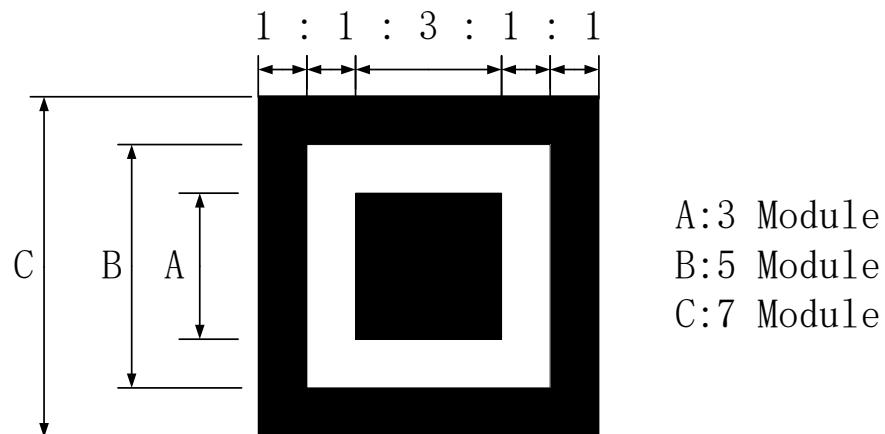


Abbildung 3.22: QR Muster.

Aus der geometrischen Sicht kann jedes Muster als drei konzentrische Quadrate betrachtet werden und besteht aus einem schwarzen (dunklen) 7×7 Modul, einem weißen (hellen) 5×5 Modul und schließlich einem dunklen 3×3 Modul. Von Kenntnissen der Geometrie ist bekannt, dass in jeder Richtung das Breiteverhältnis der alternativen Schwarz- und Weißmodul in einem Muster die Beziehung $1 : 1 : 3 : 1 : 1$ existiert, wie es in Abbildung 3.23 vorgeführt ist. Diese wichtige Eigenschaft hilft uns, die Positionen der QR Muster zu finden. In der Praxis gibt es rund um die Muster noch ein Trennmuster, das ein Funktionsmuster aller weißen (hellen) Module(ein Modul breit). Es dient als eine Grenze zwischen den Mustern und dem Datenbereich, um die Verwechslung zwischen Muster und Daten zu vermeiden.

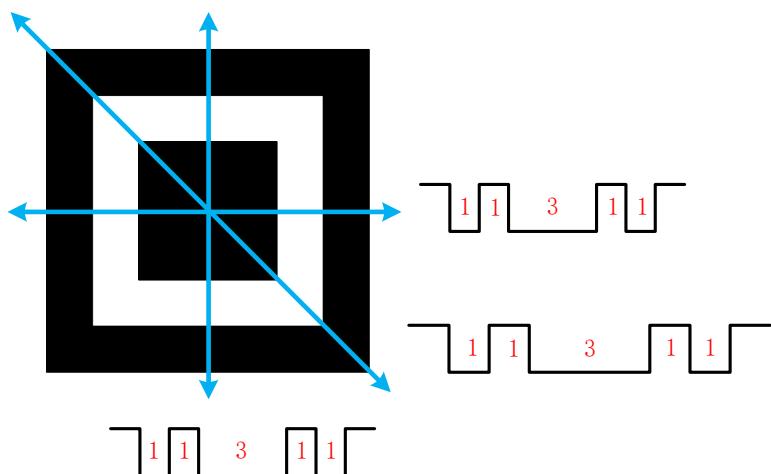


Abbildung 3.23: QR Muster Ratio.

Als nächstes werden die detaillierten Schritte der Detektion aufgeführt welche mit der Funktion „*detectFIP*“ in Matlab implementiert wurde.

Schritt 1:

Zuerst wird der Berechnungsaufwand der Analyse des ganzen Bilds geschätzt, und anschließend wird der Modulationsbereich in kleine Bereiche unterteilt, die QR Muster enthalten könnten.

Schritt 2:

Scannen jeder Zeile dieses kleinen Bereiches und Speicherung der Länge der Schwarz und Weiß Module in einen fünf Elementen Vektor. Die Länge der Module bedeutet die Anzahl aufeinanderfolgender Pixel in einer Zeile mit der gleiche Farbe. Speicherreihenfolge in diesem Vektor ist Schwarz-Weiß-Schwarz-Weiß-Schwarz. Es sollte hier angemerkt werden, dass das erste Element des Vektors die Anzahl der schwarzen Module enthält.

Schritt 3:

Immer wenn das fünfte Element des Vektors gezählt wird, werden die Werte ausgewertet, ob das Muster nahe genug an den $1 : 1 : 3 : 1 : 1$ Verhältnissen ist. Wenn die Bedingung erfüllt ist, gehen zum weitem Schritt. Ansonsten wird der Vektor um zwei Elemente nach links verschoben und die ersten zwei Elemente des Vektors werden entfernt. Anschließend wiederholen den Operation von Schritt 2.

Schritt 4:

Die Elemente vom Vektor werden verarbeitet, um das ungefähre horizontale Zentrum zu erhalten. Eine Kreuzprüfung wird an diesem Punkt vorgenommen, welche aus den Schritten 2 und 3 besteht, der Unterschied zwischen denen ist, dass der horizontale Scan durch einen vertikalen Scan ersetzt wird. Anschließend wird überprüft, ob ein vertikales Zentrum gefunden wurde. Wenn Ja wird eine Kreuz-Kreuzprüfung mit dem horizontale Scan vorgenommen, um die Ergebnisse zu optimieren. Dies wird hauptsächlich benötigt, um die reale horizontale Mitte des Musters in extremen Schräglage Fällen zu lokalisieren. Nach Speicherung des potenziellen Zentrums, werden die Elemente des Vektors geleert, um wieder im zweiten Schritt einen Scan zu machen. Ansonsten wird der Vektor um zwei nach links verschoben und die ersten beiden Elemente des Vektors weggeworfen. In Schritt 2, wird wiederum erneut begonnen zu scannen und zu zählen.

Schritt 5:

Die Ausgabe des vorherigen Schritts wird verarbeitet, falls mehrere potentielle Muster gefunden wurden, wird “*SelectBestPattern*” verwendet, um das Beste Auszuwählen. Es sollte angemerkt werden, dass wenn die möglichen Muster nach dem Ende der Erkundung nicht gefunden wurden, ein spezielles Signal zurückgegeben wird und das System zur Schritt Differenzbild Optimierung zurückkehren wird. Die Operation besteht darin dem ursprünglichen Bild, welches drei Differenzbildern besteht, ein weiteres Differenzbild hinzuzufügen.

Schritt 6:

Durch die gefunden Muster Zentren, wird eine projektive Transformation vorgenommen, um die Ecken des Bildes zu bestimmen.

Die folgende Abbildung 3.24 zeigt Flussdiagramm einer Detektion für QR Muster.

Ein Beispiel für QR Muster Detektion wird in Abbildung 3.25 gezeigt:

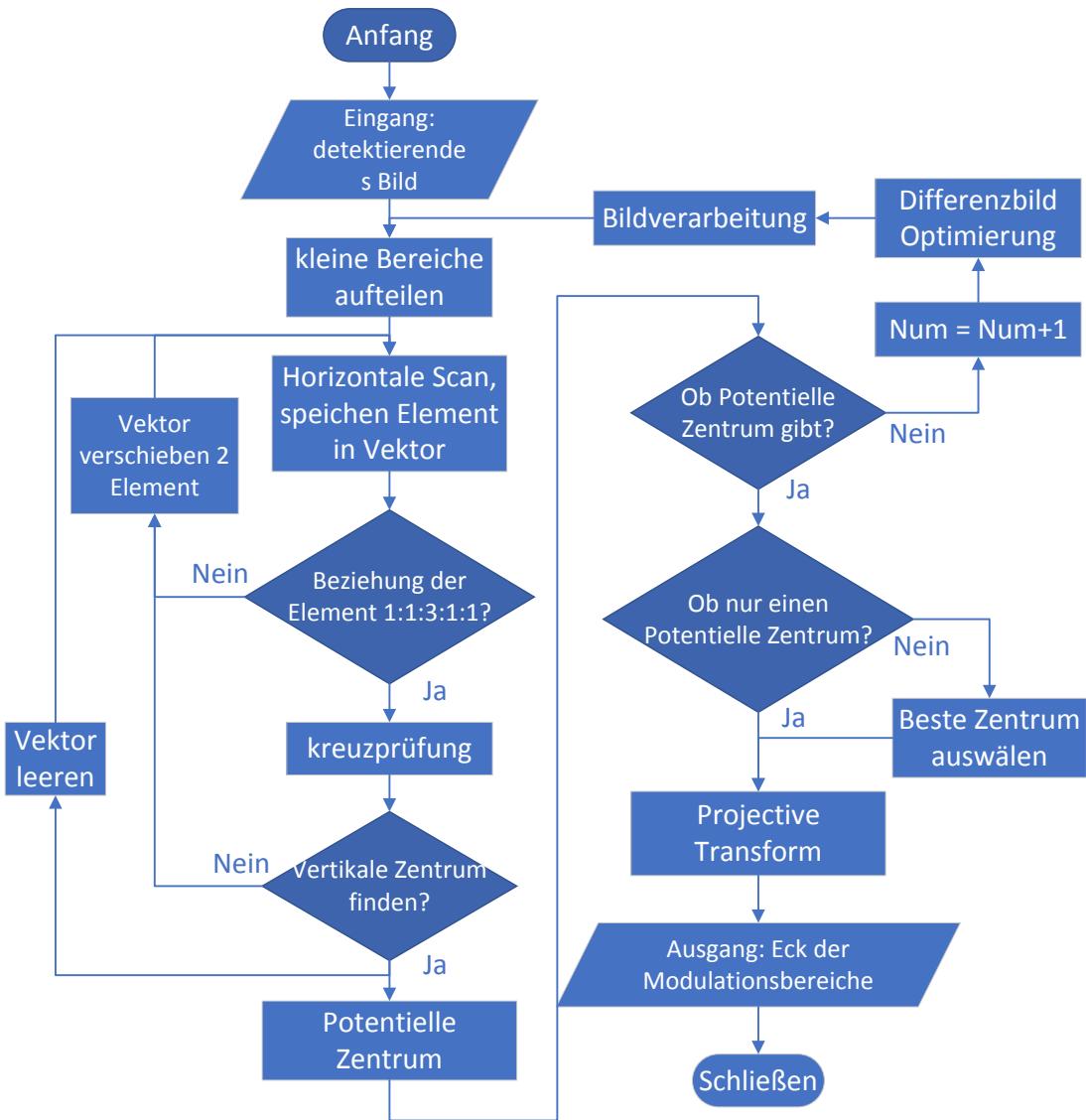


Abbildung 3.24: Flussdiagramm der QR Muster Detektion.

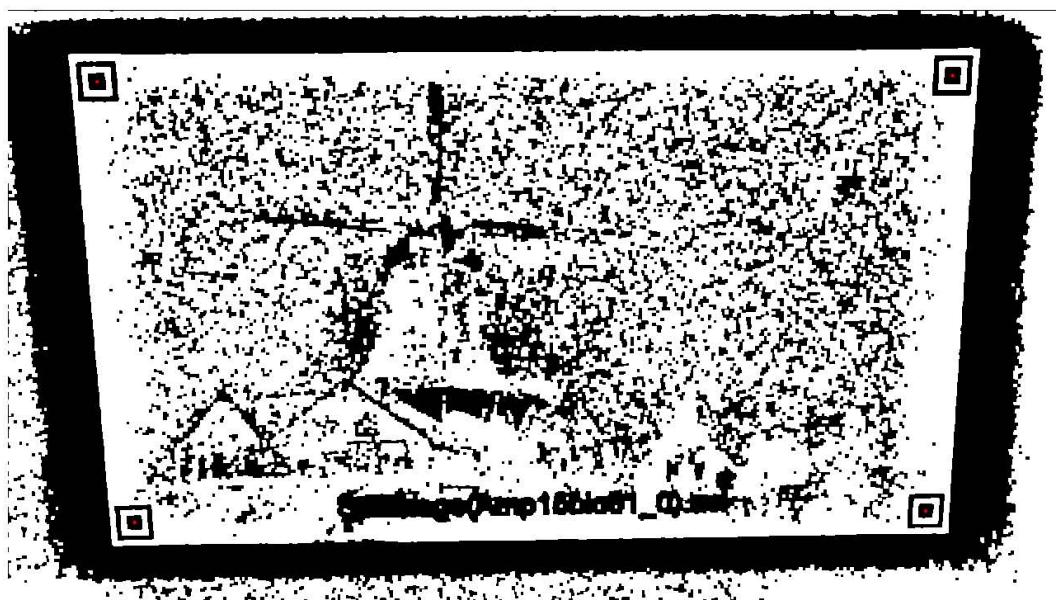


Abbildung 3.25: Beispiel einer QR Muster Detektion.

