Dreidimensionale Szenenmodellierung durch monokulare Exploration mit einem mobilen Roboter

SEMINARARBEIT

Im Rahmen des Seminars 3D-Rekonstruktion

Von

Dipl.-Kfm. Gerrit Rüdiger Staib

aus Mannheim

vorgelegt am

Lehrstuhl für Praktische Informatik IV

Prof. Dr. W. Effelsberg

Fakultät für Mathematik und Informatik

Universität Mannheim

Januar 2003

Betreuer: Dipl.-Inf. Dirk Farin

INHALT

Abkürzungsverzeichnis

bzw.	beziehungsweise
ff.	Fortfolgende
o.ä.	oder ähnliches
u.a.	unter anderem
z.B.	zum Beispiel

Abbildungsverzeichnis

1 Einleitung

1.1 Motivation der Arbeit

Roboter nehmen eine zunehmend wichtige Rolle in einer Vielzahl von Bereichen ein. Hierbei kommen einerseits stationäre Roboter im Rahmen automatisierter Produktionsvorgänge zum Einsatz. Andererseits steigt auch die Anzahl der Anwendungsmöglichkeiten für mobile Roboter. So sind mobile Roboter z.B. in Transportsystemen oder aber in Erkundungs- und Wartungssystemen denkbar.

Um diesen Aufgaben gerecht werden zu können, müssen mobile Roboter die sie umgebende Umwelt mithilfe von Sensoren erkennen können. Dies ist mit den herkömmlichen Sensoren, wie Infrarot oder Ultraschall nur sehr bedingt möglich, da es bei diesen schwierig ist, räumliche Strukturen zu erkennen. Somit bietet sich ein räumliches Erfassungssystem auf visueller Basis, welches mithilfe einer Videokamera arbeitet, an.

Die Motivation der Arbeit ist es daher, ein Verfahren zur räumlichen Erfassung mithilfe eines mobilen Roboters vorzustellen. **Das Ziel der Arbeit ist die Erstellung eines polygonbasierten 3D-Modells der Umgebung des Roboters.** Als Basis der Arbeit dient die Diplomarbeit 'Dreidimensionale Szenenmodellierung durch monokulare Exploration mit einem mobilen Roboter' von Oliver Schimmel, die April 1999 am Lehrstuhl für Rechnerarchitektur des Wilhelm-Schickard-Instituts für Informatik der Universität Tübingen vorgelegt wurde.

Der Rahmen der vorliegenden Arbeit ist das Seminar '3D-Rekonstruktion' im Wintersemester 2002/03 am Lehrstuhl für Praktische Informatik IV der Universität Mannheim.

1.2 Aufbau der Arbeit

Die vorliegende Arbeit ist in sieben Kapitel unterteilt.

Im ersten Kapitel wird eine kurze Einführung in das Thema gegeben.

Das zweite Kapitel beschreibt kurz den Versuchsaufbau und die nötige Vorverarbeitung der Bildvorlage.

Kapitel drei hat die Erkennung und Extraktion von Features (markanten, eindeutigen Bildmerkmalen) zum Inhalt.
Im vierten Kapitel wird beschrieben, wie diese Features über den Zeitverlauf verfolgt werden ('Tracking') um die Voraussetzungen für 3D-Rekonstruktion zu schaffen.
Kapitel fünf erläutert die eigentliche 3D-Rekonstruktion, die über einen kombinierten geometrischen und stochastischen Ansatz erfolgt.
Im sechsten Kapitel wird die Visualisierung der gewonnenen 3D-Rekonstruktion beschrieben.
Das siebte Kapitel enthält ein kurzes Fazit sowie eine Zusammenfassung der Ergebnisse dieser Arbeit.

2 Versuchsaufbau und Vorverarbeitung

2.1 Versuchsaufbau

Die Aufnahme der Bilder erfolgt über eine schwenkbare Kamera, die auf einen mobilen Roboter montiert ist. Hierbei werden nicht nur die Bilddaten selbst, sondern auch die jeweils aktuelle Roboterposition, Roboterrichtung und Kameraorientierung mit abgespeichert und protokolliert. Der Informationsfluss erfolgt über eine Client-Server-Struktur, die in nachfolgender Abbildung dargestellt ist:

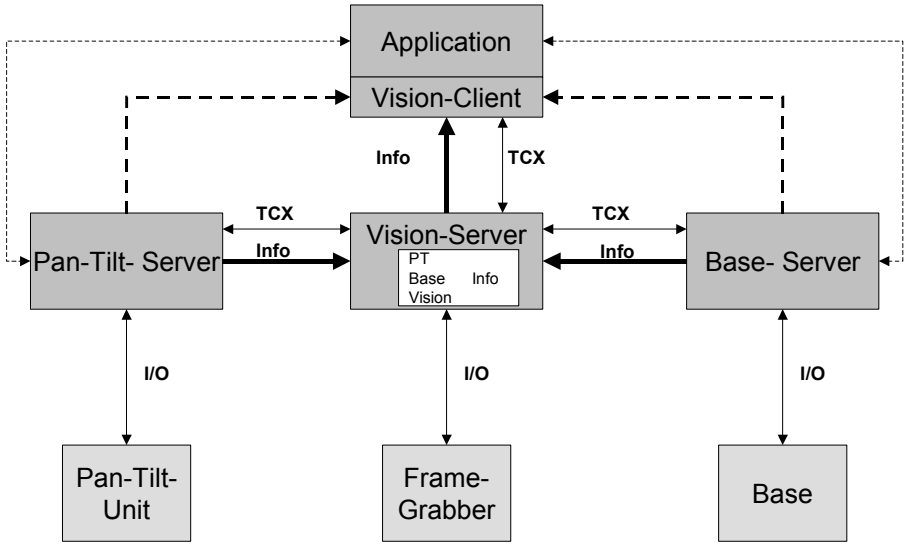


Abbildung 2-1: Client-Server-Struktur

Quelle angelehnt an [SCHIM99], Seite 13

Die Bilddaten werden via Framegrabber von der Kamera an den Visionserver übermittelt. Bei der dafür nötigen Bildaufnahme mittels Kameraoptik können dabei Skalierungsfehler und Radialverzerrungen auftreten, was im folgenden Abschnitt weiter erläutert wird. Der Pan-Tilt-Server gibt die jeweilige Kameraorientierung als Pan-Winkel ψ und Tilt-Winkel θ an.
Der Base-Server liefert die globale x- und y-Position des Roboters und die Fahrtrichtung (als Winkel η).

2.2 Vorverarbeitung

Voraussetzung für eine spätere 3D-Rekonstruktion ist die Genauigkeit der zugrundeliegenden Bilddaten. Hierbei spielt die Bildvorverarbeitung eine wichtige Rolle, um Rauschen und Kontrastschwächen zu entfernen.
Wichtige Punkte bei der Bildvorverarbeitung sind:

- Farbkorrektur
- Kontrastverstärkung
- Eliminierung des Rauschens
- Kamerakalibrierung

Eine Darstellung aller Punkte würde allerdings den Rahmen dieser Arbeit sprengen, daher wird an dieser Stelle nur auf die Kamerakalibrierung als in diesem Zusammenhang wichtigsten Punkt weiter eingegangen.
Im Rahmen der zugrundeliegenden Arbeit wird hierbei eine Kamera mit pe ie fol

