Strukturelle Ansätze für die Stereorekonstruktion

Dissertation

zur Erlangung des akademischen Grades Doktoringenieur (Dr.-Ing.)

vorgelegt an der
Technischen Universität Dresden
Fakultät Informatik

eingereicht von

Dipl.-Ing. Dmytro Shlezinger
geboren am 29.03.1970 in Kiew, Ukraine

Gutachter:

Prof. Dr.-Ing. habil. Siegfried Fuchs.	Technische Universität Dresden, Fakultät Informatik, Institut für Künstliche Intelligenz.
Prof. Dr. Georgy Gimel'farb.	University of Auckland, Tamaki Campus, Dept. of Computer Science.
Prof. Dr. Vaclav Hlavac.	Czech Technical University, Faculty of Electrical Engineering, Department of Cybernetics.

Tag der Verteidigung: 18. Juli 2005.

Dresden, den 30. August 2005

Inhaltsverzeichnis

Kapitel 1. Einführung 5

Kapitel 2. Formulierung der Aufgabe der Stereorekonstruktion als strukturelles Problem 11
2.1. Geometrische Grundlagen 11
2.2. Formulierung der Aufgabe der Stereorekonstruktion als Aufgabe der Energieminimierung 15
2.3. Gibbs-Wahrscheinlichkeitsverteilung als Oberflächenmodell 17
2.4. Bedingte Wahrscheinlichkeit 21

Kapitel 3. Theoretische Grundlagen und Algorithmen zur Lösung struktureller Probleme 25
3.1. MinSum – Monotonintervalartigkeit 25
3.2. MinSum – Maximieren der scheinbaren Qualität 27
3.3. MinSum – Überführung in eine MinCut-MaxFlow Aufgabe 33
3.4. Schätzung der marginalen Wahrscheinlichkeitsverteilungen 47
3.5. Wahl der Parameter 50

Kapitel 4. Implementierung und Experimente 57
4.1. Modell 57
4.2. Algorithmus 58
4.3. Verwendung der Maximum a-posteriori Entscheidung 67
4.4. Experimente 77

Kapitel 5. Zusammenfassung und Ausblick 95
5.1. Theoretische Fragen 95
5.2. Praktische Anwendungen 97

Literaturverzeichnis 101

KAPITEL 1

Einführung

Unter 3D-Rekonstruktion versteht man oft die Bestimmung der Oberfläche eines dreidimensionalen Objektes. Es existieren mehrere Verfahren zur Lösung dieser Aufgabe. Einige davon sind zum Beispiel: Direkte Tiefenmessung durch ein Lasergerät, Benutzung strukturierter Beleuchtung, Shape from Shading, Shape from Texture usw. Jedes dieser Verfahren hat seine Vorteile und Nachteile. Einige benötigen spezielle Geräte (ein Lasergerät, einen Streifenprojektor usw.). Die Verfahren, die nur auf Bildverarbeitungsmethoden basieren, sind in der Regel nicht robust gegen Störungen. Die Genauigkeit hängt sehr stark vom benutzten Reflexionsmodell ab (zum Beispiel bei Shape from Shading). Manchmal werden für solche Methoden auch gewisse Objektkentnisse (zum Beispiel das Farbmodell der Oberfläche bei Shape from Texture) benötigt.

Die Stereorekonstruktion und insbesondere die binokulare Stereorekonstruktion nimmt einen wichtigen Platz in der obigen Reihe ein. Dieser Ansatz gründet sich auf der folgenden Idee. Ein Punkt im dreidimensionalen Raum (nennen wir ihn Weltpunkt) wird eindeutig in zwei Bildebenen abgebildet. Kennt man die Lage dieser zwei Projektionspunkte, so kann man die Lage des Weltpunktes zurückrechnen. Somit besteht die Stereorekonstruktion in der Bestimmung von Paaren von Projektionspunkten. Diese Punkte nennt man „Korrespondierende“ Punkte und die Aufgabe nennt man „Korrespondenzproblem“.

Vor allem setzt man dabei voraus, dass korrespondierende Pixel in einem gewissen Sinn ähnlich sein sollen. Somit wird ein Ähnlichkeitsmaß definiert. Die einfachsten Methoden benutzen keine weiteren Annahmen. Verbal lassen sich diese Methoden auf folgende Art beschreiben. Man finde für jedes Pixel des ersten Bildes das ähnlichste Pixel des zweiten Bildes entsprechend dem ausgewählten Ähnlichkeitsmaß. Solche Methoden nennt man „Block Matching“. Ein sehr großer Nachteil dabei ist, dass diese Methoden meistens nicht robust gegen Störungen sind. Es gibt noch einen weiteren Schwachpunkt. Es kann passieren, dass Pixel im ersten Bild (sogar für ein ideales Stereopaar ohne Störungen) existieren, die mehrere „passende“ Korrespondenzpunkte im zweiten Bild haben (diese Situation nennen wir „Uneindeutigkeit“). In diesem Fall kann man die Korrespondenzpunkte nur auf Grund des Ähnlichkeitsmaßes nicht unterscheiden.

Einige Verfahren zur Lösung des Korrespondenzproblems basieren auf dem Begriff „Spezielle Punkte“. Das bedeutet, dass man nicht für alle Pixel des ersten Bildes die korrespondierenden Punkte bestimmt, sondern nur für diejenigen, für die das korrespondierende Pixel des zweiten Bildes einfach zu bestimmen ist. Eine einfache Variante wäre zum Beispiel, die Aufgabe nur für die Punkte des ersten Bildes zu lösen, die einen sehr großen Farbgradient haben. Mit anderen Worten, wird als Vorverarbeitung ein Kantendetektor angewendet. Danach löst man das Korrespondenzproblem nur für die Punkte, in denen die Antwort des Kantendetektors einen vorgegebenen Schwellwert überschreitet. Wenn keine Uneindeutigkeiten auftreten, sind solche einfachen Methoden in der Regel robuster gegen Störungen, als die Block Matching Verfahren. Das Problem mit den nicht eindeutigen Pixeln wird dabei allerdings nicht gut gelöst. Streng genommen, muss man die Menge der

speziellen Punkte so wählen, dass keine Uneindeutigkeiten auftreten. Dafür braucht man in der Regel zusätzliche Annahmen über die zu rekonstruierende Szene.

Ein weiterer Nachteil solcher Verfahren ist, dass man mit Hilfe der oben beschriebenen Strategie nicht die ganze Oberfläche findet, sondern nur eine Teilmenge der Weltpunkte. Danach muss man eine Approximation der realen Oberfläche finden, die durch die vorher gefundenen Punkte läuft. Letztendlich wird somit nur eine Approximation der gesuchten Oberfläche gefunden. Die Forderungen für die Robustheit und die Genauigkeit der gefundenen Oberfläche bilden offensichtlich einen Widerspruch. Je robuster das Verfahren sein soll, desto geringer wird die Anzahl der betrachteten speziellen Punkte. Somit wird die Approximation gröber.

Im Weiteren betrachten wir nur solche Methoden, die für jedes zu betrachtende Element eine Entscheidung treffen. Zu diesen Methoden gehören zum Beispiel diejenigen, in denen für jeden Punkt des ersten Bildes ein entsprechender Punkt im zweiten Bild gesucht wird. Die letzteren kann man als Aufgaben zur Suche nach einer Abbildung verstehen, die die Menge der Pixel des linken Bildes auf die Menge der Pixel des rechten Bildes abbildet (oder umgekehrt).

Um die Robustheit solcher Verfahren zu erhöhen, werden zusätzliche Beschränkungen für Korrespondenzpaare formuliert. Es ist offensichtlich, dass Korrespondenzkombinationen existieren, die keiner realen Oberfläche entsprechen. Betrachten wir als Beispiel die Aufgabe, in der genau eine Oberfläche ohne Unstetigkeiten (d.h. die Digitalisierung einer kontinuierlichen stetigen Oberfläche) gesucht wird. Sei das Bildpaar bereits rektifiziert, d.h. die korrespondierenden Punkte haben in den Bildern gleiche y-Koordinaten (ausführlicher betrachten wir dies im nachfolgenden Kapitel). Nehmen wir an, dass ein Pixel des ersten Bildes mit den Koordinaten (x_l, y_l) das korrespondierende Pixel des zweiten Bildes mit den Koordinaten (x_r, y_r) hat. Dann ist das Korrespondenzpaar $\big((x_l+1, y_l), (x_r-1, y_r)\big)$ unmöglich. Somit hängt die Entscheidung, wo der korrespondierende Punkt für ein gegebenes Pixel liegt, nicht nur vom Ähnlichkeitsmaß ab, sondern auch von den Entscheidungen, die für benachbarte Punkte getroffen werden. Formal kann man die Aufgabe zum Beispiel wie folgt formulieren. Gegeben sind:

P_l, P_r	die Mengen der Pixel des linken bzw. des rechten Bildes;
$N(p) \subset P_l$	die Nachbarschaftstruktur;
$q_p : P_r \to \mathbb{R}$	das Ähnlichkeitsmaß – für jedes Pixel $p \in P_l$ eine Funktion, die jedes Korrespondenzpaar (p, p_r), $p_r \in P_r$ bewertet (wir nehmen dabei an, dass die Werte dieser Funktion desto kleiner sind, je besser die entsprechenden Korrespondenzpaare sind);
$g_{pp'} : P_r \times P_r \to \mathbb{R}$ mit $p' \in N(p)$	die zusätzlichen Beschränkungen – für jedes Paar (p, p') von benachbarten Pixeln des linken Bildes eine Funktion, die jede Korrespondenzkombination $\big((p, p_r), (p', p'_r)\big)$, $p_r, p'_r \in P_r$ bewertet.

