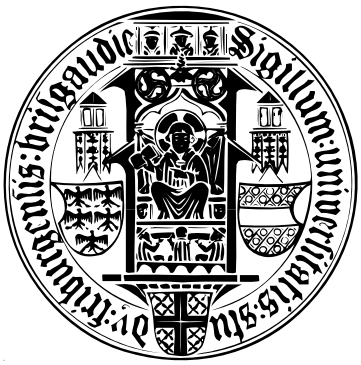
ALBERT-LUDWIGS-UNIVERSITÄT
FREIBURG
INSTITUT FÜR INFORMATIK

Lehrstuhl für Mustererkennung und Bildverarbeitung
Prof. Dr. Hans Burkhardt



Segmentierung und Lagebestimmung von 3D Objekten in Laser Range Scans

Diplomarbeit

Philip Schroll

November 2007 – April 2008

Erklärung

Hiermit erkläre ich, daß die vorliegende Arbeit von mir selbständig und nur unter Verwendung der aufgeführten Hilfsmittel erstellt wurde. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, sind als solche gekennzeichnet. Sämtliche Abbildungen wurden von mir persöhnlich erstellt, soweit keine Quellenangaben vorhanden sind. Die Diplomarbeit wurde nicht, auch nicht auszugsweise, bereits für eine andere Prüfung angefertigt.

Freiburg, den 1. Mai 2008

ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG
INSTITUT FÜR INFORMATIK
Lehrstuhl für Mustererkennung und Bildverarbeitung
Prof. Dr.-Ing. Hans Burkhardt

1. November 2007

Aufgabenstellung für die Diplomarbeit
von Herrn Philip Schroll.

Segmentierung und Lagebestimmung von 3D Objekten in Laser Range Scans

Die automatische Analyse dreidimensionaler Szenen, insbesondere die Segmentierung, Lagebestimmung und Klassifikation in einer Szene enthaltenen 3D Objekte, ist für eine Vielzahl von industriellen Anwendungen von großer Bedeutung.

In dieser Diplomarbeit sollen Szenen mit mehren Objekten mittels "Laser Range Scans" aufgenommen und automatisch analysiert werden. Dabei sollen zunächst geometrisch einfache Objekte detektiert, segmentiert und ihre Lage in der Szene bestimmt werden. Optional könnten im Laufe der Arbeit dann auch komplexere Objekte betrachtet werden.

Die Aufgabenstellung beinhaltet dabei u.A.:

- Erstellen einer Datenbank von 3D Referenzobjekten (volle 3D Information).
- Erstellen einer Datenbank von Szenen mit Objekt-Teilansichten (ein Tiefen-Scan).
- Entwurf und Implementation geeigneter Algorithmen zur Bildvorverabeitung.
- Segmentierung der Objekte unter Verwendung von assoziativen Markov-Netzwerken.
- Lagebestimmung der Objekte in der Szene.
- Evaluation der Algorithmen.

Literatur:

[1] Anguelov, D. et. Al: *Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data*, CVPR 2005

[2] Taskar B., Chatalbashev V., Koller D.: *Learning associative Markov Networks*, ICML 2004

[3] Taskar B.: *Learning structured prediction models: a large margin approach*, PhD Thesis, Stanford University, 2004

[4] Triebel R., *Three-dimensional Perception for Mobile Robots*, Diessertation Uni Freiburg 2007

Referent:	Prof. Dr.-Ing. H. Burkhardt
Betreuer:	Dipl.-Inf. J. Fehr
Ausgabedatum:	31. 10. 2007
Abgabedatum:	30. 04. 2008
Bearbeitungszeit:	6 Monate

.....

Prof. Dr.-Ing. H. Burkhardt

Danksagung

Mein Dank gilt Prof. Dr.-Ing. H. Burkhardt, der es mir ermöglichte diese Arbeit am Lehrstuhl für Mustererkennung und Bildverarbeitung anzufertigen.

Ich danke meinem Betreuer Janis Fehr für seine Anregungen und Unterstützung, sowie meiner Freundin Carolin Weisser für ihren seelischen Beistand und zuletzt dem Kaffee-Automaten im Aufenthaltsraum für die tägliche Dosis Koffein.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die Segmentierung von Bildern in inhaltlich zusammenhängende Regionen ist ein wichtiges Verfahren in der Mustererkennung. Um die Bedeutung eines Bildes erkennen zu können, müssen die Teilgebiete gefunden werden, welche von Interesse sind. Ein Roboter muss beispielsweise Hindernisse auf seinem Weg erkennen und darauf reagieren können. Aber auch im medizinischen Bereich spart die semi-automatische Segmentierung etwa von Magnet-Resonanz-Tomographie-Aufnahmen den Ärzten kostbare Zeit. Ein Verfahren zur Segmentierung von Bildern stellen Markov-Netze dar.

1.1 Markov-Netze

Markov-Felder (*MRF*) werden mit großem Erfolg für eine Vielzahl von Problemen in der Bilderkennung eingesetzt. Beispiele sind die Bildrestauration, kantenerhaltende Filterung, sowie die Schrift- und Spracherkennung. Im Fall der Schrifterkennung werden bei den meisten Methoden die einzelnen Buchstaben bzw. Wortsequenzen zuerst separiert, bevor sie klassifiziert werden können. Durch den Einsatz von Hidden-Markov-Modellen (*HMM*) können Buchstaben- oder Wortmodelle trainiert werden, wobei bei der Klassifikation über HMMs die untersuchte Sequenz im Netz segmentiert wird und dies nicht explizit vorher geschehen muss.
Die Zustände des Modells sind dann identisch mit den Wörtern eines Lexikons, aus denen die untersuchten Wortfolgen gebildet werden. Die Wahrscheinlichkeiten des Auftauchens eines Wortes in Bezug zu seinen Nachbarn werden somit durch das Markov-Modell kodiert. Wertet man die Wahrscheinlichkeit von Wortfolgen aus, kann daraus die Plausibilität eines Satzes erkannt werden. Im Fall der HMMs ist die Zustandsfolge versteckt und nur die sogenannten Emissionen sind beobachtbar. Somit muss diejenige Zustandsfolge bestimmt werden, die am wahrscheinlichsten eine bestimmte Folge von Emissionen erzeugt (siehe [Fin03]).

Bei der Verarbeitung gesprochener Sprache kann keine Trennung zwischen Segmentierung und Klassifizierung gezogen werden, da die Wortgrenzen nicht akustisch markiert sind. Auch in diesem Fall bieten Markov-Modelle einen eleganten Ausweg. Hierbei entsprechen die Ausgaben des Modells dem akustischen Sprachsignal und die Zustände stellen Laute dar. Durch die Rekonstruktion der internen Zustandsfolge für ein gegebenes Signal läßt sich die Wahrscheinlichkeit zu einer gesprochenen Wortfolge zuordnen.

1.2 Aufbau der Arbeit

Im Rahmen der Diplomarbeit wurde ein Verfahren zur automatischen Segmentierung und Lageerkennung von 3D-Objekten, sowie eine geometrische 3D-Modellierung der Objektoberflächen basierend auf den Laserscanner-Daten entwickelt. Als Scanobjekte dienten überwiegend Pappkartons, die zufällig im Scanbereich verteilt wurden. Bei der Segmentierung von wagerecht liegenden Objekten konnten diese aufgrund der hohen Tiefenwertunterschiede zwischen Objekt und Hintergrund leicht erkannt werden. Lagen die Objekte allerdings schräg zur Z-Achse, fiel eine Kantenerkennung wegen der fließenden Übergänge der schrägen Flächen schwieriger aus. Ein Vorteil der Markov-Modelle besteht darin, dass es sich zwar um einen allgemeinen Modellansatz handelt, dieser jedoch durch die Vorgabe spezifischer Eigenschaften optimal an die Problemstellung angepasst werden kann. Über die Wahl eines adäquaten Nachbarschaftssystem können die Umgebungspunkte eines Pixels mit in die Berechnung einbezogen werden. Deshalb wurde als Segmentierungsalgorithmus Markov-Felder verwendet.
In zweiten Kapitel wird die Apparatur zur Erstellung der Tiefenbilder vorgestellt, sowie der Versuchsaufbau erläutert. Die angewendeten Methoden zur Korrektur der Bildqualität werden vorgestellt, sowie alternative Methoden zur Distanzmessung mittels Lasers diskutiert und deren Vor- und Nachteile abgewogen.
Kapitel Drei stellt die gängigen Verfahren zur Bildsegmentierung vor und beschreibt deren Voraussetzungen an die zu segmentierenden Bilder, bzw. das benötigte Vorwissen über den Bildinhalt.
Das Kapitel Vier ist dem Thema der Graphischen Modelle gewidmet und geht im besonderen auf verschiedene Arten von Markov-Feldern und Energiefunktionen ein.
Kapitel Fünf beschreibt die Vorgehensweise bis zum Erhalt der segmentierten und lagebestimmten Objekte im dreidimensionalen Raum.
Im sechsten Kapitel werden die Ergebnisbilder der verwendeten Verfahren präsentiert.
Es folgt eine Zusammenfassung, sowie eine Auseinandersetzung mit den Ergebnissen.

Kapitel 2

Erstellung der Tiefenbilder

Für die Erstellung der Tiefenbilder stand uns ein 1D Laserscanner von SICK zur Verfügung. Dieser wurde an der Spitze eines Kuka Industrieroboters befestigt. Der Einsatz von Laserscannern hat den Vorteil der Unabhängigkeit von den Beleuchtungsverhältnissen. Auch in völliger Dunkelheit sind Aufnahmen möglich und die Qualität der Ergebnisbilder wird nicht durch die Beleuchtung oder eventuellen Schattenwurf des Roboterarms beeinflußt.
Der Sick LMS 400 ist ein Nahbereich-Scanner für Distanzen von $70cm$ bis $3m$, welcher die Objekte mit einer Genauigkeit von $\pm 4mm$ erfasst. Der typische systematische Messfehler schwankt in Abhängigkeit von der Entfernung der Objekte zum Scanner und der Reflexionseigenschaften der Objekte, dem Remissionsfaktor (siehe Abb. 2.1). Der Remissionsfaktor ist material- und winkelabhängig, bei nahezu parallel zum Laserstrahl verlaufenden glatten Flächen wird er ziemlich gering ausfallen. Karton hat etwa einen Remissionsfaktor von 20%, während der Wert bei polierter Stahl bis zu 200% betragen kann.

Remission	Entfernung	Statistischer Fehler (1 Sigma)	
		Typisch	Maximal
200%	700 bis 3000 mm	3 mm	
	1000 bis 2500 mm		4 mm
	< 1000 oder > 2500 mm		7 mm
78%	700 bis 3000 mm	4 mm	
	1000 bis 2500 mm		5 mm
	< 1000 oder > 2500 mm		8 mm
40%	700 bis 3000 mm	6 mm	
	1000 bis 2500 mm		7 mm
	< 1000 oder > 2500 mm		10 mm
10%	700 bis 3000 mm	9 mm	

Abbildung 2.1: Statistischer Messfehler in Abhängigkeit von der Remission und Entfernung der Objekte (Quelle: SICK [AG]).

Die Winkelauflösung beträgt $0,1° - 1°$ und die Abtastrate kann zwischen $200-500Hz$ gewählt werden. Wir wählten eine Auflösung von $0.25°$ aus, die einen guten Kompromiss zwischen der Abbildung feiner Strukturen und der Erzeugung großer Datenmengen darstellte. Der LMS 400 arbeitet mit einem Rotlicht-Laser der Klasse 2 und einer Wellenlänge von $\lambda = 650nm$, welche für das menschliche Auge sichtbar ist und bei längerem Blick in den Strahlengang schädlich sein kann. Für eine detaillierte Beschreibung des Lasers siehe [AG].

Abbildung 2.2: Aufbau der Scanaparatur mit Kuka Industrieroboter und SICK Laserscanner LMS 400-00

Abbildung 2.2 zeigt Kuka mit einem Vakuumhebeystem und dem Laserscanner (blauer Kasten) an seiner Spitze. Im Hintergrund steht der Steuerungscomputer für den Roboterarm. Zur Erfassung von Objekten wird der Roboterarm horizontal über das Objekt geführt. In diskreten Schritten werden 1D-Tiefenscans orthogonal zur Bewegungsrichtung des Armes durchgeführt. Die Scanlinie wird durch eine Winkeländerung des im Scanner rotierenden Spiegelprismas erzeugt, an dem der Laser reflektiert wird, wie in Abb. 2.3 zu sehen.

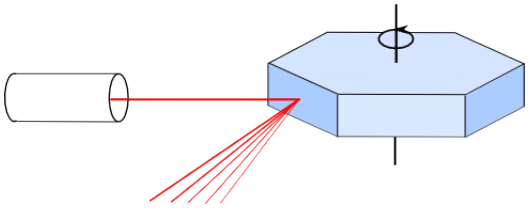


Abbildung 2.3: Generierung der Scanlinie

2.1 Versuchsaufbau

Als Scanobjekte wurden Pappkartons, Holzkisten und Styroporteile unterschiedlicher Größe und Form verwendet. Diese wurden willkürlich nebeneinander und übereinander in einem etwa $1,50 \times 1,50$ Meter messenden Bereich angeordnet. Der Roboterarm wurde in einer Höhe von $1100mm$ in diskreten Abständen von ca. $2mm$ horizontal über dem Boden bewegt. Die Scans wurden mit einer Winkelauflösung von $0.25°$ bei $200Hz$ und mit dem Laseraustrittswinkel von $70°$ durchgeführt; daraus resultiert ein Bild mit der Breite von 280 Pixeln. Die Anzahl der 1D-Scans und der Scanbereich wurden in etwa dem Abstand zwischen den 280 Scanpunkten angeglichen. Der KUKA-Industrieroboter wurde mit einer Geschwindigkeit von 30% der Maximalgeschwindigkeit betrieben. Da der Roboterarm nicht in einer kontinuierlichen Bewegung vom Start- bis zum Endpunkt, sondern in kleinen Schritten bewegt wurde, hätte eine schnellere Ansteuerung zu große Erschütterungen für den Scanner bedeutet.
Abbildung 2.4 stellt ein Beispiel von Kistenkompositionen dar. Die rote Linie etwas unterhalb der horizontalen Mittellinie stellt die Scanlinie dar.

Abbildung 2.4: Versuchsaufbau mit den zu scannenden Objekten aus Sicht das Laserscanners

2.2 Verfahren zur Entfernungsmessung

In diesem Abschnitt werden verschiedene Methoden zur optischen Entfernungsmessung mittels eines Laser vorgestellt. Handelsübliche Glühbirnen senden Licht mit unterschiedlichen Wellenlängen aus, welches zeitlich versetzt und somit phasenverschoben ist. Laser hingegen senden Licht weitgehend phasengleich und kollimiert ab. Es wird unterschieden nach Festkörperlasern, Gaslasern, Flüssigkeitslasern, Farbstofflasern und Halbleiterlasern. Für die Strichcodeverarbeitung und Entfernungsmessung werden Gaslaser (He-Ne-Laser) und Halbleiterlaser (Diodenlaser) verwendet. Die vorgestellten Verfahren beschränken sich auf solche, die mit einem in einer Richtung aktiv ausgesendeten Messstrahl arbeiten, der am Objekt reflektiert wird.

2.2.1 Triangulation

Eine Möglichkeit der Abstandsmessung ist die Triangulation. Der Laser sendet zum Erfassen der Tiefenwerte eines Objektes einen Laserstrahl aus. Abhänging von der Entfernung des Objektes wird der reflektierte Strahl an unterschiedlichen Stellen des Empfängers registriert. Der Winkel α zwischen Laser und Detektor, die Distanz D von Laser und Empfänger, sowie x_0 und f sind bekannt. Aus dem Abstand x_2 des reflektierten Laserstrahl zur Mitte der CCD-Zelle kann mit Hilfe der Winkelfunktion der Tiefenwert $x = x_0 + x_1$ errechnet werden.

$$x = D \cdot \frac{\tan\alpha + \tan\delta}{1 - \tan\alpha \cdot \tan\delta}, \quad \text{mit } \tan\delta = \frac{x_2}{f} \tag{2.1}$$

Der Winkel α zwischen x_0 und der Distanz-Geraden und der Winkel δ zwischen x_0 und x_1 werden von der Linse aus betrachtet.
Aufgrund des nichtlinearen Zusammenhangs zwischen Messabstand und der Abbildung auf dem Empfänger, ist die Auflösung im Nahbereich deutlich besser, als in der Ferne. Um die Messpannung nutzen zu können muss die Funktion linearisiert werden. Auf Grund der rein trigonometrischen Zusammenhänge kann die Messung kontinuierlich erfolgen und eignet sich somit gut zur Abstandsmessung an bewegten Objekten. Abbildung 2.5 stellt das beschriebene Messverfahren und die Linearisierung der Messpannungskurve dar.