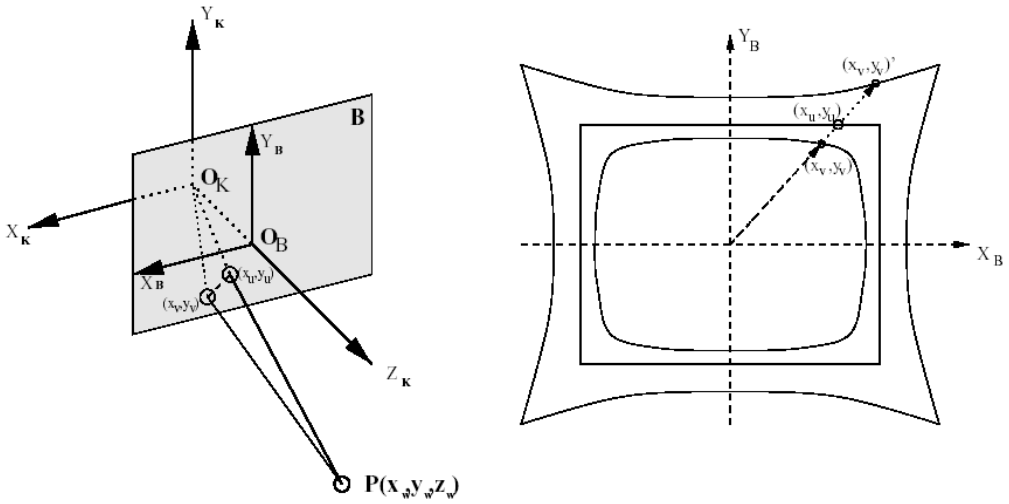


Abbildung 2-2: Radialverzerrung der Bildkoordinaten

Quelle [SCHIM99], Seite 20

Aus dem hier dargestellten Sachverhalt folgt, dass die Kamerakoordinaten in (x_k, y_k, z_k) unverzerrte Bildkoordinaten (x_u, y_u) überführt werden müssen. Hierbei wird von einer bekannten Brennweite f_k ausgegangen. Die Radialverzerrung kann als stetige Funktion beschrieben werden. Diese können nach dem Approximationssatz von Weierstraß durch gemischte Polynome der Form $p: k_0x^0 + k_1x^1 + ... + k_nx^n$ beliebig genau angenähert werden kann.

Da die meisten Linsen rotationssymmetrisch geschliffen sind, müssen nur Polynome mit ausschließlich geraden Potenzen berücksichtigt werden, wobei die Praxis zeigt, dass Polynome vierten Grades ausreichen. Die verzerrten Bildkoordinaten (x_v, y_v) können in die unverzerrten Koordinaten (x_u, y_u) durch

$$x_v = x_u \bullet (1 + D) \text{ und } y_v = y_u \bullet (1 + D) \qquad (2.1)$$

überführt werden, wobei D die Korrekturfunktion für die radiale Verzerrung ist und folgende Form hat:

$$D = k_1r^2 + k_2r^4 + ... + k_nr^{2n} \text{ mit } r=(x^2_u+y^2_u)^{1/2} \quad (2.2)$$

Die Koeffizienten werden bestimmt, indem (2.1) nach x_u bzw. y_u aufgelöst werden und die resultierenden nichtlinearen Gleichungssysteme iterativ approximiert werden.

Zusätzlich können verzerrt aufgenommene Bildspeicherkoordinaten in Koordinaten der Kalibrierungsebene umgerechnet werden (angegeben über den Winkel vom Projektionszentrum und die Bildhauptverschiebungswerte), was geringe Verschiebungen und Verdrehungen bei der Aufnahme kompensiert. Ein weiteres Verfahren ist die Kalibrierung nach Tsai, bei welchem ein Gradientenabstiegsverfahren verwendet wird. Hierbei soll der Fehler (in diesem Fall die euklidische Distanz zwischen gemessenen und festgelegten, unverzerrten Koordinaten) minimiert werden. Um dieses Ziel zu erreichen wird iterativ eine rotatorische Korrektur und eine Korrektur der Bildhaupt-verschiebung durchgeführt. Dies wird solange durchgeführt, bis keine entscheidende Verbesserung mehr erreicht wird.

Eine genauere Darstellung der angesprochenen Kalibrierungsverfahren würde den Rahmen dieser Arbeit sprengen und wird daher hier nicht durchgeführt.(Sie erfolgt in [SCHIM99] Seite 20ff.) Zur Illustration des Effektes der obigen Verfahren folgt auf der nächsten Seite eine Abbildung, die die Zustände vor Beginn der Kalibrierung, sowie nach 2000, 6000 und 10000 Iterationen zeigt.

Mit dieser Abbildung soll die Erläuterung der Vorverarbeitung abgeschlossen werden.

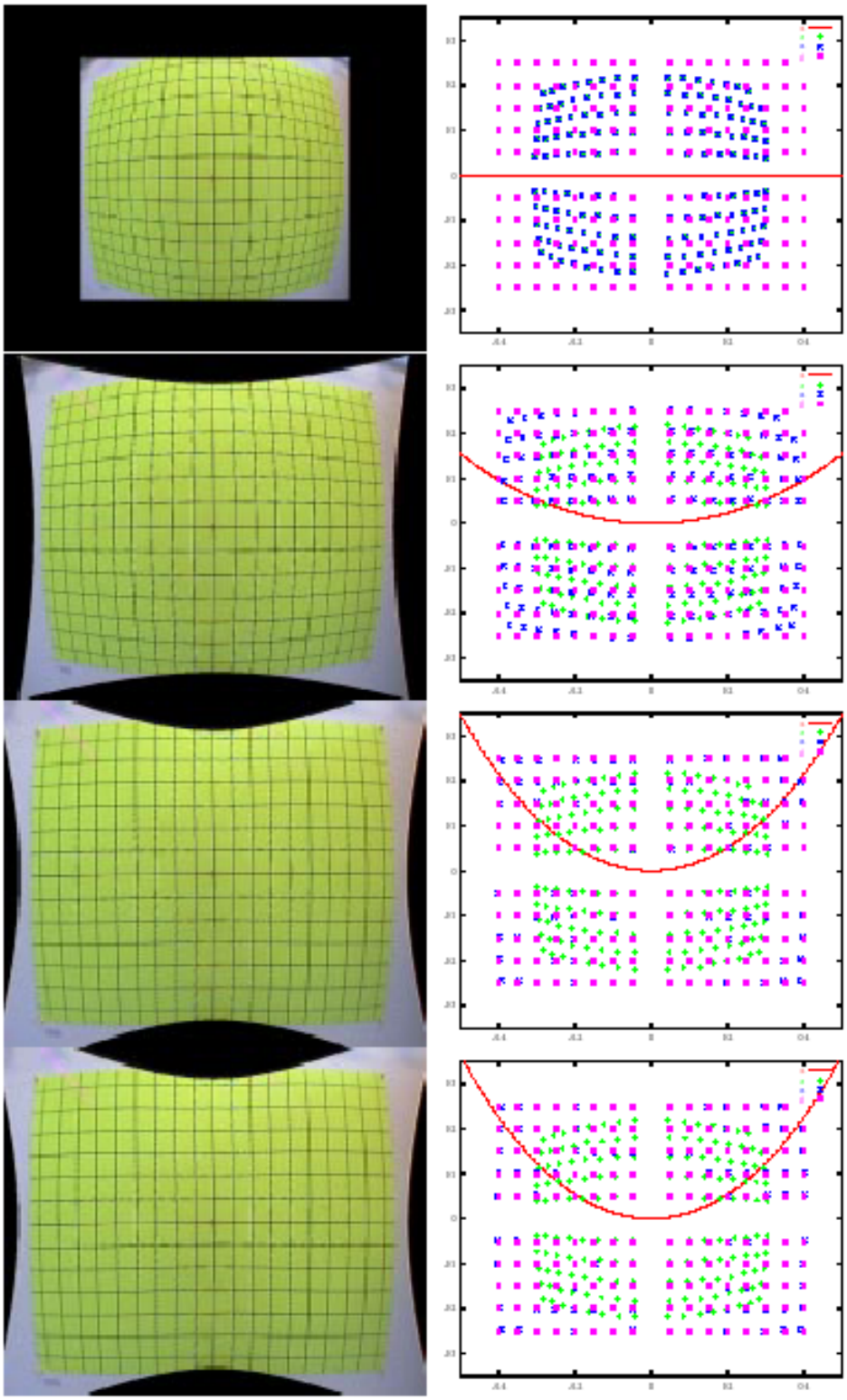


Abbildung 2-3: Kalibrierungsvorgang einer 3,8mm Linse

Quelle [SCHIM99], Seite 27

3 Feature-Extraktion

Die Feature-Extraktion soll dazu dienen, sämtliche markanten Positionen in einem Bild zu entdecken. Als markante Bildpunkte gelten vor allem:

- Auffällige Punkte (in Farbe und/oder Kontrast)
- Anfangs- und Endpunkte von Kanten
- Eckpunkte von Flächen

Bei diesen Punkten kann davon ausgegangen werden, dass sie mit einer hohen Wahrscheinlichkeit wieder erkannt werden können, was eine Voraussetzung für das spätere Feature-Tracking ('Verfolgen') darstellt.