Gesucht ist die bezüglich des Ähnlichkeitsmaßes beste Abbildung $f : P_l \to P_r$ der Menge der Pixel des linken Bildes auf die Menge der Pixel des rechten Bildes:

$$G = \min_f \Big[\sum_{p \in P_l} q_p\big(f(p)\big) + \sum_{p \in P_l} \sum_{p' \in N(p)} g_{pp'}\big(f(p), f(p')\big) \Big]. \tag{1.1}$$

Die obige Formulierung ist offensichtlich nicht die einzig mögliche. Hier besteht die Zielfunktion (der Ausdruck, der in den rechteckigen Klammern steht) aus der Summe von zwei Termen. Die Terme selbst sind Summen bestimmter „lokaler“ Funktionen q bzw. g. Derartige Formulierungen nennt man „Formulierung als Aufgabe der Energieminimierung“. Die Terme nennt man „Daten-Term“ bzw. „Modell-Term“. Im Kapitel 2 werden wir

solche Formulierungen ausführlicher betrachten. Jetzt wollen wir einen kurzen Überblick der Algorithmen zur Lösung dieser Probleme angeben.

Es ist bekannt, dass diese Aufgabe im allgemeinen Fall NP-vollständig ist. Die Worte „allgemeiner Fall“ bedeuten, dass sowohl für die Nachbarschaft als auch für die Funktion g keine Einschränkungen getroffen werden.

Ein Ausweg ist, die Aufgabe näherungsweise zu lösen. Bekannte Methoden sind zum Beispiel Simulated Annealing, Iterated Conditional Modes [**3**], eine Modifikation des Relaxation Labeling [**21**] usw. Manche dieser Methoden haben den Nachteil, dass sie sehr langsam arbeiten. Die Ergebnisse sind oft auch nicht befriedigend. Die Ursache ist, dass es in der Regel keine Sicherheit gibt, dass die gefundene Approximation das globale Extremum gut approximiert. Meistens ist die Approximationsgüte überhaupt nicht bekannt. Oder es ist bekannt, dass die Zeitkomplexität exponentiell von der Approximationsgüte abhängt. Eine interessante Lösung wurde in [**5**] vorgeschlagen. In dieser Arbeit wurde ein Algorithmus beschrieben, der die Aufgabe für bestimmte Funktionen g mit einem bekannten Approximationsfaktor löst. Eine etwas unkonventionelle Vorgehensweise zur Suche nach approximativen Lösungen ist in [**17**] beschrieben. Im Gegensatz zu anderen näherungsweisen Methoden findet das in dieser Arbeit vorgeschlagene Verfahren die Lösung exakt, aber nur für bestimmte Knoten des Basisgraphen.

Es existiert ein anderer Ausweg, nämlich entweder die Nachbarschaft oder die Klasse der Funktionen g einzuschränken. Betrachten wir die erste Möglichkeit [**11**]. Die oft in der Bildverarbeitung benutzte Nachbarschaft ist die so genannte 4-Nachbarschaft: $N(i,j) = \{(i,j+1),(i,j-1),(i+1,j),(i-1,j)\}$. Man ersetzt sie durch die Folgende: $N(i,j) = \{(i,j+1),(i,j-1)\}$. Somit wird angenommen, dass die Entscheidung für ein gegebenes Pixel nicht von Pixeln abhängt, die sich in benachbarten Zeilen befinden. In diesem Fall lässt sich die Aufgabe vereinfachen, nämlich: $G = \sum_i G_i$ mit

$$G_i = \min_{f_i} \left[\sum_j q_{(i,j)}\big(f_i(i,j)\big) + \sum_j g_{(i,j)(i,j-1)}\big(f_i(i,j), f_i(i,j-1)\big) \right].$$

Diese vereinfachte Aufgabe kann mittels Dynamischer Programmierung exakt und effizient gelöst werden.

Diese Methode erhöht die Robustheit im Vergleich zum Block Matching sehr stark. Zusätzlich ist in diesem Fall garantiert, dass die gefundene Abbildung X einer realen Oberfläche entspricht. Allerdings führt die Vereinfachung der ursprünglichen Aufgabe (1.1) dazu, dass die gefundene Oberfläche Diskontinuitäten besitzen kann, die in der tatsächlichen Oberfläche nicht existieren. Wenn die Oberfläche durch eine Funktion $z = f(x,y)$ beschrieben wird, gibt es in diesem Fall keine Garantie, dass $|f(x,y) - f(x,y \pm 1)|$ eine kleine Zahl ist. Um diesen Nachteil auszuschließen, müssen auch Beschränkungen in vertikaler Richtung des Bildes berücksichtigt werden.

Es gibt noch einen weiteren Grund, warum man die Nachbarschaft nicht einschränken sollte. Um die Werte des Ähnlichkeitsmaßes zu berechnen, wird sehr oft der Normalenvektor der Oberfläche benutzt. Bei dem zeilenweisen Ansatz erlaubt das Oberflächenmodell aber Diskontinuitäten. In diesem Fall lässt sich der Normalenvektor nicht definieren. Obwohl wir im Weiteren solche komplizierten Ähnlichkeitsmaße nicht betrachten werden, möchten wir bemerken, dass es bei dem zeilenweisen Ansatz prinzipiell nicht möglich ist, den Normalenvektor der Oberfläche zu berücksichtigen. Somit ist es notwendig, die Aufgabe der Stereorekonstruktion ohne Einschränkung der Nachbarschaft zu lösen.

Aufgaben solcher Art (wie zum Beispiel (1.1)) nennt man strukturelle Probleme. Formal lassen sie sich wie folgt formulieren. Gegeben sind:

$V = (R, E)$	ein ungerichteter Graph (nennen wir ihn Basisgraph der Aufgabe) mit der Menge R der Knoten und der Menge $E \subset R \times R$ der Kanten. Die Knoten bezeichnen wir mit $r \in R$;
K	eine endliche Zustandsmenge;
$(W, \oplus, \otimes)$	ein Semiring mit der Menge der Elemente W und den Operationen $\oplus$ und $\otimes$;
$q_r : K \to W$	für jeden Knoten r des Basisgraphen eine Bewertungsfunktion, die jedem Zustand in dem Knoten einen Wert aus der Menge W zuordnet;
$g_{rr'} : K \times K \to W$	für jede Kante $(r, r') \in E$ des Basisgraphen eine Bewertungsfunktion, die jedem Zustandspaar auf der Kante einen Wert aus der Menge W zuordnet.

Sei $f : R \to K$ eine Abbildung (Labeling) der Menge der Knoten R auf die Zustandsmenge K. Gesucht ist:

$$G = \bigoplus_f \left[\bigotimes_{r \in R} q_r\big(f(r)\big) \otimes \bigotimes_{(rr') \in E} g_{rr'}\big(f(r), f(r')\big) \right]. \tag{1.2}$$

Einige Spezialfälle davon sind (je nach Semiring $(W, \oplus, \otimes)$):

OrAnd	$(\{0,1\}, \vee, \wedge)$,
MinMax	$(\mathbb{R}, \min, \max)$,
MinSum	$(\mathbb{R}, \min, +)$,
SumProd	$(\mathbb{R}, +, \times)$.

Im Spezialfall der Stereorekonstruktion entspricht der Basisgraph oft einem zweidimensionalen Gitter (zum Beispiel der Menge der Pixel des linken Bildes), die Zustände sind Disparitäten oder Tiefenwerte (ausführlicher betrachten wir dies im Abschnitt 2.2). Die für die Stereorekonstruktion relevanten strukturellen Probleme sind MinSum (wie zum Beispiel (1.1)) und SumProd.

Einige Vorgehensweisen zu approximativen Lösungen des MinSum Problems haben wir bereits erwähnt. Eine andere Gruppe von Methoden basiert auf dem Fakt, dass manche strukturelle Probleme *für bestimmte Funktionen g* exakt lösbar sind. So formulieren die Autoren in [**5**] die Stereoaufgabe als ein MinSum Problem. Dieses Problem wird näherungsweise mit Hilfe eines iterativen Algorithmus gelöst, der in jeder Iteration ein reduziertes MinSum Problem exakt löst. Dieses wird möglich, weil man zeigen kann, dass dieses reduzierte Problem in ein so genanntes MaxFlow Problem überführbar ist. MaxFlow Probleme sind ihrerseits mit polynomialer Zeitkomplexität lösbar [**1**]. In [**22**] formulieren die Autoren das Stereokorrespondenzproblem direkt als MaxFlow Problem ohne Benutzung des Begriffs „strukturelles Problem“. Es ist aber nicht ganz klar, was für ein Modell der Oberfläche darunter gemeint ist. In [**12**] wird eine Klasse von lösbaren MinSum Problemen beschrieben. Die Autoren haben den Ansatz zur Lösung einer Segmentierungsaufgabe verwendet. Einen ausführlichen Überblick und die theoretischen Grundlagen der auf MinCut-MaxFlow basierten Verfahren findet man in [**16**]. Dabei wird allerdings nur ein Speziallfall betrachtet, nämlich wenn die Anzahl von Zustände genau zwei ist. In [**7, 26**] beschreiben die Autoren eine sehr breite Klasse von lösbaren strukturellen Problemen. Man kann leicht sehen, dass die in [**12, 16**] beschriebenen Klassen Teilklassen davon sind.

Es lässt sich zeigen, dass SumProd Probleme für die Stereorekonstruktion sogar von größerer Interesse als MinSum Probleme sind. Im Gegensatz zu MinSum Problemen, ist zur Zeit keine Klasse von SumProd Aufgaben bekannt, die mit polynomialer Zeitkomplexität lösbar sind und (unserer Meinung nach) für praktische Anwendungen relevant sind. In

dieser Arbeit benutzen wir eine approximative Methode, die auf dem so genannten Gibbs Sampler basiert. Ausführlicher betrachten wir dies im Abschnitt 3.4.

Aus theoretischer Sicht ist die vorliegende Arbeit ein Versuch, neue Methoden zur effizienten Lösung struktureller Probleme zu entwickeln. Obwohl die meisten beschriebenen Ansätze bereits bekannt sind, möchten wir sie in die Betrachtung einbeziehen, um die neuen Verfahren verständlicher darstellen zu können.