4 Zweite Methode

In diesem Kapitel wird die Realisierung des zweiten Verfahrens eingegangen. Zuerst wird die allgemeine Struktur dieser Methode beschrieben. Dies liefert ein grobes Verständnis der Struktur und des Arbeitsprinzips der gesamten Methode. Anschließend läuft eine Erklärung der Bildverarbeitungen im Detail. Diese besteht aus folgenden Teilen: Binarisierung, Morphologie Operation und Canny Kannten Detektion. In diesem Abschnitt wird erläutert, wie diese Operationen funktionieren und was der Sinn dieser Behandlungen ist. Schließlich wird eine Linienextrahierungsmethode(*Hough und Radon Transformation*) detailliert vorstellen.

4.1 Allgemeine Struktur

Anders als bei der vorherigen Methode, indem durch die Charakteristiken des QR Musters die Modulationsbereiche detektiert wurden, wird hier eine neue Strategie mit Hilfe geometrischen Eigenschaften des Modulationsbereichs verwendet. Es ist bekannt, dass im DaVid System durch Aufnahme des Inhalts auf dem Bildschirm die versteckten Daten erkannt werden können. Dann entspricht der Modulationsbereich tatsächlich dem Bildschirmbereich, der im Allgemeinen über die Form von einem Rechteck verfügt. Deswegen kann hier das Problem eines Detektierens des Modulationsbereichs in ein Problem des Detektierens eines Rechtecks umgewandelt werden. Außerdem, wegen der geometrischen Eingenschaften des Rechtecks, welches aus 2 paaren paralleler Linien besteht, lassen das Problem zu ein Linienextrahierungsproblem. Das heißt, diese Methode basiert auf dem extrahieren der Linien auf dem Bild, um den endgültigen Modulationsbereich zu erkennen und zu bestimmen. Abbildung 3.1 zeigt das Strukturdiagramm dieses Verfahrens.



Abbildung 4.1: Strukturdiagramm.

Das Objekt, welches mit dieser Methode bearbeitet wird, ist eine Reihe von Bildern, die von einer Handkamera mit Stativ aufgenommen wurde. Durch das Stativ können diese Bilder in dasselbe Koordinatensystem sein. Mit gegenseitiger Subtraktion wird eine Reihe Differenzbilder erhalten. Durch das Wissen des vorherigen Kapitels, wird die „*Energie*“ hauptsächlich aus dem Modulationsbereich gewonnen. Durch Addition der Differenzbild mit der meisten „*Energie*“ kann ein zu detektierendes Bild erhalten werden um die folgende Operation durchzuführen. Diese Teilbereiche sind schon im vorherigen Kapitel beschrieben und werden hier nicht ins Detail erklärt. Um den Modulationsbereich vom Bild hervorzuheben, wird das zu detektierendes Bild binarisiert, dadurch wird das Bild in zwei Teile geteilt, ein für das Objekt (hier der Modulationsbereich) und ein für den Hintergrund (hier der Bereich um den Modulationsbereich). Danach wird zur Beseitigung der zahlreichen Unvollkommenheiten (hier die kleinen Punkte und Lücken), die von Rauschen und Fehlern verursacht werden, wird hier eine morphologische Behandlung benötigt, welche auf der Form und Struktur des Objekts basiert. Als Nächstes soll der Modulationsbereich erkannt werden. Um diese zu implementieren, muss die rechteckige Grenze zwischen dem Modulationsbereich und dem Hintergrundbereich gefunden werden. Mit Hilfe einer Kantenextraktionsmethode, also dem Canny-Algorithmus, werden im Bild nur die Kanten übernommen, die der Grenze des Modulationsbereichs entsprechen. Die Grenze des Modulationsbereichs hat eine rechteckige Form, bestehend aus vier Linien. Das Hough- und Radon-Transformationsverfahren wird verwendet, um alle zwei geraden Linien in der horizontalen Richtung bzw. der vertikalen Richtung zu erfassen. Schließlich wird das endgültige Rechteck, also der Modulationsbereich, durch die vier geraden Linien bestimmt. Das Flussdiagramm wird in Abbildung 4.2 gezeigt. Die Details jedes Teils werden in den folgenden Abschnitten beschrieben.

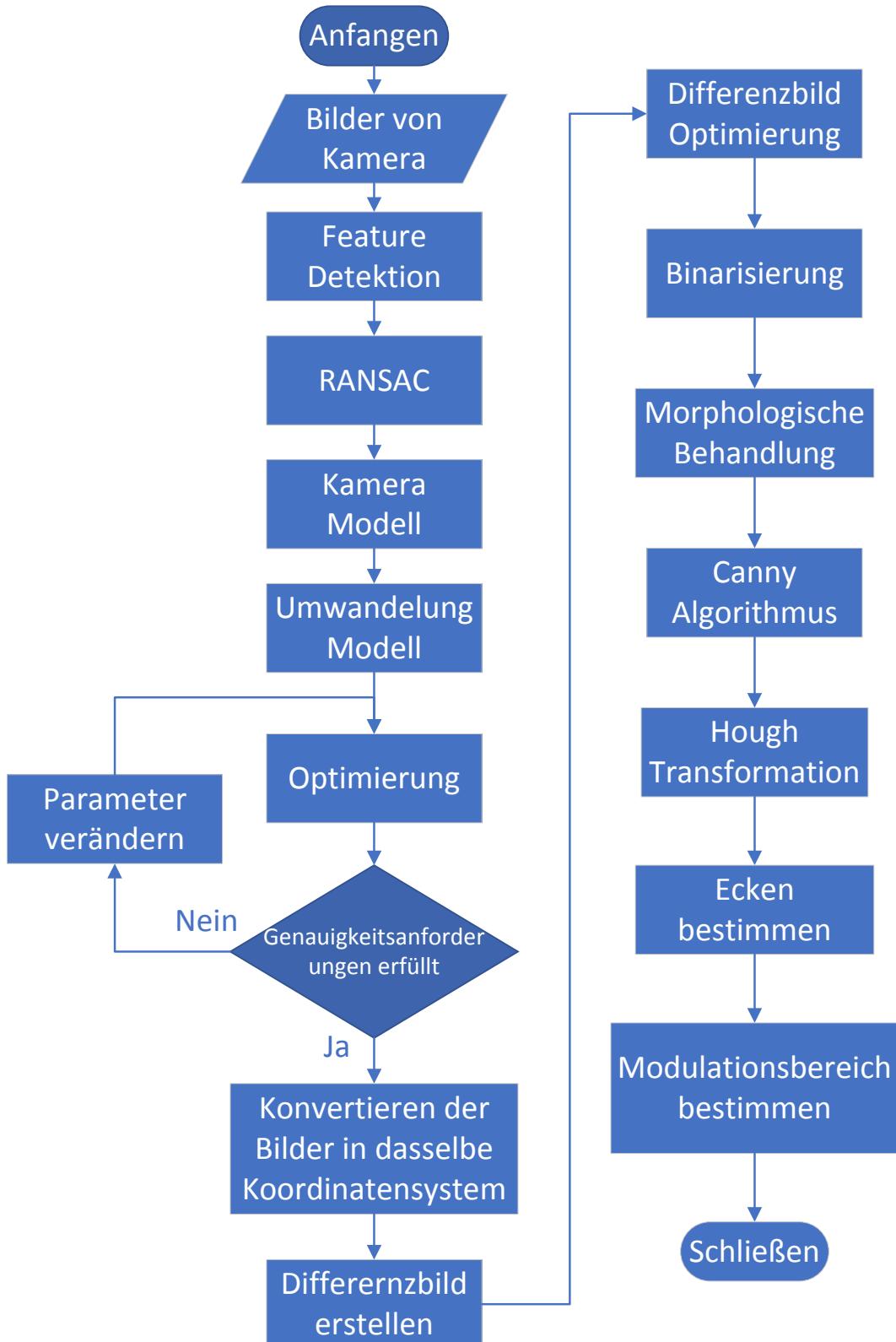


Abbildung 4.2: Flussdiagramm der zweite Methode.

4.2 Binarisierung

In diesem Abschnitt wird der Binarisierungsvorgang der Bildverarbeitung beschrieben. Um den Modulationsbereich zu detektieren, muss dieser zuerst vom Bild hervorgehoben werden. Anschließend kann mit einer Binarisierungs Operation dieser Vorgang implementiert werden, indem das Bild in zwei Teile geteilt wird, ein Teil mit dem Pixelwert 1 für den Modulationsbereich und der andere Teil mit dem Pixelwert 0 für den umgebenden Hintergrund. Schließlich kann der Modulationsbereich mit einigen weiteren Operationen wie morphologische Behandlung, Kanten Detektion behandelt werden.

In der digitalen Bildverarbeitung nehmen binäre Bilder eine ausschlaggebende Position ein. Viele Anwendungen der digitalen Bildverarbeitung können als binäre Probleme betrachtet werden. Um binäre Bilder zu verarbeiten und zu analysieren, wird zunächst das Graustufenbild digitalisieren, um ein binäres Bild zu erhalten. Wenn das Bild weiterverarbeitet wird, bezieht sich die Sammeleigenschaft des Bildes nur auf die Position des Pixels, dessen Pixelwert 0 oder 1 ist, sodass der mehrstufige Wert des Pixels nicht länger vertreten ist um die Verarbeitung zu vereinfachen. Darüber hinaus ist die Menge an Datenverarbeitung und Komprimierung gering.

Um ein ideales Binärbild zu erhalten, werden im Allgemeinen geschlossene, zusammenhängende Grenzen verwendet, um Bereiche zu definieren die sich nicht überlappen. Hier ein einfaches Beispiel zur Veranschaulichung des Binarisierungsvorgang gegeben. In einem bimodalan Histogramm eines Bildes, wie in Abbildung 4.3, gibt es zwei scharfe Wellenberge, einen für das Objekt und einen für den Hintergrund. Die Binarisierungsschwelle T wird genau an die tiefste Stelle zwischen den beiden Wellenbergen gelegt.

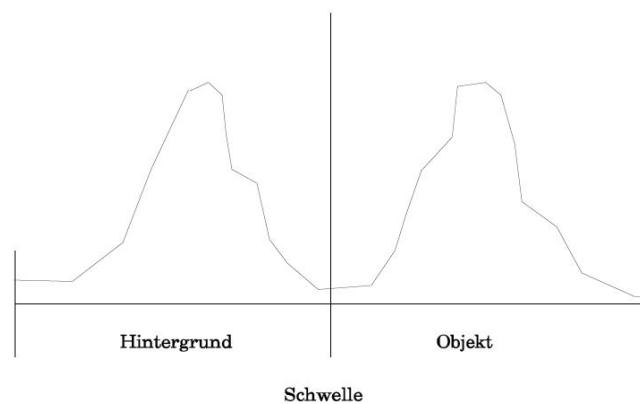


Abbildung 4.3: Histogramm eines einfachen Beispiels.

Alle Pixel, deren Graustufe größer oder gleich dem Schwellenwert T ist, werden zu einem bestimmten Objekt zugeordnet und deren Pixelwert auf 1 gesetzt. Andernfalls werden diese

Pixel aus dem Objektbereich ausgeschlossen und deren Pixelwert auf 0 gesetzt, was den Hintergrund bzw. den Ausnahmeobjektbereich anzeigt. Die Formel dazu lautet:

$$y(x) = \begin{cases} 0, & \text{für } x < T \\ 1, & \text{sonst} \end{cases} \quad (4.1)$$

In der Praxis ist das Histogramm aufgrund der Helligkeit und Inhalt des Bildes nicht der obige bimodale Fall. Dies erfordert auch, dass die geeignete Binarisierungsmethode entsprechend den tatsächlichen Bedürfnissen im Binarisierungsprozess auswählt wird. Als folgende werden einige in dieser Arbeit versuchte Binarisierungsmethoden vorgestellt.