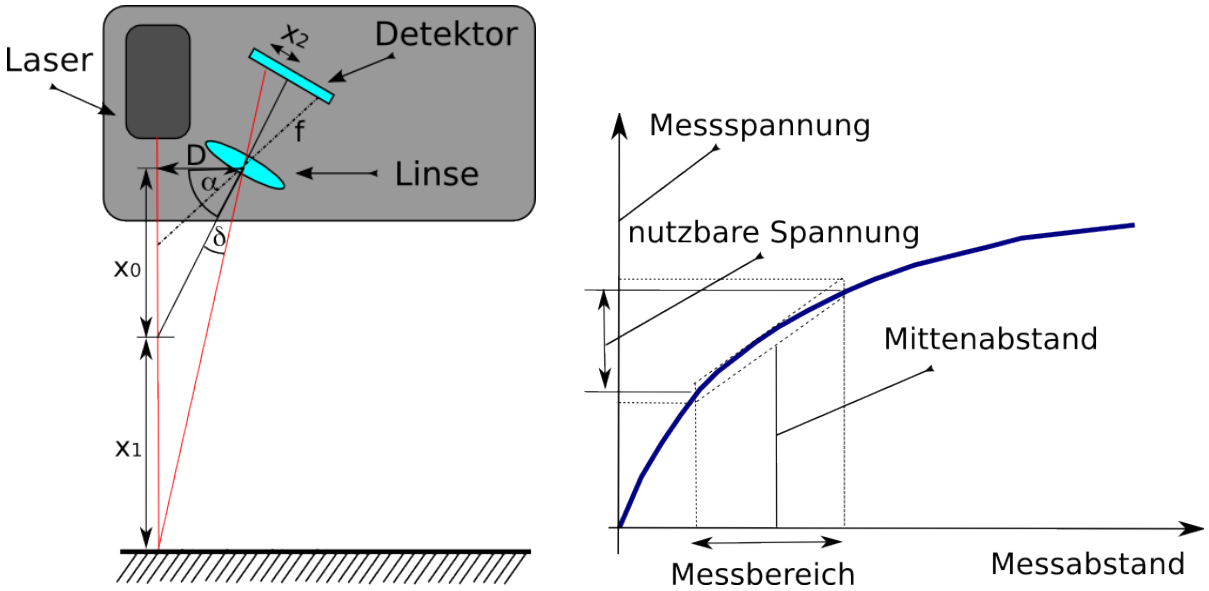


Abbildung 2.5: Funktionsweise der Laser-Triangulation

2.2.2 Laufzeitmessung

Hierbei wird die Laufzeit Δt des Laserstrahls vom Sender zum Objekt und zum Empfänger gemessen. Die Distanz d von Objekt und Laser beträgt somit $d = 0,5 \cdot c \cdot \Delta t$.
Auf Grund der enormen Lichtgeschwindigkeit von $c = 300.000\ km/s$ ist für diese Methode eine sehr genaue Zeitmessung erforderlich. Um eine Messgenauigkeit $\Delta s = 1\ cm$ zu erreichen muss die Zeit mit einer Genauigkeit von unter 0,1 ns gemessen werden.

$$\Delta t_{min} = \frac{2 \cdot \Delta s}{c} = \frac{2 \cdot 10\ mm}{300\ mm/ns} = 0.06\ ns \tag{2.2}$$

2.2.3 Phasenverschiebung

Der von uns verwendete SICK Laserscanner arbeitet auf Basis der Phasenverschiebung. Dazu wird auf den Laserstrahl eine Sinuswelle aufmoduliert. Die Entfernungsmessung geschieht aufgrund des Phasenlaufzeitunterschiedes der hinlaufenden Welle und der vom Objekt reflektierten Welle (siehe Abb. 2.6). Dieser Unterschied ist proportional zur Distanz.

Wird die Laserfrequenz selbst zur Überlagerung genutzt kann keine absolute Weglänge gemessen werden, sondern nur die relative Änderung bei Verschiebung des Objektes. Dazu wird die Summe von hinlaufender und zurücklaufender Welle periodisch moduliert. Bei Verschiebung um eine halbe Lichtwellenlänge durchläuft das Signal genau eine Periode. Die Entfernung resultiert aus der Anzahl der Durchgänge multipliziert mit der Wellenlänge des Lichts.

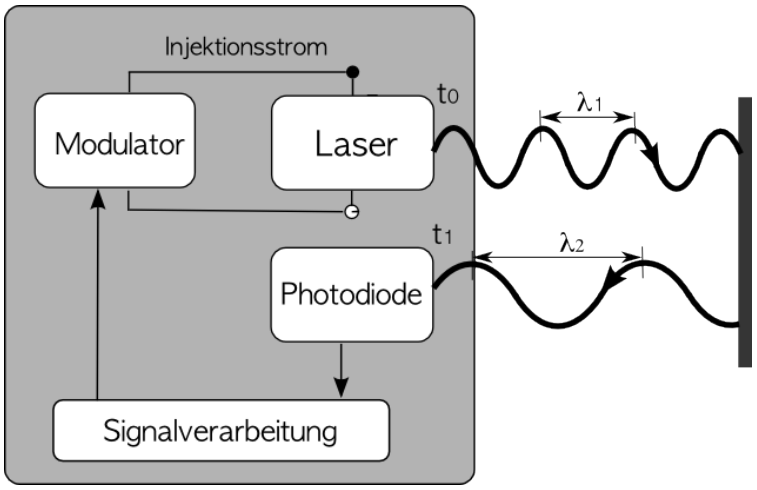


Abbildung 2.6: Funktionsweise der Entfernungsmessung mittels Phasenverschiebung

Aus der Laufzeit Δt des Lichts und der aufmodulierten Wellenlänge λ_1 resultiert ein Phasenunterschied ϕ zwischen Sender und Empfänger, aus der die Entfernung berechnet werden kann.

$$\phi = \frac{\Delta t}{T} \cdot 2\pi \qquad \text{, Periodendauer } T = \lambda_1 / c \tag{2.3}$$

Die Distanz d wird dann wie folgt berechnet:

$$d = \frac{c \cdot T}{4\pi} \cdot (\phi + 2\pi \cdot n) \qquad \text{, } n = |\text{Durchläufe}| \tag{2.4}$$

Bei diesem Verfahren wird die maximal messbare Entfernung zum Laser und die Genauigkeit der Messung durch die Periodendauer bestimmt. Bei hoher Genauigkeit, also kleiner Wellenlänge kann auch nur eine kurze Entfernung gemessen werden, wohingegen bei weiten Entfernungen die Messgenauigkeit sinkt.
Um diese Problematik zu umgeben wird eine Welle aufmoduliert und die Phasenlage dieser Welle anstatt der Laserwellenlänge zur Berechnung genutzt.

$$L = \Delta\phi \cdot \frac{1}{2} \cdot \frac{\lambda_1 \cdot \lambda_2}{\|\lambda_1 - \lambda_2\|} \tag{2.5}$$

Da die exakte Bestimmung der Wellenlänge nicht möglich ist, muss sie als Referenz benutzt werden:

$$L = L_{ref} \cdot \frac{\Delta\phi}{\Delta\phi_{ref}} \tag{2.6}$$

Der Vorteil dieser Methode gegenüber der Laufzeitmessung ist die höhere Auflösung, die mit geringerem messtechnischen Aufwand zu realisieren ist. Die Amplitude dieses Signals kann zur Berechnung der Remission der Objektoberfläche genutzt werden, um verschiedene Oberflächenmaterialien zu unterscheiden.

2.3 Oberflächenbeschränkungen

Bei der Verwendung von Laserscannern gibt es gewisse Einschränkungen bzgl. der Oberflächenbeschaffenheit der zu scannenden Objekte. Der Laserstrahl sollte von dem Objekt diffus, also ohne Vorzugsrichtung reflektiert werden, um garantiert von der Photodiode empfangen zu werden. Dazu muss die Objektoberfläche eine gewisse Rauheit aufweisen und darf nicht spiegelnd sein, glänzende Oberflächen haben bei gleicher Entfernung unter unterschiedlichen Auftreffwinkeln unterschiedliche Remissionswerte.
Ebenso spielt die Materialfarbe eine wichtige Rolle. Das Laserlicht besteht im Gegensatz zum Tageslicht nicht aus einer Summe von Farben, sondern ist monochromatisch. Wenn dieses Laserlicht auf ein Objekt trifft, kann dieses Objekt lediglich Licht eben dieser Wellenlänge reflektieren. Das heißt rote Objekte reflektieren das Laserlicht gut, während grüne Objekte das Laserlicht eher absorbieren. Schwarze Objekte absorbieren Licht aller Wellenlängen und erscheinen somit für den Laser unsichtbar. Als Beispiel dient Abb. 2.7, welche das Tiefenbild eines Modellhauses mit schwarzem Dach zeigt, wobei das Dach ist zu großen Teilen nicht zu sehen ist. Außerdem fällt die Heterogenität der Hausfassade auf, wenn kein Glättungsfilter verwendet wird. Dies liegt an der Messungenauigkeit des Laserscanners.

Abbildung 2.7: Tiefenbild eines Modellhauses mit schwarzem Dach, welches nur teilweise erkannt wurde.

2.4 Fehlerkorrektur

Bei der Aufnahme der Objekte ist das Ergebnis durch das „Schwenken" des Lasers verfälscht. Die Aufnahme eines Pappkartons zeigt eine deutliche Krümmung des Bodens, die real nicht vorhanden ist. Zudem wird der Ungleichverteilung der Scanpunkte bei Kontakt mit Hindernissen nicht Rechnung getragen (siehe Abb. 2.8 Kiste links).

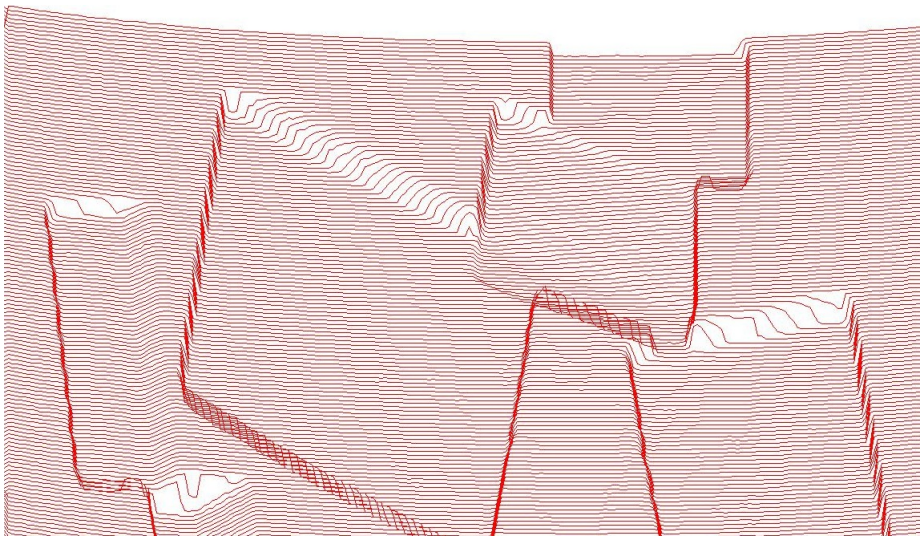


Abbildung 2.8: Tiefenbild eines Kartons ohne Fehlerkorrektur

Abbildung 2.9 verdeutlicht die zunehmende Diskrepanz zwischen den gemessenen Höhenwerten h und der exakten Entfernung y mit wachsendem Abstand b zum Mittelpunkt. Die y-Werte stellen somit die von uns gesuchten winkelunabhängigen Tiefenwerte dar.

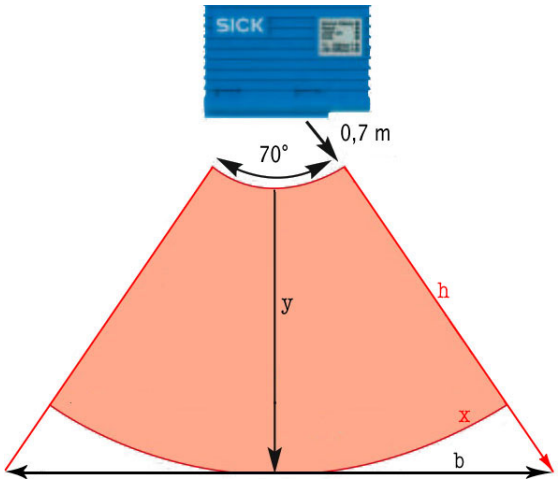


Abbildung 2.9: Darstellung der Differenz zwischen realer Entfernung y und ungenormter gemessener Entfernung h.

Um diesen Fehler zu vermeiden wurden die Tiefenwerte auf eine Ebene projiziert. Das Array D mit den 280 Tiefenwerten d wurde zu diesem Zweck in zwei Arrays mit den korrespondierenden X- und Y-Werten transformiert. Dafür wurde folgende Normalisierung verwendet:

$$norm(d) = \frac{\sqrt{(2.58^2 + \left(\frac{d*1.8}{Dh}\right)^2}}{2.58} \qquad \text{, mit } Dh = \text{halbe Größe von } D \tag{2.7}$$

Der Y-Wert errechnet sich aus dem Quotienten von D und dem Normierungsfaktor:

$$Y(Dh+i) = \frac{D(Dh+i)}{norm(\|i\|)} \qquad i = \begin{cases} d & \text{, falls } i \leq Dh \\ -(d-Dh) & \text{, falls } i > Dh \end{cases}$$

Die X-Werte werden folgendermaßen aus den Y-Werten errechnet:

$$X(Dh+d) = Cal + \sqrt{D(Dh+d)^2 - Y(Dh+d)^2} \qquad \text{, falls d} \leq \text{Dh}$$

$$X(Dh-(d-Dh)) = Cal - \sqrt{D(Dh-(d-Dh))^2 - Y(Dh-(d-Dh))^2} \qquad \text{, falls d} > \text{Dh}$$

Der Kalibrationswert Cal stellt dabei die Entfernung vom Laser zum Boden dar und wird vor jeder Messserie einmal auf leerem Boden ermittelt. Er sorgt dafür, dass die X-Koordinaten bei Null beginnen und nicht beim Überschreiten der Mitte wieder absinken.
Im nächsten Schritt mßsen die Abstände zwischen den Scanpunkten normiert werden, um jedem Pixel im Ergebnisbild einen Tiefenwert zuordnen zu können.
Ohne Objekte in der Scanlinie sind die Scanpunkte am Boden gleichverteilt, trifft der Laserstrahl aber auf einen Gegenstand, ändert sich der Abstand zum vorherigen Scanpunkt. Trifft der Laserstrahl von vorne auf ein quadratisches Objekt, dann „klettert" er am Objektrand hoch, ohne eine horizontale Bewegung zu vollführen. Dies führt bei äquidistanter Interpretation der Scanpunkte zu einer scheinbaren Schrägen. Diese würde bei der späteren Segmentation Schwierigkeiten bereiten, da bei einer Draufsicht die Seitenwände nicht zu sehen sein dürften.
Beim Übergang von der hinteren Kante zum Boden erhöht sich die Distanz zum vorherigen Scanpunkt, da der Laserstrahl einen längeren Weg bis zum Boden hat.

Um die Punkte äquidistant zu verteilen wird ein neues Array erstellt. Um dieses Array zu füllen wird in dem X-Array nach dem Wert $n \times \frac{\max(x)}{|X|}$ gesucht, mit $n = 0, \ldots, n = |X|$. Existierte der X-Wert wurde der korrespondierende Y-Wert in das neue Array geschrieben. Ist dieser nicht vorhanden, der Laser demnach ein Objekt getroffen hat, wird der Y-Wert des nächst kleinere vorhandenen X-Wertes Richtung Rand gespeichert. Würde der Punkt mit dem geringsten Abstand gewählt, erschien die Kiste breiter, als sie in Wirklichkeit ist. An einer Kante wird der letzte gemessene Tiefenwert so lange weiterpropagiert, bis ein Oberflächenpunkt des Objektes erreicht wurde (siehe Abb. 2.10).

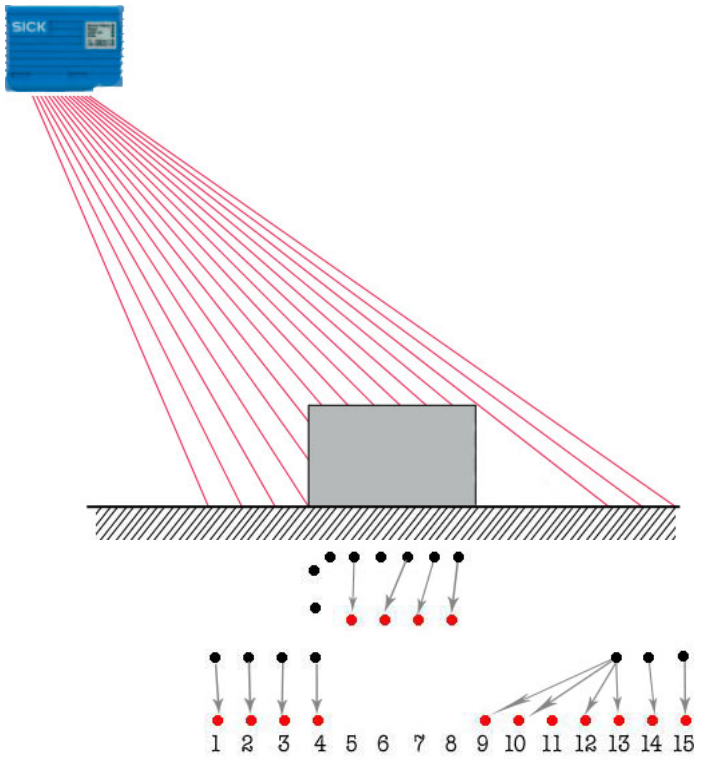


Abbildung 2.10: Distanzverschiebung zwischen den einzelnen Scanpunkten, sowie die Normierung der Abstände der Scanpunkte (schwarz sind die Originalwerte, rot die äquidistant normierten Werte)

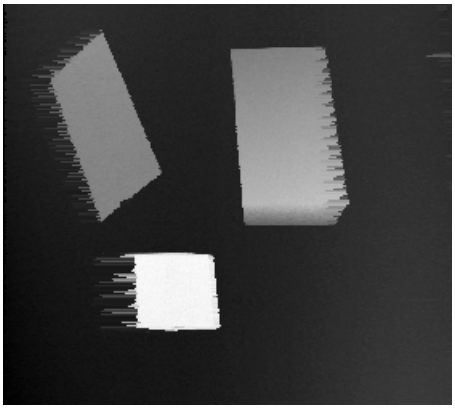


Abbildung 2.11: Schlierenbildung beim scannen ohne Medianfilter

Probleme traten auf, wenn der Laserstrahl eine Kante streifte, wie etwa bei Punkt Nr. 13 in Abb. 2.10. Dann wurde der verfälschte Tiefenwert bis zur Objekt-Oberkante (Nr. 8) weiterpropagiert und es entstanden Schlieren. Dies konnte behoben werden, indem der vorletzte, sicher richtig erkannte Kantenwert (Nr. 14) verwendet wurde. Um ein homogeneres Ergebnis zu erzielen wurde dies durch einen 5×1- Medianfilter mit folgendem Aussehen realisiert:

x_{-1}	x	$\mathbf{x}_1$	x_2	x_3

bzw.

x_{-3}	x_{-2}	$\mathbf{x_{-1}}$	x	x_1

, wenn $x < Dh/2$

Dabei stellt der $\mathbf{x}$-Wert in der Mitte den ausgewählten Tiefenwert dar. Der Medianfilter ist ein Rangordnungsfilter, welcher diskrete Bildstörungen eliminiert. Die Nachbarpixel werden sortiert und der betrachtete Pixel durch den in der Mitte liegenden Wert aus der Nachbarschafts-Liste ersetzt. Das Ergebnis ist in Abb. 2.12 zu sehen, hier sind im Gegensatz zu Abb. 2.9 steile vertikale Seitenflächen zu erkennen. Zudem ist der Boden mit dem im folgenden beschriebenen Methode eingeebnet wurden.