3.1 Der Moravec-Operator

Als grundlegende Methode für die Feature-Extraktion wird in dieser Arbeit der sogenannte Moravec-Operator verwendet. Grundlage des 1977 von Paul Moravec definierten Operators ist die Auswertung eines *n* x *n* –Faltungskerns (Bildausschnitt) auf die dort vertretenen Grauwerte. Hierbei werden die Varianzen der Grautöne entlang der vier Hauptrichtungen (Horizontale, Vertikale und die zwei Hauptdiagonalen) des Faltungskerns betrachtet. Die Varianzen werden ermittelt, indem die Grauwertdifferenzen entlang der Hauptrichtungen summiert werden. Die Formeln für diese Berechnung lauten wie folgt:

$$sum_{vert}(x,y) = \sum_{r,s} |p_{x+r,y+s} - p_{x+r,y+s+1}| \quad (3.1)$$

$$sum_{hor}(x,y) = \sum_{r,s} |p_{x+r,y+s} - p_{x+r+1,y+s}| \quad (3.2)$$

$$sum_{diag1}(x,y) = \sum_{r,s} |p_{x+r,y+s} - p_{x+r+1,y+s+1}| \quad (3.3)$$

$$sum_{diag2}(x,y) = \sum_{r,s} |p_{x+r,y+s} - p_{x+r+1,y+s-1}| \quad (3.4)$$

Die aktuelle Pixelposition wird hier durch (x,y) beschrieben. Liegt eine (im Bild befindliche) Kante entlang einer der vier Hauptrichtungen, so ist die entsprechende Summe null.

Die Verwendung des Operators sollte von einer lokalen Maximumsuche innerhalb des Faltungskerns gefolgt werden. Der Grauwert des Features sollte

dabei nicht nur ein lokales Maximum sein, sondern auch über einem festen Schwellwert liegen, um lokale Grauwertmaxima, die durch spezielle Oberflächenstrukturen (z.B. Marmorierungen) bedingt sind, herauszufiltern.
Eine genauere Darstellung des Moravec-Operators findet sich in [MORAV77]
Die Ergebnisse der Grundform des Moravec-Operators sind nur sehr begrenzt zuverlässig. Aus diesem Grunde werden in der zugrundeliegenden Arbeit Erweiterungen des Operators eingeführt, um dieses Problem zu vermindern. Einige wichtige Erweiterungen sollen im folgenden kurz beschrieben werden.

3.2 Erweiterungen des Moravec-Operators

3.2.1 Die Einführung zusätzlicher Hauptrichtungen

Eine vergleichsweise wenig aufwendige Erweiterung, die zu einer deutlichen Verbesserung des Ergebnisses führt, ist die Einführung zusätzlicher Hauptrichtungen. Für optimale Ergebnisse wären unendlich viele Hauptrichtungen wünschenswert, was allerdings aufgrund der sehr hohen benötigten Rechenzeit nicht verwirklicht wird. Stattdessen werden vier zusätzliche Hauptrichtungen eingeführt, die zwischen den bestehenden platziert werden (siehe Abbildung unten). Auch diese wenigen zusätzlichen Hauptrichtungen führen zu einer erheblichen Verbesserung des Ergebnisses.

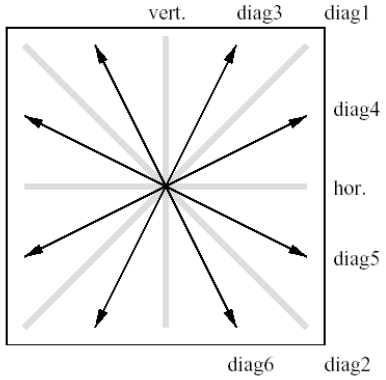


Abbildung 3-1: Moravec-Erweiterung der Hauptrichtungen

Quelle [SCHIM99], Seite 33

3.2.2 Gewichteter Faltungskern

Der Moravec-Operator weist bei größeren Faltungskernen (ab n>7) eine unerwünschte Eigenschaft auf: Die erkannten Feature-Punkte weichen von den eigentlich gewünschten Punkten ab. Insbesondere werden die Eckpunkte

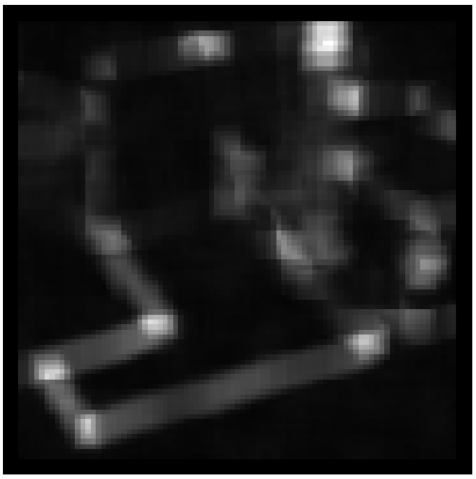
erkannter Objekte zu weit nach innen versetzt. Dies behindert die Erkennung korrekter Kanten und verfälscht die Ergebnisse für die Weiterverarbeitung. Um diesem Phänomen entgegenzuwirken wird ein gewichteter Faltungskern benutzt. Hier wird die Faltungsmatrix in einen inneren und einen äußeren Bereich unterteilt. Der innere Bereich hat dabei die Seitenlänge n/2. Im äußeren Bereich werden weiterhin die in 3.1 beschriebenen Gleichungen zur Berechnung verwendet, wohingegen der innere Kern mit der doppelten (bzw. einer beliebigen, jedoch mehr als einfachen) Gewichtung in das Ergebnis einfließt. Als Folge dieser Gewichtung verschiebt sich der Feature-Mittelpunkt in Richtung der zu erkennenden Eckposition.

3.2.3 Weitere Erweiterungen und Auswirkungen

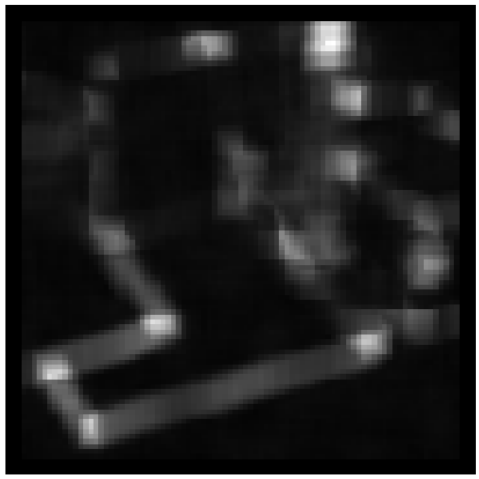
Weitere Möglichkeiten zur Erweiterung des Moravec-Operators sind:

- Erweiterung auf Farbe
- Zusätzliche Schwellwertkontrolle
- Binäre Summenbildung
- Verifikation durch Rotation