Grundlegende globale Schwelle Methode

Die grundlegende globale Schwellenmethode ist eine Erweiterung der festen Schwellenmethode, die vorher beschrieben wurde. Wenn die Histogrammspitzen und -täler des Bildes offensichtlich sind und Doppelpeaks aufweisen, ist diese Methode effektiver. Basiert auf der visuellen Überprüfung des Histogramms und der Erhaltung des Schwellenwerts durch eine iterative Methode. Die grundlegende Algorithmus ist wie folgt:

1. Wahl eines Parameters t und einen anfänglichen Schwellenwert T_0 , wobei der Durchschnitt der maximalen Grauwerte I_{max} und minimalen Grauwerte I_{min} verwendet wird.

$$T_0 = (I_{max} + I_{min})/2 \quad (4.2)$$

2. Segmentieren des Bildes mit dem Schwellenwert T_0 . Dadurch besteht das Bild aus zwei Teilen: G_1 besteht aus den Pixeln mit dem Grauwert größer als T_0 und G_2 aus denen, mit dem Grauwert kleiner oder gleich T_0 .

3. Berechnen des durchschnittlichen Grauwerts aller Pixel in u_1 und u_2 und den neuen Schwellenwert T_1 .

$$T_1 = (u_1 + u_2)/2 \quad (4.3)$$

4. Falls $|T_0 - T_1| < t$, dann ist T_1 der optimale Schwellenwert. Andernfalls wird T_1 zu T_0 zugewiesen und die Schritte 2–4 werden wiederholt, bis der optimale Schwellenwert gefunden ist.

Abbildung 4.6 zeigt ein Binarisierungs Beispiel mit globalen Schwellenwertmethode.



Abbildung 4.4: Binarisierung mit globalen Schwellenwertmethode.

Grundlegende lokale Schwellenwert Methode

Ein Bildgebungsfaktor ungleichmäßiger Helligkeit bewirkt, dass ein Histogramm, welches ansonsten für eine effiziente Segmentierung geeignet wäre, ein Histogramm wird, das nicht effektiv mit einem einzigen globalen Schwellenwert segmentiert werden kann.

Das Verfahren zur Verarbeitung besteht darin, das Bild weiter in Unterbilder zu unterteilen, um verschiedene Unterbilder mit unterschiedlichen Schwellenwerten zu segmentieren. Diese Methode wird als grundlegende adaptive Schwellenwert-Binarisierungsmethode bezeichnet. Das Hauptproblem bei diesem Ansatz besteht darin, das Bild zu unterteilen und den Schwellenwert für das resultierende Teilbild abzuschätzen. Da die Schwelle für jedes Pixel von dem Pixel in der Untergruppe abhängt ist die Position im Bild, also solche Schwellen, adaptiv.

Das Verfahren zum Unterteilen von Unterbildern wird für das Bild übernommen. Hier werden drei Arten von Unterteilungsverfahren ausgewählt, Unterbilder einer Größe von 32×32 , 4×4 , 16×16 Pixeln werden jeweils geteilt und die durchschnittliche Graustufe der Unterbilder wird als ein Schwellenwert für die Binarisierung ausgewählt. Das Ergebnis wird in Abbildung 4.5 gezeigt.

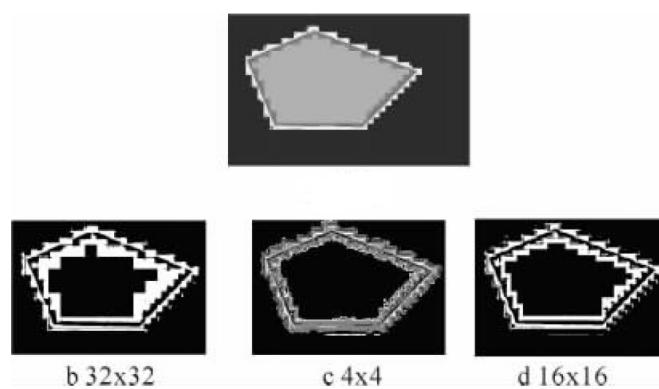


Abbildung 4.5: Binarisierung mit lokalen Schwellenwertmethode.

Otsu adaptive Schwellenmethode

Otsu[8], auch bekannt als die maximale Interklassenvarianz, wurde 1979 vom japanischen Gelehrten Otsu vorgeschlagen und ist eine adaptive Schwellenwertbestimmungsmethode, die auch als Otsu bekannt ist.

Die Grundidee des Otsu-Algorithmus ist, das Histogramm des Bildes, gemäß der Varianz zwischen dem Vordergrund und dem Hintergrund zu verwenden, um den optimalen Schwellenwert dynamisch zu bestimmen. Die Anzahl der Pixeln in einem Bild ist N und die Graustufe $L(0, 1, \dots, L - 1)$. Die Anzahl der Pixel mit dem Grauwert i ist n_i , dadurch ist die Wahrscheinlichkeit von i $p_i = \frac{n_i}{N}$. Für das Bild stellt das T Segmentierungsschwellenwert zwischen Vordergrund und Hintergrund dar. Der Vordergrund entspricht den Grauwerten von 0 zu $T - 1$, dagegen der Hintergrund den Grauwerten von T zu $L - 1$.

Die Wahrscheinlichkeit des Vordergrundgebiets W_0 und des durchschnittlichen Grauwerts U_0 sind

$$w_0 = \sum_{i=0}^{T-1} p_i, \quad u_0 = \sum_{i=0}^{T-1} ip_i / w_0 \quad (4.4)$$

Die Wahrscheinlichkeit der Hintergrundpunkte W_1 und des durchschnittlichen Grauwerts U_1 sind

$$w_1 = \sum_{i=T}^{L-1} p_i = 1 - w_0, \quad u_1 = \sum_{i=T}^{L-1} ip_i / w_1 \quad (4.5)$$

Dann ist der gesamte durchschnittliche Grauwert des Bildes

$$u = w_0 u_0 + w_1 u_1 \quad (4.6)$$

Die Infra-Klassen-Varianz ist definiert als

$$\sigma^2 = w_0(u_0 - u)^2 + w_1(u_1 - u)^2 = w_0 w_1 (u_0 - u_1)^2 \quad (4.7)$$

Wenn der Schwellenwert T sich zwischen 0 und $L - 1$ befindet und σ^2 das Maximum ist, ist das entsprechende T der optimale Schwellenwert. Beispielsweise wird in Abbildung 4.6 gezeigt.



Abbildung 4.6: Binarisierung mit Ostu.

Die Ostu-Methode verwendet Grauerthistogramme zur Bestimmung des Schwellenwerts. Es ist eine automatische nicht-parametrische Schwellenauswahlmethode. Diese Methode ist einfach durchzuführen, wird nicht von der Kontrast- und Helligkeitsänderung unter bestimmten Bedingungen beeinflusst und kann das Objekt zufriedenstellend vom Hintergrundbereich trennen.

Abbildung 4.7 zeigt Binärbild nach einer Binariesierung Operation mit Ostu.



Abbildung 4.7: Binärbild mit Ostu.

4.3 Morphologie

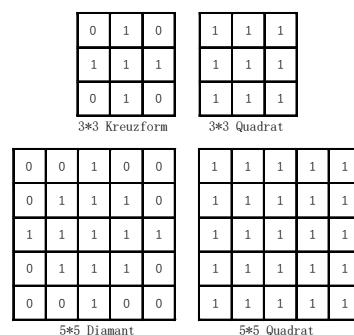
Nach der Binarisierung kann das Bild über zahlreiche Unvollkommenheiten verfügen. Insbesondere ist das Binärbild, welches durch einfache Schwellenwerte generiert wurde, durch Rauschen und Fehlern verzerrt. Die morphologische Bildverarbeitung verfolgt das Ziel, diese Unvollkommenheiten zu beseitigen indem Form und Struktur des Objekts berücksichtigt werden.

Morphologische Operationen arbeiten auf der Grundlage der Mengenoperation und hängen von der relativen Ordnung des Pixelwerts ab. Diese Eigenschaft macht es besonders geeignet für die Verarbeitung von Binärbildern. Natürlich ist die Mengenoperation gültig für alle Graustufenbilder. Hier in dieser Arbeit werden nur binarisierte Bilder benutzt. Es erstellt ein neues Binärbild, bei dem das Pixel nur dann einen nicht-Null Wert hat, wenn die Operation an dieser Position im Eingabebild erfolgreich war.

Die Eingabedaten für die mathematischen morphologischen Operationen sind zwei Bilder: das zu bearbeitende Bild A und ein Strukturelement B. B ist normalerweise eine kleine Pixelmatrix mit jeweils einem Wert von Null oder Eins. Einige Eigenschaften des Strukturelements werden wie folgt gewählt.

- Die Matrixdimensionen geben die Größe des Strukturelements an.
- Das Muster, das aus Einsen und Nullen besteht, gibt die Form des Strukturelements an.
- Ein Ursprung des Strukturelements ist üblicherweise ein Pixel innerhalb des Strukturelements, es verlassen auch außerhalb des Strukturelements liegen.

Eine übliche Anwendung besteht darin, dass der Ursprung ungerader Dimensionen der Pixelmatrixstellen und den Ursprung als das Zentrum der Matrix definiert werden. Einige grundlegende Strukturelemente sind wie folgt:



0	1	0
1	1	1
0	1	0

3*3 Kreuzform

1	1	1
1	1	1
1	1	1

3*3 Quadrat

0	0	1	0	0
0	1	1	1	0
1	1	1	1	1
0	1	1	1	0
0	0	1	0	0

5*5 Diamant

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

5*5 Quadrat

Abbildung 4.8: Einige grundlegende Strukturelemente.

Zuerst werden die zwei grundlegenden morphologischen Operatoren, Erosion und Dilatation, beschrieben. Anschließend werden die darauf basierenden Operationen, die als Öffnung und Schließung bekannt sind, vorgestellt.

Dilatation

Die Dilatation Operation bewirkt, dass das Objekt nach groß wächst. Das Wachstum hängt von der Art und Form des Strukturelements ab. Die Formel einer Dilatation wird dargestellt als:

$$A \oplus B = \{z \mid (\widehat{B})_z \cap A \neq \emptyset\} \quad (4.8)$$

Hier ist $(\widehat{B})_z$ das Strukturelement B, welcher seinen Ursprung reflektiert und um z verschoben ist. Die entsprechende Dilatation für das Bild A mit B ist die Menge aller Verschiebungen z, wo \widehat{B} und A mindestens ein gemeinsames Element haben. Das Ergebnis der Dilatation ist dazu da, um Pixel um die Grenze des Objekts hinzuzufügen. Außerdem wird die Dilatation Operation im wesentlichen verwendet, um die Löcher (fehlende Pixel) in einem soliden Objekt zu füllen. Dieser beeinflusst die Intensität an einem Bereich und kann als ein Unschärfe Effekt bezeichnet werden, nämlich ein räumlicher Tiefpassfilter der beim linearen Filtern des Bildes verwendet wird. Abbildung 4.9 zeigt eine typische Dilatation Operation.

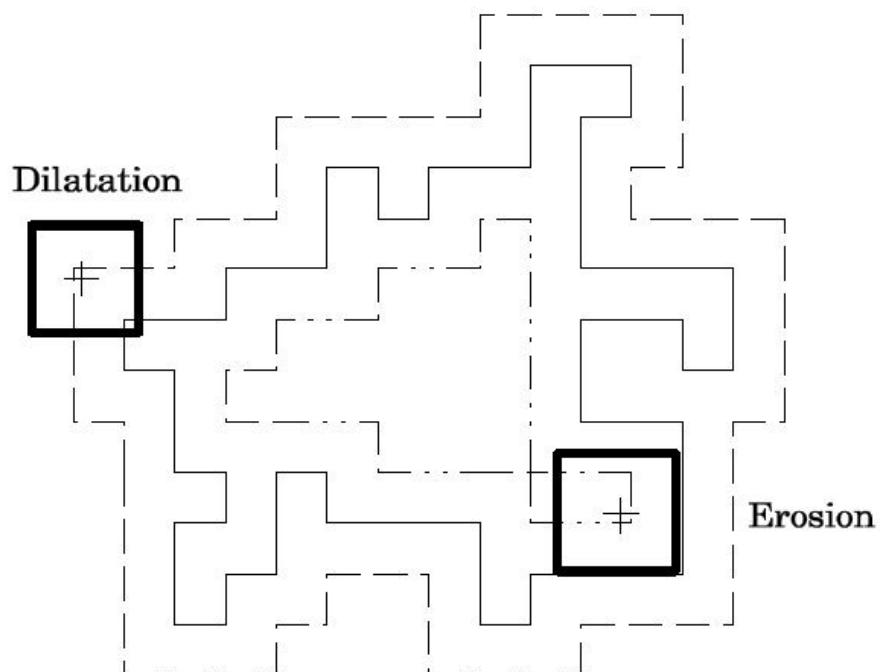


Abbildung 4.9: Dilatation und Erosion.