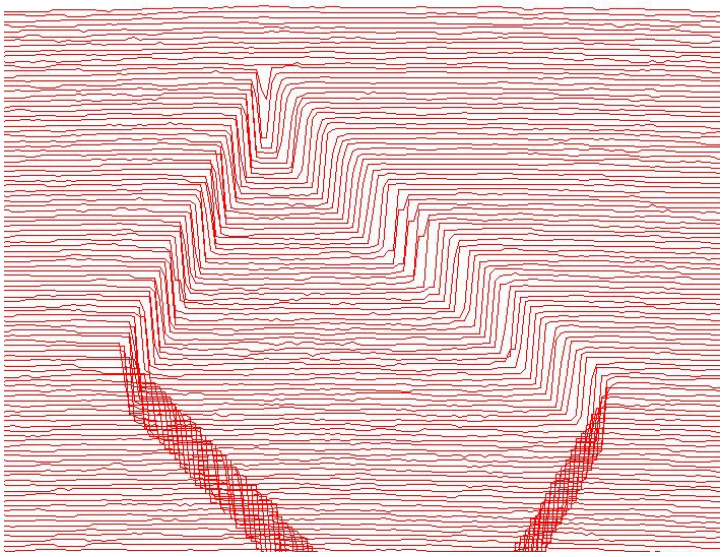


Abbildung 2.12: Tiefenbild eines Kartons mit Fehlerkorrektur

2.4.1 Bodennormierung

Da uns zur Justierung des Roboterarms nur eine Wasserwaage aus dem Baumarkt zur Verfügung stand, fiel die Anpassung in exakt horizontaler Lage schwierig. Zudem erwies sich der Boden als uneben. Deshalb entschlossen wir uns anhand der Eckpunkte des Scanbildes den Hintergrund nachträglich auf einen Wert zu normieren.
Allerdings musste nun bei der Aufnahme der Objekte darauf geachtet werden, dass die Eckpunkte frei von Kisten waren, damit die Tiefenwerte des Bodens richtig bestimmen werden konnten. Zur Ermittlung der Durchschnitts-Eckpunkte wurde der Mittelwert aus den 9 Randpunkten jeder Ecke berechnet und im folgenden mit oli (oben links), ore, uli, ure bezeichnet. Als nächstes wurden die Höhendifferenzen δ_0 zwischen den Eckpunkten $(0,0)-(280,0)$ und δ_{250} zwischen $(0,250)-(280,250)$ berechnet und jeweils durch die Bildbreite von 280 Pixeln geteilt. Somit stellt δ_0 die Differenz zwischen den einzelnen Pixeln von Y_0 und δ_{250} die Differenz zwischen den Pixeln von Y_{250} dar. Nun konnten die Höhenunterschiede entlang der Bildlänge für jeden Pixel mit Hilfe der δ-Werte auf gleiche Weise ermittelt und beglichen werden.

Listing 2.1: Bodennormierung

```
1 for (int x=1; x<realim.shape()(1); x++)
2     {
3     int oben = (int)(oli - x*(oli-ore)/xend);
4     int unten= (int)(uli - x*(uli-ure)/xend);
5     for (int y=0; y<realim.shape()(0); y++)
6       realim(y,x)+=y*(oben-unten)/yend + x*(oli-ore)/xend;
7     }
```

Abbildung 2.13 veranschaulicht die Berechnung zur Glättung des Untergrundes, die rote Fläche stellt dabei den neu berechneten homogenen Hintergrund dar.

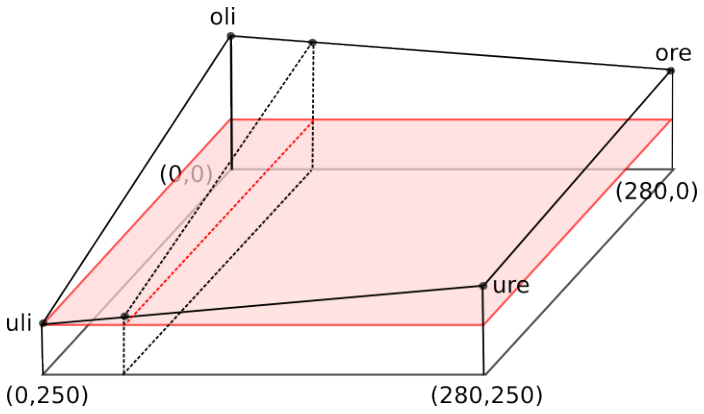


Abbildung 2.13: Normierung eines schiefen Bodens auf einen Hintergrundwert.

Kapitel 3

Segmentierung

Die Segmentierung ist ein wichtiger Teil der Bildanalyse mit dem Ziel ein Signal bzw. Bild verstehen zu können. Dabei gibt es mehrere Möglichkeiten Objekte voneinander zu unterscheiden, etwa durch unterschiedliche Form, Farbe oder Textur. Das Ziel der Segmentierung ist die Unterteilung des Bildes in inhaltlich zusammenhängende disjunkte Mengen. Das Homogenitätskriterium definiert dabei, welche Pixel einer Region angehören.
Im folgenden werden einige Verfahren zur automatischen Segmentierung vorgestellt. Eine gebräuchliche Unterteilung findet zwischen Pixel-, Kanten- und Regionenorientierten Verfahren statt. Besteht ein Vorwissen über die zu segmentierenden Elemente kann gezielt nach bestimmten Formen oder Texturen gesucht werden. Dieses Verfahren nennt man modellbasiert bzw. texturorientiert. Die Gemeinsamkeit aller Segmentierverfahren besteht darin, dass jedem Bildpunkt Eigenschaften zugewiesen werden und diese benutzt werden um die Pixel in Gruppen zu unterteilen.

3.1 Pixelorientierte Verfahren

Pixelorientierte Verfahren sind einfache und schnelle Verfahren, bei denen für jeden einzelnen Pixel die Zugehörigkeit zu einer Region bestimmt wird. Als Beispiel dient das Schwellwert-Verfahren, bei dem die Bildmatrix pixelweise durchlaufen wird und jeder Bildpunkt abhängig von einem vorher festgesetzten Schwellwert einer Region zugeteilt wird. Der Schwellwert kann anhand des Histogramms der Farbwerte festgelegt werden. Meist wird dabei zwischen Hintergrund und Vordergrund unterschieden, der Schwellwert sollte dann zwischen und möglichst weit von den beiden Farbmaxima entfernt liegen.
Es können allerdings auch mehrere Schwellwerte gesetzt werden, abhängig von der Anzahl der zu segmentierenden Objekte. Zudem ist es möglich statt globaler Schwellwerte lokale zu verwenden. Besteht Vorwissen über das Bild, etwa die Helligkeitsverteilung oder die Posi-

tion der Lichtquelle, kann das Bild in Regionen unterteilt werden und für jede Region ein separater Schwellwert ermittelt werden.
Dieses Verfahren ist bei einem globalen Schwellwert sehr schnell, aber auch stark von der Qualität des Ausgangsbildes abhängig. Bei weichen Kanten, verrauschten Bildern oder Bildmaterial mit staken Helligkeitsänderungen kommt es zu Falschsegmentierungen.

3.2 Kantenorientierte Verfahren

Bei diesen Verfahren wird das Bild nach Kanten durchsucht, da diese meist eine sinnvolle Grenzen zwischen Regionen darstellen. Um geschlossene Konturen um die zu segmentierenden Objekte zu gewährleisten, bieten sich gradientenbasierte Methoden an. Beispiele sind der Sobel- oder Laplace-Operator, welche die Silhouette der Objekte hervorheben. Allerdings werden auch bei diesen Algorithmen harte Kanten für eine saubere Konturerkennung benötigt. Abbildung 3.1 zeigt das Resultat der Anwendung des Sobelfilters auf eines unserer Scanbilder. Der Sobelfilter stellt eine diskrete Annäherung an die lokale Ableitung eines Bildes dar und ist folgendermaßen definiert:

$$S_x = \begin{matrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{matrix} \qquad S_y = \begin{matrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{matrix} \tag{3.1}$$

Um die Gradienten in X und in Y-Richtung zu erhalten wird das Ausgangsbild A erst mit S_x und danach mit S_y gefaltet. Das richtungsunabhängige Ergebnis erhält man durch folgende Berechnung, in der die Faltung durch $\star$ ausgedrückt wird: $S = \sqrt{(A \star S_x)^2 + (A \star S_y)^2}$.

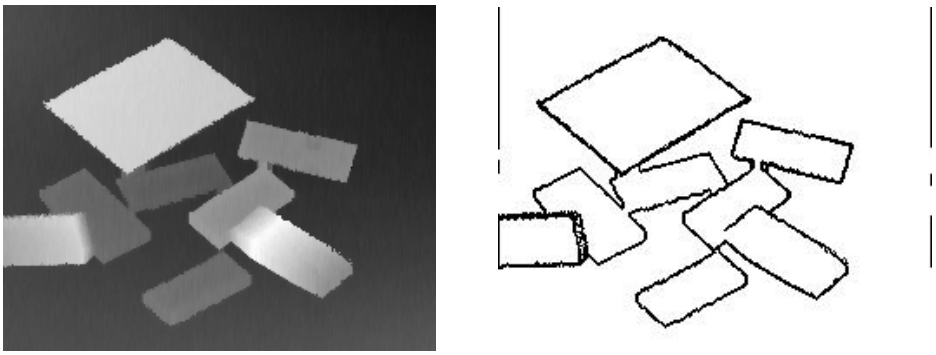


Abbildung 3.1: Ergebnis des Sobelfilters (rechts) angewandt auf ein Tiefenbild (links)

3.3 Regionenorientierte Verfahren

Die regionenorientierten Verfahren betrachten Punktmengen als Gesamtheit und versuchen dadurch zusammenhängende Objekte zu finden. Beispiele sind das Graph-Cut-Verfahren oder Region-Growing-Prinzip. Beim Region-Growing werden benachbarte Punkte einer Region hinzugefügt, wenn die Distanz zwischen dem Grauwert des betrachteten Pixels und dem Mittelwert der Pixel der benachbarten Region unter einem Schwellwert liegt. Durch das Hinzufügen von Punkten, welche schon anderen Regionen zugeordnet wurden, ist ein Verschmelzen von Regionen möglich.
Alternativ gibt es auch noch das Wasserscheidenverfahren (siehe [VS91]), welches versucht zusammenhängende Gebirgskämme zu finden indem Täler geflutet werden. Die Gebirgskämme werden durch Gradientenebenen dargestellt.
Hierbei kann es allerdings zu Übersegmentierung mit vielen kleinen Segmenten kommen.

3.4 Modellbasierte Verfahren

Bei modellbasierten Verfahren wird ein Modell der zu segmentierenden Region erstellt, nach dem dann das Bild durchsucht wird.
Ein prominenter Vertreter dieser Methode ist die Hough-Transformation, welche 1972 von Duda und Hart vorgestellt wurde [DH72]. Diese kann zum Erkennen von Linien oder Kreisen bzw. Ellipsen genutzt werden. Wird nach runden Formen gesucht, muss allerdings im Vorhinein der Radius definiert werden, bzw. die Hough-Transformation auf allen möglichen Radiengrößen durchgeführt werden.
Um Geraden zu detektieren wird das Bild in den Parameterraum transformiert. Gewöhnlich werden Geraden durch die Formel $y = m \cdot x + b$ beschrieben, wobei m die Steigung und b den y-Achsenabschnitt darstellt. Um das Problem der Formalisierung vertikaler Linien zu umgehen wird die Hesse Normalenform als Geradengleichung verwendet.

$$\text{Hesse Normalenform: } r = x \cdot \cos\varphi + y \cdot \sin\varphi \tag{3.2}$$

Wird ein Punkt in den Parameterraum transformiert, stellt die sinusartige Kurve im Hough-Raum sämtliche durch den Punkt verlaufenden Geraden dar. Der Schnittpunkt zweier solcher Kurven im Parameterraum beschreibt die durch die beiden Punkte laufende Gerade im Ortsraum (siehe [GW92]). Modellbasierte Verfahren haben allerdings den Nachteil sehr rechenintensiv zu sein. Abbildung 3.2 zeigt die gefundenen Geraden in dem Gradientenbild aus Abb. 3.1.

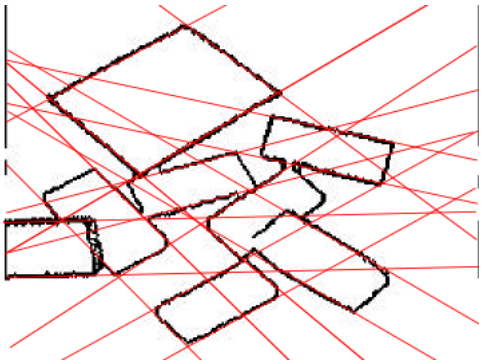


Abbildung 3.2: Mit Hough detektierte Linien eines Binärbildes (manuell eingezeichnet).

3.5 Texturorientierte Verfahren

Häufig sind zusammengehörige Regionen nicht anhand der gleichen Farbe oder Form, sondern anhand gleicher Texturen auszumachen, etwa bei Wolken, welche keine feste Form oder homogene Farbgebung besitzen.
Die Granularität einer Textur kann mittels Tamura's Feature wie folgt berechnet werden. Die Vorgehensweise zur Generierung des Tamura's Feature besteht in der Berechnung des Durchschnittpixels $A_k(x, y)$ für jeden vorhanden Pixel in der Bildmatrix. Anhand dieser Werte wird die Differenz der angrenzenden Durchschnittswerte in 2^{k-1} Schritten in horizontaler und vertikaler Richtung ermittelt $E_{k,h}(x, y)$ und $E_{k,v}(x, y)$. Zuletzt wird die Matrix $S_{best}(x, y) = 2^k$ ermittelt, welche für jedes Pixel das k sucht, das die größte Veränderung bewirkt. Abbildung 3.3 zeigt zwei Texturen und die dazugehörigen S_{best}-Matrizen. Der Durchschnitt der S_{best}-Matrix stellt die Granularität des Bildes dar.

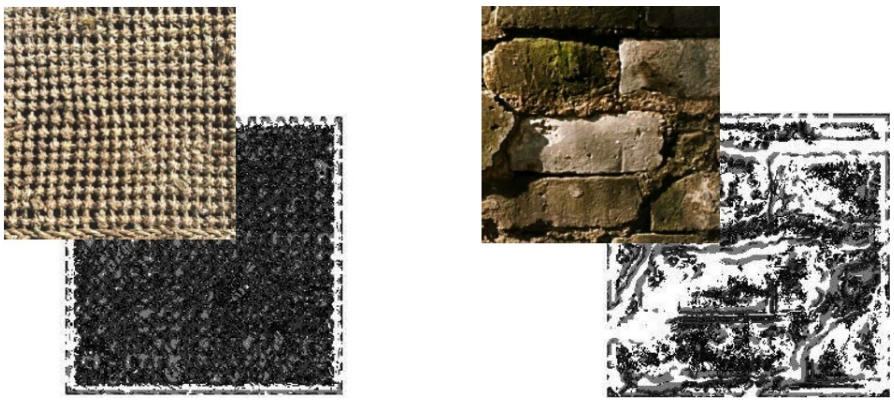


Abbildung 3.3: Vergleich von Texturen und deren S_{best}-Matrix. Quelle [Bau]

Textur-Erkenntnisse erlangt man auch durch Frequenzanalyse, beispielsweise mittels der Gabor Features (siehe [Bau]). Frequenzen erlauben Aussagen über ein Bild, große homogene Bereiche im Bild schlagen sich in niedrigen Frequenzanteilen nieder, feine Details und genaue

Auflösung von Farbunterschieden in hohen Frequenzen.
Alternativ bieten sich auch Markov-Zufallsfelder an, die in dieser Arbeit verwendet und im folgenden Kapitel genauer beschrieben werden.

3.6 Die Qual der Wahl

Jedes der beschriebenen Verfahren hat seine Vor- und Nachteile und muss mit Bedacht auf das zu untersuchende Bild gewählt werden.
Schwellwertverfahren segmentieren nur bei horizontal liegenden Kisten, da diese einen homogenen Tiefenwert aufweisen. Sie können aber bei der Entfernung des normierten Untergrundes dienen, indem die Entfernung zum Boden als Schwellwert benutzt wird.
Kantenbasierte Verfahren stellen die Kontur der Kisten gut dar, können aber bei schräg liegenden Kisten nicht die Berührungsgeraden zwischen den schrägen Flächen detektieren, da in diesem Fall die Kanten nicht sehr ausgeprägt sind. Allerdings wurden auch die Kantenbilder als ein Kriterium bei der Segmentierung durch die Markov-Netze benutzt.
Modellbasierte Verfahren, wie das Hough-Verfahren versagen ebenso bei schrägen Kanten. Außerdem werden nicht die Anfangs- und Endpunkte der Kanten-Geraden erkannt, sondern nur deren Richtung.
Das Region-Growing-Verfahren hingegen kann angewandt auf das Gradientenbild nur die Schrägen erkennen, da dort die Gradienten in die gleiche Richtung zeigen, bei horizontal liegenden Kisten fluktuieren die Gradienten aufgrund der Messfehler des Laserscanners.
Bei Verwendung von Markov-Netzen (*MRF*) können die lokalen Potentiale der MRFs den Bilddaten angepasst werden, um so ein optimales Ergebnis zu erziehlen, weswegen wir uns für diese Methode entschieden haben.