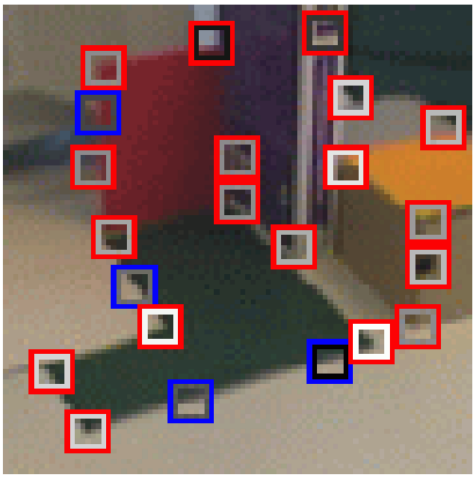
Eine genauere Darstellung dieser Erweiterungen würde allerdings den Rahmen dieser Arbeit sprengen, daher wird an dieser Stelle nur auf deren Existenz hingewiesen (Genaue Erläuterungen zu diesen finden sich in [SCHIM99], Seite 31 ff.). Zum Abschluss der Beschreibung der Erweiterungen des Moravec-Operators sollen dessen Resultate mit denen der Grundform verglichen werden. Die nächste Abbildung stellt die jeweiligen erkannten Features gegenüber. In der linken Spalte finden sich dabei jeweils die Ergebnisse des erweiterten Moravec-Operators, in der rechten die der Grundform. Die obere Zeile legt einen 7x7-Faltungskern zugrunde, die untere einen 9x9-Faltungskern. Aus der Abbildung wird die deutlich bessere Positionierung der vom erweiterten Moravec-Operator erkannten Features ersichtlich, wohingegen bei der Grundform der in 3.2.2 beschriebene Fehler auftritt.



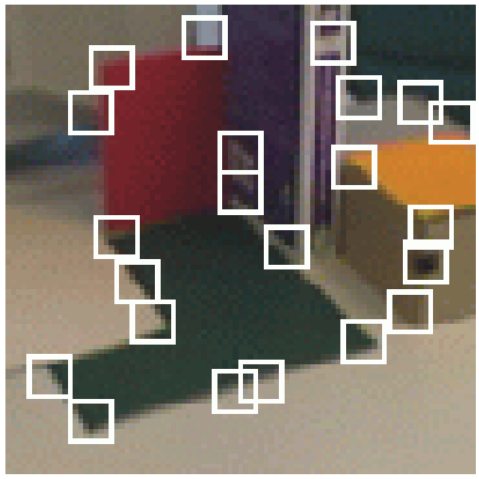
Moravec mit Erweiterungen, 7x7 Matrix



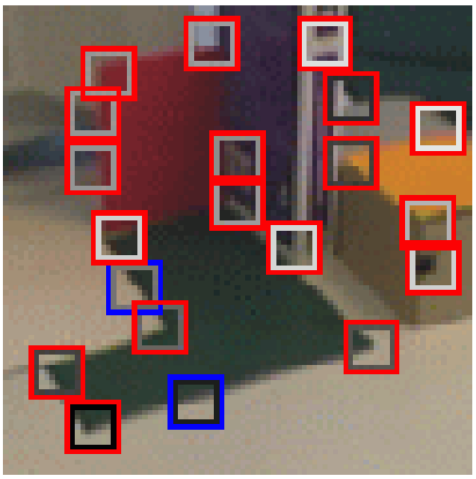
Original Moravec Operator, 7x7 Matrix



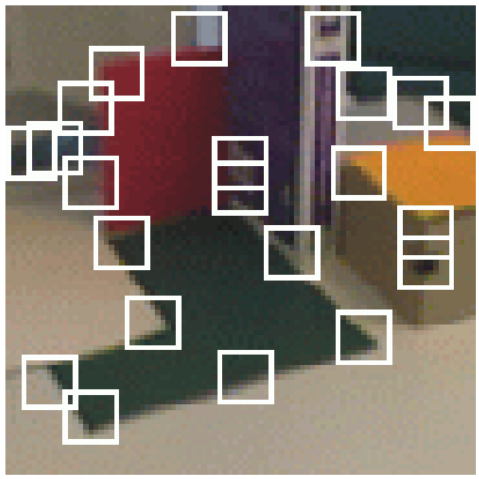
Features nach Maximumsuche im oberen Bild



Features nach Maximumsuche im oberen Bild



wie oben nur mit 9x9-Faltungsmatrizen



Original-Operator mit 9x9-Faltungsmatrizen

Abbildung 3-2: Beispiele für Feature-Extraktion

Quelle [SCHIM99], Seite 41

4 Feature-Tracking

Das Feature-Tracking beschäftigt sich mit dem Verfolgen extrahierter Features innerhalb aufgenommener Bildfolgen. Somit müssen die Feature-Informationen der Einzelbilder logisch untereinander verknüpft werden. Es wird also untersucht, ob Features in Folgebildern mit bereits gefundenen Features übereinstimmen. Wenn dies der Fall ist, bilden diese Features einen sogenannten **Feature-Stream**.

Bei den nachfolgend beschriebenen Verfahren wird von annähernd ruckfreien Filmsequenzen ohne große Sprünge in der Aufnahmezeit und mit relativ gleichmäßigen Kamerabewegungen ausgegangen.

Im folgenden wird nun zunächst das verwendete Verfahren zur **Feature-Korrelation** erläutert, welches dazu dient, die Wahrscheinlichkeit der Übereinstimmung von Feature-Punkten in Folgebildern zu ermitteln. Danach wird das Verfahren des **bidirektionalen Matching** beschrieben, mithilfe dessen gängige Fehlerquellen bei Featurestreams reduziert werden können.

4.1 Feature-Korrelation

Die Feature-Korrelation wird, in der vorliegenden Arbeit, über eine Kombination von **Positionskorrelation** und **Korrelation der lokalen Struktur** ermittelt.

Diese zwei Teilansätze werden kurz dargestellt, bevor auf deren Verknüpfung eingegangen wird.

4.1.1 Positionskorrelation

Bei der Positionskorrelation werden Feature-Positionen zweier Folgebilder verglichen. Danach wird eine Wahrscheinlichkeit ermittelt, dass die Feature-Position im Folgebild mit der aus dem Ursprungsbild vorherbestimmten übereinstimmt. Hierzu wird für jede Feature-Position p im Stream zum Zeitpunkt t(i) (wobei i die Nummer der Bildaufnahme ist) eine voraussichtliche Position zum Zeitpunkt t(i+1) bestimmt. Die voraussichtliche Position p^* ergibt sich aus p und einem gerichteten Geschwindigkeitsterm $\vec{v}$. Der Geschwindigkeitsterm ergibt sich dabei aus einer Art gleitenden Durchschnittsbildung bereits korrelierter Positionen. Der so vorbestimmte Punkt p^* muss nun mit allen Punkten x_i des Folgebildes verglichen werden.

Die euklidische Distanz $dist(x_i,p^*):=|| x_i - p^*||_2$ wird daraufhin über eine Bewertungsfunktion abgebildet und es wird damit eine Wahrscheinlichkeit W_p ermittelt, dass p und p^* übereinstimmen.

4.1.2 Korrelation der lokalen Struktur

Ein weiteres Kriterium für Übereinstimmung von Features kann über den Vergleich ihrer visuellen Struktur gewonnen werden. Nun wird hier ein n x n-Faltungskern, der den Feature-Punkt im Ausgangsbild umgibt, betrachtet und mit dem entsprechenden Faltungskern um einen Feature-Punkt im Folgebild verglichen. Nun werden die Grauwerte jeweils entsprechender Positionen in den jeweiligen Faltungskernen paarweise multipliziert. Diese Produkte werden summiert und durch die Summe ihrer Quadrate geteilt. Als Funktion für die Übereinstimmungswahrscheinlichkeit ergibt sich somit:

$$W_V = (\sum_{r,s} a_{x+r,y+s} \bullet b_{x+r,y+s}) / (\sum_{r,s} a^2_{x+r,y+s} + \sum_{r,s} b^2_{x+r,y+s})$$

Hierbei sind a_x bzw. b_x Punkte innerhalb des Ausgangs- bzw. Folge-Faltungskerns, dessen Mittelpunkt durch die Position (x,y) gegeben wird.