Aus praktischer Sicht ist das Ziel der Arbeit, ein Beispiel anzugeben, für welches die beschriebenen Verfahren bessere Resultate liefern, als nicht strukturelle Ansätze. Außerdem möchten wir zeigen, dass diese Methoden insbesondere für die Stereorekonstruktion gut geeignet sind.

Die Arbeit ist wie folgt gegliedert:

- Im Kapitel 2 betrachten wir die Formulierung der Aufgabe der Stereorekonstruktion als strukturelles Problem. Dabei werden verschiedene Varianten diskutiert, die zu verschiedenen strukturellen Problemen führen.
- Im Kapitel 3 geben wir Verfahren zur Lösung der entstehenden strukturellen Probleme an.
- Im Kapitel 4 beschreiben wir, wie man die angegebenen Ansätze für die Stereorekonstruktion verwenden kann. Hier betrachten wir auch verschiedene Details, die bei der Implementierung nützlich sind. Im selben Kapitel werden auch die experimentellen Ergebnisse präsentiert.
- In der Zusammenfassung diskutieren wir offene Fragen.

KAPITEL 2

Formulierung der Aufgabe der Stereorekonstruktion als strukturelles Problem

2.1. Geometrische Grundlagen

Stereoverfahren wurden in Literatur schon oft beschrieben (siehe zum Beispiel [**13**]). Deswegen wiederholen wir deren grundlegende Idee nur ganz kurz. Das Basisprinzip der Stereorekonstruktion ist in der Abbildung 2.1 skizziert. Sei $P = (X,Y,Z)$ ein Weltpunkt im dreidimensionalen Raum. Der Punkt wird von zwei Kameras beobachtet. Wenn alle Parameter der Kameras bekannt sind (die inneren Parameter und die Lagen der Kameras im dreidimensionalen Raum), kann man anhand der Koordinaten (X,Y,Z) die Koordinaten der Punkte auf dem linken bzw. rechten Bildschirm berechnen, in die der Weltpunkt abgebildet wird: $p_l = (x_l, y_l)$ bzw. $p_r = (x_r, y_r)$. Diese Punkte nennen wir Bildpunkte und das Paar (p_l, p_r) nennen wir Korrespondenzpaar. Die Abbildungen des Weltpunktes auf die Bildpunkte bezeichnen wir mit $T_l : \mathbb{R}^3 \to \mathbb{R}^2$ bzw. $T_r : \mathbb{R}^3 \to \mathbb{R}^2$. Die umgekehrte Bestimmung ist auch möglich. Kennt man die Koordinaten der zwei korrespondierenden Bildpunkte (und natürlich alle Parameter der Kameras), so kann man die dreidimensionale Lage des

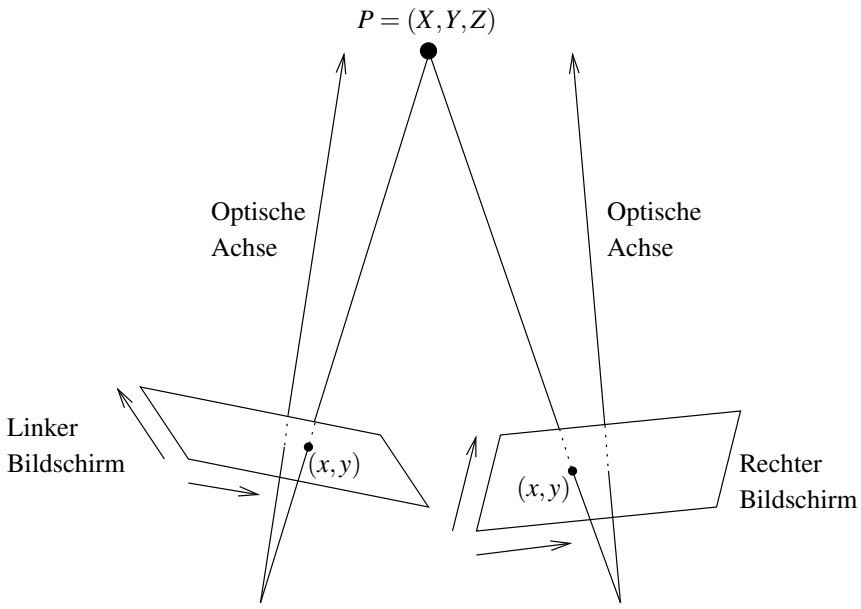


ABBILDUNG 2.1. Grundprinzip der Stereorekonstruktion

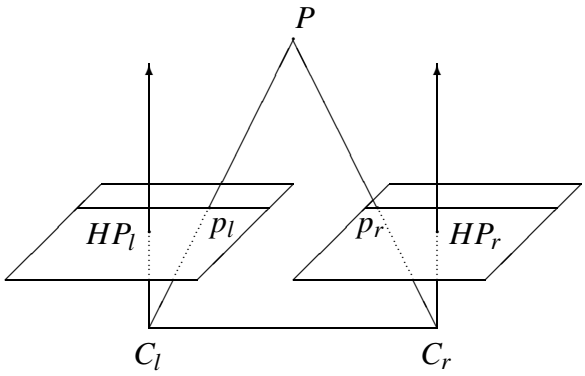


ABBILDUNG 2.2. Paralleles Stereo

Weltpunktes zurückrechnen. Somit besteht die Aufgabe der Stereorekonstruktion in der Suche nach solchen Korrespondenzpaaren.

An dieser Stelle möchten wir einen wichtigen Speziallfall betrachten, nämlich die so genannte rektifizierte Stereorekonstruktion (man nennt sie auch parallele Stereorekonstruktion). Diese Situation entsteht bei einer bestimmten Konfiguration der Kameraparameter. Eine solche Konfiguration ist in der Abbildung 2.2 schematisch gezeigt. In diesem Fall werden die folgenden geometrischen Bedingungen erfüllt:

- Die inneren Kameraparameter sind gleich. Diese Parameter sind die Brennweite, die Größe der Pixel und der Hauptpunkt (in der Abbildung mit HP_l bzw. HP_r bezeichnet).
- Die optischen Achsen sind zu einander parallel und stehen senkrecht zu der Linie, die die Projektionszentren der Kameras verbindet (die Strecke C_lC_r in der Abbildung).

Man kann leicht sehen, dass korrespondierende Punkte in diesem Fall die gleiche Bildkoordinate y haben.

Wenn alle Kameraparameter bekannt sind, gibt es keinen prinzipiellen Unterschied zwischen dem rektifizierten und dem nicht rektifizierten Fall. Deswegen betrachten wir im Weiteren nur den rektifizierten Fall. Die für die Bestimmung der Abbildungen T_l und T_r benötigten Regeln und Formeln geben wir im Kapitel 4 an.

Die so formulierte Aufgabestellung (die Suche nach Korrespondenzpaaren) ist offensichtlich noch zu allgemein. Man muss sie weiter präzisieren. Vor allem ist klar, dass bei gegebenen Kameraparametern nicht jedes Viertupel (x_l, y_l, x_r, y_r) einem Weltpunkt entspricht. Somit muss man nicht alle Viertupel von Koordinaten betrachten, sondern nur eine Teilmenge davon, d.h. die Menge der potenziell erlaubten Korrespondenzpaare. Zweitens, ist auch Folgendes zu berücksichtigen. Eine beliebige Teilmenge von Korrespondenzpaaren entspricht einer Wolke von Weltpunkten, die sich nicht immer als Oberfläche eines Objektes interpretieren lässt. Folglich ist es notwendig, zusätzliche Einschränkungen einzuführen, die garantieren, dass die gefundene Teilmenge der Korrespondenzpaare als die Oberfläche eines Objektes interpretierbar ist.

Oft spezifiziert man die gesuchte Teilmenge der Korrespondenzpaare auf folgende Art. Diese Teilmenge soll aus Korrespondenzpaaren (p_l, p_r) bestehen, so dass jedes Pixel des linken Bildes p_l genau ein mal in der Teilmenge vorhanden ist. Mit anderen Worten, ist für jedes Pixel des linken Bildes der passende Partner im rechten Bild zu bestimmen. Die

benötigte Teilmenge kann auch umgekehrt definiert werden, so dass für jedes Pixel des rechten Bildes ein Partner im linken Bild gesucht wird. Solche Einschränkungen sind noch nicht genügend, um die Interpretierbarkeit dieser Menge als die Oberfläche eines Objektes zu garantieren (siehe Kapitel 1). Das ist nur mit Hilfe von weiteren Einschränkungen für die Korrespondenzkombinationen möglich. Außerdem sieht man in den obigen Spezifizierungen eine bestimmte Unsymmetrie. Präziser gesagt, werden hier nicht alle Teilmengen der Korrespondenzpaare betrachtet, die als Oberfläche eines Objektes interpretierbar sind, sondern nur einige davon.

Ein eleganter Ausweg basiert auf dem Begriff „Matching“ [**23**]. Ein ähnliches Verfahren wird auch in [**15**] vorgeschlagen. Ein sehr großer Vorteil dabei ist, dass diese Vorgehensweise erlaubt, die Punkte der Oberfläche zu berücksichtigen, die nur monokular sichtbar sind. Leider, führen solche Ansätze ohne weitere Annahmen zu NP-vollständigen Aufgaben.

Eine andere Spezifizierung der Teilmenge der Korrespondenzpaare, bei der solche Probleme (Interpretierbarkeit als Oberfläche, Unsymmetrie) nicht entstehen, betrachten wir im nachfolgenden Abschnitt. Jetzt möchten wir eine andere Frage diskutieren. Bei jeder der bisher diskutierten Spezifizierungen existieren natürlich sehr viele Teilmengen von Korrespondenzpaaren, die die eingeführten Einschränkungen erfüllen (weil es tatsächlich viele Oberflächen gibt). Unter allen Teilmengen muss man die in einem gewissen Sinn beste wählen. Das bedeutet, dass ein Maß benötigt wird, bezüglich dessen die Suche nach der besten Oberfläche durchgeführt werden soll. Die einfachste Variante ist anzunehmen, dass dieses Maß die Summe der Qualitäten aller Elementen ist, aus denen die Teilmenge besteht. Die Qualität eines Korrespondenzpaares (man nennt sie auch die lokale Qualität) bezeichnen wir als $A(p_l, p_r)$.