Kapitel 4

Graphische Modelle

Graphische Modelle vereinigen die Methoden der Graphentheorie und der klassischen Wahrscheinlichkeitstheorie und sind so in der Lage mit unvollständigem Wissen umzugehen. Der Vorteil von Graphischen Modellen ist die einfache Darstellung von Abhängigkeiten der durch die Knoten repräsentierten Zufallsvariablen.
Gerichtete Graphen modellieren ein Bayesianisches Netz und ungerichtete Graphen stellen ein Zufallsfeld dar. Ein Zufallsfeld enthält die Wahrscheinlichkeitsverteilung für eine Menge von Zufallsvariablen. Dabei wird jeder Pixel einem Knoten X zugeordnet, dieser erhält eine Zufallsvariable x mit dem Grauwert des Pixels und eine Potenzialfunktion. Über das Zufallsfeldmodell kann dann der wahrscheinlichste Wert für die Variablen in Abhängigkeit vom Vorwissen errechnet werden.
Unsere Bildvektoren mit Daten $D = \{x_1, \ldots, x_n\}$ bestehen aus der Menge $|M| = 256^{250\times280}$. M besteht hierbei aus 256 Graustufen, welche die Tiefenwerte repräsentieren und einer Ausdehnung von 250 Pixeln in der Höhe und 280 Pixeln in der Breite. Die einzelnen Pixel x_i werden dabei als unabhängige und gleichverteilte Realisierungen einer unbekannten Zufallsvariablen mit der Wahrscheinlichkeitsverteilung $P(X)$ angesehen.
Vorwissen wird im Bayesschen Modell als Wahrscheinlichkeitsfunktion dargestellt und Folgerungen müssen auf den vorliegenden Daten D basieren (bedingte Wahrscheinlichkeit). Da eine Wahrheitswerttabelle auf Grund der Größe der Ausgangsmenge unpraktikabel ist, werden die bedingten Wahrscheinlichkeiten in Graphen codiert. Ist die Anzahl der Eltern jeder Variablen eines Bayes-Netzes begrenzt, so folgt aus der Kettenregel, dass die Anzahl der Parameter, die zur Darstellung des Bayes-Netzes nötig sind (also die Wahrscheinlichkeiten), nur linear mit der Größe des Netzes wächst, anstatt wie die Gesamtverteilung exponentiell.

4.1 Bayes-Netze

Bayes Netzwerke repräsentieren die gemeinsame Verteilung der Variablen $V = \{V_1, \dots V_n\}$. Ein Bayes-Netz besteht aus zwei Komponenten,

- Einem gerichteten, azyklischen Graphen $G = (V, E)$, mit der Knotenmenge V und einer Kantenmenge E, wobei eine Kante je zwei Knoten verbindet.
- Einer Menge lokaler bedingter Wahrscheinlichkeitsverteilungen $P(V_i|elter(V_i))$ der einzelnen Variablen V_i.

Existiert eine gerichtete Kante von Knoten V_i nach $V_j : V_i \rightarrow V_j$ so heißt der Knoten V_j Kindknoten und der Knoten V_i Elternknoten. Die Menge der Eltern eines Knotens V_i wird als $elter(V_i)$ bezeichnet.
Die Grundlage der Wahrscheinlichkeitsverteilung ist die Markov-Bedingung, dass jede Variable V_i unabhängig von allen anderen Variablen, außer ihren Kinder ist, gegeben ihrer Eltern aus G. V_i ist somit bedingt unabhängig von V_j gegeben V_k, wenn gilt:

$$P(V_i, V_j|V_k) = P(V_i|V_k) \cdot P(V_j|V_k) \text{ bzw. } P(V_i|V_j, V_k) = P(V_i|V_k)$$

Die gemeinsame Wahrscheinlichkeitsverteilung wird durch die Kettenregel berechnet:

$$P(V_1 = v_1, \dots, V_n = v_n) = \prod_{j=1}^{n} P(V_i = v_i)|elter(V_i) = v_{elter(V_i)})$$

Friedman und Goldszmidt [FG96] haben ein berühmtes Beispiel für ein Bayesianisches Netz in Form einer Story vorgestellt (siehe Abb. 4.1). Die Zufallsvariablen sind ***E**rdbeben, Ein**b**ruch, **A**larm, Anruf vom **N**achbar* und ***R**adiomeldung*. Ein Einbruch in unser Haus ist unabhängig von einem Erdbeben, Die Alarmauslösung und die Radiomeldung sind unabhängig gegeben Erdbeben. Wegen der Unabhängigkeit der Variablen kann die Verbundwahrscheinlichkeit P(B,E,A,N,R) folgendermaßen reduziert

$$P(B, E, A, N, R) = P(A|E, B) \cdot P(R|E) \cdot P(E) \cdot P(B)$$

und als Bayesianisches Netzwerk dargestellt werden.
Die gerichteten Kanten sind dabei so konstruiert, dass sie eine kausale Abhängigkeit von der Ursache zur Wirkung codieren.

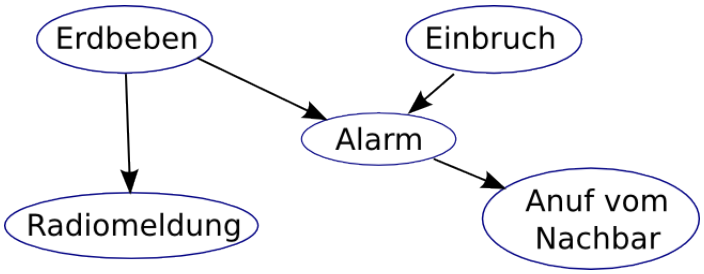


Abbildung 4.1: Beispiel eines Bayesianischen Netzwerks

4.2 Markov-Zufallsfelder

In der Bildverarbeitung sieht man sich oft mit Problemen konfrontiert, die nicht deterministisch beschreibbar sind, wie etwa dem Hintergrundrauschen. Bei optischen Sensoren stellt sich dieses durch Helligkeitsschwankungen dar, welche durch Spannungsspitzen der elektrischen Schaltkreise entstehen.
Diese Phänomene können durch Zufallsfelder modelliert werden. Sie lösen die feste Zuweisung der Farbwerte zu den Bildpunkten und stellen sie als wahrscheinlichkeitsabhängige Zuordnung dar. Die Komponenten eines Zufallsvektors $X = (x_1, \ldots, x_n)$ stehen von nun an für die Farbwerte und sind den Gitterpunkten der Pixel zugeordnet.
Ein *Markov-Netz* ist ein stochastischer Prozess, der in diskreten Zeitschritten abläuft und durch einen ungerichteten Graph $G = (X, P)$, sowie ein Nachbarschaftssystem δ definiert ist. Dabei bezeichnet X, wie schon erwähnt die Menge von Zuständen und P eine Matrix, die de Übergangswahrscheinlichkeiten enthält. Die Wahrscheinlichkeit P_{ij} vom Zustand i in den Zustand j zu wechseln hängt nur von i ab, und nicht von vorher durchlaufenen Zuständen. Da ein Markov-Netz aus einem ungerichteten Graph besteht, gibt es keine Vorgänger oder Nachfolger und die Wahrscheinlichkeit muss durch lokale Funktionen bestimmt werden.
Dabei müssen folgende Voraussetzungen bzgl. der Zufallsvariablen in X erfüllt sein:

- *Positivität:* Die Wahrscheinlichkeitsverteilung P auf S muss positiv sein:

$$P(x) > 0$$

- *Die lokale Markov-Eigenschaft* muss gelten:

$$P(X_s = x_s | X_t = x_t, t \neq s) = P(X_s = x_s | X_t = x_t, t \in \delta\{s\})$$

Markov-Netze bieten somit die Möglichkeit, die in den Daten vorhandene räumliche Information und deren Korrelation bei der Analyse zu berücksichtigen. Dabei geht man davon aus, dass sich benachbarte Positionen wenig unterscheiden und gewisse Regelmäßigkeiten vorhanden sind.

Sind zwei Knoten $x_s, x_t \in \delta$ *Nachbarn*, wird dies auch folgendermaßen dargestellt $x_s \sim x_t$. Abbildung 4.2 zeigt einige Beispiele von 2D-Nachbarschaften auf regulären Gittern, die Nachbarschaften in drei Dimensionen sind analog.

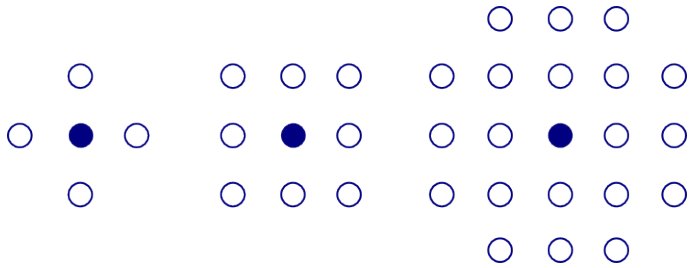


Abbildung 4.2: a) 4- b) 8- c) 20-Nachbarschaft der blauen Knoten in einem regulären zweidimensionalen Netz.

In einem kompletten Graph sind alle Knoten Nachbarn aller anderen Knoten, ein Knoten kann allerdings nicht Nachbar seiner selbst sein: $s \notin \delta\{s\}$.
Da die Nachbarschaftsbeziehung keine Informationen darüber liefert, ob die Nachbarn eines Pixels symmetrisch angeordnet sind, oder überhaupt eine geschlossene Region bilden, wird als nächstes die Clique definiert.
Eine *Clique* von G ist entweder ein einzelner Knoten oder ein kompletter Subgraph von G – Abbildung 4.3 zeigt die Cliquen einer 8-Nachbarschaft, die ersten vier Objekte stellen die Cliquen einer 4-Nachbarschaft dar.

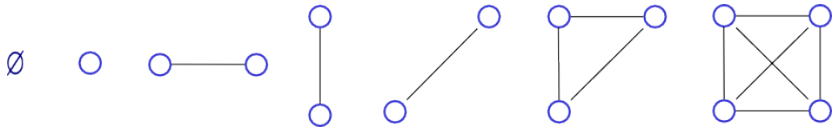


Abbildung 4.3: Cliquen einer 8-Nachbarschaft

4.2.1 Bedingte Zufallsfelder

Bedingte Zufallsfelder (*CRF*) wurden zuerst von Lafferty et al. (2001) [LMP01] als Modell für überwachtes Lernen vorgestellt. Dabei wird der Klassifikator mit Hilfe einer klassifizierten Stichprobenmenge an Merkmalen trainiert. Die Stichprobe selbst muss zuvor manuell klassifiziert werden. Anstatt den Lernprozess als Verbundwahrscheinlichkeit über die Originaldaten x und die Beobachtung y zu modellieren, werden in CRFs die Bedingten Wahrscheinlichkeiten $P(x|y)$ gelernt.

4.2.2 Assoziative Markovfelder

Assoziative Zufallsfelder (*AMN*) sind ein Sonderfall von CRF's und wurden von [TCK] vorgestellt. Die Wahrscheinlichkeit wird folgendermaßen definiert:

$$P(y|x) = \frac{\prod_{c \in C} \phi_c(x_c, y_c)}{\sum_{y'} \prod_{c \in C} \phi_c(x_c, y'_c)} \tag{4.1}$$

wobei ϕ_c die Cliquenpotentiale und x_c, y_c die Features und Label der Cliquen darstellt. Je größer das Cliquenpotential, um so wahrscheinlicher sind die Label y für die Features x korrekt. Der Nenner wird auch Partitionsfunktion Z genannt und kann nicht in polynomieller Zeit berechnet werden.
Die Größe der Cliquen wird meist auf zwei begrenzt, für solche paarweisen MRF's kann die Gleichung folgendermaßen vereinfacht werden:

$$P(y|x) = \frac{1}{Z} \prod_{i=1}^{N} \varphi(x_i, y_i) \prod_{(ij) \in E} \psi(x_{ij}, y_i, y_j) \tag{4.2}$$

wobei φ das Knotenpotential und ψ das Kantenpotential repräsentieren. Im Gegensatz zu MRF's ermöglichen AMN's verschiedenen Labels verschiedene Anziehungskraft auf Labels der Umgebung auszuüben.

4.2.3 Hidden Markov Model

Das Verborgene Markov-Modell (*HMM*) ist eine stochastische Funktion eines Markovprozesses, welches sich durch zwei Zufallsprozesse beschreiben lässt. Die Erweiterung bzgl. der MRF's besteht darin, dass jeder Zustand mit einer Verteilungsfunktion verknüpft wird. Dabei stellt eine Markov-Kette, in der die Zustände und Übergangswahrscheinlichkeiten gespeichert sind den ersten versteckten Zufallsprozess dar. Der zweite Zufallsprozess erzeugt in jedem Schritt ein Symbol gemäß der Wahrscheinlichkeitsverteilung des Zustandes, in dem es sich gerade befindet und wechselt anschließend nach der Übergangswahrscheinlichkeit in einen neuen Zustand.
Die Vorgabe einer Topologie bietet die Möglichkeit physikalische Eigenschaften des Prozesses einfließen zu lassen und die Zustandsübergänge einzuschränken. Bei der automatischen Spracherkennung etwa hat sich das *Links-Rechts-Modell* als sinnvoll herausgestellt.

4.3 Gibbs-Felder

Mittels der Markovschen Eigenschaft kann ein Zufallsfeld lokal charakterisiert werden. Die globale Charakterisierung eines Zufallsfeldes X erfolgt über die Gibbssche Verteilung:

$$P(x) = \frac{1}{Z} e^{-\beta E(x)} \tag{4.3}$$

mit

$$Z = \sum e^{-\beta E(x)} \text{ und } \beta = \frac{1}{kT}$$

Hierbei stellt T die Temperatur, k die Bolzmann-Konstante und $E(f)$ die Energiefunktion. Die Energiefunktion kann als die Summe der Potentiale beschrieben werden:

$$E_A(x) = \sum_{A \subset S} U_A(x)$$

wobei die Teilmenge A einer Clique entspricht. Die Energiefunktionen werden im folgenden Abschnitt detailliert besprochen.

4.4 Energiefunktion

Bei vielen Aufgaben in der Bildverarbeitung, etwa dem Segmentieren ist das Kennzeichnen (engl. *labeling*) von Pixeln erforderlich. Dabei sollte die Vergabe der Label L in Abhängigkeit der Daten, sowie des Vorwissens über das zu labelnde Bild geschehen. Die Energiefunktion kann somit aus einem Datenteil $E_{data}(L)$ und einem Term der das Vorwissen einbringt $E_{prior}(L)$ beschrieben werden:

$$E(L) = E_{data}(L) + \lambda \cdot E_{prior}(L) \tag{4.4}$$

λ stellt die Gewichtung zwischen Vorwissen und Daten her. Der Datenterm bestraft Label, die nicht zu den Originaldaten passen, während E_{prior} Label bestraft, die nicht mit dem Vorwissen korrelieren. Wenn das Vorwissen über die Daten nicht genau formalisiert werden kann, wird meist gewisse eine Homogenität der Umgebung vorausgesetzt. Dabei geht man davon aus, dass innerhalb der Objektgrenzen die Pixelwerte einen fließenden Übergang besitzen und nur an Objektkanten Sprünge zwischen benachbarten Pixeln auftreten. Der Term E_{prior} bzw. E_{smooth} muss so gewählt werden, dass L nicht überall homogen ist, damit keine Informationen über Objektgrenzen verloren gehen.

4.4.1 Ising-Modell

Das Ising-Modell ist nach Ernst Ising benannt [Isi25] und beschreibt ein idealisiertes Bild eines Ferromagneten. Dabei wird vorausgesetzt, dass das magnetische Moment, bzw. Spin S der Atome nur die diskreten Zustände $S = +1$ (parallel) und $S = -1$ (antiparallel) annehmen kann. Ein komplettes System von Spins mit gegebenen Ausrichtungen heißt *Konfiguration*. Der allgemeine Energieausdruck für eine solche Situation ist durch das Heisenberg-Modell gegeben

$$H(x) = -\frac{1}{kT}\left(J\sum_{i\sim j} x_i x_j - m\,B\sum_i v_i\right) \tag{4.5}$$

T ist die Temperatur, k die Bolzmann-Konstante, m und J sind Materialkonstanten.
Im ferromagnetischen Fall $J > 0$ erhalten die parallelen Spins wenig Energie und haben deshalb eine hohe Wahrscheinlichkeit sich parallel auszurichten.
Die Gleichung besagt, dass es energetisch günstiger ist, wenn zwei benachbarte Spins die gleiche bzw. entgegengesetzte Orientierung besitzen (erster Term) und wenn die Spins parallel zu einem äußeren Magnetfeld B liegen (zweiter Term). Durch das externe Feld kann jeder Konfiguration für jeden Pixel noch ein zusätzliches Gewicht zugeordnet werden. Hierdurch kann ein Vorwissen über die räumliche Verteilung der Konfigurationen in die Energiefunktion integriert werden.

Bei hohen Temperaturen verhalten sich die Spins bzw. Pixel unabhängig voneinander, sinkt die Temperatur steigt die Tendenz der Spins eine energetisch günstige Ausrichtung zu wählen. Es werden also eher große homogen Flächen entstehen, aufgrund der nur zwei Spinrichtungen können ausschließlich Binärdaten verwendet werden.
Abbildung 4.4 zeigt beispielhaft eine Ising-Konfiguration, das blaue Pixel würde sich bei niedrigen Temperaturen in ein schwarzes verwandeln.