Erosion

Der Operationseffekt einer Erosion ist genau das Gegenteil von dem einer Dilatation. Die Erosion Operation bewirkt, dass das Objekt nach klein werden. Die Erosion eines Bildes A ist durch das Strukturelement B definiert als

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \quad (4.9)$$

Hier ist die Erosion des Bildes A die Menge aller Verschiebungen z, wo das Strukturelement B durch Verschiebungen, zu Teilmenge des Bildes A gehört. Diese Operation führt zu einem Verlust von Grenzpixeln des Objekts.

Die Erosion Operation entfernt solche Strukturen, die eine kleinere Größe als das strukturierende Element haben. Zudem kann es verwendet werden, um die verrauschte „Verbindung“ zwischen zwei Objekten zu entfernen. Da die unerwünschten Pixel entfernen werden, entspricht der Effekt einem Schärfen des Objekts. Abbildung 4.9 zeigt eine typische Erosion Operation.

Öffnung

Die Öffnung Operation eines Bildes ist eine Kombination aus Erosion und Dilatation, d.h. auf eine Erosion Operation, folgt eine Dilatation Operation. Praktisch werden Bild A durch beide Operationen mit dem gleichen Strukturelement B ausgeführt. Die Formel einer Öffnung Operation ist definiert als

$$A \circ B = (A \ominus B) \oplus B \quad (4.10)$$

Die Grenzen des geöffneten Objekts sind die Punkte, wo das Strukturelement B die äußersten Punkte der Grenze vom Objekt erreicht, während B innerhalb dieser Grenze entlangfahren. Feine strukturierte Details, kleiner als das Strukturelement, werden demnach bei der Öffnung Operation entfernt und dünne Verbindungen zwischen größeren Teilen aufgelöst. Eine Öffnung Operation wird in Abbildung 4.10 gezeigt.

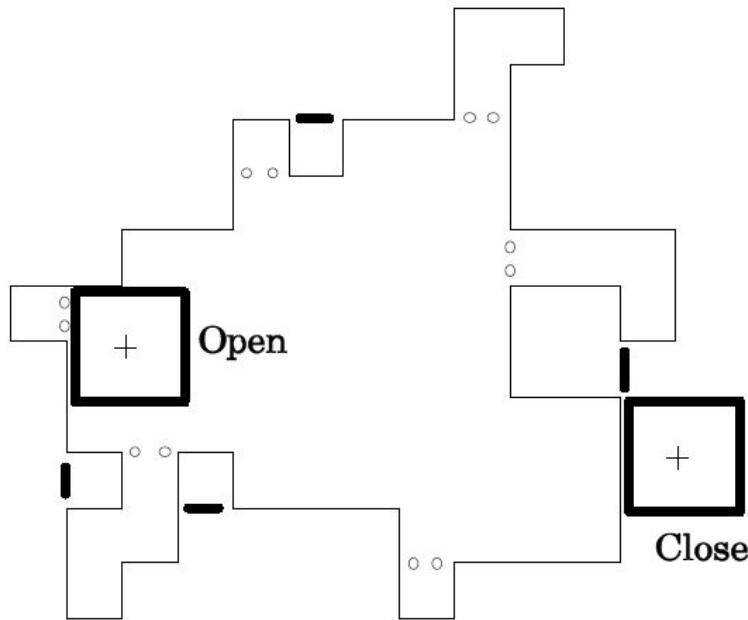


Abbildung 4.10: Öffnung und Schließung.

Schließung

Genau wie die Öffnung Operation ist die Schließung Operation auch eine Kombination aus Erosion und Dilatation. Der Unterschied dazwischen liegt in der Reihenfolge der Operationen, d.h. zuerst wird eine Dilatation Operation, danach eine Erosion Operation mit dem gleichen Strukturelement ausgeführt. Die Schließung eines Bildes A durch das Strukturelement B ist definiert als

$$A \bullet B = (A \oplus B) \ominus B \quad (4.11)$$

Die Grenze des geschlossenen Objekts sind die Punkte, wo das Strukturelement B, die äußersten Punkte der Grenze von Objekt erreicht, während B außerhalb dieser Grenze entlangfahren. Kleinere Risse, Lücken und feine Details werden dagegen aufgefüllt und mit den großen Teilen zusammengeschlossen. Abbildung 4.10 zeigt den Vorgang einer Schließung Operation.

Öffnung und Schließung Operation besitzen folgende Eigenschaften:

- Öffnung und Schließung sind idempotent.
- Die Öffnung Operation ist anti-extensiv.

- Die Schließung Operation ist extensiv.
- Öffnung und Schließung sind dual bezüglich der Komplementierung.
- Ein Bild wird als B-Geöffnet bezeichnet, wenn es bei der gleichen Öffnung Operation unverändert bleibt.
- Ein Bild wird als B-Geschlossen bezeichnet, wenn es bei der gleichen Schließung Operation unverändert bleibt.

Das zu detektierendes Bild aus Abbildung 4.7 nach einer morphologischen Operation wird in Abbildung ?? gezeigt.



Abbildung 4.11: Bild nach einer morphologischen Operation.

4.4 Canny detection

Nach der morphologischen Operation erhalten wir ein Binärbild, welches grob in zwei Teile geteilt werden kann. Ein Modulationsbereich mit dem Pixelwert 1 im mittleren Bereich und einen umgebenden Hintergrundbereich mit dem Pixelwert 0 im äußeren Bereich. Um die nächste Linie Detektion zu implementieren, wird eine Kanten Detektion, weil Kanten oft mit den Grenzen von Objekten in einer Szene verknüpft werden, durchgeführt. In anderen Worten, ist es die Grenze zwischen Modulationsbereich und Hintergrundbereich. [9]

Es gibt immer einige gut Kantenextraktionsmethode, wie Sobel, Canny, Laplacian, Prewitt, Roberts uvm. In dieser Arbeit wird die leistungsfähigste Kantenerkennungsmethode bzw. der Canny Algorithmus benutzt. Die Canny-Methode unterscheidet sich von den anderen Kantenerkennungsmethoden darin, dass sie zwei verschiedene Schwellenwerte verwendet (um starke und schwache Kanten zu erkennen) und die schwachen Kanten in der Ausgabe nur dann einschließt, wenn sie mit starken Kanten verbunden sind. Diese Methode kann daher im Vergleich zur anderen Methode weniger Wahrscheinlichkeit verfügen, durch Rauschen getäuscht zu werden, und mehr Wahrscheinlichkeit verfügen, echte schwache Kanten zu erkennen.

Das Ziel vom Canny Algorithmus ist es, einen optimalen Kantenextraktionsalgorithmus zu finden. Drei Kriterien für die optimale Kantendetektion werden vorgeschlagen:

- Gute Erkennung: Der Algorithmus kann so viele tatsächliche Kanten wie möglich im Bild erkennen.
- Gute Positionierung: Identifizieren der Kanten so nah wie möglich an den tatsächlichen Kanten im Bild.
- Minimale Antwort: Kanten in einem Bild können nur einmal identifiziert werden, und mögliches Bildrauschen sollte nicht als Kanten erkannt werden.

Die Implementierung des Canny Algorithmus funktioniert so:

1. **Gaußsche Unschärfe.** Das Hauptziel ist, Rauschen zu entfernen. Da Rauschen Hochfrequenzsignale ähnelt, wird es leicht als eine falsche Kante erkannt. Die Anwendung Gaußscher Unschärfe ist dazu da, um Rauschen zu entfernen und die Erkennung falscher Kanten zu reduzieren. Es sollte beachtet werden, dass ein zu großer Radius einige schwache Kanten nicht erkennen.
2. **Berechnen der Größe und Richtung des Gradienten.** Die Kanten des Bildes können in verschiedene Richtungen zeigen. Daher verwendet der klassische Canny Algorithmus vier Gradientenoperatoren, um die Gradienten in horizontaler, vertikaler und diagonaler Richtung zu berechnen. Jedoch werden im allgemeinen diese vier Gradientenoperatoren nicht verwendet, sondern mit Kantenunterschiedsoperatoren (wie Rober, Prewitt, Sobel) die Differenz G_x und G_y in horizontaler und vertikaler Richtung berechnet. Die Berechnung der Gradientengröße und Richtung funktioniert wie folgt:

$$G = \sqrt{G_x^2 + G_y^2} \quad (4.12)$$

$$\theta = \arctan2(G_y, G_x)$$

Der Gradientenwinkel θ liegt im Bereich Radianten $-\pi$ bis π , anschließend approximiert dieser in vier Richtungen, die horizontale, vertikale und zwei diagonale Richtungen ($0^\circ, 45^\circ, 90^\circ, 135^\circ$) repräsentieren. Dieser kann durch $\pm i\pi/8$ ($i = 1, 3, 5, 7$) geteilt werden, sodass der in jeweils einem Bereich fallende Gradientenwinkel einen spezifischen Wert ergibt, der eine von vier Richtungen repräsentiert. Es folgt ein Beispiel einer Sobel Operator.

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * A \quad \text{and} \quad G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A \quad (4.13)$$

- 3. Nicht-maximale Unterdrückung.** Nicht-maximale Unterdrückung ist eine Kantenverfeinerungsmethode. Die Gradientenkanten, die normalerweise abgeleitet werden, sind nicht einen, sondern mehrere Pixel breit, während Kriterium 3 erfordert, dass die Kante nur eine genaue Punktbreite hat. Nicht-maximale Unterdrückung kann dazu beitragen, den lokalen maximalen Gradienten beizubehalten, während alle anderen Gradientenwerte unterdrückt werden. Dies bedeutet, dass nur die schärfste Position in der Gradientenänderung beibehalten wird. Der Algorithmus funktioniert wie folgt:

- Vergleichen der Gradientenstärke des aktuellen Punkts mit der Gradientenstärke der positiven und negativen Gradientenrichtungspunkte.
- Wenn die Gradientenstärke des aktuellen Punktes im Vergleich zur Gradientenstärke anderer Punkte in der gleichen Richtung am größten ist, wird der Wert auf 1 beibehalten. Ansonsten wird dieser auf 0 gesetzt. Zum Beispiel, wenn die Gradientenrichtung des aktuellen Punktes in die Richtung von 90° direkt darüber zeigt, muss es mit den Pixeln in der vertikalen Richtung direkt darüber und darunter verglichen werden.

Es ist erwähnenswert, dass die positiven und negativen Richtungen keine unterschiedlichen Bedeutungen haben. Zum Beispiel ist die südöstliche Richtung sowie die nordwestliche Richtung als eine Richtung der Diagonalen zu betrachten. Vorher approximieren die Gradientenrichtung horizontal, vertikal und zwei Diagonalen in vier Richtungen, so dass jedes Pixel in einer dieser vier Richtungen entsprechend mit seiner eigenen Gradientenrichtung verglichen wird, um zu bestimmen ob es beibehalten wird.

- 4. Doppelte Schwelle.** Ein allgemeiner Kantenerkennungsalgorithmus welcher einen Schwellenwert verwendet, um kleine Gradientenwerte zu entfernen die durch Rauschen oder Farbänderungen verursacht werden, während große Gradientenwerte beibehalten werden. Der Canny Algorithmus wendet einen doppelten Schwellenwert, d.h. einen hohen

und einen niedrigen Schwellenwert an, um Kantenpixel zu unterscheiden. Wenn der Kantenpixelgradientwert größer als der hohe Schwellenwert ist, wird er als ein starker Kantenpunkt betrachtet. Wenn der Kantengradientenwert kleiner als der hohe Schwellenwert und größer als der niedrige Schwellenwert ist, wird er als ein schwacher Kantenpunkt markiert. Punkte, die unterhalb der niedrigen Schwelle liegen werden unterdrückt.

5. **Hysteresis Grenzenverfolgung.** Bisher könnten die starken Kantenpunkte als echte Kante angesehen werden. Dagegen könnten schwache Kantenpunkte echte Kanten sein, jedoch könnten diese ebenfalls eine Ursache von Rauschen oder Farbänderungen sein. Um genaue Ergebnisse zu erhalten, sollten Kantenpunkte zweiter Art entfernt werden. Im Allgemein werden schwache Kantenpunkte und starke Kantenpunkte die durch reale Kanten verursacht werden, als verbunden betrachtet, während schwache durch Rauschen verursachten Kantenpunkte, keine Verbindung besitzen sind. Der sogenannte Hysteresis Grenzenverfolgung Algorithmus untersucht die angrenzenden Pixel eines schwachen Kantenpunktes, solange ein starker Randpunkt vorhanden ist, wird dieser schwache Randpunkt als echter Kantenpunkt betrachtet und wird dessen Wert als 1 geblieben.