In der Literatur findet man sehr viele verschiedenen Vorschläge, wie die lokale Qualität aussehen soll (oder kann). Die einfachste Variante ist, dafür die quadratische Differenz zwischen den Grauwerten der Pixel p_l und p_r zu benutzen: $A(p_l, p_r) = \big(I_l(p_l) - I_r(p_r)\big)^2$ (mit $I(p)$ ist hier der Grauwert des Pixels p bezeichnet). Um die Robustheit der lokalen Qualität zu erhöhen, werden zusätzlich kleine Umgebungen der Pixel p_l und p_r betrachtet – d.h. Fenster einer vordefinierten Größe $(2d{+}1) \times (2d{+}1)$ mit den Zentren in den Pixeln p_l bzw. p_r (im Weiteren mit $F = \{\triangle p = (i, j) : -d \leq i \leq d, -d \leq j \leq d\}$ bezeichnet). Somit lässt sich die lokale Qualität wie folgt ausdrücken:

$$A(p_l, p_r) = \sum_{\triangle p \in F} \left[I_l(p_l + \triangle p) - I_r(p_r + \triangle p)\right]^2. \tag{2.1}$$

Weitere Verbesserungen des Ähnlichkeitsmaßes ergeben sich durch die Berücksichtigung verschiedener Details des Reflexionsmodells der Oberfläche, des Beleuchtungsmodells, des Prozesses der Erzeugung der Bilder usw. So kann man zum Beispiel die folgende einfache Methode benutzen, um einen Unterschied in den Beleuchtungen der Bilder zu berücksichtigen. Wir nehmen an, dass sich die Grauwerte der Pixel in den korrespondierenden Fenstern um eine Zahl unterscheiden können, die sich innerhalb des Fensters nicht ändert. Folglich wird für jedes Korrespondenzpaar (p_l, p_r) eine „Grauwertsverschiebung“ C_v gesucht, so dass nach der Anwendung dieser Verschiebung die quadratische Differenz (2.1) minimal wird. Die erreichte minimale quadratische Differenz wird als lokale Qualität benutzt:

$$A(p_l, p_r) = \min_{C_v} \sum_{\triangle p \in F} \left[I_l(p_l + \triangle p) + C_v - I_r(p_r + \triangle p)\right]^2.$$

Man kann leicht sehen, dass die obige Formel zum folgenden Ausdruck äquivalent ist:

$$A(p_l,p_r) = \sum_{\triangle p\in F}\left[I_l(p_l+\triangle p)-I_r(p_r+\triangle p)\right]^2 - \Big[\sum_{\triangle p\in F}(I_l(p_l+\triangle p)-I_r(p_r+\triangle p))\Big]^2/n \tag{2.2}$$

mit $n=(2d+1)\cdot(2d+1)$ – die Anzahl der Pixel im Fenster F. Obwohl der obige Ausdruck die Grauwertsverschiebung C_v explizit nicht enthält (diese Konstante wird nicht benötigt, um den Wert von A zu berechnen), geben wir auch die Formel für den Wert dieser Größe an. Der Wert ist die Differenz der mittleren Grauwerte der Bilder innerhalb der ausgewählten Fenster:

$$C_v = \frac{1}{n}\sum_{\triangle p\in F} I_r(p_r+\triangle p) \;-\; \frac{1}{n}\sum_{\triangle p\in F} I_l(p_l+\triangle p).$$

Ein weiterer Schritt besteht darin, dass zusätzlich eine „Grauwertskalierung“ C_s zugelassen ist. Somit wird eine lineare Farbtransformation (beschrieben durch zwei Konstanten C_v und C_s) gesucht, so dass nach der Anwendung dieser Transformation die quadratische Differenz (2.1) minimal wird:

$$A(p_l,p_r) = \min_{C_v,C_s}\sum_{\triangle p\in F}\left[I_l(p_l+\triangle p)\cdot C_s + C_v - I_r(p_r+\triangle p)\right]^2.$$

Die gesuchte Größe stimmt (bis auf eine Normierung mit der Streuung des zweiten Bildes) mit dem Korrelationskoeffizient überein:

$$A(p_l,p_r) = \frac{SLR - SL\cdot SR/n}{\sqrt{SLL-SL^2/n}\,\sqrt{SRR-SR^2/n}} \tag{2.3}$$

mit

$$\begin{aligned} SLR &= \sum_{\triangle p\in F} I_l(p_l+\triangle p)\cdot I_r(p_r+\triangle p)\\ SL &= \sum_{\triangle p\in F} I_l(p_l+\triangle p)\\ SR &= \sum_{\triangle p\in F} I_r(p_r+\triangle p)\\ SLL &= \sum_{\triangle p\in F} I_l(p_l+\triangle p)^2\\ SRR &= \sum_{\triangle p\in F} I_r(p_r+\triangle p)^2. \end{aligned}$$

Bei allen oben beschriebenen Ansätzen entsteht das folgende Problem. Da die Bildpunkte $p_l = T_l(X,Y,Z)$ und $p_r = T_r(X,Y,Z)$ Projektionspunkte sind, müssen sie nicht ganzzahlige Koordinaten haben. In dem Fall muss man präzisieren, was unter dem Ausdruck $I(p)$ zu verstehen ist. Die einfachste Variante wäre, den Grauwert des nächsten Pixels zu benutzen, das ganzzahlige Koordinaten hat. Man kann den benötigten Grauwert mit Hilfe der bilinearen Interpolation finden. In [**4**] wird eine andere Methode vorgeschlagen, die erlaubt, solche „Sampling“-Probleme zu umgehen.

Die oben beschriebenen Ähnlichkeitsmaße folgen aus dem so genannten Lambertschen Reflexionsmodell. Dieses Modell basiert auf der Annahme, dass jeder Punkt der Oberfläche selbst eine Lichtquelle ist. Im Kontext der Aufgabe der Stereorekonstruktion führt dies zu der Annahme, dass bei einem idealen Stereopaar (ohne Störungen) die Grauwerte der korrespondierenden Pixel gleich sind. In der Realität ist das meistens nicht

der Fall. Die Grauwerte hängen von vielen geometrischen und physikalischen Parametern ab, nämlich: der Lage der Beleuchtungsquelle, dem Blickwinkel, der Orientierung der Oberfläche (Normalenvektor der Oberfläche im betrachteten Weltpunkt), physikalischen Eigenschaften der Oberfläche usw. Wenn bei der konkreten Applikation solche Daten zu Verfügung stehen, kann man (und soll man) sie benutzen, um ein Ähnlichkeitsmaß zu konstruieren, welches der Realität besser entspricht.

Ein neues Ähnlichkeitsmaß zu entwickeln oder die bereits dafür existierenden Ansätze zu studieren ist allerdings nicht unser Ziel. Wir wollen uns mit anderen Fragen beschäftigen, nämlich ob sich die Aufgabe der Stereorekonstruktion als strukturelles Problem formulieren lässt und ob die entstehenden strukturellen Probleme exakt lösbar sind. Uns wird interessieren, ob die strukturellen Ansätze im Allgemeinen, unabhängig von der konkreten Anwendung im Vergleich zu anderen bekannten Verfahren (zum Beispiel Block Matching, Zeilenweise Stereo usw.) bessere Resultate liefern. Deswegen werden wir im Weiteren nur die einfachen Ähnlichkeitsmaße (2.1), (2.2) und (2.3) benutzen, die nicht mit speziellen Anwendungen verbunden sind.

2.2. Formulierung der Aufgabe der Stereorekonstruktion als Aufgabe der Energieminimierung

Im Allgemeinen kann die Vorgehensweise wie folgt gegliedert werden. Zunächst definiert man, was unter dem Begriff „Oberfläche“ zu verstehen ist. Dann ist es notwendig, die Menge aller möglichen Oberflächen vernünftig einzuschränken. Mit anderen Worten, formuliert man ein Modell der gültigen Oberflächen. Dann ordnet man jeder Oberfläche eine Qualität zu, die „angibt“, wie gut die Oberfläche dem gegebenen Stereopaar entspricht. Schließlich kann man die Aufgabe der Stereorekonstruktion als folgende Optimierungsaufgabe formulieren: Man finde die Oberfläche, die zu der gegebenen Menge gehört und die beste Qualität hat. Diese Qualität bezeichnet man oft als Energie der Oberfläche. Und eine derartige Formulierung nennt man Formulierung als Aufgabe der Energieminimierung. Im Weiteren werden wir auch die Begriffe Qualität, Maß und Bewertungsfunktion benutzen.

In dieser Arbeit betrachten wir eine etwas vereinfachte Variante der Stereorekonstruktion, in der nur *eine* (in einem gewissen Sinn stetige) Oberfläche im 3D-Raum gesucht wird. Wir nehmen zusätzlich an, dass jeder Punkt der Oberfläche binokular sichtbar ist (es gibt keine Verdeckungen). Somit kann man die Oberfläche als eine Funktion $f : \mathbb{R}^2 \to \mathbb{R}$ verstehen, die jedem Paar (X,Y) von Weltkoordinaten einen Wert der Z-Weltkoordinate zuordnet. Sei der verarbeitete dreidimensionale Raum (sowohl der Definitionsbereich als auch der Wertebereich der Funktion f) diskret, d.h. durch ein orthogonales Gitter (i, j, k) modelliert, wobei i, j und k ganze Zahlen sind. Wir bezeichnen die Menge aller Paare (i, j) als R. Ein Element $r \in R$ dieser Menge nennen wir Pixel. Die Menge aller möglichen Werte von k bezeichnen wir als K - die Menge der Zustände, in denen sich das Pixel befinden kann. Somit ist die Funktion f (ihre diskrete Version) eine Abbildung $f : R \to K$, die jedem Pixel r einen Wert aus der Menge K zuordnet. Eine fixierte Abbildung nennen wir Labeling oder Mapping oder Zustandsfeld. Die Menge aller möglichen Zustandsfelder bezeichnen wir als K^R.