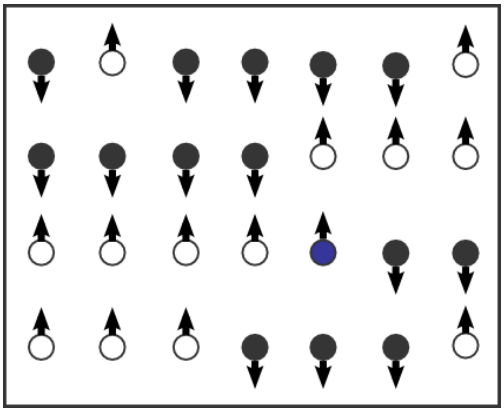


Abbildung 4.4: Beispiel einer Ising-Konfiguration ($S_{schwarz} = -1, S_{weiss} = +1$)

4.4.2 Potts-Modell

Das Potts-Modell ist eine Verallgemeinerung des Ising-Modells auf n Labels und wurde von Renfray B. Potts in seiner Ph.D. thesis vorgestellt [Pot52] und hat folgende Form:

$$H_\gamma(x|y) := G_\gamma(x) + D(x|y) = \gamma \sum_{i \sim j} \delta_{(x_i, x_j)} + \sum_{i=1}^{n} (x_i - y_i)^2 \tag{4.6}$$

Der erste Term $G_\gamma(x)$ ist der Regularisierungsterm, er bewertet die Anzahl der ungleichen Nachbarn. Der zweite Term $D(x|y)$ ist der quadratische Datenterm, er misst den Abstand der Daten, also die Datentreue des Signals x zu den Daten y. Der Parameter $\gamma > 0$ steuert das Gewicht von Glattheit zu Datentreue.
Die Minimierung des Potts-Funktionals zu gegebenem y in x bedeutet eine Darstellung $\hat{x}$ zu den Daten y zu finden, die einen bestmöglichen Kompromiss zwischen Glattheit und Datentreue findet.
Diese Minimierung kann i.a nur approximativ erfolgen, weshalb im folgenden Abschnitt einige gängige Methoden vorgestellt werden. Dadurch kann die Laufzeit allerdings sehr lang werden. Zudem können die berechneten Minima meist nicht mit Sicherheit als globale Minima identifiziert werden.

4.5 Optimierungsprobleme

Um die exakte Wahrscheinlichkeitsverteilung zu erhalten können Schätzer verwendet werden. Da es meist nicht in polynomieller Zeit machbar ist den exakten Wert der a posteriori Verteilung zu berechnen, werden im folgenden Verfahren vorgestellt, die durch zufällige Auswahl von Zufallsvariablen aus der Wahrscheinlichkeitsverteilung das beste Ergebnis approximieren. Dabei werden Markov-Ketten von Zuständen durch wiederholtes Anwenden von Markov-Prozessen erzeugt, was bedeutet das der ausgewählte Zustand nur von seinem Vorgänger abhängt. Die Markov-Prozesse müssen so ausgewählt werden, dass sie eine Abfolge von Zuständen darstellen, die mit der Bolzmann-Wahrscheinlichkeit konform ist, also ein Gleichgewicht im System darstellt. Dazu müssen die Markov-Prozesse *ergodisch* sein, es muss möglich sein ausgehend von jedem Zustand des Systems alle anderen Zustände zu erreichen. Das Ziel besteht darin einen Zufallsvektor zu finden, der proportional zu der gesuchten a posteriori Verteilung ist.

4.5.1 Maximum a posteriori Schätzer (MAP)

Das Lernen einer Verteilung besteht im Finden eines möglichst guten Parametersatzes. Das beste Modell ist daher dasjenige, welches am besten zu den Daten y passt, also die Wahrscheinlichkeit $P(x|y)$ maximiert.
Die a posteriori Wahrscheinlichkeit kann mit dem Bayesschen Theorem beschrieben werden:

$$P(x|y) = \frac{P(x) \cdot P(y|x)}{P(y)},$$

Die Verteilung $P(y)$ stellt dabei den Informationsstand über die Daten dar, der als *a priori-Wahrscheinlichkeit* bezeichnet wird. $P(x|y)$ wird als *a posteriori-Wahrscheinlichkeit* bezeichnet und $P(y|x)$ ist der *Likelihood*.
Da die Daten als konstant angesehen werden folgt daraus die Definition des *MAP*

$$\hat{x} = \arg\max_x P(x|y) \propto \arg\max_x P(x) \cdot P(y|x)$$

was der Verbundwahrscheinlichkeit $arg\max_c P(y, x)$ entspricht.

4.5.2 Metropolis

Der Metropolis-Algorithmus ist nach Nicholas Metropolis benannt, der ihn 1953 vorstellte [MRR+53]. Das Verfahren nähert eine Wahrscheinlichkeitsverteilung durch Wiederholung folgender Schritte an:

- Wähle einen Zustand $y = x_i + (2r - 1)\delta$ aus, wobei r eine Zufallszahl zwischen 0 und 1 und δ ein beliebiger fest gewählter Abstand ist.

- Berechne die Energie-Differenz $\Delta E = E(y) - E(x_i)$, wenn diese größer als Null ist, dann akzeptiere die neue Konfiguration mit der Wahrscheinlichkeit $P_A = \min\left(1, e^{-\beta \Delta E}\right)$, wenn $\Delta E < 0$ akzeptiere den neuen Zustand immer.

Der Metropolis Algorithmus akzeptiert jeden neuen Zustand, der energetisch günstiger ist, als der alte. Damit er aber auch lokale Maxima verlassen kann wird mit einer gewissen Wahrscheinlichkeit auch ein energetisch schlechterer Zustand akzeptiert.

4.5.3 Gibbs Sampler

Gibbs-Sampling ist wie auch der Metropolis-Algorithmus ein iteratives Verfahren, um eine Folge von Stichproben der gemeinsamen Wahrscheinlichkeitsverteilung zweier oder mehrerer Zufallsvariablen zu erzeugen, um die unbekannte Verbundverteilung anzunähern. Er wurde erstmals im Jahre 1984 von S. Geman und D. Geman vorgestellt [GG84] und kann als Spezialfall des Metropolis-Algorithmus angesehen werden.
Gibbs-Sampling eignet sich besonders dann, wenn die gemeinsame Verteilung eines Zufallsvektors unbekannt, jedoch die bedingte Verteilung einer jeden Zufallsvariable bekannt ist. Das Funktionsprinzip besteht darin, wiederholend eine Variable auszuwählen und gemäß ihrer bedingten Verteilung einen Wert in Abhängigkeit der Werte der anderen Variablen zu erzeugen. Die Werte der anderen Variablen bleiben in diesem Iterationsschritt unverändert. Der Gibbs Sampler passt den Zustand also lokal an die stationäre Verteilung an.
Sei $x = (x_1, \ldots, x_n)$ der aktuelle Zustand, dann erfolgt die Berechnung folgendermaßen:

- Wähle eine zufällige Komponente I

- Ist $I = 1$, dann ersetze die i-te Komponente x_i mit der Wahrscheinlichkeit

$$r_{x_{-i}, y_i} = \frac{P(x_1, \ldots, x_i - 1, y_i, x_{i+1}, \ldots, x_n)}{P(x_{-i})}$$

  durch y_i mit $x_{-i} = \{(y_1, \ldots, y_n) : y_j = x_j \forall i \neq j\}$

4.5.4 Simulated Annealing

Simulated Annealing wurde 1983 von Kirkpatrick et al. vorgestellt [KGV83] und ist ebenfalls eine Adaption des Metropolis-Algorithmus.
Dieses Verfahren versucht den physikalischen Prozeß des Abkühlens nachzuempfinden. Wird Metall erhitzt und langsam wieder abgekühlt, wird es weicher und enthält weniger Unregelmäßigkeiten. Im erhitzten Material sind die einzelnen Atome aus ihrem Gitter gelöst und können sich frei bewegen. Mit abnehmender Temperatur nimmt die Beweglichkeit der Atome ab und neue Bindungen entstehen. Dabei tendieren die Atome dazu Zustände mit möglichst geringer Energie einzunehmen.
Der Algorithmus erzeugt analog dazu bei hohen Temperaturen ständig neue zufällige Konfigurationen unabhängig von ihrer Güte. Mit abnehmender Temperatur sinkt die Wahrscheinlichkeit zufälliger Konfigurationsänderungen und es werden nur noch Konfigurationen gewählt, die zu einem besseren Ergebnis führen. Dies führt dazu, dass Simulated Annealing im Gegensatz zu anderen Verfahren aus lokalen Minima wieder herausfinden kann. Nach unendlich langer Abkühlzeit wird immer das globale Minimum gefunden, in der Praxis mit einer endlichen Abkühlzeit ist das nicht immer der Fall. Es muss also ein Kompromiss zwischen der Wahrscheinlichkeit das globale Minimum zu finden und der Geschwindigkeit geschlossen werden.
Die Nachbarn der augenblicklichen Konfiguration s_i werden nicht systematisch auf ihren Zielfunktionswert untersucht. Aus allen Nachbarn in $N(s_i)$ wird vielmehr mittels einer gleichverteilten Wahrscheinlichkeit P_{ij}^G ein Nachbar s_j ausgewählt. Lösung s_i wird auch zugunsten des schlechteren Nachbarn s_j mit der Metropoliswahrscheinlichkeit P_{ij}^A verworfen:

$$P_{ij}^A(T) = \begin{cases} 1 & \text{, falls } f(s_j) < f(s_i) \\ \exp\left(-\frac{f(s_j)-f(s_i)}{T}\right) & \text{, falls } f(s_j) \geq f(s_i) \end{cases}$$

Ist die Temperatur T sehr hoch, so ist auch die Wahrscheinlichkeit nahe 1 einen beliebigen Nachbarn auszuwählen.
Die Wahrscheinlichkeitsverteilung eines Markov-Zufalls-Feldes ist durch die Gibbsverteilung

$$P(X = \omega) = \frac{1}{Z(T)} \exp\left(-\frac{E(\omega)}{T}\right)$$

gegeben, wobei $Z(T)$ ein Normalisierungsfaktor ist.

Kapitel 5

Segmentierung und Lagebestimmung

In dieses Kapitel werden die Verfahren zur Segmentierung und Lagebestimmung der Objekte im dreidimensionalen Raum erklärt. Im ersten Schritt wird die Segmentierung der Objektoberflächen gezeigt. Anschließend werden die Eckpunkt der segmentierten Oberflächen gesucht und anhand der Tiefeninformation die Oberflächennormale in allen drei Raumrichtungen berechnet. Anhand der Oberflächennormale kann die dreidimensionale Gestalt der Objekte rekonstruiert werden. Durch einen Kollisionstest wird im letzten Schritt ermittelt, welche Kistenoberflächen der selben Kiste angehören. Bei positivem Kollisionstest werden zur Lageoptimierung die lokalen Basen der beiden Kisten gemittelt und die resultierende Kiste anhand der neuen Basis ausgerichtet.

5.1 Implementation

Die Segmentierung wurde mit Hilfe eines Markov-Netz durchgeführt. Als Features wurden der Orginal-Tiefenwert, der Gradient in X-, sowie in Y-Richtung und die Gradientenmagnitude gewählt.
Zu Beginn versuchten wir die auf den Tiefenwerten basierenden Label der unteren Objektkanten entlang gleicher Gradienten bis zur Schnittlinie der Flächen durchzupropagieren. Zu diesem Zweck wurde noch das Feature Kante verwendet, welches während der Berechnung der Potentiale gesetzt wurde. Zur Initialisierung des Kantenfeatures wurden alle Knoten mit einem gewissen Gradientenmagnitudenwert auf *Kante* gesetzt. Besaß ein Kanten-Knoten A einen Nachbarknoten B mit gleichem Gradienten oder Höhenwert wurde B ebenfalls zu einem Kanten-Knoten und übernahm das Label von A. Dadurch, dass an der gesamten Objektunterkante das selbe Label vorhanden war, sollten die seitlichen unterschiedlichen Label von der Menge der anderen unterdrückt werden. Um die Ausbreitung der falschen Label weiter zu reduzieren wurden bei Aufeinandertreffen von zwei Kanten-Knoten immer das jeweils

kleinere Label – welches somit eher an der Unterkante zu finden ist – weiterverwendet. Dies hatte allerdings eine extrem langsame Ausbreitung der Kantenknoten zur Folge, da nur jeweils direkte Nachbarn interagieren und bei einer zufälligen Wahl der Knoten nur diejenigen, welche schon ein gesetztes Kantenfeature besaßen dieses weiterpropagieren konnten (siehe Abb. 5.1).

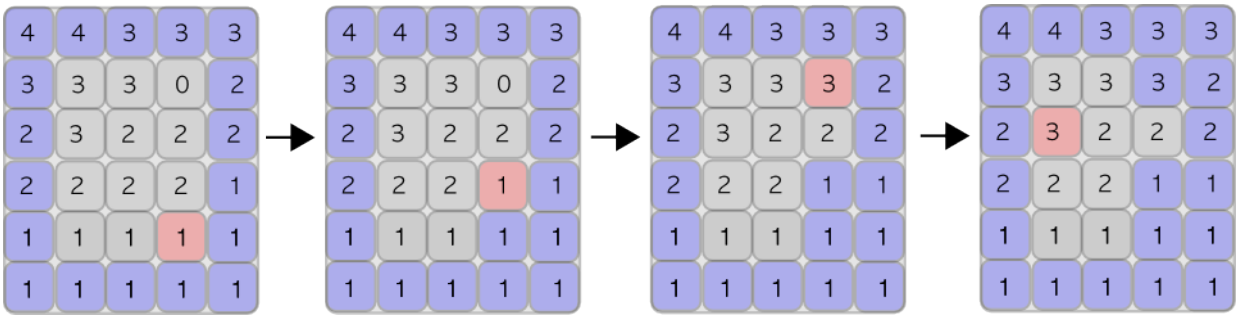


Abbildung 5.1: Die blauen Pixel stellen die Kanten-Knoten dar, die roten Pixel die aktuell untersuchten Knoten. Wenn gleich viele oder mehr benachbarte Kanten-Knoten bei gleichen Gradienten für ein kleineres Label stimmt, wird dieses übernommen und der untersuchte Knoten wird selbst zum Kanten-Knoten.

Deshalb entschlossen wir uns eine Regions-Klasse zu verwenden in welcher Nachbar-Knoten mit gleichen Features gespeichert wurden. Um zu garantieren, dass jeder segmentierten Fläche ein eindeutiges Label zugeordnet wurde haben wir uns eine Klasse gebaut, die bei Aufruf ein fortlaufendes Label zurückliefert und einen Kontainer zur Speicherung von Knoten bereitstellt. Zur Erstellung eindeutiger Label wurde das Netz zeilenweise durchlaufen und jedem Pixel eine Region zugeordet. Besaßen zwei benachbarte Pixel unterschiedliche Regionen, aber gleiche Gradienten und in etwa die gleiche Höhe, wurden die beiden Regionen verschmolzen. Gleichzeitig wurden die Höhen- und Gradientendifferenzen als Potentiale im Markov-Netz gespeichert. Durch das Verschmelzen von Regionen musste die Zufallskomponente bei der Wahl von gleichen Regionen abgestellt werden, da sonst bei hinreichend großen Iterationschritten alle Knoten in einer Region gelandet wären. Durch die Messungenauigkeit des Laserscanners und die großen Höhendifferenzen bei schräg liegenden Kisten wurden zwei Kantenpotentiale benötigt, eins welches die Gradienten-Ähnlichkeit abbildete und eins welches die Differenz der Tiefenwerte widerspiegelte. Als Nachbarschaftssystem wurde das gewählt, welches alle 8 Nachbarknoten berücksichtigt, um unabhängig von der Objektlage auch in Diagonalrichtung gute Ergebnisse zu erziehlen. Durch das Verschmelzen von Regionen musste so im Schnitt jeder Knoten nur einmal betrachtet werden, was einen enormen Geschwindigkeitsgewinn mit sich brachte.

5.2 Segmentierung

Anhand der 2D-Gradienten $Dir_{x,y}$ und der Grauwertdifferenz um den Flächenschwerpunkt entlang des Gradienten Dir_z ist es möglich die 3D-Normalen n der Flächen zu bestimmen:

$$n = \begin{pmatrix} -\frac{Dir_z \cdot Dir_y}{Mag_{xyz}} \\ -\frac{Dir_z \cdot Dir_x}{Mag_{xyz}} \\ \frac{Mag_{xy}}{Mag_{xyz}} \end{pmatrix} \tag{5.1}$$

mit $Mag_{xy} = \sqrt{Dir_x^2 + Dir_y^2}$ und $Mag_{xyz} = \sqrt{Mag_{xy}^2 + (Dir_z \cdot Dir_x)^2 + (Dir_z \cdot Dir_y)^2}$
Abbildung 5.2 zeigt die 3D-Darstellung der 2D-Tiefenwerte und der errechneten Normalen, sowie die einzelnen Komponenten der 3D-Gradienten.