4.1.3 Kombination der beiden Ansätze

Zur endgültigen Bewertung der Übereinstimmungswahrscheinlichkeit zweier Feature-Punkte in unterschiedlichen Bildern müssen die, aus den oben beschriebenen Ansätzen gewonnenen, Einzelwahrscheinlichkeiten miteinander verknüpft werden. Hierzu wird eine gewichtete Summe gebildet in der Form:

$$W = \alpha \bullet W_p + (1 - \alpha) \bullet W_V \; ; \; 0 \leq \alpha \leq 1$$

Über die Festlegung des Parameters α kann somit der Einfluss von Positionskorrelation und visueller Korrelation reguliert werden.

4.2 Bidirektionales Matching

Mit Bidirektionalem Matching wird ein Verfahren bezeichnet, welches (unter Verwendung der in 4.1 beschriebenen Methoden zur Bestimmung der Übereinstimmungswahrscheinlichkeit) Feature-Punkte eines Streams in zwei Richtungen (vorwärts und rückwärts) auf Übereinstimmung überprüft.

Hiermit können Spezialfälle (und damit mögliche Fehlerquellen) abgefangen werden:

- Stream-Fortsetzung (‚match')
- Stream-Ende (‚end of track')
- Geister (‚ghosts')
- Entstehung eines neuen Streams (‚new stream')
- Rauschen (‚single peak')

Dabei liegt der Vorgehensweise der im folgenden Struktogramm dargestellte Algorithmus zugrunde:

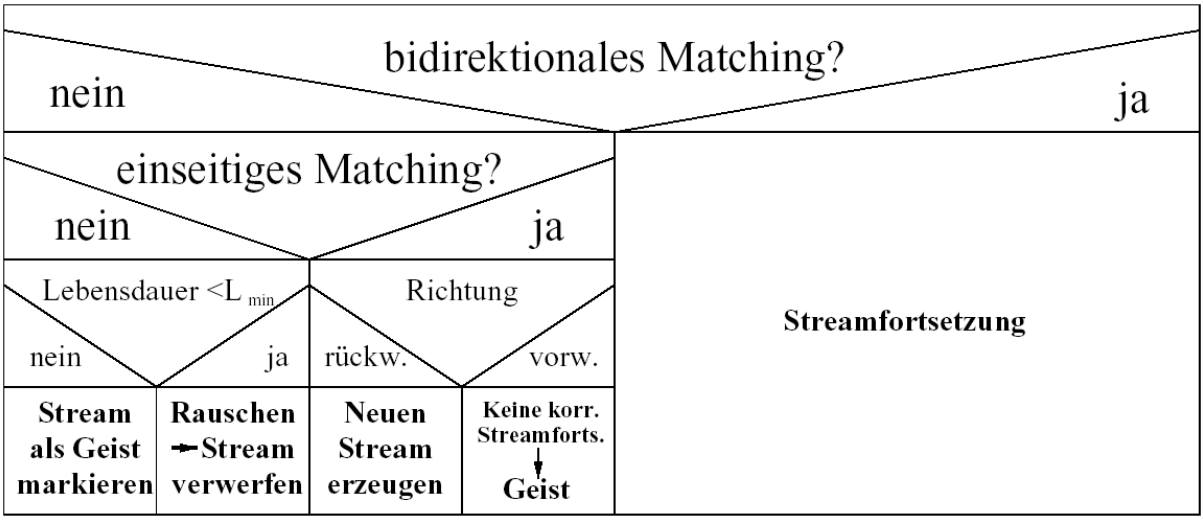


Abbildung 4-1: Ablauf der Feature-Kategorisierung

Quelle [SCHIM99], Seite 50

Nun soll auf die einzelnen Spezialfälle noch näher eingegangen werden.

4.2.1 Stream-Fortsetzung

Eine Stream-Fortsetzung ist gegeben, wenn bei Features eine gegenseitige Korrelationsübereinstimmung (=bidirektionales Matching) festgestellt wird.
Formal ausgedrückt müssen dann folgende Bedingungen gelten:

$$W(p_{i,t}, p_{j,t+1}) = \max_{p_{k,t+1} \in P_{t+1}} W(p_{i,t}, p_{k,t+1}) \qquad (4.1)$$

sowie

$$W(p_{i,t}, p_{j,t+1}) = \max_{p_{k,t} \in P_t} W(p_{k,t}, p_{j,t+1}) \qquad (4.2)$$

Um eine möglichst große Anzahl fortgesetzter Streams zu erreichen, werden iterativ Übereinstimmungen zwischen Features gesucht, bis keine mehr zu erreichen sind. Nicht zuordenbare Punkte werden als ‚new streams' oder

‚ghosts' gespeichert oder ganz verworfen. In der folgenden Abbildung ist die Stream-Fortsetzung durch bidirektionales Matching noch einmal grafisch dargestellt:

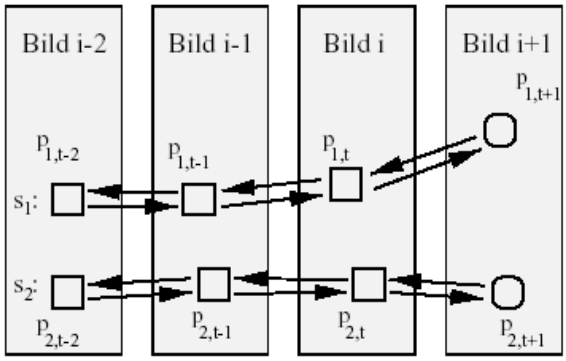


Abbildung 4-2: Stream-Fortsetzung durch bidirektionales Matching

Quelle: [SCHIM99], Seite 50

4.2.2 Geister

Geister entstehen, wenn Features nicht entsprechend der Bedingungen (4.1) und (4.2) eingeordnet werden können. Der Grund hierfür kann sein, dass die Features kurzzeitig perspektivisch verdeckt sind oder die Sichtbarkeit durch Rauschen eingeschränkt wurde.

Für Geister gilt also unverändert (4.1), aber anstatt (4.2) gilt:

$$W(p_{i,t}, p_{j,t+1}) \neq \max_{p_{k,t} \in P_t} W(p_{k,t}, p_{j,t+1}) \qquad (4.3)$$

(In einem weiteren Spezialfall gilt (4.2), das Maximum liegt aber unter einem vorher festgelegten Schwellwert.)

Anders ausgedrückt kann zwar im ‚neuen' Bild kein Feature mit dem entsprechenden Feature im ‚alten' Bild in Übereinstimmung gebracht werden, aber eine voraussichtliche Position des Features im neuen Bild kann geschätzt werden. Auch zur Geister-Entstehung folgt eine grafische Darstellung:

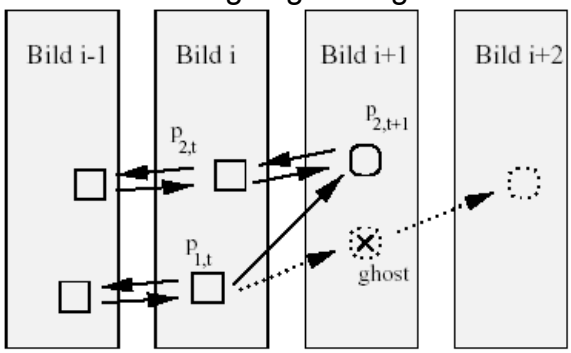


Abbildung 4-3: Entstehung von Geistern

Quelle [SCHIM99], Seite 51

4.2.3 Ende eines Streams

Ein Feature-Stream gilt als beendet wenn er n-mal hintereinander nicht ‚gematcht' werden konnte.