Der nächste Schritt ist die oben beschriebene Menge der Oberflächen einzuschränken. Vor allem ist klar, dass die Entscheidung für ein Pixel r von Entscheidungen abhängen soll, die für andere Pixel angenommen werden. Die einfachste Variante ist anzunehmen, dass die Entscheidung für ein gegebenes Pixel r nur von Entscheidungen in den benachbarten Pixeln abhängt. Somit sollen die benachbarten Pixel in einer gewissen Beziehung stehen.

Diese Beziehung soll ihrerseits der intuitiven Vorstellung von den Eigenschaften der zu rekonstruierenden Oberfläche entsprechen.

Betrachten wir dies anhand eines Beispiels ausführlicher. Es sei bekannt, dass die gesuchte Oberfläche keine Diskontinuitäten besitzt. Unter Diskontinuitäten werden wir große Sprünge der Z-Koordinate verstehen. Anders gesagt, muss $|f(r)-f(r')| \leq \delta$ für alle Paare (r,r') von benachbarten Pixeln gelten, wobei δ ein vorgegebener Schwellwert ist (der maximale erlaubte Sprung der k-Koordinate). Drücken wir diese Forderung auf folgende Art aus. Wir werden die Menge R als die Menge der Knoten eines Graphen verstehen. Diesen Graphen nennen wir Basisgraph der Aufgabe. Die Menge E der Kanten dieses Graphen besteht aus Paaren (r,r') von benachbarten Pixeln. Somit gilt $(r,r') \in E$, wenn für die entsprechenden Pixel (i,j) und (i',j') $|i-i'|+|j-j'|=1$ erfüllt ist (4-Nachbarschaft). Wir definieren für jede Kante des Graphen eine Funktion $g_{rr'} : K \times K \to \mathbb{R}$, die den Wert 0 annimmt, falls $|k-k'| \leq \delta$ gilt, und den Wert ∞ andernfalls. Somit lässt sich die obige Forderung wie folgt ausdrücken. Die Oberfläche (das Labeling) f ist erlaubt, wenn

$$G'(f) = \sum_{(r,r')\in E} g_{rr'}\big(f(r),f(r')\big) = 0$$

gilt, und nicht erlaubt, wenn obige Summe den Wert ∞ annimmt. Solche Funktionen $g_{rr'}$ nennen wir „scharfe Glattheitsforderung“:

$$g_{rr'}(k,k') = \begin{cases} 0 & \text{wenn } |k-k'| \leq \delta \\ \infty & \text{sonst.} \end{cases} \tag{2.4}$$

Der nächste Schritt ist, jeder gültigen Oberfläche eine Qualität zu zuordnen. Die einfachste Vorgehensweise ist anzunehmen, dass diese Qualität die Summe gewisser „lokalen“ Qualitäten ist. Führen wir für jedes Pixel eine Funktion $q_r : K \to \mathbb{R}$ ein, die jeden Zustand bewertet. Als diese Bewertungsfunktion benutzen wir das im Abschnitt 2.1 eingeführte Ähnlichkeitsmaß. Da jedem Paar $\big(r,f(r)\big)$ genau ein Weltpunkt (X,Y,Z) entspricht und jedem Weltpunkt genau ein Paar von Projektionspunkten $p_l = T_l(X,Y,Z)$ und $p_r = T_r(X,Y,Z)$ entspricht, gilt $q_r\big(f(r)\big) = A\big(T_l(r,f(r)),T_r(r,f(r))\big)$. Somit ist die Qualität eines „erlaubten“ Labelings

$$Q(f) = \sum_r q_r\big(f(r)\big).$$

Für alle Oberflächen (einschließlich der ungültigen) definieren wir die Qualität als $G(f) = Q(f) + G'(f)$. Man kann leicht sehen, dass diese Qualität für ungültige Oberflächen den Wert ∞ annimmt.

Offensichtlich ist die oben beschriebene Funktion g nicht die einzige Möglichkeit. Man benutzt oft andere Funktionen g, die anderen Eigenschaften der zu rekonstruierenden Oberfläche entsprechen. Für die Stereorekonstruktion geeignete Funktionen sind zum Beispiel:

$$g_{rr'}(k,k') = c \cdot |k-k'| \quad \text{lineare Metrik,} \tag{2.5}$$

$$g_{rr'}(k,k') = c \cdot (k-k')^2 \quad \text{quadratischer Abstand.} \tag{2.6}$$

Bei diesen Funktionen sind Diskontinuitäten prinzipiell erlaubt. Große Diskontinuitäten sind allerdings nicht erwünscht (werden bestraft). Eine interessante Variante ist das so genannte Potts Modell:

$$g_{rr'}(k,k') = \begin{cases} a & \text{wenn } k = k' \\ b & \text{sonst} \end{cases} \tag{2.7}$$

mit $a < b$. In diesem Fall wird angenommen, dass die gesuchte Funktion f stückweise konstant sein soll.

Jede Auswahl hat ihre Nachteile und Vorteile. Man kann nicht sagen, welche der Funktion am besten ist. Das steht in einem starken Zusammenhang mit den Eigenschaften des zu rekonstruierenden Objektes. Eine Analyse, welche Funktionen in welchen Fällen bessere Ergebnisse liefern, führen wir im Kapitel 4 durch.

Fassen wir alle oben eingeführten Begriffe zusammen und definieren wir das entstehende strukturelle Optimierungsproblem.
Gegeben sind:

$V = (R,E)$	der Basisgraph, R ist die Menge der Knoten, E ist die Menge der Kanten;
K	die Zustandsmenge;
$q_r : K \to \mathbb{R}$	für jeden Knoten die lokale Bewertungsfunktion, die jedem Zustand in dem Knoten einen reellen Wert (die Ähnlichkeit) zuordnet;
$g_{rr'} : K \times K \to \mathbb{R}$	für jede Kante die Modellfunktion, die jedem Zustandspaar in der Kante einen reellen Wert zuordnet.

Gesucht ist das Labeling f^* (eine Abbildung $f : R \to K$, die jedem Knoten des Basisgraphen einen Wert aus der Menge K zuordnet) mit der besten Qualität

$$f^* = \arg\min_f \left[\sum_{r \in R} q_r\big(f(r)\big) \; + \sum_{(r,r') \in E} g_{rr'}\big(f(r), f(r')\big) \right]$$

bzw. der Wert dieser Qualität

$$G = \min_f \left[\sum_{r \in R} q_r\big(f(r)\big) \; + \sum_{(r,r') \in E} g_{rr'}\big(f(r), f(r')\big) \right]. \tag{2.8}$$

Im speziellen Fall der Aufgabe der Stereorekonstruktion ist der Basisgraph ein Gitter, die lokalen Bewertungsfunktionen q_r entsprechen dem Ähnlichkeitsmaß, und die Modellfunktionen $g_{rr'}$ entsprechen der Glattheitsforderung (zum Beispiel (2.4) – (2.7)).

2.3. Gibbs-Wahrscheinlichkeitsverteilung als Oberflächenmodell

In diesem Abschnitt betrachten wir eine etwas andere Formulierung der Aufgabe der Stereorekonstruktion. Wir werden (nach wie vor) die Oberflächen durch Labelings f eines Graphen $V(R,E)$ repräsentieren, in dem jeder Knoten r einem Ort (i, j) im 3D-Raum entspricht. Die Zustandsmenge K ist die Menge aller möglichen Werte der Koordinate k (siehe Abschnitt 2.2). Um das Oberflächenmodell zu spezifizieren, soll jeder Oberfläche eine Qualität zugeordnet werden. Ein natürliches Maß dafür ist die Wahrscheinlichkeit dieser Oberfläche in einem Wahrscheinlichkeitsmodell. Unserer Meinung nach ist das einfachste nicht triviale Wahrscheinlichkeitsmodell eine Gibbssche Wahrscheinlichkeitsverteilung zweiter Ordnung. In diesem Modell wird die a-priori Wahrscheinlichkeit der Oberfläche (des Labelings) f durch

$$P(f) = \frac{1}{Z} \prod_{(r,r') \in E} g_{rr'}\big(f(r), f(r')\big) \tag{2.9}$$

mit dem Normierungsfaktor

$$Z = \sum_f \left[\prod_{(r,r') \in E} g_{rr'}\big(f(r), f(r')\big) \right] \tag{2.10}$$

definiert. Die Funktionen $g_{rr'} : K \times K \to \mathbb{R}$ sind die Parameter der Verteilung. Einige oft benutzte Funktionen sind zum Beispiel:

$$g_{rr'}(k,k') = \begin{cases} 1 & \text{wenn } |k-k'| \le \delta \\ 0 & \text{sonst} \end{cases} \qquad \text{scharfe Glattheitsforderung,} \tag{2.11}$$

$$g_{rr'}(k,k') = \exp(c \cdot |k-k'|) \qquad \text{lineare Metrik,} \tag{2.12}$$

$$g_{rr'}(k,k') = \exp\bigl(c \cdot (k-k')^2\bigr) \qquad \text{quadratischer Abstand,} \tag{2.13}$$

$$g_{rr'}(k,k') = \begin{cases} a & \text{wenn } k = k' \\ 1 & \text{sonst} \end{cases} \qquad \text{Potts Modell.} \tag{2.14}$$

Die bedingte Wahrscheinlichkeitsverteilung $P(X|f)$, wobei X die Beobachtung ist (bei gegebenem Stereopaar), drücken wir auf folgende Art aus:

$$P(X|f) = \prod_{r \in R} q_r\bigl(f(r), X\bigr). \tag{2.15}$$

Hier sind q_r gewisse lokale Funktionen, die desto größere Werte annehmen, je besser das Ähnlichkeitsmaß für das entsprechende Voxel $\bigl(r, f(r)\bigr)$ ist (ausführlicher betrachten wir dieses im nachfolgenden Abschnitt). Der Kürze halber werden wir im Weiteren diese Funktionen in der Form $q_r\bigl(f(r)\bigr)$ schreiben. Somit ergibt sich die Wahrscheinlichkeit eines Paares (X, f) (das ist in unserem Fall das elementare Ereignis) als

$$P(X,f) = \frac{1}{Z} \prod_{(r,r') \in E} g_{rr'}\bigl(f(r), f(r')\bigr) \cdot \prod_{r \in R} q_r\bigl(f(r)\bigr). \tag{2.16}$$