Zur Implementierung dieses Schritts werden alle verbundenen schwachen Kanten untersucht. Wenn ein Punkt einer verbundenen schwachen Kante mit einem starken Kantenpunkt verbunden ist, wird die schwache Kante beibehalten, ansonsten wird diese schwache Kante unterdrücken. Bei der Suche kann der Algorithmus "*Breitezuerst*" oder "*Tiefenzuerst*" verwendet werden. Nach der gesamten Suche des Bildes, werden die Nicht-Kantenpunkte entfernt, d.h. der Wert dieser auf 0 gesetzt wird.

Abbildung 4.12 zeigt ein Ergebnis mit der Canny Kantenextraktion.

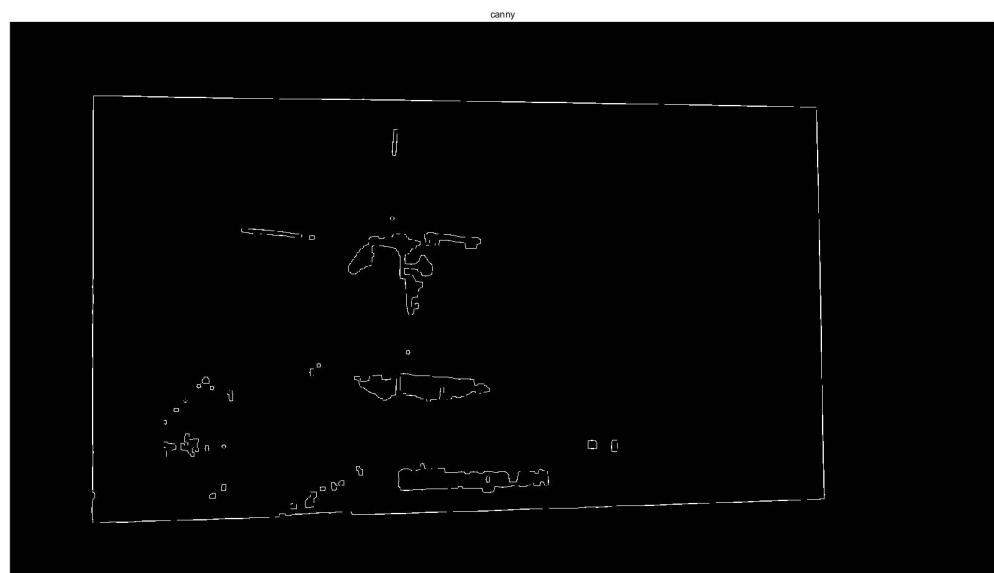


Abbildung 4.12: Binärbild nach der Canny detektion.

4.5 Cross Dilatation

Es könnte herausfinden, dass viele Linien aufgrund der Kamera-Verzerrung nicht perfekt gerade sind, wie Abbildung zeigt. Dies kann die Erkennung des Rechtecks in praktischen Anwendungen beeinträchtigen. Um dieses Problem zu lösen und die robuste Leistung der Methode zu erhöhen, Cross Dilatation wird eingeführt, welche jedes Pixel (x, y) zu einem Kreuzmuster zu erweitern. Alle Pixel, einschließlich die ausgedehnten Pixeln, werden zu die nächste Operation behandelt.

Es ist angenommen, dass die Sphäre des Einflusses jedes Pixels mit einer Region dargestellt wird, die um das Pixel herum zentriert ist und sondern nicht das Pixel selbst. Dann wenn jede Pixel auf eine Linien kann die Region erreichen, die durch das nächste Pixel der Linien beeinflusst wird, dann könnte die gesamte Linie extrahiert werden. Um die Sphäre des Einflusses zu bezeichnen, wird jedes Pixel (x, y) zu einem Kreuzmuster erweitert, indem ein benachbartes Pixel in der vier Richtung des Pixels 1 setzen, wie in Abbildung 4 gezeigt.

4.6 Hough Transformation

Hier werden die Grenzen des modulierten Rechtecks als Linien betrachtet. Um das Problem für eine Rechteck Detektion lösbar zu machen, müssen zunächst Linie extrahiert werden. Um dies zu tun, müssen die Kanten die auf einer Linie im Bild liegen erkannt werden. Die Hough Transformation ist eine beliebte Methode zum Extrahieren von Linien aus einem Bild. Diese ist in der Lage die benötigten Informationen zu liefern, indem die Spitzen an den Punkten der geraden Linien denen des binären Bildes entsprechen. Der Aufwand einer Hough Transformation hängt von der Größe des Bildes und der Anzahl der analysierten Winkel ab. Da die Winkelauflösung für die weitere Verarbeitung wichtig ist, ist eine maximale Kameraneigung vorgeschrieben, welcher in dieser Arbeit $\pm 10^\circ$ beträgt.

Um die Funktionsweise des HT Algorithmus zu beschreiben, müssen einige Definition eingefügt werden. In der Hough Transformation, kann jede Linie in der xy Ebene parametrisch beschreiben werden als:

$$\rho = x \cos \theta + y \sin \theta \quad (4.14)$$

Hier bedeutet ρ die Entfernung vom Ursprung der Koordinate zur Linie, θ der Winkel zwischen ρ und der positiven Richtung der x -Achse, (x, y) sind die Punkte auf der geraden Linie, wie Abbildung 4.13 zeigt.

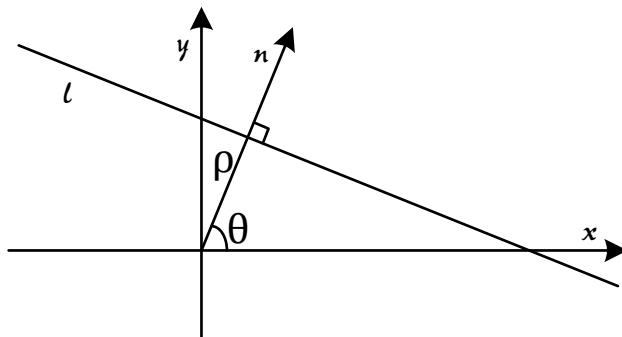


Abbildung 4.13: ρ, θ Parameterraum.

Gleichung 4.14 gilt für jedes Pixel auf dem Bild, d.h. für jeden Punkt auf dem Bild kann eine entsprechende trigonometrische Kurve nach der Hough-Transformation im Parameterraum (θ, ρ) gefunden werden. Es werden zufällig zwei Punkte auf dem Bild ausgesucht, die zwei trigonometrischen Kurven entsprechen welche unvermeidlich einen Schnittpunkt (θ_0, ρ_0) erzeugen

werden. Dieser Schnittpunkt wird dann in die Gleichung der Linie eingefügt, um eine gerade Linie zu bestimmen (und zwar die durch die zwei Punkte auf dem Bild festgelegte Linie).

Das heißt, die entsprechenden trigonometrischen Kurven, die von Punkten auf derselben Linie auf dem Bild erzeugt werden, schneiden sich an einem Punkt (θ_0, ρ_0) im Parameterraum. Je mehr Punkte auf der Linie sind, desto mehr Kurven schneiden sich an diesem Punkt. Wie der Algorithmus einer Hough Transformation wie folgendermaßen darstellt:

1. Erstellung eines Parameterraums mit einer geeigneten Quantisierungsstufe für Entfernung ρ und Winkel θ .
2. Erstellung eines Akkumulator Array $A(\rho, \theta)$ für alle (ρ, θ) .

$$A(\rho, \theta) = 0 \quad (4.15)$$

3. Für jeden Nicht-Hintergrundpunkt (x, y) wird im Bild ρ mit jedem θ berechnet, ob diese Gleichung erfüllt wird: $\rho = x \cos \theta + y \sin \theta$. Wird das Akkumulator-Array um 1 erhöht:

$$A(\rho, \theta) = A(\rho, \theta) + 1 \quad (4.16)$$

4. Suche des Spitzenwerts im Array A , welche die Linie im Parameterraum angibt.

Eine Beispiel Implementierung der Hough Transformation ist in Abbildung 4.14 zu sehen.



Abbildung 4.14: Houghdetektion.

4.7 Radon Transformation

In diesem Abschnitt wird eine erweiterte Form von Hough Transformation bzw. Radon Transformation vorgestellt. Die Hough Transformation transformiert eine gegebene Kurve im Bildraum als ein Punkt im Parameterraum entsprechend dem Parameterausdruck der Kurve ,dann durch Suchen nach Peaks in dem Parameterraum diesen Kurve in dem Bildraum bestimmen. Radon Transformation projiziert den Bildraum in den $\rho\theta$ -Raum in Form eines Linienintegrals (entspricht dem Parameterraum der Linie). Es kann verstanden werden, dass die Hough Transformation eine diskrete Form der Radon Transformation ist. Aufgrund der soliden mathematischen Basis ist die Radon Transformation im Gegensatz dazu raffinierter und genauer.

Die Idee der Radon-Transformation besteht darin, die ursprüngliche Funktion räumlich zu transformieren, das heißtt, die ursprünglichen in der XY-Ebene Punkte werden auf die AB-Ebene abgebildet. Dann sind alle Punkte einer geraden Linie in der XY-Ebene an einer demselben Punkt der AB-Ebene. Zeichnen die akkumulierte Dicke der Punkte auf der AB-Ebene auf, und können die Existenz der Linien auf der XY-Ebene kennen.

Wie bei der Hough-Transformation werden die Parameter ρ und θ zuerst verwendet, um die gerade Linie auf der xy-Ebene darzustellen. Diese Parametrisierte Linie wird in Abbildung 4.15 gezeigt und die Formel läuft als:

$$L(\rho, \theta) = \{(x, y) : x \cos \theta + y \sin \theta = \rho\} \quad (4.17)$$

Angenommen, es gibt eine Funktion $f(x, y)$, wie in Abbildung 4.16 gezeigt, dann ist das Integral der Funktion über die gerade Linie L:

$$\int_L f(x, y) ds \quad (4.18)$$

Wobei ds das Differential der Linie ist.

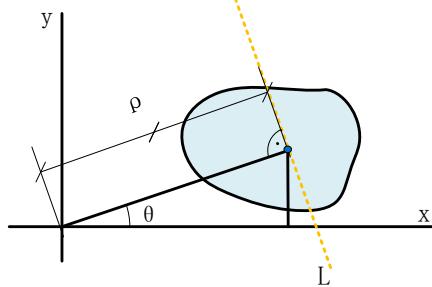


Abbildung 4.15: Parametrisierte Linie.

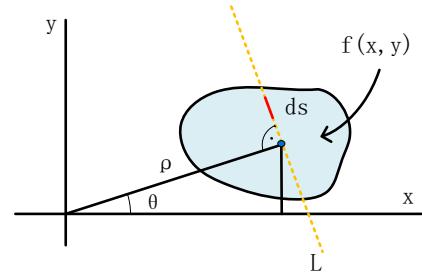


Abbildung 4.16: Integral der Funktion $f(x, y)$.

Die obigen Integrale für x, y sind leicht zu lösen. Eine der Lösungstechniken ist die Verwendung der Delta-Funktion. Das obige Integral kann geschrieben werden als:

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - \rho) dx dy \quad (4.19)$$

Deswegen wenn einen Satz von (ρ, θ) bestimmen, können einen Integral Wert entlang $L(\rho, \theta)$ erhalten. Somit wird die Radon-Transformation zum Linienintegral der Funktion $f(x, y)$ umgewandelt, wie in Abbildung 4.17 gezeigt. Die Höhe g repräsentiert das Linienintegral von $f(x, y)$ auf der Linie L .

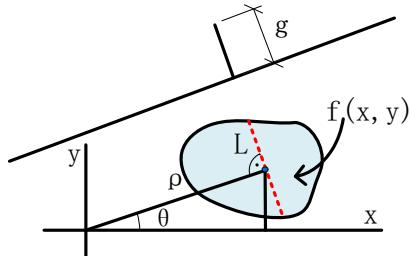


Abbildung 4.17: Radon Transformation als Li- nienintegral.

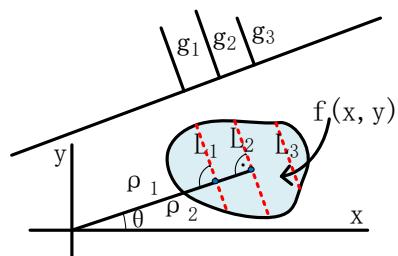


Abbildung 4.18: Parallele Linienintegral mit glei- chen θ .