4.2.4 Stream-Entstehung

Ein neuer Stream entsteht, wenn neue Punkte nicht mehr mit alten in Übereinstimmung gebracht werden können. Die folgende Abbildung visualisiert den Vorgang bei der Stream-Entstehung:

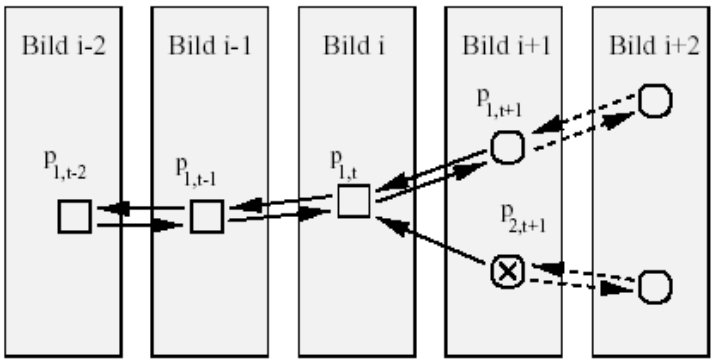


Abbildung 4-4: Entstehung eines neuen Streams

Quelle: [SCHIM99], Seite 52

4.2.5 Rauschen und perspektivische Verdeckung

Erkannte Features können grundsätzlich ihre Ursache auch im Rauschen bzw. in perspektivischen Phänomenen, wie partielle Verdeckung, haben. Diese Arten von Features können allerdings über ihre (kurze) Lebensdauer identifiziert und danach eliminiert werden. Problematisch sind hier nur länger anhaltende Verdeckungen, die in diesem Schritt nicht erkannt werden können.

5 Rekonstruktion der 3D-Information

Im folgenden wird die Umwandlung erkannter Feature-Positionen und –Bewegungen (angegeben in zweidimensionalen Informationen auf der Bildebene) in 3D-Informationen beschrieben. Hierzu werden zunächst die zugrundeliegenden Koordinatensysteme eingeführt, danach wird auf die eigentliche Rekonstruktion, bestehend aus Strahlengangberechnung, Schnittpunktberechnung und Objektpunktbestimmung eingegangen.

5.1 Koordinatensysteme

Zur 3D-Rekonstruktion werden hier 5 Koordinatensysteme verwendet:

- Das **Welt-Koordinatensystem W**: Es steht für die Umgebung des Roboters und wird bei dessen Initialisierung festgelegt und danach nicht mehr verändert.
- Das **Roboter-Koordinatensystem R**: Es hat seinen Ursprung im Mittelpunkt des Roboters in Bodenhöhe. Die x-Achse zeigt hier in Fahrtrichtung, die y-Achse horizontal orthogonal dazu, die z-Achse gibt die (konstante) Höhe des Roboters wieder. Die Fahrtrichtung bezüglich W wird durch den Winkel ϕ_{base} gegeben.
- Das **Pan-Tilt-Koordinatensystem PT**: Sein Ursprung liegt im Schnittpunkt der Schwenk- und Neigeachse der Kamera. Seine z-Achse entspricht der Pan-Achse, die y-Achse der Tilt-Achse, die x-Achse liegt orthogonal zu den beiden anderen. Pan-Winkel ψ und Tilt-Winkel θ liegen jeweils bei 0, wenn die Kamera horizontal in Fahrtrichtung zeigt.
- Das **Kamerakoordinatensystem K** hat den Ursprung im Projektionszentrum der Kameralinse. Die z-Achse entspricht dabei der Projektionsgeraden durch den Ursprung. Die x- und y-Koordinaten entsprechen den projizierten Bildkoordinaten
- Das **Bildebenenkoordinatensystem BE** liegt parallel zur x-y-Ebene der Kamera, ist allerdings um den Abstand z_0 in z-Richtung von der Kamera verschoben (BE hat nur x- und y-Achsen).

In der folgenden Abbildung werden die eben besprochenen Koordinatensysteme noch einmal grafisch dargestellt.

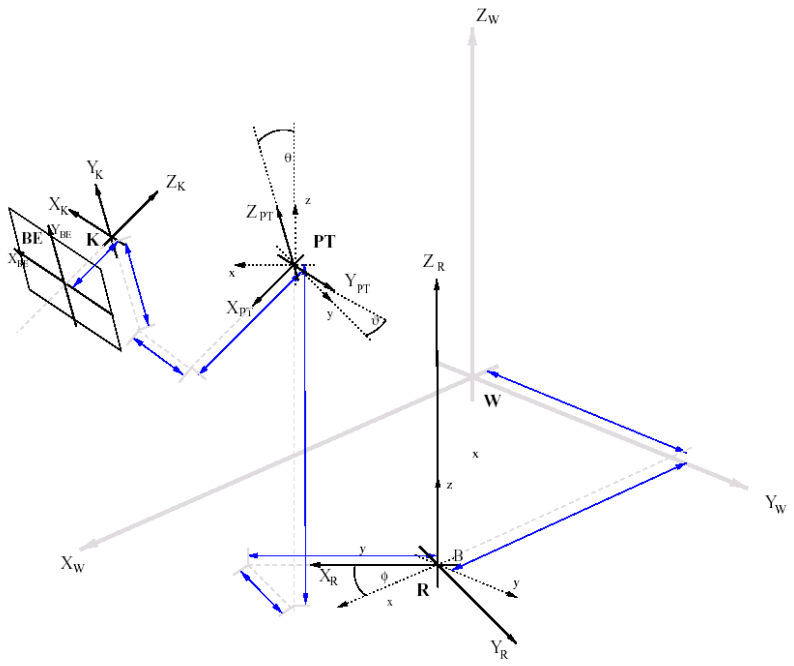


Abbildung 5-1: Welt-, Roboter-, Pan-Tilt-, Kamera- und Bildebenen-Koordinatensystem

Quelle: [SCHIM99], Seite 55

Zur Berechnung von Welt-3D-Informationen ist es u.a. nötig die Koordinaten bezüglich des Kamera-Koordinatensystems in Welt-Koordinaten zu überführen. Dies geschieht in mehreren Schritten:

1. Die Überführung von Kamerakoordinaten in Pan-Tilt-Koordinaten. Dies geschieht durch die Translation $p_{PT} = {}^{PT}T_K \bullet p_k$ mit

$$ {}^{PT}T_K = \begin{bmatrix} 1 & 0 & 0 & x_k \\ 0 & 1 & 0 & y_k \\ 0 & 0 & 1 & z_k \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

2. Die Überführung der Pan-Tilt-Koordinaten in Roboterkoordinaten. Hierzu verwendet man die folgende Translation: $p_R = {}^{R}T_{PT} \bullet p_{PT}$, wobei gilt:

$$ {}^{R}T_{PT} = \begin{bmatrix} \cos(\psi)\cos(\theta) & -\sin(\psi) & \cos(\psi)\sin(\theta) & x_{PT} \\ \sin(\psi)\cos(\theta) & \cos(\psi) & \sin(\psi)\sin(\theta) & y_{PT} \\ \sin(\theta) & 0 & \cos(\theta) & z_{PT} \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

3. Die Umwandlung der Roboterkoordinaten in Weltkoordinaten. Die dazugehörige Translation lautet $p_W = {}^{W}T_R \bullet p_R$. . ${}^{W}T_R$ wird auf der folgenden Seite noch dargestellt.

$$ {}^{W}T_R = \begin{bmatrix} \cos(\phi_{base}) & -\sin(\phi_{base}) & 0 & x_{base} \\ \sin(\phi_{base}) & \cos(\phi_{base}) & 0 & y_{base} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Mit diesem letzten Schritt ist die Umrechnung der Kamerakoordinaten in Weltkoordinaten abgeschlossen.

5.2 Ermittlung der Strahlengänge

Nachdem im vorangegangenen Abschnitt dargestellt wurde, wie 3D-Koordinaten, die relativ zur Kamera angegeben sind, in 3D-Weltkoordinaten überführt werden können, muss jetzt noch erläutert werden, wie (zweidimensionale) Bildebenen-Positionen erkannter Features in dreidimensionale Koordinaten relativ zur Kamera umgewandelt werden können. Der erste Schritt hierbei ist die Ermittlung der Strahlengänge.