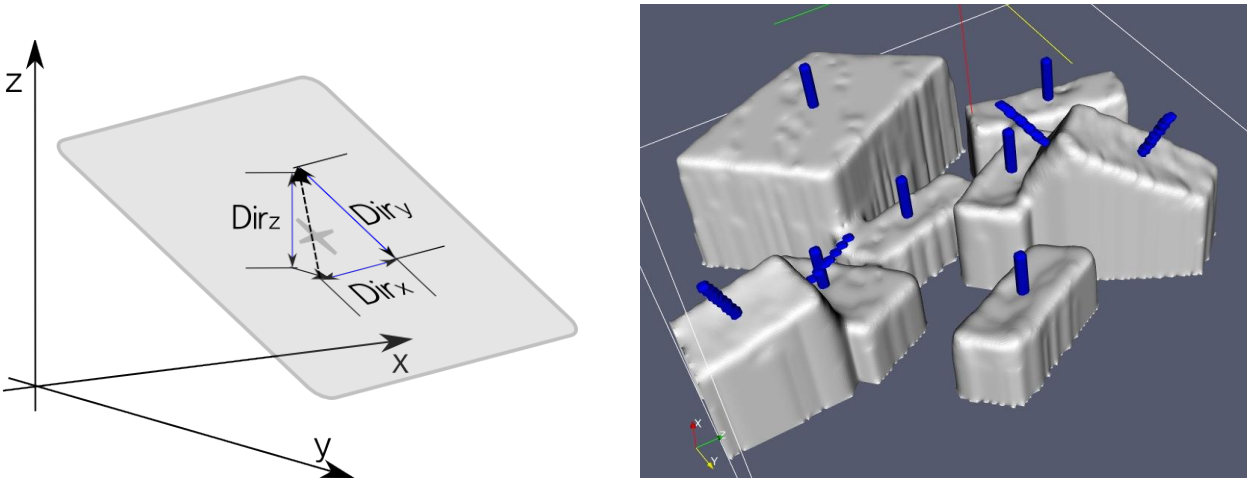


Abbildung 5.2: Gradientenkomponenten (links) und gerenderte 3D-Darstellung der segmentierten Objekte und der daraus errechneten Normalen

Abbildung 5.3 (oben links) zeigt die Verteilung der Regionen nach der Labelgenerierung. Im nächsten Schritt wurde das Markov-Netz mit einem deterministischen Gibbssampler durchlaufen, wobei die Gradientenähnlichkeit und Höhendifferenz von 8 Nachbarpixeln und deren Regionen verglichen wurden. Besaß die Mehrheit der Nachbarn ähnliche Gradienten oder Höhen wie der betrachtete Pixel, wurde dieser und seine Region mit der am häufigsten vorkommenden Region verschmolzen. In Abb. 5.3 (unten rechts) ist das Resultat der Regions-Generierung und -Zuordnung. Die verbliebenen Regionen repräsentieren zutreffend die zu segmentierenden Flächen. Die vereinzelt auftretenden Löcher in den Regionen sind Messfehler. Es werden im folgenden nur Regionen mit einer Mindestzahl an Knoten betrachtet.

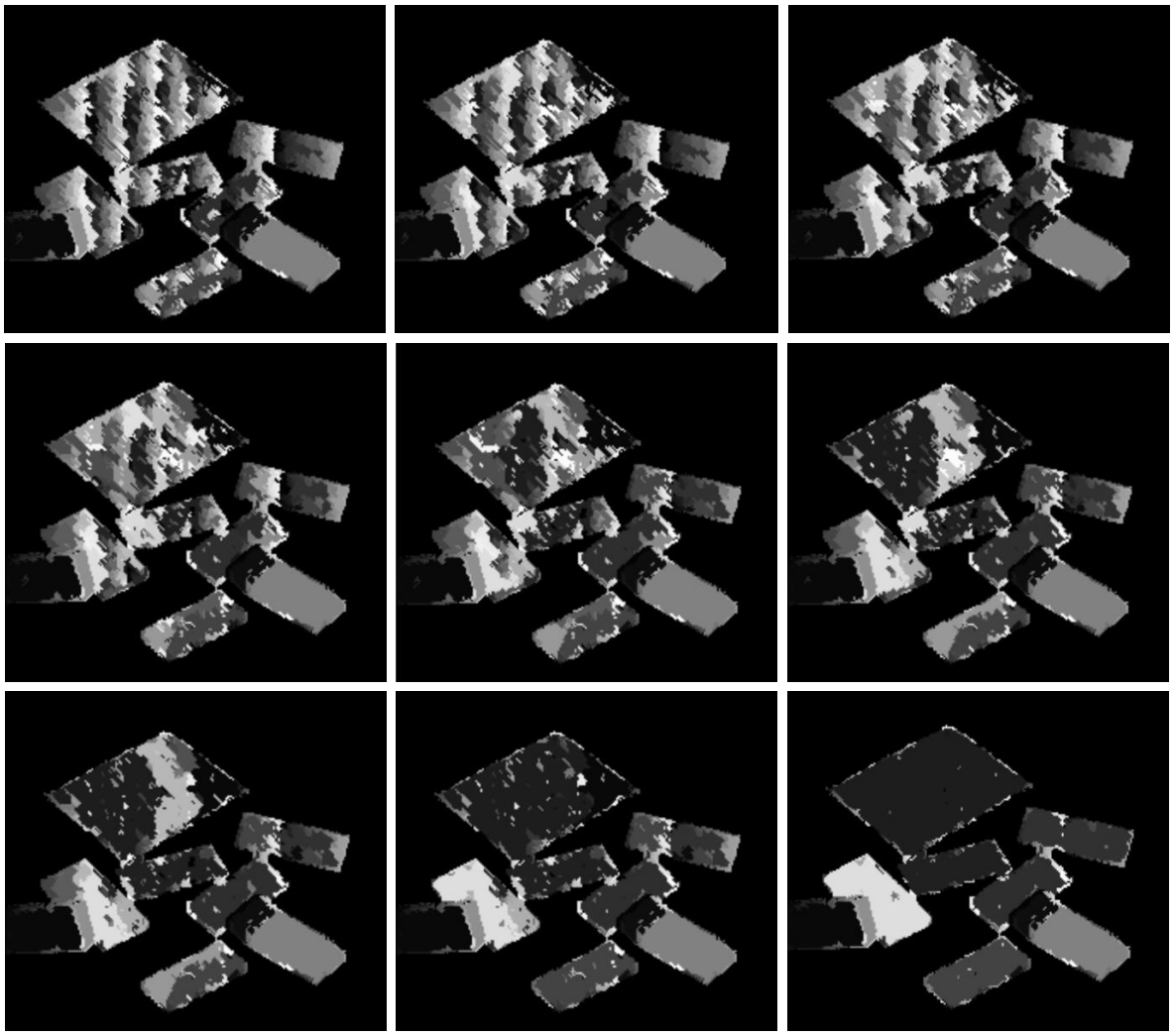


Abbildung 5.3: Zwischenschritte bis zur endgültigen Labelverteilung

5.3 Oberflächenrekonstruktion

Im nächsten Schritt wurden die Ecken des jeweiligen Objektes identifiziert. Dazu wurde zunächst der Punkt mit dem größten Abstand zum Flächenschwerpunkt Z gesucht, im folgenden mit E_1 bezeichnet. Durch Vorzeichenwechsel des Vektors $\overline{ZE_1}$ wurde die gegenüberliegende Ecke E_3 identifiziert. Anschließend wurde die Region nach den Punkten durchsucht, die mit dem Zentrum und E_1 verbunden einen Winkel nahe 90 Grad beschrieben und am Rand der Region lagen, also mindestens ein Hintergrundpixel in ihrer Nachbarschaft vorhanden war. Bei rechteckigen Kisten sollten im Idealfall genau zwei Punkte gefunden werden, einer auf der Hälfte der Längsseite L_1 und einer auf der Hälfte der Breitseite L_2. Es stelle sich heraus, dass derjenige Punkt auf der Längsseite am zuverlässigsten die Ausdehnung der Kiste beschrieb.
Um bei einer eventuellen Teilüberdeckung des Objekts nicht eine zu geringe Breite zu erhal-

ten, wurde entlang der entgegengesetzten Richtung des normierten Vektors $\overline{ZL_1}$ nach dem Rand der Kiste gesucht. Das Doppelte der maximalen Länge beider Vektoren wurde als neue Breite übernommen (siehe 5.4 links).

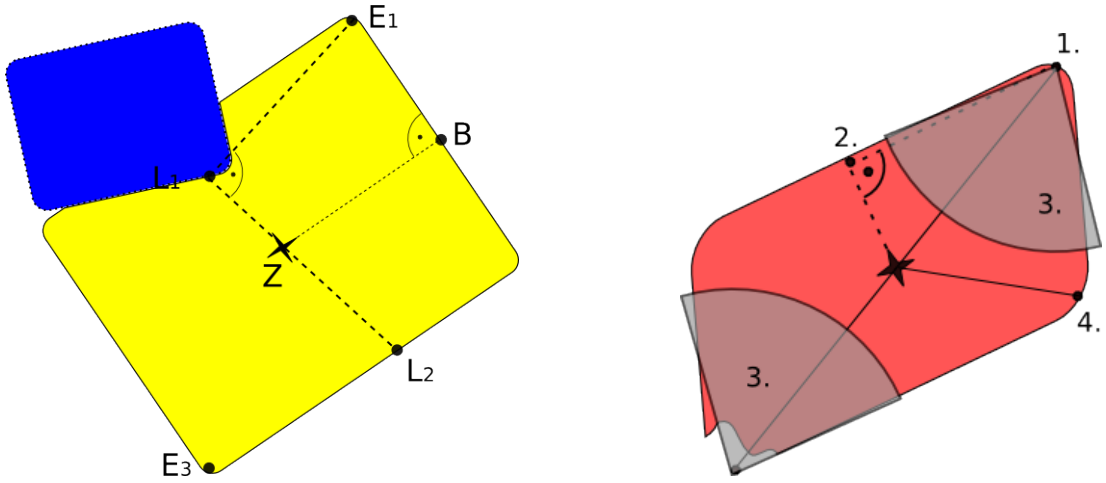


Abbildung 5.4: Vorgehensweise zum Auffinden der Objektecken (links bei einen überdeckten Objekt und rechts bei einem Objekt mit Segmentierungsfehlern)

Anhand der Länge und Breite war es möglich die Kiste als Rechteck zu rekonstruieren. Da die Abstände der Längen und Breitenabtastrate des Lasers nicht exakt übereinstimmten, wurden die quadratischen Kisten bei Lagen nahe 45 Grad zum Koordinatensystem als Rauten abgebildet. Um die Form der zu segmentierenden Objekte nicht auf Rechtecke einzugrenzen wurde der Eckpunkt und er ihm gegenüberliegende Punkt im Umkreis von $\frac{3}{4}$ der Kistenbreite abgedeckt und erneut nach dem vom Zentrum am weitesten entfernten Punkt gesucht. So wurde der Nachbarpunkt des Eckpunktes E_1 gefunden (siehe 5.4 rechts).
Der Wert $\frac{3}{4}$ hat sich empirisch als bester Kompromiss zwischen der minimalen Abdeckung von einer halben Objektbreite und der maximalen Abdeckung von 90 Prozent der Kistenbreite herausgestellt.
Da das Skalarprodukt aus Normale n und Vektor v zur jeweiligen Kante Null ergeben muss, konnte nach folgender Formel die Höhe der einzelnen Eckpunkt ermittelt werden:

$$v(2) = -\frac{(n(0) \cdot v(0) + n(1) \cdot v(1))}{n(2)}$$

5.3.1 Dilatation

Um die Lücken in den erkannten Oberflächen zu schließen wurde eine Dilatation durchgeführt. Dies war wichtig, um zu verhindern, dass ein Element, welches nicht am Rand der Region lag, aber ein fehlsegmentierten Pixel aus einer anderen Region in seiner Nachbarschaft besaß, fälschlicherweise als Randpixel interpretiert wurde.

Das Strukturelement SE hatte folgende Form

$$SE = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix} \tag{5.2}$$

Die *Dilatation* ist die Menge der Punkte x, für die das SE die durch das Bild definierte Menge X berührt, wenn sich sein Bezugspunkt an der Stelle X befindet.

$$dilatation_{SE}(X) = \{x | SE_x \cap X \neq 0\} \tag{5.3}$$

Alternativ kann der dilatierte Wert auch als der größte Grauwert in einer durch das SE definierten Umgebung definiert werden:

```
1 for (int x=1; x < A.shape()(0)-1; x++)
2   for (int y=1; y < A.shape()(1)-1; y++)
3     B(x,y)= max( in(Range(x-1,x+1),Range(y-1,y+1))*SE );
```

Abbildung 5.5 zeigt ausgewählte Regionen, welche durch die Dilatation homogenisiert wurden. Die roten Markierungen stellen den jeweiligen Flächenschwerpunkt Z, den Punkt mit der größten Entfernung zum Schwerpunkt E_1 und den Punkt auf der Längsseite des Objektes, der orthogonal zu der Geraden $\overline{ZE_1}$ liegt dar.

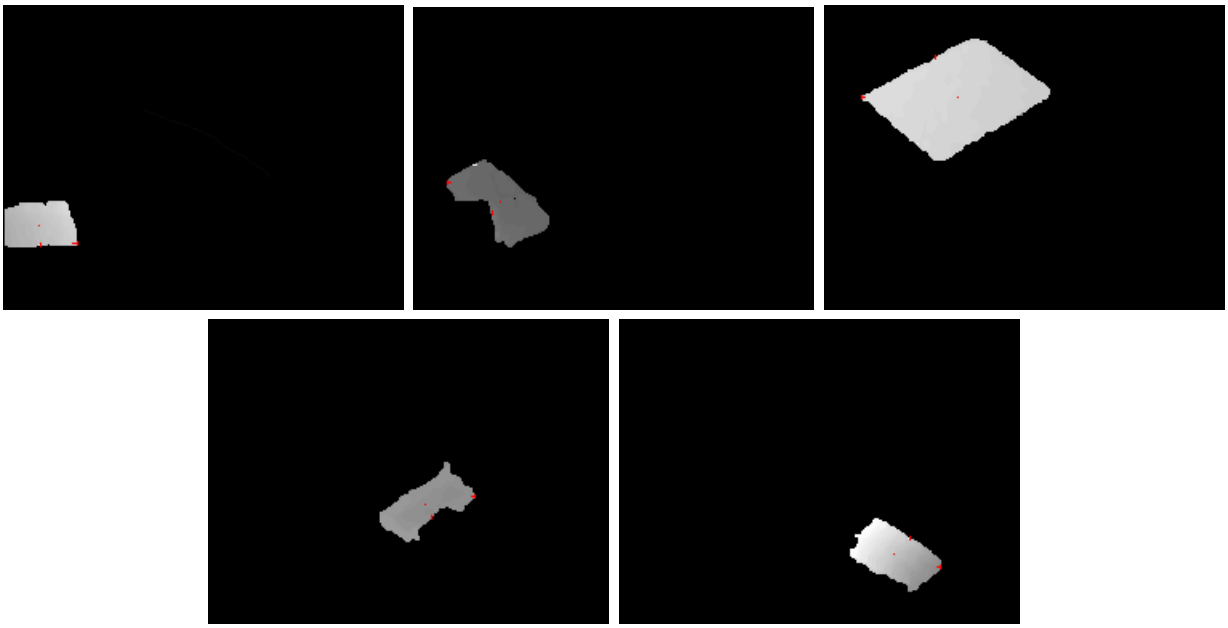


Abbildung 5.5: Segmentierte und dilatierte Flächen. Die roten Markierungen zeigen den Schwerpunkt, einen Eckpunkt und den Punkt orthogonal zu den Beiden.

5.4 Kollisionstest

Bei Schräglage der Kisten in Z-Richtung wurde mehr als eine Seite einer Kiste eingescannt. Um herauszufinden welche rekonstruierten Kisten eigentlich zwei oder drei Teilansichten ein und derselben Kiste waren, wurden sämtliche Kisten auf Kollision mit den Schwerpunkten von vermeintlichen Kollisionspartnern gerprüft. Um eine Liste aller möglichen Kollisionspartner zu erhalten wurde um jede Kiste eine Boundingbox gelegt. Als Radius der Boundingbox wurde die maximale Distanz von Zentrum zu den Eckpunkten gewählt. Kisten, deren Boundingboxen sich schnitten wurden als potentielle Kandidaten ermittelt.
Um herauszufinden ob der Schwerpunkt einer Kiste in dem Volumen einer anderen liegt wurden die Kisten in den Ursprung verschoben und die Eckpunkte, sowie der Vektor zum Schwerpunkt der Nachbarkiste D in das lokale Koordinatensystem der größeren Kisten transformiert. Durch die Transformation ins lokale Koordinatensystem wurde die Scherung ausgeglichen und die Eckpunkte lagen hier alle gleich weit vom Zentrum entfernt. Die Kiste mit dem größeren Volumen wurde gewählt, da hier durch die größere Oberfläche die Wahrscheinlichkeit einer besseren Segmentation gegeben war.

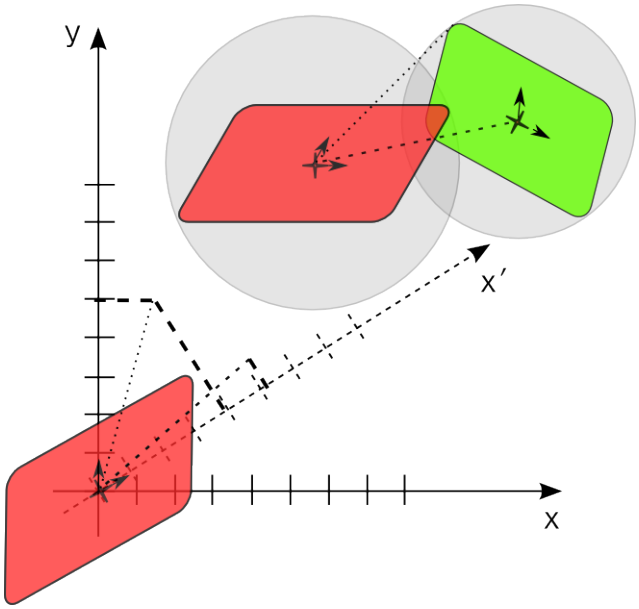


Abbildung 5.6: Transformation der Weltkoordinaten in lokale Koordinaten

Nun musste nur noch getestet werden, ob der Betrag der X-, Y- oder Z-Richtung größer als der jeweilige Betrag von D war. Falls D in allen Komponenten kleiner als die Kistenkoordinaten war, so lag das Zentrum der kleinen Kiste innerhalb der großen Kiste. Zur Optimierung der größeren Kiste wurde nun der Mittelwert beider lokalen Koordinatensysteme ermittelt und mit den lokalen Eckpunkten multipliziert.