Wenn viele Linien parallel zu L sind, haben sie das gleiche θ , die radialen Koordinaten ρ sind verschieden. Für jede solche parallele Linie machen ein Linienintegral von $f(x, y)$, werden viele Projektionslinien erzeugt, wie in Abbildung 4.18 gezeigt. Das heißt, die

Radon-Transformation eines Bildes in einem bestimmten Winkel erzeugt N Linienintegralwerte (Radon-Transformation), und jeder Zeilenintegralwert entspricht einer Radialkoordinate. Die Radon-Transformationswerte bei verschiedenen Winkeln werden kombiniert, um ein Radon-Diagramm zu bilden.

5 Implementierung

In diesem Kapitel soll der zuvor vorstellt Verfahren implementiert werden. Zuerst wird die experimenteller Umgebung auf PC, einschließlich die benutzte Geräte, Software einführt. Anschließend besteht die Implementierung der beiden vorherigen Methoden mit Matlab, welche die Anzeige jedes Schritt und die Demonstration(oder hier Anzeige?) der Endergebnisse enthalten. Schließlich läuft die Implementierung des Detektionsverfahrens auf einer Smartphone-GPU.

5.1 Experiment Umgebung

Hier wird die Experimentumgebung auf PC-Seite bzw. Test Geräte, Software in diese Arbeit erläutert. Google Pixel wird benutzt, um die Bild aufzunehmen. Dessen Parameter sind in Tabelle 3.1 verfügbar. Die Anwendungssoftware “*VLCReceiver*” wird mit automatischer Einstellungsmodus benutzt. Entsprechend dem möglichen Einsatzbereich des David-Systems wurden im Experiment die Bildschirme mit verschiedener Größen und Umgebungen ausgewählt. Die in dem Experiment verwendeten Bildschirme sind in der Tabelle 5.1 gezeigt. Die aufgenommen Bilder wird auf der PC-Seite mit Matlab 2017b unter der Lizenz der TU Dortmund verarbeitet.

Tabelle 5.1: Verwendeter Bildschirm.

Hersteller	Model	Beleuchtung	Auflösung	Größe	Frequenz
Lenovo Laptop	T440P	LED	1920 × 1280	14"	60
Samsung	SMB2440MH	LED	1920 × 1280	24"	60
Ausu	VG248	LCD	1920 × 1280	24"	60
LG	55EF950V	OLED	2160 × 3840	55"	120
Projektor	1234	1234	1920	14"	60

5.2 Implementierung der ersten Verfahren

Hier ist der Implementierungsprozess von Methode 1. Das Bild stammt aus einer Reihe von acht Bildern, die mit einer Smartphone in Handy aufgenommen wurden. Entsprechend den

Eigenschaften von Bildregistration in dieser Arbeit sind die von der handgehaltenen Kamera aufgenommenen Objekte alle statische Videos. Dies bedeutet, dass sich der Inhalt des Videos nicht ändert. Im tatsächlichen Betrieb entspricht es einer statischen elektronischen Werbetafel. Die Modulationsamplitude werden als 6 erstellt, anschließend Daten Datenblock als $4 \times 4\text{Pixel}$, QR Muster Größe als $72 \times 72\text{Pixel}$.

Aufgrund von Handschütteln während der Aufnahme entsteht eine leichte Verschiebung zwischen den Bildern. Um diese Effekt deutlicher anzusehen, wird das erste Bild und das achte Bild zum Vergleich verwendet, wie in Abbildung 6.1 zeigt. Abbildung 5.2 zeigt das Bild nach der Bildregistration. Wie zu sehen ist, ist der Abstand gut komprimiert.

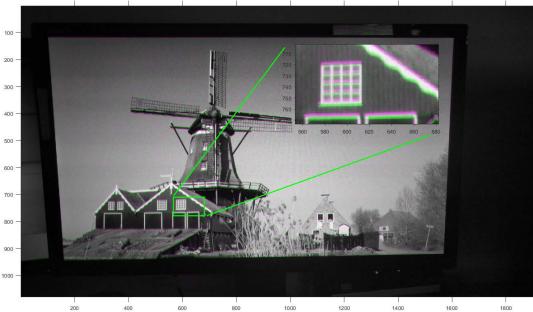


Abbildung 5.1: Bilder vor Bildregistration.

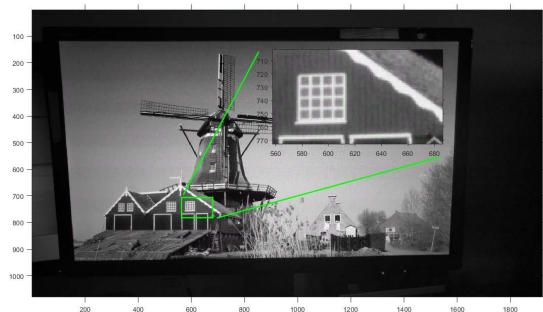


Abbildung 5.2: Bilder nach Bildregistration.

Aufgrund der verschiedene Bildwiederholfrequenz werden die QR Muster in den Differenzbildern einige unerwartete Effekte aufweisen könnten, wie in Tabelle 3.2 gezeigt. Um diese zu lösen, eine Absolutwertoperation wird durchführt. Danach mit Hilfe des Berechnung der „Energie“ jedes Differenzbild lassen sich ein neu zu detektierendes Bild entstehen. Es wird in Abbildung 5.3 gezeigt.



Abbildung 5.3: Ein zu detektierendes Bild.

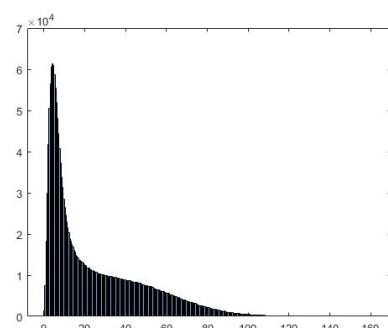


Abbildung 5.4: Histogramm.

Anschließend wird dies Bild zu Bildverarbeitung Modul machen. Abbildung 5.5 und Abbildung 5.6 zeigen die Ergebnis durch die Binarisierung bzw. die morphologische Operation.

Mit QR Muster Detektion kann die Zentrum des Muster bestimmen werden, wie in Abbildung 5.7 zeigt. Deshalb lassen sich der Modulationsbereich befinden. In Abbildung Abbildung?? wird es durch ein rotes Rechteck angezeigt. Schließlich mit einer projektive Transformation kann die Endergebnis erhalten, wie in Abbildung 6.6 gezeigt.



Abbildung 5.5: Binär.

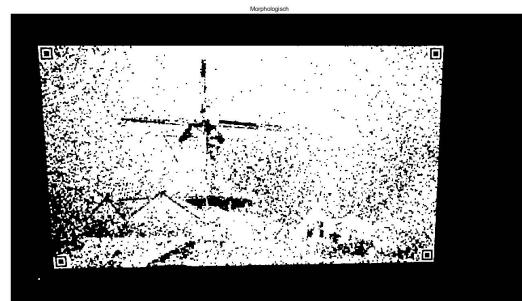


Abbildung 5.6: Morphologisch.

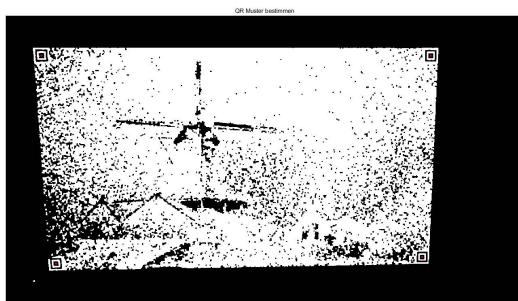


Abbildung 5.7: QR Muster.



Abbildung 5.8: Modulationbereich.



Abbildung 5.9: Ergebnis der 1. Methode.

Als Nächst einige Ergebnisse der Verwendungen von die 1. Methode wird angegeben.

Bilder

5.3 Implementierung der zweiten Verfahren

Hier wird der Implementierung für zweite Methode vorstellen. Das Bild stammt aus einer Reihe von acht Bildern, die durch Google Pixel mit Stativ aufgenommen wurden. Deswegen wird hier Bildregistration nicht verwendet. Dies bedeutet, dass sich der Inhalt des Videos verändern lassen. Die Modulationsamplitude werden als 6 erstellt, anschließend Daten Datenblock als $4 \times 4\text{Pixel}$.

Durch das Wissen des 4. Kapitels, wird von diese Bildern eine Reihe Differenzbilder enthalten. 3 mit größte “Energie” Differenzbilder werden addiert, um ein zu detektierendes Bild zu entstehen, wie in Abbildung 5.10 gezeigt. Anschließend mit Verwendung der Binarisierungsmethod, hier lauft Ostu Schwellen methode, wird ein Binärbild bekommen, indem der Modulationsbereich von Bild herausgezogen wird. Um die kleinen Punkte und Lücken zu beseitigen, lassen sich eine morphologische Operation durchführen. Das hier benutzte Strukturelement ist ein 5×5 Quadrat. Diese Effekt der beider Operationen werden in Abbildung 5.11 und Abbildung 5.12 gezeigt. Danach um die Einfluss der Kamera-Verzerrung zu verhindern, wird die Binärbild

mit einer Cross Dilatation machen, also $n_{around} = 1$. Das Struktur der Cross Dilatation wird in Abbildung gezeigt und Abbildung 5.14 zeigt diese Operation. Schließlich lassen das Bildzentrum auf den Ursprung setzen , und durch Vergleich der Integralwerte kann die längste Linien, die oben, unten, links und rechts legt, gefunden werden. Außerdem um die Berechnungskosten zu reduzieren, werden die Linien innerhalb von $\pm 10^\circ$ erkannt.



Abbildung 5.10: Ein zu detektierendes Bild.

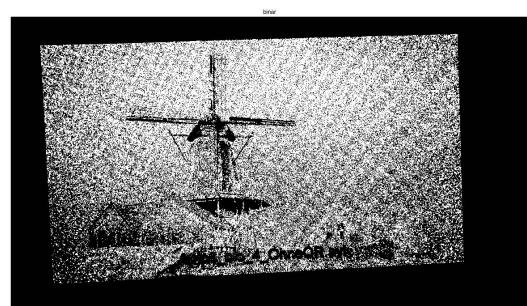


Abbildung 5.11: Binarisierung.



Abbildung 5.12: Morphologisch.

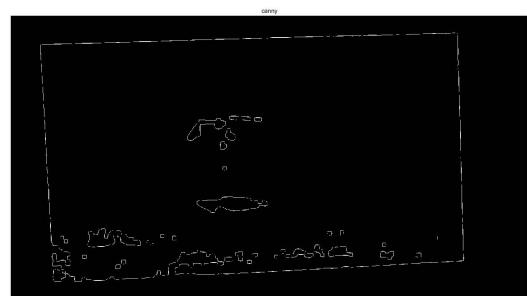


Abbildung 5.13: Canny Detektion.

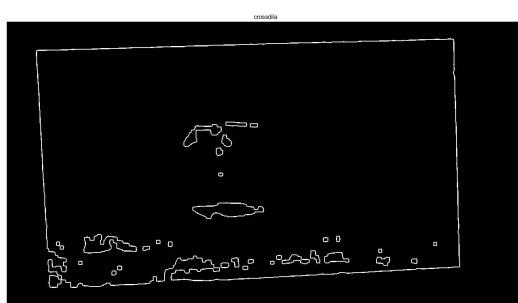


Abbildung 5.14: Cross Dilatation.



Abbildung 5.15: Radon Detektion.



Abbildung 5.16: Ergebnis der 2. Methode.

Als Nächst einige Ergebnisse der Verwendungen von die 2. Methode wird angegeben.

Bilder

5.4 Implementierung auf einer Smartphone-GPU

In diese Abschnitt wird die Implementierung auf Smartphone-GPU erläutert. Hier wird Smartphone Xiaomi Mi3 benutzt. Dessen Parameter sind in Tabelle 5.2 verfügbar. Durch RenderScript, als ein Framework für High Performance Computing auf der Android-Plattform, kann die vorherige Methode auf Smartphone-GPU parallel implementiert wird. Die zweite Methode ist hier implementiert. Die verwendet Software auf der PC-Seite läuft Android Studio 3.1.4.

Tabelle 5.2: Grundlegende Parameter für Mi3.