Nachdem die Wahrscheinlichkeitsverteilung definiert ist, kann die Aufgabe der Stereorekonstruktion als Aufgabe der Bayesschen Entscheidung formuliert werden. Gegeben sei eine Menge der Entscheidungen D. Weiterhin sei eine Kostenfunktion $C : D \times K^R \to \mathbb{R}$ gegeben, die jedem Paar (Entscheidung, Labeling) einen reellen Wert zuordnet. Gesucht ist die Entscheidung $d^* \in D$, die den Erwartungswert der Kosten minimiert:

$$d^* = \arg\min_{d \in D} \sum_f P(f|X) C(d,f) = \arg\min_{d \in D} \sum_f P(X,f) C(d,f). \tag{2.17}$$

Betrachten wir zunächst den einfachsten möglichen Fall, nämlich wenn die Menge D die Menge aller Labelings ist ($D = K^R$) und die Kostenfunktion die so genannte Deltafunktion ist:

$$C(d,f) = \begin{cases} 1 & \text{wenn } d \ne f \\ 0 & \text{sonst.} \end{cases} \tag{2.18}$$

In diesem Fall ist die Bayessche Entscheidung das a-posteriori wahrscheinlichste Zustandsfeld:

$$d^* = \arg\max_f P(f|X) = \arg\max_f P(X,f) = \arg\max_f \prod_{(r,r') \in E} g_{rr}\bigl(f(r), f(r')\bigr) \cdot \prod_{r \in R} q_r\bigl(f(r)\bigr).$$

Diese Aufgabe lässt sich in die im Abschnitt 2.2 definierte MinSum Aufgabe (2.8) überführen:

$$d^* = \arg\min_f \left[\sum_{(r,r') \in E} \hat{g}_{rr'}\bigl(f(r), f(r')\bigr) + \sum_{r \in R} \hat{q}_r\bigl(f(r)\bigr) \right]$$

mit $\hat{g}_{rr'}(k,k') = -\ln g_{rr'}(k,k')$ und $\hat{q}_r(k) = -\ln q_r(k)$. Somit ist die Aufgabe der Energieminimierung ein Speziallfall der Aufgabe der Bayesschen Entscheidung, wobei die Kostenfunktion die Deltafunktion (2.18) ist.

Es ist leicht zu sehen, dass die Deltafunktion nicht die beste mögliche Kostenfunktion ist. In diesem Fall wird für eine falsche Entscheidung unabhängig davon bezahlt, „wie weit“ diese Entscheidung vom wahren Zustandsfeld entfernt ist. Eine realistischere Kostenfunktion kann zum Beispiel eine additive Kostenfunktion folgender Art sein. Sei D' die Menge der Entscheidungen in *einem* Knoten. Somit ist die Menge der Entscheidungen $D = D'^R$ und die Entscheidung d ist die Funktion $d : R \to D'$. Bezeichnen wir die Entscheidung im Knoten r mit $d(r) \in D'$. Eine Kostenfunktion $C : D'^R \times K^R \to \mathbb{R}$ heißt additiv, wenn

$$C(d,f) = \sum_{r\in R} c\big(d(r), f(r)\big) \tag{2.19}$$

gilt. Hier ist die Funktion $c : D' \times K \to \mathbb{R}$ eine lokale Kostenfunktion, die jedem Paar (Entscheidung in einem Knoten, Label in diesem Knoten) einen reellen Wert zuordnet. Setzt man (2.19) in (2.17) ein, so ergibt sich die Bayessche Entscheidung als

$$d^* = \arg\min_{d\in D} \sum_f \left[P(f|X) \cdot \sum_{r\in R} c\big(d(r), f(r)\big) \right]. \tag{2.20}$$

Diesen Ausdruck kann man wie folgt äquivalent umformen:

$$\begin{aligned} d^* &= \arg\min_{d\in D} \sum_{r\in R} \sum_f \big[P(f|X) \cdot c\big(d(r), f(r)\big) \big] = \\ &\arg\min_{d\in D} \sum_{r\in R} \sum_{k\in K} \sum_{f:f(r)=k} \big[P(f|X) \cdot c\big(d(r),k\big) \big] = \\ &\arg\min_{d\in D} \sum_{r\in R} \sum_{k\in K} \left[c\big(d(r),k\big) \cdot \sum_{f:f(r)=k} P(f|X) \right] = \\ &\arg\min_{d\in D} \sum_{r\in R} \sum_{k\in K} \big[c\big(d(r),k\big) \cdot P\big(f(r)=k \,|X\big) \big]. \end{aligned}$$

Somit ergibt sich die Bayessche Entscheidung aus

$$d^*(r) = \arg\min_{d(r)\in D'} \sum_{k\in K} \big[c\big(d(r),k\big) \cdot P\big(f(r)=k \,|X\big) \big] \quad \forall r \in R. \tag{2.21}$$

Die marginale a-posteriori Wahrscheinlichkeit dafür, dass das Labeling f in dem Knoten r durch den Zustand k durchgeht, ergibt sich dabei als

$$P\big(f(r)=k \,|X\big) = \frac{P\big(X, f(r)=k\big)}{\sum_{k'} P\big(X, f(r)=k'\big)} \tag{2.22}$$

mit

$$\begin{aligned} &P\big(X, f(r)=k\big) = \\ &\frac{1}{Z} \sum_{f:f(r)=k} \left[\prod_{(r',r'')\in E} g_{r'r''}\big(f(r'), f(r'')\big) \cdot \prod_{r'\in R} q_{r'}\big(f(r')\big) \right]. \end{aligned} \tag{2.23}$$

Die Bayessche Entscheidung für solche additiven Kostenfunktionen ist wie folgt zu treffen:

a) Die marginalen a-posteriori Wahrscheinlichkeitsverteilungen der Zustände $k \in K$ sind für alle Knoten $r \in R$ mit Hilfe der Formeln (2.22) und (2.23) zu berechnen;
b) Bayessche Entscheidungen mit Hilfe dieser Werte und (2.21) sind für alle Knoten voneinander unabhängig zu treffen.

Eine ausführliche Analyse derartiger Probleme findet man in [**19**].

Betrachten wir als Beispiel die additive Deltafunktion

$$c\big(d(r),f(r)\big)=\begin{cases}1 & \text{wenn } d(r)\neq f(r)\\ 0 & \text{sonst}\end{cases} \tag{2.24}$$

mit $d(r)\in K$. Die Bayessche Entscheidung in diesem Fall ist die Wahl des a-posteriori wahrscheinlichsten Zustandes in jedem Knoten:

$$d^*(r)=\arg\max_{k\in K}P\big(f(r)=k\,|X\big)=\arg\max_{k\in K}P\big(X,f(r)=k\big). \tag{2.25}$$

Im konkreten Fall der Aufgabe der Stereorekonstruktion kann man die Menge der Entscheidungen bzw. die Kostenfunktion noch auf folgende Art erweitern. Jeder Zustand $k\in K$ entspricht einem Wert der Koordinate Z im realen dreidimensionalen Raum. Somit haben alle reellen Werte $d(r)\in[k_{min},k_{max}]$ auch eine physikalische Bedeutung – ein Wert von Z. Das heißt, dass man als Antwort nicht unbedingt einen ganzzahligen Wert $d(r)\in K$ geben muss, sondern einen reellen Wert $d(r)$ im Bereich von $[k_{min},k_{max}]$. Wir werden die Menge der Entscheidungen als die Menge aller möglichen Funktionen folgender Art verstehen: $d:R\to\mathbb{R},\ d(r)\in[k_{min},k_{max}]\ \forall r\in R$. Die Kostenfunktion kann auch entsprechend erweitert werden. In jedem Knoten zahlt man eine Strafe für die falsche Entscheidung abhängig davon, „wie weit“ (im räumlichen Sinn) die Entscheidung vom wahren Zustand in dem Knoten entfernt ist:

$$c\big(d(r),f(r)\big)=\big(d(r)-f(r)\big)^2. \tag{2.26}$$

Diese Funktion nennen wir additive quadratische Kostenfunktion. Die Bayessche Entscheidung $d^*(r)$ ist in diesem Fall der Mittelwert von k:

$$\begin{aligned}d^*(r)=&\arg\min_{d(r)\in[k_{min},k_{max}]}\sum_{k\in K}P\big(f(r)=k\,|X\big)\cdot\big(d(r)-k\big)^2=\\ &\sum_{k\in K}k\cdot P\big(f(r)=k\,|X\big).\end{aligned} \tag{2.27}$$

Eine weitere Entwicklung der Menge der Entscheidungen bzw. der Kostenfunktion besteht darin, dass noch eine spezielle Entscheidung eingeführt wird, die wir „Rückweisung“ nennen (bezeichnen wir diese mit RW). Diese Erweiterung kann wie folgt motiviert werden. Wenn es für bestimmte Knoten nicht möglich ist, eine (in einem gewissen Sinn) „sichere“ Entscheidung zu treffen, sollte man für diese Knoten die Antwort „Unentschieden“ geben (und eine entsprechende Strafe ε dafür zahlen).