Hierbei wird auf das (in 2.2 beschriebene) Kalibrierungsverfahren zurückgegriffen, bei dem die radiale Verzeichnung direkt in den Koordinaten der Kalibrierungsebene berechnet wird, wobei deren Abstand z_0 zum Projektionszentrum bekannt ist. Die Sichtstrahlen werden durch den Ortsvektor q_k und den Richtungsvektor r_k bezüglich des Kamerakoordinatensystems K definiert. Diese haben die Form:

$$ \vec{q_k} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} $$

und

$$ \vec{r_k} = \begin{pmatrix} z_0 \\ p_x \bullet (1+k_1 r^2+\ldots) \\ p_y \bullet (1+k_1 r^2+\ldots) \end{pmatrix} $$

Hierbei geben p_x und p_y jeweils die Feature-Koordinaten auf der Bildebene an und werden mit der Korrekturfunktion (siehe Gleichung 2.2) multipliziert.

Der Ortsvektor kann über den in 5.1 Weg in Weltkoordinaten überführt werden. Beim Richtungsvektor wird anstatt der kompletten Matrizen nur jeweils der

Rotationsanteil (die 3x3-Matrizen die jeweils von den 3 linken Spalten und 3 oberen Zeilen der in 5.1 dargestellten Matrizen gebildet werden) zur Transformation verwendet, um ihn in Weltkoordinaten zu überführen.
Der Kamerakoordinatenursprung O_k sowie die Projektionsrichtung kommen direkt vom Pan-Tilt-Server.

5.3 Schnittpunktbestimmung

Als nächster Schritt zur Ermittlung der 3D-Koordinaten eines erkannten Features muss der Schnittpunkt aller zu diesem Feature führenden Sichtstrahlen bestimmt werden. Dies ist nicht unproblematisch, da sich die berechneten Strahlen zumeist nicht genau schneiden sondern windschief zueinander stehen. Somit wird der Mittelpunkt der kürzesten Verbindung zwischen den Strahlen verwendet. Zur Ermittlung der kürzesten Verbindung werden die Strahlen wie Geraden behandelt (daher muss nach Bestimmung des ‚Schnittpunkts' noch getestet werden, ob der Punkt auch vor der Projektionsebene liegt!). Diese haben die Form:
g_1: $x = \vec{a} + s\,\vec{u}$ sowie g_2: $x = \vec{b} + t\,\vec{v}$
Der Vektor *w* soll die kürzeste Verbindung zwischen den Geraden sein, d.h. er muss zu beiden orthogonal stehen und beide schneiden. Formal bedeutet dies:
$\vec{u} \bullet \vec{w} = 0$
$\vec{v} \bullet \vec{w} = 0$
$\vec{w} = \vec{a} - \vec{b} + s\,\vec{u} - t\,\vec{v}$
Nach Auflösen der Gleichungen erhält man für den ‚Schnittpunkt':
$0S = (\vec{a} + s\,\vec{u} + \vec{b} + t\,\vec{v})/2$
Die folgende Abbildung visualisiert den Vorgang:

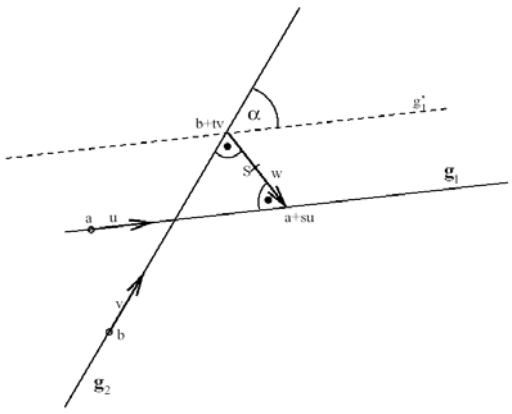


Abbildung 5-2: Räumliche Schnittpunktbestimmung zweier Geraden

Im weiteren muss der ‚Schnittwinkel' zwischen $g_1$und g_2 betrachtet werden, da bei spitzen Schnittwinkeln ein enormer Drift des Schnittpunkts entstehen kann. Daher muss die Güte eines Schnitts bewertet werden. Hierbei wird zum einen die Distanz $|\vec{w}|$ sowie, mithilfe eines Winkelkorrektheitsmaßes, der Schnittwinkel α (zwischen g_1 und g_2) bewertet.

Dies ergibt

$v_d = 1 / (1 + \beta \bullet |\vec{w}|)$

sowie

$v_\alpha = sin^2\ \alpha$

Und damit die Gesamtwahrscheinlichkeit: $v = v_d \bullet v_\alpha$

Diese gibt die Wahrscheinlichkeit an, dass es sich bei dem ermittelten Punkt um einen korrekten Schnittpunkt handelt.

5.4 Objektpunktbestimmung

Im letzten Schritt der Bestimmung der 3D-Position eines Features muss nun aus der Menge der gemäß 5.3 für einen Feature-Stream ermittelten Schnittpunkte der Sichtstrahlen der optimale Punkt herausgegriffen werden. Hierzu wird aus der Schnittpunktwolke unter Miteinbeziehung der für die einzelnen Punkte ermittelten Gesamtwahrscheinlichkeiten *v* ein geometrischer Mittelpunkt ermittelt.

Somit gilt für den Objektpunkt:

$$P_{3D} = (\sum_{i,j} v_{ij}\ S_{ij}) / \sum_{i,j} v_{ij}$$

6 Visualisierung der 3D-Information

Nachdem im letzten Kapitel dargestellt wurde, mit welchen Methoden die 3D-Positionen einzelner Punkte bestimmt werden, muss nun zwischen diesen Punkten ein Zusammenhang mittels Verbindungslinien oder –flächen erzeugt werden, um eine Visualisierung des rekonstruierten Raumes zu ermöglichen.

6.1 Kanten- und Liniendetektion

Gemäß [ENCAR98] lautet die Definition für eine Kante:
„Eine Kante ist eine Struktur, ein Pixel breit, im Zentrum des Grauwertgefälles zwischen zwei aneinandergrenzenden Regionen, die sich in ihren Graustufen hinreichend unterscheiden."

Feature-Streams zwischen denen auf mindestens δ Bildern eine Kante als Verbindungslinie zu erkennen ist, werden im folgenden als Nachbarn bezeichnet. Für die Feststellung einer solchen Nachbarschaftsbeziehung sind Operatoren nötig, die eine vektorisierte Form für die extrahierten Kanten liefert.
Im vorliegenden Fall kann dabei der Umstand ausgenutzt werden, dass über die extrahierten Feature-Punkte die möglichen Anfangs- und Endpunkte bereits gegeben sind. Es erfolgt hier ein Linientest mithilfe eines modifizierten Bresenham-Algorithmus (siehe [BRESE65]) zum Linienzeichnen.
Hierzu wird zunächst aus den drei RGB-Farbkanälen des Originalbildes mittels des Canny-Operators (siehe [ENCAR98]) ein Gradientenbild aus dem Maximum der einzelnen Gradientenbilder erzeugt. Dieses Bild wird durch die Anwendung des Gauß-Operators ausgedehnt. Nun ergibt sich ein ausgedehntes Kantenpotenzial. Die Punkte dieser potenziellen Kanten werden nun zunächst daraufhin untersucht, ob sie einen geforderten Grauwert übersteigen. Danach wird betrachtet wie hoch der Anteil der Punkte der betrachteten Kante ist, die diese Grauwert-Bedingung erfüllen. Liegt der Anteil über einem geforderten Schwellwert, wird der Kandidat als Linie bezeichnet. Abschließend wird die ‚Qualität' der Linie ermittelt, indem geprüft wird, auf wie vielen Bildern die Linie zu erkennen ist (je größer die Anzahl der Bilder, um so höher die entsprechende Qualität). Die folgende Abbildung gibt noch einmal einen Überblick über das gesamte Verfahren:

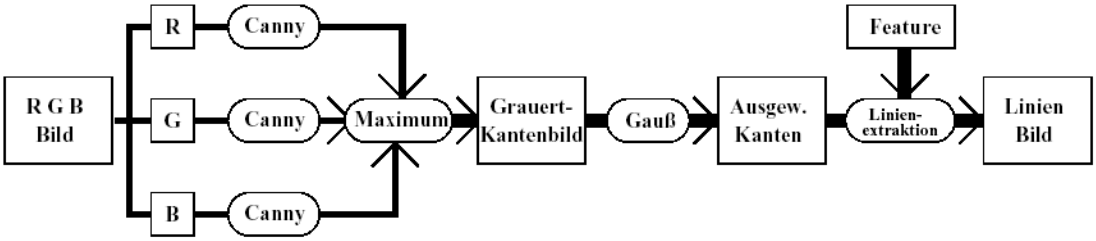


Abbildung 6-1: Ablauf der Linienextraktion

Quelle: [SCHIM99], Seite 64

Bevor es zu einer Weiterverarbeitung kommen kann, müssen noch ‚problematische' Linien eliminiert werden, die sonst im Folgenden zu Fehldarstellungen führen könnten. Hierzu werden alle Linien einem Schnittpunkttest und einem Ähnlichkeitstest unterzogen (siehe [SCHIM99]). Fällt einer der beiden Tests positiv aus, wird die Linie verworfen.