Wurde keine Kollision der Schwerpunkte ermittelt wurde im nächsten Schritt geprüft, ob eine Überdeckung einer Kiste durch die andere vorlag. Dazu wurde der Vektor vom Zentrum der größeren Kiste zum höchsten Eckpunkt der benachbarten Kiste ebenfalls ins lokale Koordinatensystem der größeren Kiste transformiert. Lagen die $X-$und $Y-$ Koordinaten innerhalb der größeren Kiste aber die Höhe außerhalb so wurde die Kiste durch ihren Nachbarn überdeckt.
Abbildung 5.6 verbildlicht die Vorgehensweise, in diesem Fall wird erkannt, dass die beiden Kisten nicht zusammengehören. Die beiden gestrichelten Geraden stellen die Zentrendistanz, sowie die Distanz vom Zentrum bis zum höchsten Punkt der benachbarten Kiste dar. Die lokalen Basen sind durch Pfeile vom Kistenzentrum ausgehend dargestellt.

5.4.1 Koordinatentransformation

Eine Koordinatentransformation wird nach folgender Formel berechnet:

$$A \cdot \vec{x} = A' \cdot \vec{x}' \Rightarrow A'^{-1}A \cdot \vec{x} = A'^{-1}A' \cdot \vec{x}' \Rightarrow A'^{-1}A \cdot \vec{x} = \vec{x}'$$

$\vec{x}$ stellt die Eckpunkte der Objekte in karthesischen Koordinaten dar. Die Matrix A stellt die Basis des Karthesischen Koordinatensystems dar, also die Einheitsmatrix I und kann somit weggelassen werden. $\vec{x}'$ und A' stehen analog für lokale Basis und lokale Objektkoordinaten. Die Gleichung kann somit folgendermaßen zusammengefasst werden:

$$A'^{-1} \cdot \vec{x} = \left(\begin{pmatrix} a'_{00} \\ a'_{10} \\ a'_{20} \end{pmatrix} \quad \begin{pmatrix} a'_{01} \\ a'_{11} \\ a'_{21} \end{pmatrix} \quad \begin{pmatrix} a'_{02} \\ a'_{12} \\ a'_{22} \end{pmatrix} \right)^{-1} \cdot \vec{x} = \vec{x}' \tag{5.4}$$

A^{-1} stellt die Inverse Matrix dar und kann für den dreidimensionalen Fall aus den cramerschen Regeln wie folgt berechnet werden:

$$A^{-1} = \begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix}^{-1} = \frac{1}{\det(A)} \begin{pmatrix} a_{11}a_{22} - a_{12}a_{21} & a_{02}a_{21} - a_{01}a_{22} & a_{01}a_{12} - a_{02}a_{11} \\ a_{12}a_{20} - a_{10}a_{22} & a_{00}a_{22} - a_{02}a_{20} & a_{02}a_{10} - a_{00}a_{12} \\ a_{10}a_{21} - a_{11}a_{20} & a_{01}a_{20} - a_{00}a_{21} & a_{00}a_{11} - a_{01}a_{10} \end{pmatrix}^{-1}$$

Die Determinante für eine reguläre $n \times n$-Matrix ist folgendermaßen definiert:

$$\det(A) = \sum_{\sigma \in S_n} \left(sgn(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)} \right)$$

Die Summe wird über alle Permutationen σ der Zahlen $\{1, \ldots, n\}$ berechnet. sgn bezeichnet das Vorzeichen der Permutation. Ein positives Vorzeichen stellt dabei eine gerade Anzahl von Vertauschungen dar.
Um die Ergebnisse mit den Originaldaten vergleichen zu können wurden sie in einer .vtk Datei gespeichert, die mittels folgendem Code erstellt wurde.

Listing 5.1: list2vtk

```
1  void list2vtk(string Dateivtk, list<vector<Array<float,1> > > normalen, int Typ)
2    {
3    int size=normalen.size();
4    int x=6;
5    if (Typ==12) x=10;                    //Typ 9 = Flaeche, Typ 12 = Quader
6    ofstream vtkFile(Dateivtk.c_str());
7    vtkFile<<"# vtk DataFile Version 2.0\ngenerate Normals\n";
8    vtkFile<<"ASCII\nDATASET UNSTRUCTURED_GRID\n";
9    vtkFile<<"\nPOINTS "<<x*size+4<<" float\n";
10
11   list<vector<blitz::Array<float,1> > >::iterator it;
12   for (it=normalen.begin(); it != normalen.end(); it++ )
13      {
14      blitz::Array<float,1> Pos(3), Nrml(3);
15      Pos=(*it)[0];
16      Nrml=(*it)[1]*20;
17
18      vtkFile<<Pos(2)<<" "<<Pos(0)<<" "<<Pos(1)<<"  ";
19      vtkFile<<Pos(2)+Nrml(2)<<" "<<Pos(0)+Nrml(0)<<" "<<Pos(1)+Nrml(1)<<"\n";
20      for (int i=2; i<x; i++)
21        vtkFile<<(*it)[i](2)<<" "<<(*it)[i](0)<<" "<<(*it)[0](1)<<"\n";
22      }
23   vtkFile<<"0 280 0  0 0 0  0 0 250  0 280 250\n";        // Bodenkoordinaten
24
25   //Zusammengehoerende Koordinaten
26   vtkFile<<"\nCELLS "<<2*size+1<<" "<<(x+2)*size+5;
27   for (int i=0; i<size; i++)
28      {
29      vtkFile<<"\n2 "<<x*i<<" "<<x*i+1<<"\n"<<x-2;
30      for (int ii=2; ii<x; ii++) vtkFile<<" "<<x*i+ii;
31      }
32   vtkFile<<"\n4 "<<x*size<<" "<<x*size+1<<" "<<x*size+2<<" "<<x*size+3<<"\n"
33   vtkFile<<"\nCELL_TYPES "<<2*size+1<<"\n";
34   for (int i=0; i<size; i++)
35      vtkFile<<"3 "<<Typ<<"\n";
36
37   vtkFile<<"9\n";
38   vtkFile<<"\nPOINT_DATA "<<x*size+4<<"\nSCALARS MyScalars float 1\n";
```

```
39    vtkFile<<"LOOKUP_TABLE default\n";
40    for (int i=1; i<(size+1); i++ )
41        for (int ii=0; ii<x; ii++) vtkFile<<x*i<<" ";
42    vtkFile<<"0 0 0 0";     // Bodenfarbe
43
44    vtkFile.close();
45    }
```

In Abb. 5.7 können nun die Ergebnisse der Rekonstruktion direkt mit den segmentierten Daten verglichen werden. Als Bildbetrachter diente Paraview [Par08]. Die Oberflächen wurden mit unterschiedlichen Farben versehen, um sie besser voneinander unterscheiden zu können. Die segmentierten Daten sind als Gittermodell dargestellt.

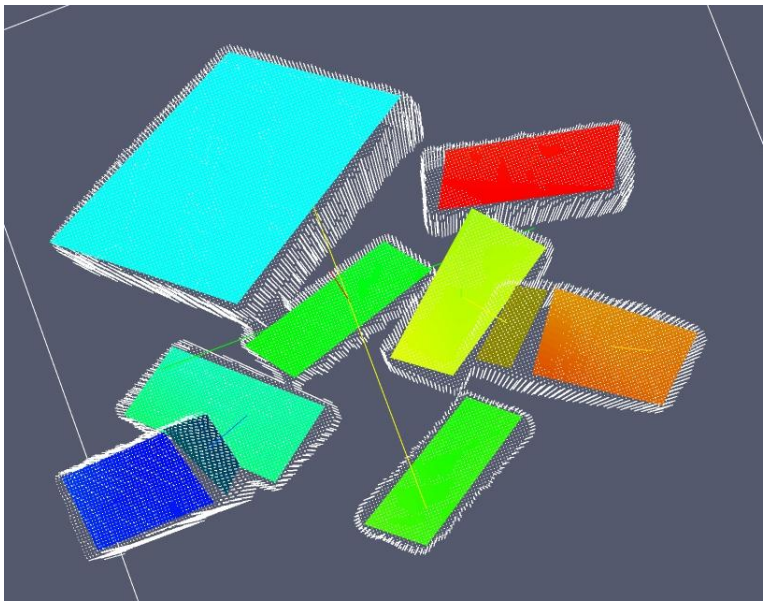


Abbildung 5.7: Visualisierung der segmentierten Oberflächen, sowie der Originaldaten als Gittermodell

Kapitel 6

Ergebnisse

Es wurden 3 bis 13 Kisten nach der Beschreibung im Versuchsaufbau eingescannt und die Tiefenbilder wie im letzten Kapitel beschrieben bearbeitet.
Bei optischer Evaluation der Algorithmen ist eine hohe Übereinstimmung zwischen den segmentierten Daten und den rekonstruierten Oberflächen der Objekte zu erkennen. Kleine Abweichungen konnten durch nicht exakt gefundenen Eckpunkte entstehen. Außerem traten Fehlrekonstruktionen bei nur teilweise eingescannten Objekten in Randlagen auf (siehe Abb. 6.1b unten rechts). Zu beachten ist allerdings auch, dass die gerenderten Daten nicht die exakte Position der einzelnen Pixel widerspiegeln und Abweichungen zwischen den segmentierten Daten zu den Originaldaten auftreten können. Dennoch scheint die Ausrichtung im Raum und der Schwerpunkt der Kisten korrekt gefunden worden zu sein. Dies ist für eine spätere Ergreifung der Kisten mittels des Roboterarms von entscheidender Bedeutung, damit die Vakuumgreifer die Objekte sicher transportieren können.

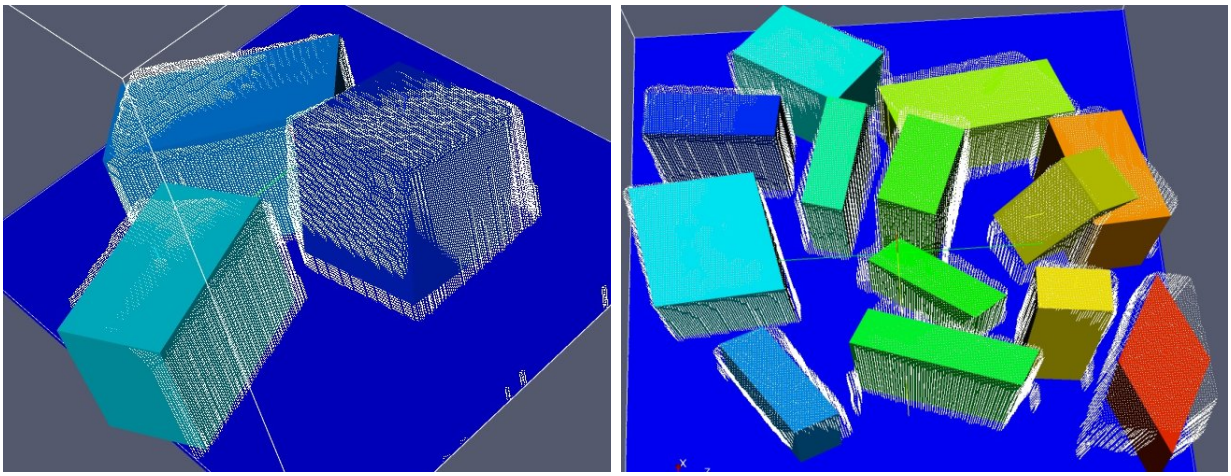


Abbildung 6.1: a) b) Die segmentierten Daten sind als Drahtgitter gerendert ihre rekonstruierten Kistenmodelle sind als bunte Quadrate abgebildet

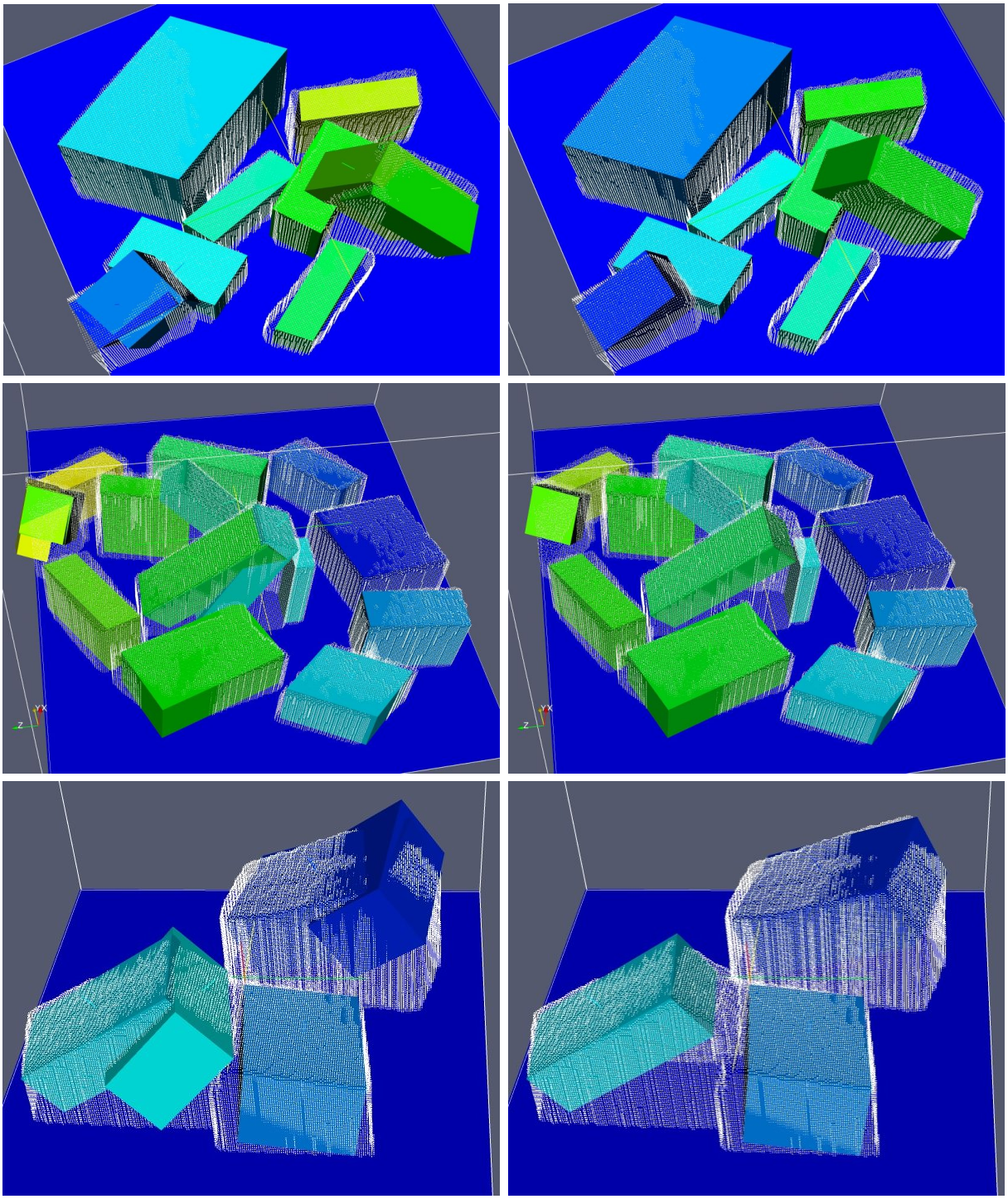


Abbildung 6.2: Die segmentierten Daten sind als Drahtgitter gerendert, ihre rekonstruierten Kistenmodelle sind als bunte Quadrate abgebildet (links jeweils alle segmentierten Flächen und die daraus rekonstruierten Quadrate, rechts wurden die redundanten Quadrate entfernt und die Lage der dazugehörigen Objekte justiert.)

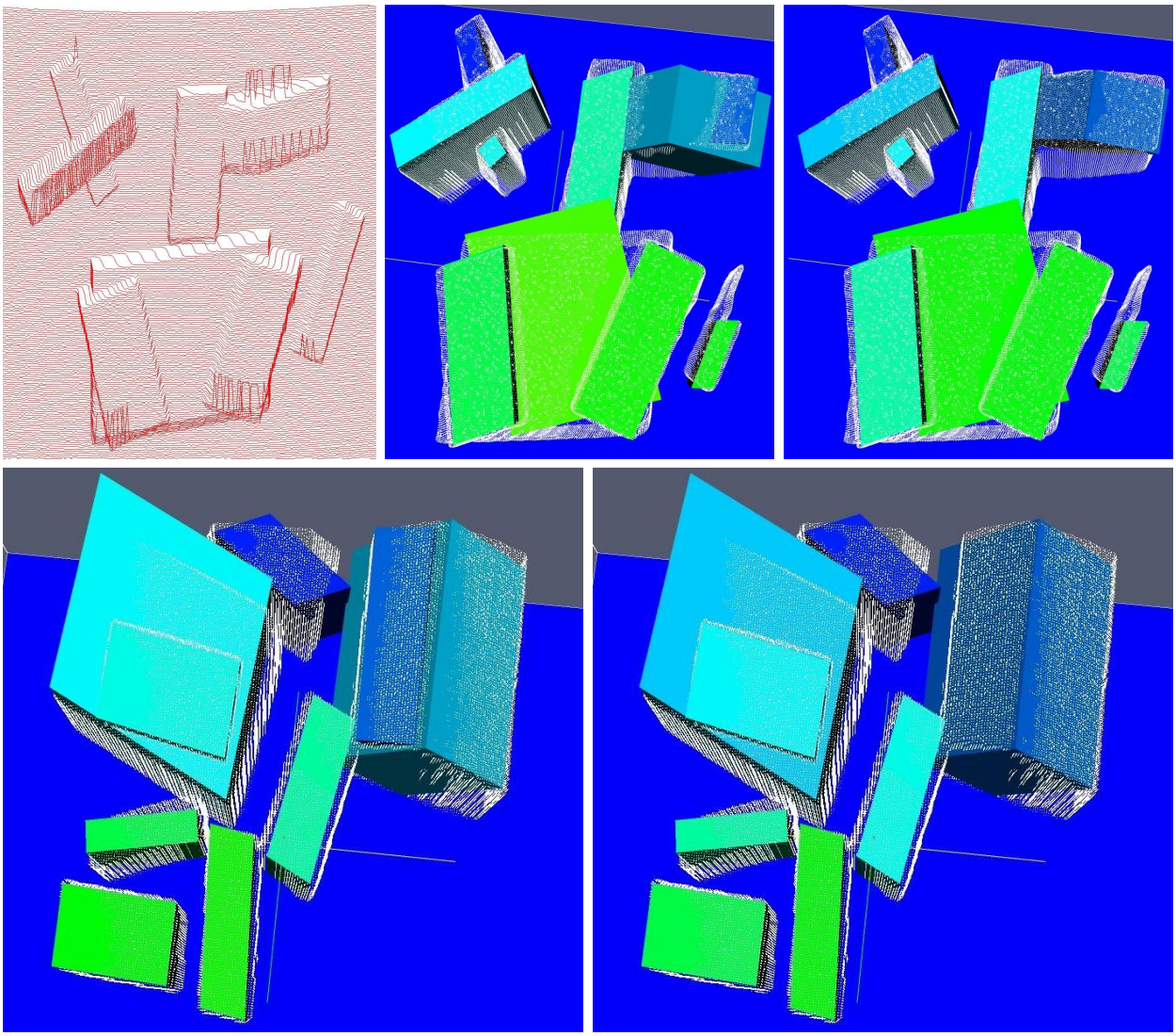


Abbildung 6.3: Fehlerhafte Segmentierung bei zu großen Teilen überdeckten Kisten. In der oberen Reihe wurde ein Bild ohne Korrektur des Tiefenbildes verwendet, weswegen die Kiste am rechten äußeren Rand nur teilweise rekonstruiert werden konnte. Die fehlende Korrektur ist ebenso an den schräg interpretierten senkrechten Kistenflächen zu erkennen, welche bei einer Draufsicht nicht sichtbar sein dürften (rechts sind jeweils die Ergebnisbilder).