Smartphone	GPU	CPU	RAM
Mi3	Qualcomm Snapdragon 800 MSM8274AB	Krait 400, 2300MHZ	2 GB, 800 MHZ

RenderScript

RenderScript stellt eine native High-Performance-Computing-API bereit, die in C (C99-Standard) geschrieben ist. Mit Renderscript können Anwendungssoftware automatisch verschiedene

Operationen parallel über alle verfügbaren Prozessorkerne ausführen. Es bietet auch Unterstützung für verschiedene Verarbeitungsarten wie CPU, GPU oder DSP. Renderscript ist nützlich für die Grafikverarbeitung, mathematische Modelle oder jede andere Anwendung, die viele mathematische Berechnungen erfordert. Darüber hinaus kann auf alle diese Funktionen zugegriffen werden, um verschiedene Architekturen oder eine unterschiedliche Anzahl von Prozessorkernen zu unterstützen, ohne Code zu schreiben. Es ist auch nicht notwendig, die Anwendungssoftware für verschiedene Prozessortypen zu kompilieren, da der RenderScript-Code kompiliert wird, während er auf dem Gerät ausgeführt wird.

Die Vorteile für RenderScript sind wie folgend gelegt:

1. Convenience: Renderscript ist für die Ausführung auf vielen Geräten in verschiedenen Prozessor- (CPU-, GPU- und DSP-Instanzen) Architekturen ausgelegt. Alle unterstützten Architekturen sind nicht für jedes bestimmte Gerät spezifisch, da sein Code zur Laufzeit auf dem Gerät kompiliert und zwischengespeichert wird.
2. Effizient: Renderscript stellt parallel eine Hochleistungs-Computing-API zur Verfügung, die den Kernel über das gesamte Gerät verteilt.
3. Einfach zu verwenden: Renderscript vereinfacht die Entwicklung, wo dies möglich ist, z. B. das Abbrechen von JNI-Code.

Die Ergebnisse der Implementierung auf MI-3 wird in Abbildung 5.17 gezeigt.



Abbildung 5.17: Ergebnis auf Smartphone.

6 Evaluierung

In diesem Kapitel soll der zuvor implementierte Verfahren evaluiert werden. lassen sich eine Reihe von Testanalysen durchführen, um die Methode auf die Genauigkeit und die Leistungsfähigkeit zu testen.

6.1 Evaluierung der 1.Verfahren

Hier wird das 1. Verfahren mit dessen Genauigkeit und Leistungsfähigkeit evaluiert. Als in Kapitel 3 vorstellt, diese Methode ist hauptlich für den Fall, bei der Aufnahmen die Smartphone in der Hand gehalten. Dann brauchen eine Bildregistration Operation, um die Bildern in dasselbe Korordinatensystem zu transformieren. Die Funktion dieses Moduls wirkt sich direkt auf die Entstehung der Differenzbildern aus und hat einen wesentlichen Einfluss auf die spätere Erkennung. Deswegen wird zuerst eine Analysierung dafür durchgeführt.

Um Leistung intuitiv auszudrücken, lassen sich einen Satz Parameter setzen. Das Modulationsamplitude erstellt als 6, Datenblock als $4 \times 4\text{Pixel}$, QR Muster Größe als $72 \times 72\text{Pixel}$. In der Reihe von Bildern wird das erste Bild und das letzte Bild verglichen, wie in Abbildung 6.1 zeigt.

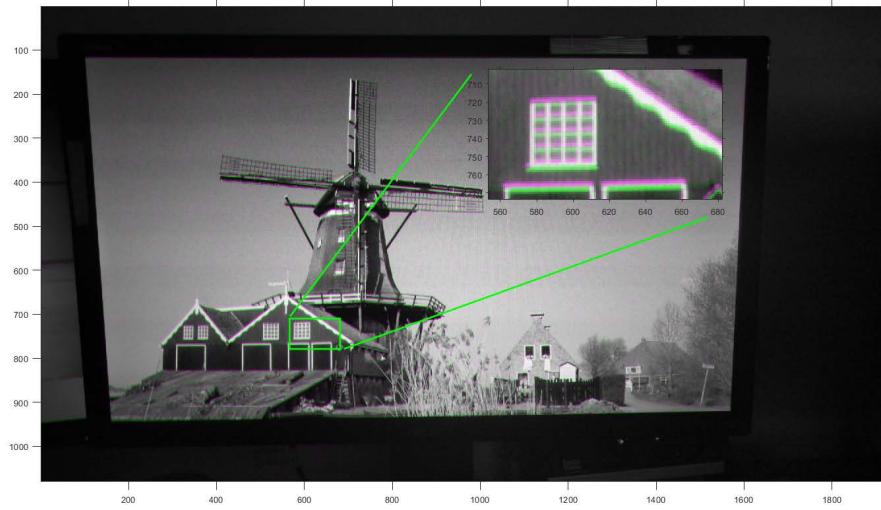


Abbildung 6.1: Verschiebung durch Handshake.

Aus der Abbildung ist deutlich zu erkennen, dass das Bild durch Handshake eine deutliche Verschiebung erfahren hat. Die durchschnittliche Entfernung der entsprechenden Pixel zwischen den zwei Bildern hier beträgt 2.3213 Pixel. Durch die Bildregistration, die in dieser Arbeit vorstellt, kann dieses Problem effektiv gelöst werden. Abbildung 6.2 zeigt die Konvergenzkurve des Fehlers J bei der Summenwertbildung im Quadrat aller Korrespondenzpunkte.

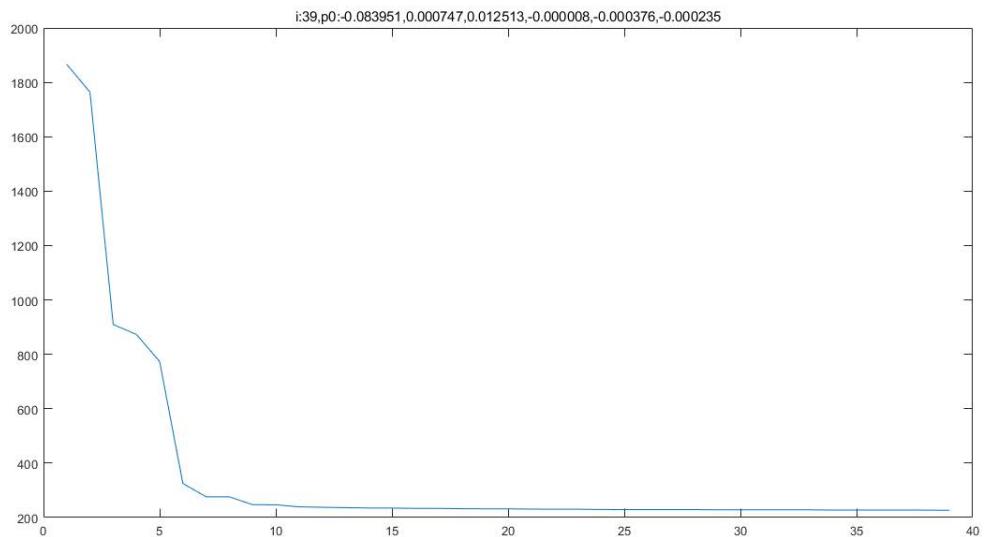


Abbildung 6.2: Konvergenzkurve von J .

Die vertikale Koordinate repräsentiert den Fehlerwert J und die horizontale Koordinate reprä-

sentiert die Anzahl der Iterationen des Algorithmus. Als zeigt in der Abbildung, wird J nach etwa 10 Iterationen konvergiert. Die detailliert sammelte Daten wird in Tabelle 6.1 gegeben.

Tabelle 6.1: Daten des Fehlers.

Verglichen Bilder	Anzahl der Punkt	Ursprünglich Fehler	Mittelwert	Ergebnis Fehler	Mittelwert
1-2	849	350.5674	0.4129	224.9265	0.2649
1-3	723	826.8834	1.1437	236.0301	0.3265
1-4	619	662.3621	1.0701	208.4760	0.3368
1-5	500	723.2307	1.4465	207.1174	0.4142
1-6	835	1607.4002	1.9250	231.7138	0.2775
1-7	802	1059.4579	1.3210	260.3104	0.3246
1-8	803	1863.9972	2.3213	246.7970	0.3073

Abbildung 6.3 zeigt diese Effekt, dass das Bild in Abbildung 6.1 durch die Bildregistration entstehen. Wie zu sehen ist, ist der Fehler gut komprimiert. Abbildung 6.3 zeigt die Abweichung in horizontaler und vertikaler Richtung zwischen den entsprechenden Punkten nach der Operation. Es ist ersichtlich, dass die Abweichung in beiden Richtungen ca. 95% unter 0.5 Pixel sind. Die durchschnittlichen Offsets in horizontaler und vertikaler Richtung betragen 0.1746 bzw. 0.1956 Pixel. Also die neue durchschnittliche Entfernung beträgt 0.3073 Pixel.

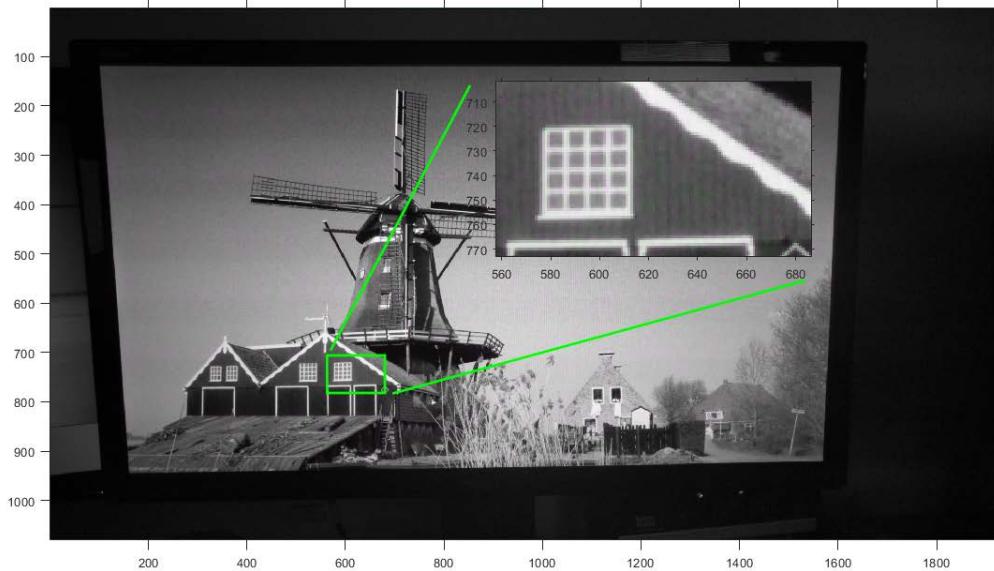


Abbildung 6.3: Bild nach Registration.

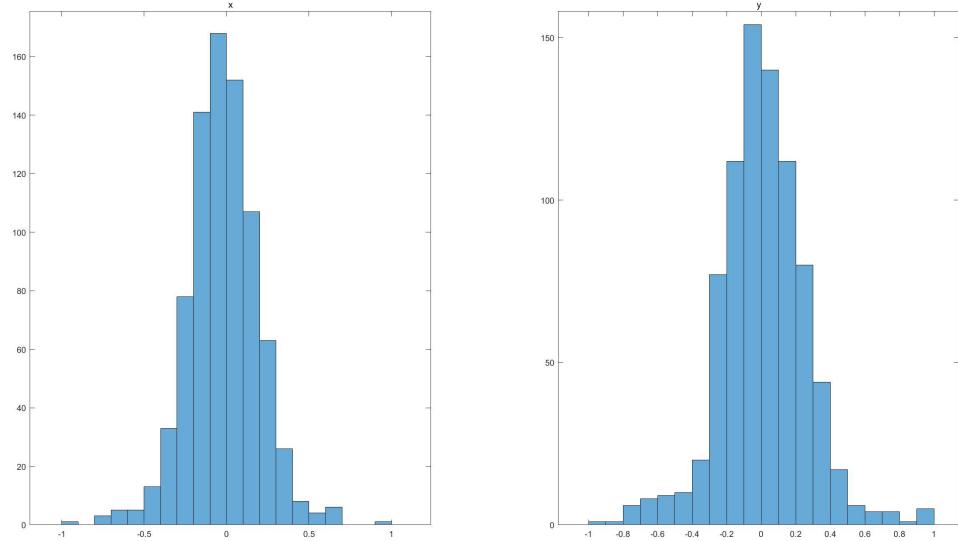


Abbildung 6.4: Histogramm.

Als nächstes folgt eine Gesamtbewertung von Methode 1. Hier wird ein Bild speziell für das Testen verwendet, dh das Gitter wird zum Originalbild hinzugefügt, und die Größe des Gitters wird im Voraus bestimmt. Hier wird ein Gitter von 200×200 ausgewählt. Abbildung 6.3 zeigt diese spezielle Bild.