6.2 Dreidimensionale Darstellung

Durch die im vorangegangenen Abschnitt dargestellte Linienextraktion wurde ein dreidimensionales Gitternetz erzeugt. Weiterhin gibt es jedoch noch Punkte, die völlig frei im Raum stehen bzw., nur mit einem Nachbarn verbunden sind. Um ein komplettes Netz über die vorhandenen Objekte legen zu können wird daher zunächst eine Triangulierung im zweidimensionalen Raum des projizierten Bildes vorgenommen. Die dabei erhaltenen Dreiecke müssen danach allerdings noch nachbearbeitet und in realistischere Konstellationen gestellt werden.

6.2.1 Triangulierung

In der vorliegenden Arbeit wurde der Delaunay-Algorithmus als Triangulierungsmethode gewählt. Grund hierfür war die Eigenschaft dieser Methode, möglichst großflächige Dreiecke zu liefern. Eine genaue Darstellung der Methode würde den Rahmen dieser Arbeit sprengen, daher wird an dieser Stelle auf [ENCAR98] verwiesen, wo eine ausführliche Beschreibung erfolgt.

6.2.2 Flipping-Algorithmus

Die im vorangegangenen Abschnitt vorgenommene Triangulierung ist nicht grundsätzlich deckungsgleich mit den wirklichen Kanten. Somit muss versucht

werden, die erhaltene Triangulierung mit diesen optimal in Übereinstimmung zu bringen. Hierzu wird auf das Ergebnis, des in 6.1 durchgeführten Linientests, zurückgegriffen. Nun wird jedes Dreieck (der Triangulierung) mit allen im Linientest festgestellten Kanten auf mögliche Schnittpunkte getestet. Liegt ein Schnittpunkt vor (in der folgenden Abbildung z.B. zwischen ΔABC bzw. ΔACD und der Kante *BD*), so stimmt die Triangulierung hier nicht mit der Realität überein. Durch ‚Flippen' der Dreieckskante AC zur Kante BD entstehen zwei neue Dreiecke ΔABD und ΔBCD, wobei *BD* nun mit der Realität übereinstimmt. Dieser Schnitttest zwischen Kantenmenge und Dreiecksmenge ist iterativ zu wiederholen, bis keine Übereinstimmungen mehr zu finden sind.

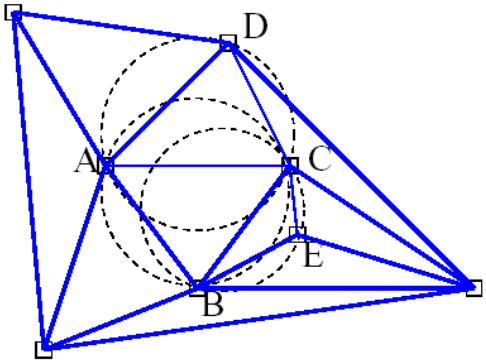


Abbildung 6-2: Delaunay-Triangulierung

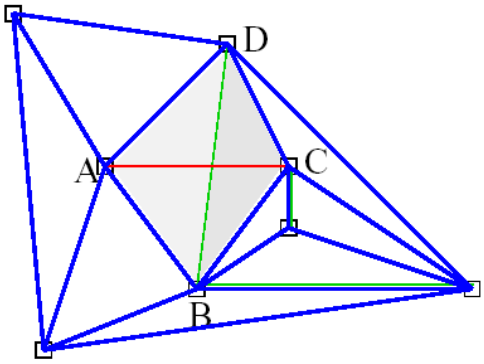


Abbildung 6-3: Flipping-Algorithmus-Anpassung

Quelle: jeweils [SCHIM99], Seite 71

Die dann resultierende Triangulierung stimmt sehr viel besser mit der Realität überein, ist allerdings selten absolut korrekt, da Probleme bei der Projektion von 3D auf 2D bzw. durch Rauschen o.ä. bestehen.

Die (mittels Triangulierung und Flipping) aufbereiteten Visualisierungsinformationen können nun als Grundlage für eine Tiefenkarte oder eine direkte 3D-Visualisierung dienen. Bei der Tiefenkarte wird jedem Dreieck ein Grauwert zugeordnet, um seine Höhe zu visualisieren. (Diese Darstellung wird ähnlich in Landkarten verwendet.)
Die direkte 3D-Visualisierung wandelt die gefundenen Informationen in 3D-Grafikformate um. In der zugrundeliegenden Arbeit wurde hierfür VRML (Virtual Reality Modeling Language) verwendet. Grund hierfür war vor allem die Aktualität und Verbreitung dieses Formats. Die generierte VRML-Datei wirkt dabei als eine Art Mittler zwischen Rekonstruktion und Visualisierung.

7 Zusammenfassung und Ausblick

Ziel der Arbeit war die Darstellung von Algorithmen und Methoden zur dreidimensionalen Rekonstruktion von Innenräumen mittels Kameraaufnahmen von einem mobilen Roboter aus. Hierbei sollten die, durch die Bildaufnahmen verlorengegangenen, Tiefeninformationen wiederhergestellt werden.
Im ersten Schritt ist dabei eine Feature-Extraktion durchzuführen. Hierzu wurde ein modifizierter Moravec-Operator verwendet, der sehr zuverlässige Feature-Punkte liefert.
Anschließend mussten die gefundenen Features im Zeitablauf ‚getrackt' werden. Dabei wurde eine kombinierte Analyse mit visuellen und Bewegungselementen genutzt, die ebenfalls gute Ergebnisse liefert.
Über diese Feature-Bewegungen und die Konstruktion der zugehörigen Sichtstrahlen wurden die 3D-Positionen der Features bestimmt (Genauigkeit 5-10cm). Diese Punktinformationen können durch Kantenextraktion, Triangulierung und Flipping auf eine dreidimensionale Darstellung mit Flächen ausgedehnt werden. Hieraus kann eine Tiefenkarte bzw. eine 3D-Szene generiert werden.
Die in der dargestellten Arbeit verwendeten Algorithmen sind sehr stark auf Genauigkeit und weniger auf schnelle Verarbeitung ausgelegt. Durch eine stärkere Parallelisierung wären Geschwindigkeitszugewinne denkbar. Eventuell könnte dies zu einer Echtzeitverarbeitung führen, was völlig neue Anwendungsgebiete, wie z.B. Roboternavigation, ermöglichen würde.

8 Weiterführende Literatur

BRESE65 Bresenham, J.E.: Algorithm for Computer Control of a Digital Plotter.In: IBM System Journal, Band 4, Nr. 1, S. 25-30, 1965.

ENCAR98 Encarnaçâo, Straßer, W., Klein, R.: Graphische Datenverarbeitung II. Oldenbourg, 1998.

MORAV77 Moravec, H. P.: Towards Automatic Visual Obstacle Avoidance. In: 5.
Int. Joint Conf. Artificial Intell., Band 2, S. 584, 1977.

SCHIM99 Schimmel, Oliver, Dreidimensionale Szenenmodellierung durch monokulare Exploration mit einem mobilen Roboter, Tübingen, 1999