Betrachten wir zuerst die so erweiterte additive Deltafunktion. In diesem Fall ist die Menge der Entscheidungen $D=(K\cup RW)^R$. Die Kostenfunktion wird wie folgt definiert:

$$c\big(d(r),f(r)\big)=\begin{cases}1 & \text{wenn } d(r)\in K, d\neq f(r)\\ 0 & \text{wenn } d(r)=f(r)\\ \varepsilon & \text{wenn } d(r)=RW.\end{cases} \tag{2.28}$$

Die Bayessche Entscheidung für diesen Fall ist die Lösung (das Argument) der folgenden Aufgabe:

$$\min\left[\min_{d(r)\in K}\sum_{k\in K}P\big(f(r)=k\,|X\big)\cdot c\big(d(r),k\big),\ \varepsilon\right].$$

Die Entscheidung ist wie folgt zu treffen. Man betrachte den a-posteriori wahrscheinlichsten Zustand k^* und vergleiche seinen Wahrscheinlichkeitswert mit der Zahl $1-\varepsilon$. Falls

der Wahrscheinlichkeitswert dieses Zustandes größer als diese Zahl ist, entscheide man sich für k^*, sonst für die Rückweisung:

$$d^*(r) = \begin{cases} k^* = \arg\max_{k \in K} P\big(f(r) = k \mid X\big) & \text{wenn } P\big(f(r) = k^* \mid X\big) > 1 - \varepsilon \\ RW & \text{sonst.} \end{cases} \tag{2.29}$$

Man kann leicht sehen, dass die Antwort „Rückweisung“ bei $\varepsilon > 1 - 1/|K|$ unmöglich ist.

Schließlich betrachten wir die mit dem Zustand „Rückweisung“ erweiterte additive quadratische Kostenfunktion (2.26). Die Menge der Entscheidungen ist $D = ([k_{min}, k_{max}] \cup RW)^R$. Die Kostenfunktion ist

$$c\big(d(r), f(r)\big) = \begin{cases} \big(d(r) - f(r)\big)^2 & \text{wenn } d(r) \in [k_{min}, k_{max}] \\ \varepsilon & \text{wenn } d(r) = RW. \end{cases} \tag{2.30}$$

Die Bayessche Entscheidung für diesen Fall ist die Lösung (das Argument) der folgenden Aufgabe:

$$\min \left[\min_{d(r) \in [k_{min}, k_{max}]} \sum_{k \in K} P\big(f(r) = k \mid X\big) \cdot \big(d(r) - k\big)^2 \,,\ \varepsilon \right].$$

Die Entscheidung ist wie folgt zu treffen. Man berechne die Streuung von k und vergleiche diese mit der Zahl ε. Falls die Streuung kleiner als ε ist, entscheide man für den Mittelwert von k, sonst für die Rückweisung:

$$d^*(r) = \begin{cases} d' & \text{wenn } \sigma^2 < \varepsilon \\ RW & \text{sonst} \end{cases} \tag{2.31}$$

mit

$$d' = \sum_{k \in K} P\big(f(r) = k \mid X\big) \cdot k$$
$$\sigma^2 = \sum_{k \in K} P\big(f(r) = k \mid X\big) \cdot (k - d')^2.$$

Es ist leicht zu sehen, dass die Antwort „Rückweisung“ bei $\varepsilon > (k_{max} - k_{min})^2/4$ unmöglich ist.

Man kann sehen, dass bei allen additiven Kostenfunktionen die Hauptschwierigkeit darin liegt, die marginalen a-posteriori Wahrscheinlichkeitsverteilungen $P\big(f(r) = k \mid X\big)$ zu berechnen. Die dafür zu lösende SumProd Aufgabe (2.23) ist im allgemeinen Fall (wie alle anderen strukturellen Probleme) NP-vollständig. Im Gegensatz zu den anderen strukturellen Problemen (MinMax oder MinSum) ist es sogar nicht bekannt, ob eine Klasse von Funktionen g existiert (abgesehen von ganz trivialen Fällen), für die man die Aufgabe ohne Beschränkungen für die Nachbarschaftstruktur exakt lösen kann. Im nachfolgenden Kapitel geben wir einen effizienten Algorithmus zur Schätzung dieser Wahrscheinlichkeitsverteilungen an.

2.4. Bedingte Wahrscheinlichkeit

In diesem Abschnitt betrachten wir etwas ausführlicher, wie die Funktionen q_r (siehe (2.15)) im Kontext der Aufgabe der Stereorekonstruktion aussehen sollen. Dabei ist Folgendes zu berücksichtigen. Die bedingte Wahrscheinlichkeitsverteilung $P(X|f)$ muss eine

gültige Wahrscheinlichkeitsverteilung sein, d.h.

$$
\begin{aligned}
&P(X|f) = \prod_r q_r\big(f(r)\big) \geq 0 \quad \forall\, X, f \\
&\sum_X P(X|f) = \sum_X \prod_r q_r\big(f(r)\big) = 1 \quad \forall\, f.
\end{aligned} \tag{2.32}
$$

Andererseits sollen die Funktionen q_r in einem gewissen Sinn dem Ähnlichkeitsmaß entsprechen (siehe Abschnitt 2.1). Das bedeutet, dass die Werte $q_r(k)$ desto größer sein sollen, je kleiner die entsprechenden Werte des Ähnlichkeitsmaßes $A(p_l, p_r)$ sind (mit $p_l = T_l(r,k)$ und $p_r = T_r(r,k)$). Da jedem Voxel (r,k) genau ein Paar von Projektionspunkten (p_l, p_r) entspricht, werden wir im Weiteren das Ähnlichkeitsmaß auch in der Form $A(r,k)$ schreiben.

Man kann die anhand der Bilder berechneten Werte von $A(r,k),\ \forall r \in R, k \in K$ als Merkmalsatz verstehen, der aus dem Bildpaar X extrahiert wird. Auf Grund dieses Gedankens ersetzen wir die bedingte Wahrscheinlichkeitsverteilung $P(X|f)$ des Bildpaares X (unter der Bedingung eines gegebenen Labelings f) durch die Wahrscheinlichkeitsverteilung $P(A|f)$ des Merkmalsatzes $A : R \times K \to \mathbb{R}$, der dem Bildpaar X entspricht. Die Abhängigkeit der q-Werte von den entsprechenden Werten des Ähnlichkeitsmaßes werden wir als eine monoton fallende positive Funktion $Q : \mathcal{A} \to \mathbb{R}_+$ verstehen, wobei $\mathcal{A}$ der Wertebereich des Ähnlichkeitsmaßes ist, d.h. $q_r(k) = Q\big(A(r,k)\big)$. Somit werden wir uns im Weiteren mit der Frage beschäftigen, wie die für die Stereorekonstruktion geeignete Funktion Q aussieht.

Bezeichnen wir mit $A_f = (a_1, \ldots, a_r, \ldots, a_{|R|})$ den Vektor aus $|R|$ Komponenten, in dem die r-te Komponente dem Knoten r entspricht und der Wert dieser Komponente $a_r = A\big(r, f(r)\big)$ ist. Weiterhin nehmen wir an, dass $P(A|f) = P(A_f|f)$ gilt, d.h. dass die bedingte Wahrscheinlichkeit des Merkmalsatzes A unter der Bedingung f nur von den Werten des Ähnlichkeitsmaßes auf dem Labeling f abhängt. Somit kann man die Bedingung (2.32) wie folgt ausdrücken:

$$
\begin{aligned}
\sum_{A_f \in \mathcal{A}^{|R|}} P(A_f|f) = \sum_{A_f \in \mathcal{A}^{|R|}} \prod_r Q(a_r) = \prod_r \sum_{a_r \in \mathcal{A}} Q(a_r) = \Big[\sum_{a \in \mathcal{A}} Q(a)\Big]^{|R|} = 1 \\
\Rightarrow \sum_{a \in \mathcal{A}} Q(a) = 1.
\end{aligned} \tag{2.33}
$$

Um die Funktion Q abzuleiten, führen wir einen zusätzlichen Begriff ein, den wir „Färbung der Oberfläche“ nennen. Sei $I : R \to \mathcal{X}$ eine Funktion, die jeder Position r eine Farbe $x \in \mathcal{X}$ zuordnet. Wir betrachten dabei nur das einfachste Farbmodell der Oberfläche, in dem die Farbwerte in den Positionen r uniform und unabhängig generiert werden. Somit haben alle Farbwerte x die gleiche Wahrscheinlichkeit $P_x = 1/|\mathcal{X}|$. Die weitere Annahme besteht darin, dass bei fixiertem Farbwert $x = I(r)$ und Zustand $k = f(r)$ die Farbwerte in den Bildern $x_l = I_l\big(T_l(r,k)\big)$ und $x_r = I_r\big(T_r(r,k)\big)$ voneinander unabhängig generiert werden. Als Wahrscheinlichkeitsmodell für $P(x_l|x)$ und $P(x_r|x)$ benutzen wir die Gaussche Wahrscheinlichkeitsverteilung. Somit kann man die Wahrscheinlichkeit eines Paares von Farbwerten (x_l, x_r) unter der Bedingung eines gegebenen Farbwertes x der Oberfläche wie

folgt schreiben:

$$
\begin{aligned}
P(x_l,x_r|x) &= P(x_l|x)\cdot P(x_r|x) = \\
&\frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(x_l-x)^2}{2\sigma^2}\right]\cdot\frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(x_r-x)^2}{2\sigma^2}\right] = \\
&\frac{1}{2\pi\sigma^2}\exp\left[-\frac{(x_l-x_r)^2}{4\sigma^2}\right]\cdot\exp\left[-\frac{(x-\bar{x})^2}{\sigma^2}\right], \qquad (2.34)
\end{aligned}
$$

wobei $\bar{x} = (x_l + x_r)/2$ die „mittlere“ Farbe ist. Der Parameter σ ist die Streuung der Gaussschen Wahrscheinlichkeitsverteilungen. Die Wahrscheinlichkeit des Paares von Farbwerten (x_l,x_r) ergibt sich somit als

$$
\begin{aligned}
P(x_l,x_r) &= \sum_x P(x)\cdot P(x_l,x_r|x) = \\
&P_x\cdot\frac{1}{2\pi\sigma^2}\exp\left[-\frac{(x_l-x_r)^2}{4\sigma^2}\right]\cdot\sum_x\exp\left[-\frac{(x-\bar{x})^2}{\sigma^2}\right]\approx \\
&P_x\cdot\frac{1}{2\sqrt{\pi}\sigma}\exp\left[-\frac{(x_l-x_r)^2}{4\sigma^2}\right]. \qquad (2.35)
\end{aligned}
$$

Ersetzt man 2σ durch σ und bezeichnet man die irrelevanten Teile mit einer Konstante, so erhält man

$$
P(x_l,x_r) = C\frac{1}{\sigma}\exp\left[-\frac{(x_l-x_r)^2}{\sigma^2}\right]. \qquad (2.36)
$$

Somit gilt

$$
\begin{aligned}
q_r\big(f(r)\big) &= P(x_l,x_r) = \\
&Q(a) = C\frac{1}{\sigma}\exp\left[-\frac{a}{\sigma^2}\right] \qquad (2.37)
\end{aligned}
$$

mit $a = A\big(r,f(r)\big) = (x_l-x_r)^2$, $x_l = I_l\big(T_l(r,f(r))\big)$, $x_r = I_r\big(T_r(r,f(r))\big)$.