Abbildung 6.5: GitterBild.

Die detektierend Modulationsbereich durch die 1. Methode wird in Abbildung 6.3 gezeigt. Aus

der Abbildung wurde der Offset jedes Gitterpunkts markiert. Der Untergraph in der oberen rechten Ecke der Abbildung ist eine Teilvergrößerung des roten rechteckigen Bereichs. Die durchgezogene Linie repräsentiert die Position des Gitters im Idealfall, und die gestrichelte Linie repräsentiert die tatsächliche Position des Gitters, nachdem das Verfahren 1 detektiert wurde. Der rote Pfeil zeigt den Offset des Punktes an. Der daraus berechnete durchschnittliche Offset beträgt 1.7680 Pixel.

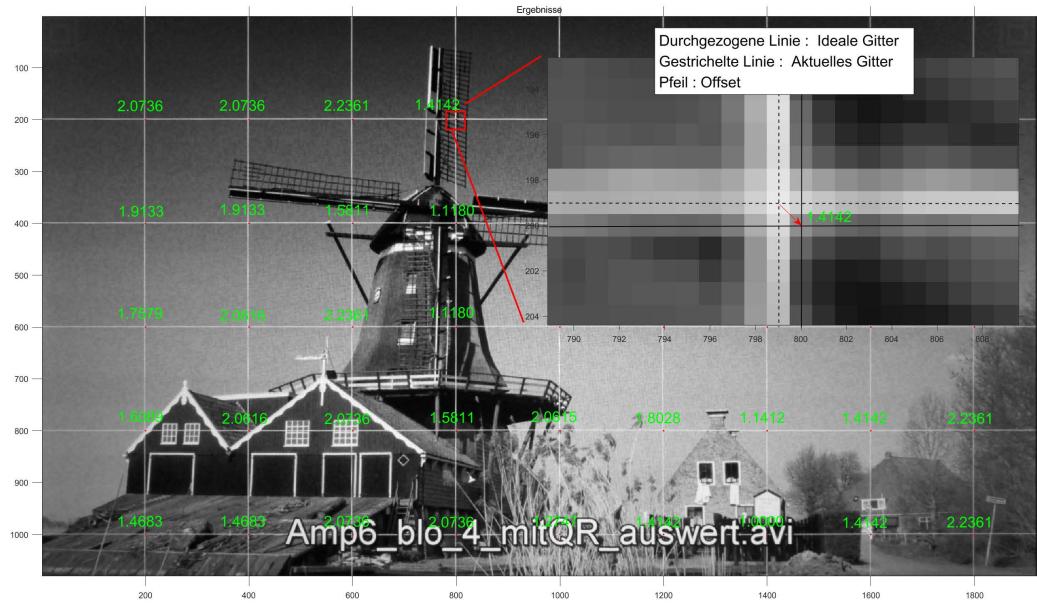


Abbildung 6.6: Ergebnis1.

6.2 Evaluierung der 2.Verfahren

In dieser Abschnitt wird das 2. Verfahren evaluiert. Als in Kapitel 4 vorstellt, diese Methode ist hauptlich für den Fall, bei der Aufnahmen des Bilder die Smartphone auf ein Stativ gelegt. Dann wird hier die Bildregistration Operation nicht mehr gebraucht. Die Gitterbilder aus dem vorherigen Abschnitt werden auch hier verwendet. Der Parameter ist eingestellt auf: Modulationsamplitude als 6, Datenblock als $4 \times 4\text{Pixel}$, QR Muster Größe als $72 \times 72\text{Pixel}$. Die Ergebnisse nach Methode 2 sind in der Abbildung 6.7 dargestellt. Der berechnete durchschnittliche Offset beträgt 1.3468 Pixel.

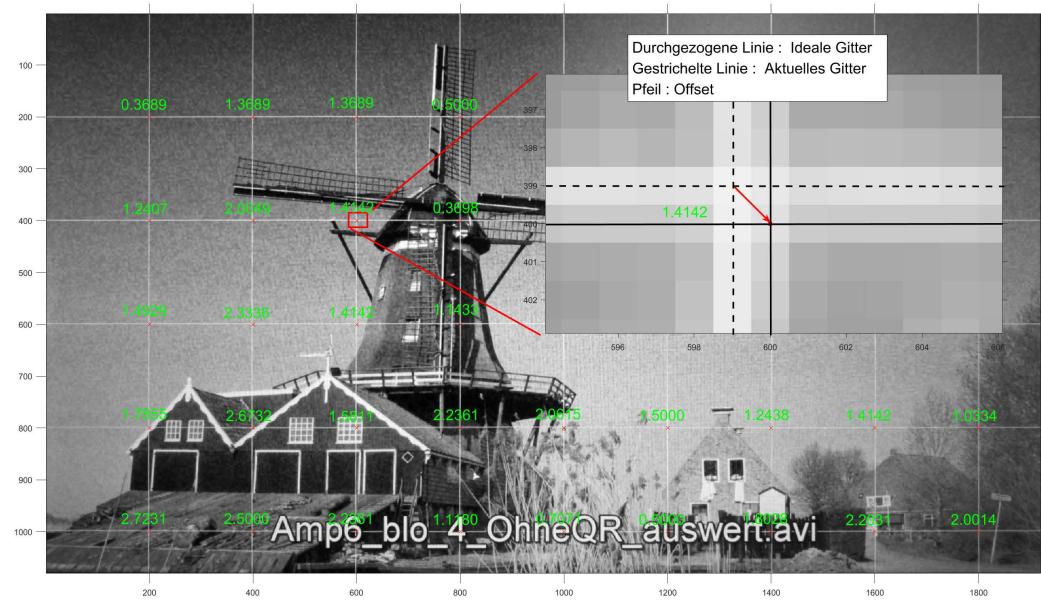


Abbildung 6.7: Ergebnis2.

7 Zusammenfassung

Anhang A

Erster Anhang

A.1 Section

Jetzt nur noch schreiben! :)

Abbildungsverzeichnis

2.1 Eine beispielhafte Implementierung des DaVid-Systems	3
2.2 Schematische Darstellung vom DaVid-System	4
2.3 Blockschaltbild der Signalverarbeitung in zeitlicher differentieller Modulation	5
2.4 Ein Beispiel einer modulierten Bildfolge	7
3.1 Strukturdiagramm	10
3.2 Flussdiagramm der ersten Methode	12
3.3 Flussdiagramm der Bildregistrierung	13
3.4 Scale space	16
3.5 Extremwert Erkennung	16
3.6 Dominante Orientierung feststellen	17
3.7 Merkmalspunkt Deskriptor	17
3.8 SURF Merkmal	18
3.9 Linien Detektion	19
3.10 Ohne RANSAC	21
3.11 Mit RANSAC	21
3.12 Modell einer Lochkamera	22
3.13 Transformation vom Weltkoordinatensystem zum Kamerakoordinatensystem	23
3.14 Rotation um Z-Achse	24
3.15 Transformation vom Kamerakoordinatensystem zum Bildkoordinatensystem	25
3.16 Konvertierung vom Bildkoordinatensystem zum Pixelkoordinatensystem	26
3.17 Rotationsbewegungsmodell	29
3.18 Transformation in die selben Koordinate	31
3.19 Flussdiagramm für Optimierung	32
3.20 Differenzbild Flussigdiagramm	35
3.21 Ein zu detektierendes Bild	36
3.22 QR Muster	37
3.23 QR Muster Ratio	38
3.24 Flussdiagramm der QR Muster Detektion	40
3.25 Beispiel einer QR Muster Detektion	41
4.1 Strukturdiagramm	43

4.2 Flussdiagramm der zweite Methode	44
4.3 Histogramm eines einfachen Beispiels	45
4.4 Binarisierung mit globalen Schwellenwertmethode	47
4.5 Binarisierung mit lokalen Schwellenwertmethode	47
4.6 Binarisierung mit Ostu	49
4.7 Binärbild mit Ostu	49
4.8 Einige grundlegende Strukturelemente	50
4.9 Dilatation und Erosion	51
4.10 Öffnung und Schließung	53
4.11 Bild nach einer morphologischen Operation	54
4.12 Binärbild nach der Canny detektion	58
4.13 ρ, θ Parameterraum	59
4.14 Houghdetektion	60
4.15 Parametrisierte Linie	62
4.16 Integral der Funktion $f(x, y)$	62
4.17 Radon Transformation als Linienintegral	62
4.18 Parallelle Linienintegral mit gleichen θ	62
 5.1 Bilder vor Bildregistration	65
5.2 Bilder nach Bildregistration	65
5.3 Ein zu detektierendes Bild	65
5.4 Histogramm	65
5.5 Binär	66
5.6 Morphologisch	66
5.7 QR Muster	66
5.8 Mudulaitonbereich	66
5.9 Ergebnis der 1. Methode	67
5.10 Ein zu detektierendes Bild	68
5.11 Binarisierung	68
5.12 Morphologisch	68
5.13 Canny Detektion	68
5.14 Cross Dilatation	68
5.15 Radon Detektion	68
5.16 Ergebnis der 2. Methode	69
5.17 Ergebnis auf Smartphone	70
 6.1 Verschiebung durch Handshake	72
6.2 Konvergenzkurve von J	72
6.3 Bild nach Registration	73

Abbildungsverzeichnis

6.4 Histogramm	74
6.5 GitterBild	74
6.6 Ergebnis1	75
6.7 Ergebnis2	76

Tabellenverzeichnis

3.1	Parameter der Kameras im Vergleich	28
3.2	Die mögliche Differenzbilder	34
5.1	Verwendeter Bildschirm	64
5.2	Grundlegende Parameter für Mi3	69
6.1	Daten des Fehlers	73

Quellcodeverzeichnis

Literatur

- [1] R. Kays, C. Brauers und J. Klein, „Modulation concepts for high-rate display-camera data transmission“, in *2017 IEEE International Conference on Communications (ICC)*, Mai 2017, S. 1–6.
- [2] R. Kays, C. Brauers und J. Klein, „DaViD : Data Transmission Using Video Devices-An Innovative System for Smart Media Applications“, 2016.
- [3] R. Kays, „Modulation concepts for visible light communication using video displays“, in *2015 IEEE 5th International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*, Sep. 2015, S. 388–392.
- [4] R. Kays, „Visible light communication using TV displays and video cameras“, in *2015 IEEE International Conference on Consumer Electronics (ICCE)*, Jan. 2015, S. 554–555.
- [5] H. Bay, A. Ess, T. Tuytelaars und L. V. Gool, „SURF:Speeded Up Robust Features“, Bd. 110, 2008, S. 346–359.
- [6] M. A. Fischler und R. C. Bolles, „Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography“, *Commun. ACM*, Jg. 24, Nr. 6, S. 381–395, Juni 1981. Adresse: <http://doi.acm.org/10.1145/358669.358692>.
- [7] Z. Zhengyou, „A Flexible New Technique for Camera Calibration“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jg. 22, S. 1330–1334, Dez. 2000.
- [8] N. Otsu, „A Threshold Selection Method from Gray-Level Histograms“, *IEEE Transactions on Systems, Man, and Cybernetics*, Jg. 9, Nr. 1, S. 62–66, Jan. 1979.
- [9] J. Canny, „A Computational Approach to Edge Detection“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jg. PAMI-8, Nr. 6, S. 679–698, Nov. 1986.
- [10] R. Reinhold und R. Kays, „Improvement of IEEE 802.15.4a IR-UWB for time-critical industrial wireless sensor networks“, in *2013 IFIP Wireless Days (WD)*, Nov. 2013, S. 1–4.
- [11] T. K. Moon, *Error Correction Coding, Mathematical Methods and Algorithms*, 1. Aufl. Hoboken, NJ: Wiley-Interscience, Juni 2005, 800 S.

- [12] *Ieee standard for local and metropolitan area networks, part 15.4: Low-rate wireless personal area networks*, IEEE Std 802.15.4-2011, 2011.

Eidesstattliche Versicherung

Blaubär, Käpt'n Kevin
Name, Vorname

123456
Matr.-Nr.

Ich versichere hiermit an Eides statt, dass ich die vorliegende Bachelorarbeit mit dem Titel
Tasty Kanalmodell für die drahtlose Kommunikation zwischen Gebäuden und Außeninstallationen
selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäß Zitate kenntlich gemacht.
Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Dortmund, 4. September 2018
Ort, Datum

Unterschrift

Belehrung:

Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -)

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin“) zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Dortmund, 4. September 2018
Ort, Datum

Unterschrift