6.1 Laufzeit

Der Scanvorgang dauert je nach eingestellten Geschwindigkeit des Roboterarms 2 - 5 Minuten. Die schnellste Zeit wird bei 50 % der Maximalgeschwindigkeit erzielt, erhöht aber auch das Risiko von Scanabbrüchen aufgrund zu hoher Vibrationen. Die Segmentierung und Lageerkennung benötigt abhängig von der Anzahl der Objekte 2 - 3 Minuten. Das Auffinden des Schwerpunktes und der Flächengradienten wird im Markov-Netz durch Addition der Werte bei jedem neuen Knoten einer Region bewerkstelligt und fällt so nicht ins Gewicht.

Kapitel 7

Zusammenfassung und Ausblick

Ziel dieser Arbeit war die Segmentierung der Scanbilder in einzelne Kisten anhand von 2D-Tiefenbildern, sowie die Ermittlung der Ausdehnung und Lage der Objekte im dreidimensionalen Raum. Durch die Lage der Objekte konnten freistehende Kisten von nichtfreistehenden Kisten unterschieden werden. So ist eine Möglichkeit zur späteren Implementierung einer automatischen Kistenergreifung durch den Roboterarm geboten. Dabei muss jedoch beachtet werden, dass falls überdeckte Kisten vorhanden nach jeder entfernten Kiste – welche eine andere überdeckt hat – ein neuer Tiefenscan vorgenommen werden.

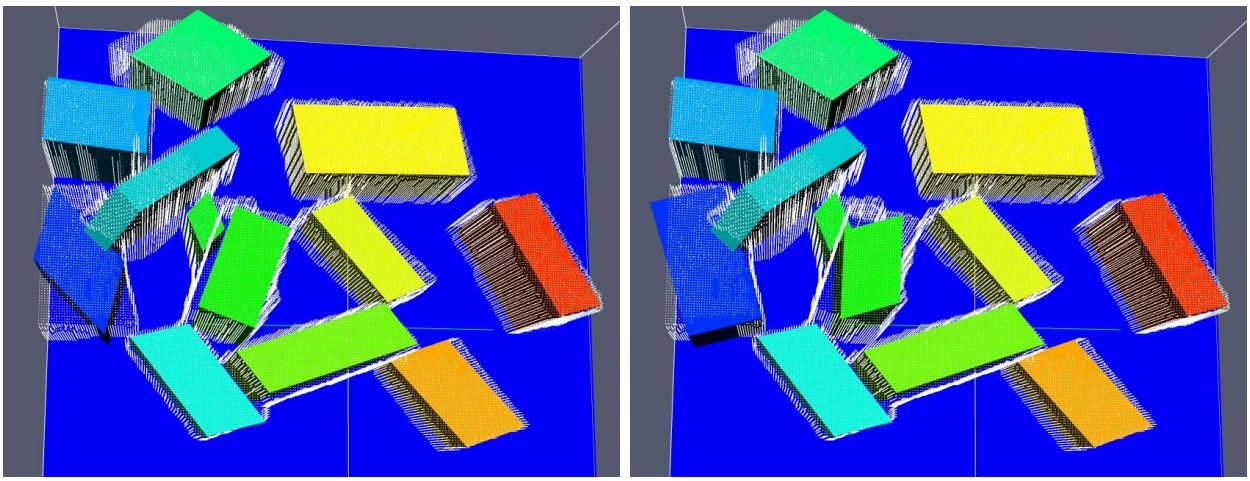


Abbildung 7.1: Rekonstruktion der Kisten mit jeweils auf- und abgerundeten Schwerpunktkoordinaten.

Probleme traten bei Kisten mit Teilüberdeckung oder am Rand liegenden Kisten, welche nur teilweise eingescannt wurden auf.
Abbildung 7.1 zeigt zwei qualitativ unterschiedliche Ergebnisse, wenn der Schwerpunkt jeweils ein Pixel nach rechts bzw. links verschoben wurde. Hier ist also noch Optimierungspotential vorhanden. Unterschiedliche Eckpunkte können somit mehrmals mit leicht verschobenen Schwerpunkten berechnet werden. Anschließend können die Ergebnisse dann auf Passgenauigkeit mit den Originaldaten geprüft werden. Das Modell, welches am besten mit den Ausgangsdaten korreliert ist dann das Modell der Wahl. Da die segmentierten und gerenderten Gittermodelle aber nicht die Originaldaten abbilden, müssen zum Erhalt der Originaldaten die einzelnen Objekte vermessen und auf die Scanauflösung umgerechnet werden, was ein nicht zu unterschätzenden Aufwand darstellt. Unter der Annahme, dass der Laserscanner die Objekte wirklichkeitsgetreu abbildet, bzw. bei gleicher Distanz der Scanpunkte in Längs- und Querrichtung müssten die von den Tiefenbilder dargestellten Objekte manuell vermessen werden.
In Abb. 7.2 links sind die Messfehler an Objektkanten und insbesondere an schrägen Objektkanten zu sehen. Durch den Einsatz eines Medianfilters konnten diese jedoch weitgehend eliminiert werden, ohne die Konturen zu verwischen.

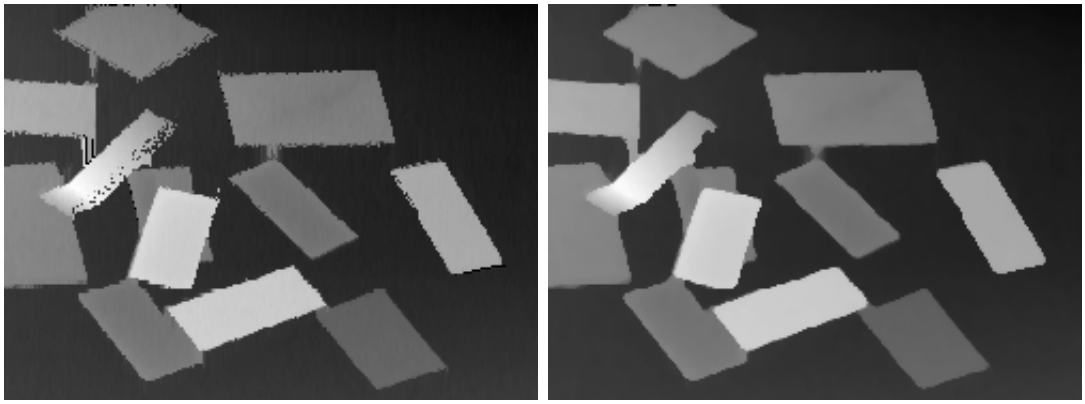


Abbildung 7.2: Tiefenbild der Objekte aus 7.1 (links), sowie das mit einem 2×2-Median geglättete Tiefenbild.

7.1 Übersicht

Um einen umfassenden Überblick der einzelnen Schritte von den Tiefenbildern bis zur fertigen Rekonstruktion zu erhalten werden diese nochmals in Abb. 7.3 dargestellt.

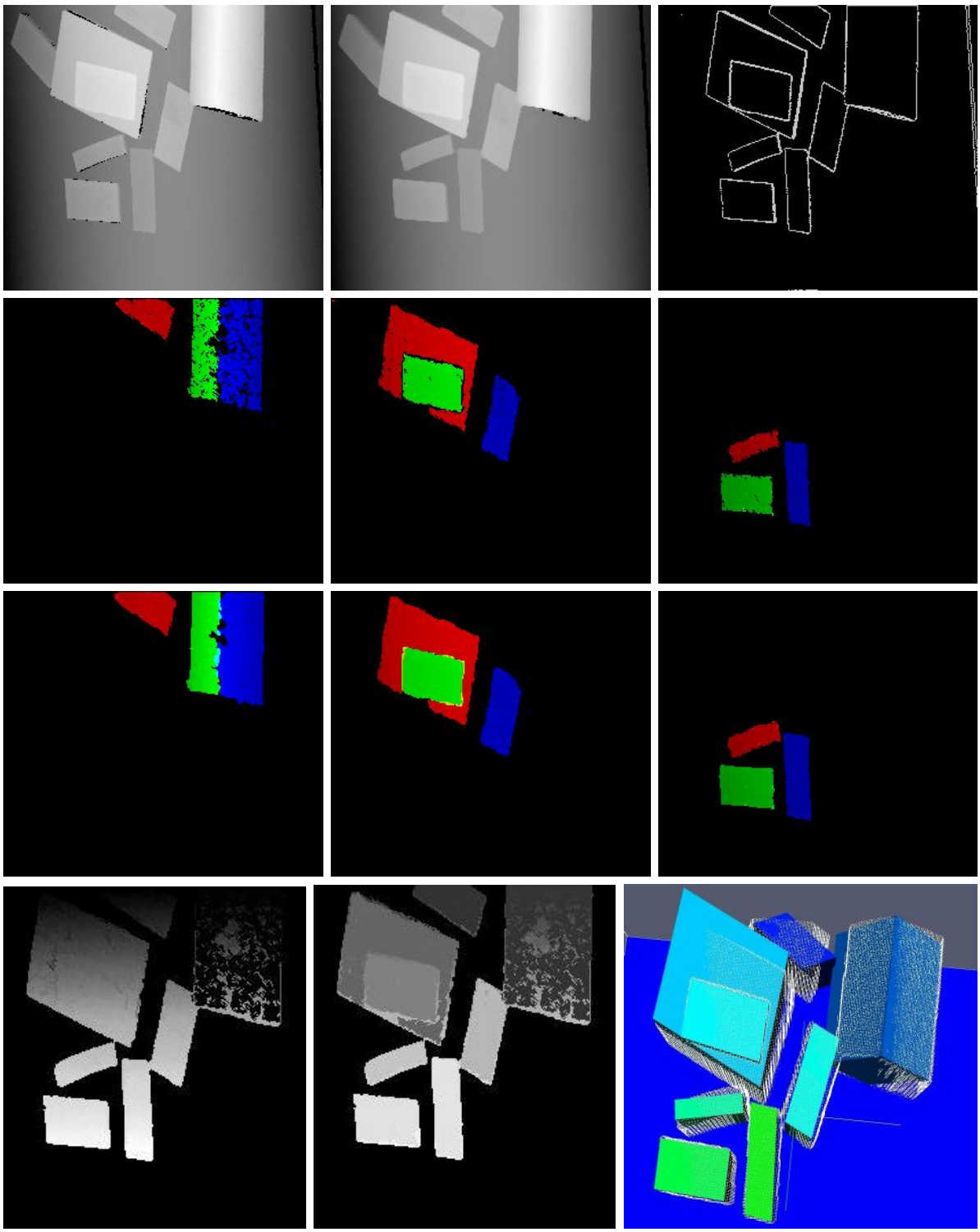


Abbildung 7.3: Zusammenfassend sind alle Schritte abgebildet (von oben links nach unten rechts): Originalbild, Median, Kantenbild, Segmentierung, Dilatation, Labelverteilung und Rekonstruktion im Vergleich mit dem segmentierten Gittermodell.

Anhang A

Verwendete Software und Bibliotheken

A.1 ImageJ

ImageJ [Ima08] ist eine Java-basierte und damit plattformübergreifende OpenSource Software zur Bildbearbeitung. Diese Software stellt eine Vielzahl von Funktionen und Filtern bereit und lässt sich durch Plugins den eigenen Wünschen anpassen. Anhand dieses Programms wurden verschiedene Filter auf den Ausgangsbildern erprobt, um eine schnelle qualitative Bestimmung des Filters zu erhalten, der die besten Ergebnisse bei der Weiterverarbeitung lieferte. Dieser wurde dann nachprogrammiert.

A.2 Inkscape

Inkscape [Ink08] ist ein freies vektorbasiertes Grafik- und Zeichenprogramm, welches sich zum Erstellen von Diagrammen und beweglichen Verbindungen eignet. Desweiteren werden geometrische Formen wie Rechteck, Stern, Ellipse oder Spirale zur Verfügung gestellt. Diese können unter anderem rotiert oder geschert werden. Zur Erzeugung von Sinuskurven können Spiralteile benutzt werden.

A.3 Paraview

Paraview [Par08] ist ein OpenSource Programm, welches zur Visualisierung große Datensätze dient. Es ermöglicht unter anderem das Rendern von Daten und 3D-Datensätzen, sowie dem Darstellen von .vtk-Dateien.

A.4 GDImage

GD [GD08] ist eine quelloffene Bibliothek zum Erstellen von PNG, JPEG, GIF und anderen Bilddateien. Zudem können Linien und Kreise erstellt, sowie verschiedene Schriftgrößen verwendet werden. Mit Hilfe dieser Bibliothek wurden die Linienbilder der Tiefenwerte erstellt.

A.5 Blitz++

Blitz++ [Bli08] ist eine Bibliothek mit C++ Funktionen, welche die Arbeit mit Matrizen und Vektoren erleichtert. Diese Bibliothek stellt templatebasierte und geschwindigkeitsoptimierte Operationen auf Matrizen und Vektoren zur Verfügung.

A.6 NCetCDF

Network Common Data Format (NetCDF) [Net08] ist ein Dateiformat, welches sich für den Austausch wissenschaftlicher Daten eignet. Die Daten selbst sind als n-dimensionale Arrays abgelegt, was einen schnellen Zugriff ermöglicht.

A.7 libRMF

libRMF [lib08] ist eine universitätsinterne Bibliothek, welche ein Random Markov Field zur Verfügung stellt.

Literaturverzeichnis

[AG] SICK AG. Betriebsanleitung laser-messsystem lms 400. www.sick.cz/cz/novinky/identifikace/lms400/data/cs.toolboxpar.0005.file.tmp/LMS400 manual_DE.pdf.

[Bau] Sven Baumann. Seminar cbir - albert ludwig universität freiburg. http://lmb.informatik.uni-freiburg.de/lectures/bild_seminar/data/04-tex.pdf.

[Bli08] Blitz++ user's guide (version 0.9) und bibliothek. www.oonumerics.org/blitz, 2008.

[CG92] George Casella and Edward I. George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

[DH72] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, 1972.

[FG96] Nir Friedman and Moises Goldszmidt. *Learning Bayesian Networks with Local Structure*. 1996.

[Fin03] Gernot A. Fink. *Mustererkennung mit Markov-Modellen*. B. G. Teubner, Stuttgart – Leipzig – Wiesbaden, 2003.

[GD08] Gdimage, programm und dokumentation. www.libgd.org, 2008.

[GG84] S. Geman and D. Geman. *Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1984.

[GW92] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1992.

[Ima08] Imagej, programm und dokumentation. http://rsb.info.nih.gov/ij, 2008.

[Ink08] Inkscape, programm und dokumentation.
www.inkscape.org, 2008.

[Isi25] Ernst Ising. *Beitrag zur Theorie des Ferromagnetismus*. Z. Phys. 31, 1925.

[KGV83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science, Number 4598, 13 May 1983*, 220, 4598:671–680, 1983.

[lib08] librmf, bibliothek.
http://lmb.informatik.uni-freiburg.de/lmbwiki/index.php/Janis_Software_Stuff#libRMF, 2008.

[LMP01] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[MRR+53] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[Net08] Netcdf (bibliothek und datenformat).
www.unidata.ucar.edu/software/netcdf, 2008.

[Par08] Paraview user's guide (version 1.6) und programm.
www.paraview.org/HTML/Index.html, 2008.

[Pot52] Renfrey B. Potts. *Some Generalized Order-Disorder Transformations, Proceedings of the Cambridge Philosophical Society*. Vol. 48 pp. 106-109, 1952.

[TCK] B. Taskar, V. Chatalbashev, and D. Koller. Learning associative markov networks.

[Vek99] Olga Veksler. *Efficient graph-based energy minimization methods in computer vision. PhD thesis, Department of Computer Science, Cornell University*. 1999.

[VS91] Lee Vincent and Pierre Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE PAMI, 1991*, 13(6):583–598, 1991.