Die obige Reihe von Formeln (2.33)–(2.37) hat mehrere Schwachpunkte. Vor allem ist das die Annahme, dass die Zahlen $A\big(r,f(r)\big)$ bei fixiertem Labeling f *voneinander unabhängig* generiert werden. Darauf basieren die Annahme $P(A|f) = P(A_f|f)$ und der Übergang von „$\sum\prod$“ zu „$\prod\sum$“ in der Formel (2.33). Das Farbmodell der Oberfläche ist offensichtlich auch nicht besonders gut. Die Gaussche Wahrscheinlichkeitsverteilung, mit der die Farbwerte x_l und x_r anhand des Farbwertes x generiert werden, entspricht einer gewissen Annahme über das Modell des Rauschens. Streng genommen, ist diese Annahme nicht genügend gut begründet. Eine weitere Vereinfachung besteht im Folgenden. Der mit „$\approx$“ bezeichnete Übergang (bei der Summierung über alle Werte von x in der Formel (2.35)) gilt exakt nur, wenn der Wertebereich von x unendlich ist. In diesem Fall geht aber die Wahrscheinlichkeit P_x gegen Null. Außerdem gilt alles oben gesagte nur für den Fall, dass die quadratische Differenz der Grauwerte als Ähnlichkeitsmaß benutzt wird. Es ist nicht ganz klar, wie die Funktionen q_r für die anderen Ähnlichkeitsmaße aussehen sollen.

Schließlich wollen wir auch bemerken, dass die Gausschen Wahrscheinlichkeitsverteilungen nur dann die in (2.34) benutzte Form haben, wenn die Größen x bzw. x_l und x_r *Zahlen* sind. Bei der Berechnung des Ähnlichkeitsmaßes wird aber nicht nur *ein* Farbwert, sondern die Farbwerte der Pixel innerhalb eines Fensters benutzt. Der Farbwert eines Pixels ist oft selbst ein Dreitupel der Farbkomponenten – Rot, Grün und Blau. Das heißt, dass die Größen x, x_l und x_r tatsächlich *Vektoren* sind. In diesem Fall soll die Formel (2.37) wie

folgt aussehen:

$$Q(a) = C\frac{1}{\sigma^l}\exp\left[-\frac{a}{\sigma^2}\right], \tag{2.38}$$

wobei $l = 3 \times$ (Anzahl der Pixel im Fenster) die Dimension der Farbvektoren ist. In diesem Fall stimmt aber die Annahme über Unabhängigkeit der Farbwerte auf der Oberfläche f (das Farbmodell $I(r)$ der Oberfläche) offensichtlich nicht, weil sich in diesem Fall gewisse Fenster überlappen.

In dieser Arbeit möchten wir nicht näher betrachten, wie die korrekten Funktionen $Q(a)$ für verschiedene Ähnlichkeitsmaße aussehen sollen. In der Praxis benutzen wir für alle Ähnlichkeitsmaße dieselbe Funktion (2.38), wobei aber die Konstante l nicht die Dimension des Farbvektors ist, sondern aus den Experimenten bestimmt wird. Dabei vermuten wir, dass der Faktor C (der dafür sorgt, dass die Forderung (2.32) erfüllt ist) von σ nicht stark abhängt.

Man muss auch bemerken, dass bei einem fixierten Wert von σ der Vorfaktor C/σ^l für die Ermittelung der Bayesschen Entscheidung irrelevant ist, weil es sich in diesem Fall um eine Konstante bezüglich der Optimierung handelt. Die Abhängigkeit des Vorfaktors von σ spielt allerdings eine gewisse Rolle beim *Lernen* von σ. Ausführlicher betrachten wir dies im nachfolgenden Kapitel.

KAPITEL 3

Theoretische Grundlagen und Algorithmen zur Lösung struktureller Probleme

Wie bereits festgestellt, sind alle in der Einführung vorgestellten strukturellen Probleme im allgemeinen Fall NP-vollständig. In manchen speziellen Fällen, nämlich wenn die Nachbarschaftstruktur beschränkt ist, sind derartige Probleme exakt mit polynomialer Zeitkomplexität lösbar. Zum Beispiel kann man diese Aufgaben mit Hilfe der dynamischen Programmierung lösen, wenn der Basisgraph der Aufgabe eine Kette ist [**2**]. Andere bekannte exakt lösbare Fälle sind zum Beispiel: Bäume, k-Bäume [**25**], Einfache Netze [**26**].

Ein anderer Weg ist, solche Klassen der Funktionen g zu finden, die es erlauben, strukturelle Probleme ohne Beschränkungen an die Nachbarschaftstruktur exakt zu lösen. Eine solche Klasse von lösbaren MinSum Aufgaben betrachten wir im nächsten Abschnitt.

3.1. MinSum – Monotonintervalartigkeit

Der Begriff „Monotonintervalartigkeit“ wurde zum ersten mal in [**26**] eingeführt. In [**7**] wird diese Eigenschaft unter dem Namen „Bikonkavität“ benutzt. Der Kürze und Bequemheit halber werden wir den Begriff „Monotonie“ benutzen. Betrachten wir zunächst die Definition dieser Eigenschaft.

Sei die Menge der Zustände K vollständig geordnet. Das bedeutet, dass für jedes Paar von Zuständen k_1 und k_2 eine Relation „Größer“ definiert ist. Im Weiteren werden wir diese Relation „Höher“ nennen, um sie nicht mit dem Begriff „größerer Wert“ zu verwechseln. Somit heißt der Ausdruck $k_2 \geq k_1$: „Der Zustand k_2 liegt höher als der Zustand k_1“. Folglich kann man die Zustandsmenge als Teilmenge der ganzen Zahlen verstehen, d.h. alle Zustände können durchnummeriert werden. Im Weiteren sei $K = \{1, \ldots, |K|\}$.

Seien k_1 und k_2 zwei Zustände im Knoten r, so dass $k_2 \geq k_1$ gilt. Ähnlich, seien k'_1 und k'_2 zwei Zustände im Knoten r', so dass $k'_2 \geq k'_1$ gilt. Die Funktion $g_{rr'}$ heißt monoton, wenn für jedes derartige Viertupel k_1, k_2, k'_1, k'_2

$$g_{rr'}(k_1,k'_1) + g_{rr'}(k_2,k'_2) \leq g_{rr'}(k_1,k'_2) + g_{rr'}(k_2,k'_1) \tag{3.1}$$

gilt. Die MinSum Aufgabe ist monoton, wenn alle Funktionen $g_{rr'}$, $(r,r') \in E$ monoton sind. Dabei können die Funktionen q_r beliebig sein.

Diese Definition lässt sich in einer etwas einfacheren Form ausdrucken. Sei $k_2 = k_1+1$, d.h. „Der Zustand k_2 liegt höher als der Zustand k_1 und unter allen solchen Zuständen (außer k_1 selbst) ist k_2 der niedrigste“. Dann ist (3.1) zu der folgenden Definition äquivalent. Die Funktion $g_{rr'}$ heißt monoton, wenn für alle $k = 1 \ldots |K|-1$ und $k' = 1 \ldots |K|-1$

$$g_{rr'}(k,k') + g_{rr'}(k+1,k'+1) \leq g_{rr'}(k,k'+1) + g_{rr'}(k+1,k') \tag{3.2}$$

gilt.

Aus (3.1) folgt (3.2) direkt. Es ist einfach zu sehen, dass auch die umgekehrte Aussage richtig ist, d.h. (3.1) aus (3.2) folgt. Betrachten wir dazu für jedes Viertupel von Zuständen

(k_1,k_2,k_1',k_2') die Zahl

$$\alpha(k_1,k_2,k_1',k_2') = g_{rr'}(k_1,k_1') + g_{rr'}(k_2,k_2') - g_{rr'}(k_1,k_2') - g_{rr'}(k_2,k_1').$$

Man kann sehen, dass für alle Viertupel mit $k_2 \geq k_1$ und $k_2' \geq k_1'$

$$\alpha(k_1,k_2,k_1',k_2') = \sum_{k_3=k_1}^{k_2-1} \sum_{k_3'=k_1'}^{k_2'-1} \alpha(k_3,k_3+1,k_3',k_3'+1) \tag{3.3}$$

gilt. Wenn die Funktion $g_{rr'}$ entsprechend der Definition (3.2) monoton ist, sind alle Summanden in der obigen Formel nicht positiv. Somit ist die Zahl $\alpha(k_1,k_2,k_1',k_2')$ auch nicht positiv. Das bedeutet, dass die Funktion $g_{rr'}$ entsprechend der Definition (3.1) monoton ist.

Bevor wir Verfahren zur Lösung solcher Aufgaben betrachten, zeigen wir, dass die in [**12, 16**] beschriebenen Klassen Spezialfälle von monotonen Funktionen sind.

In [**16**] betrachten die Autoren Funktionen $E^{i,j} : \{0,1\} \times \{0,1\} \to \mathbb{R}$ von zwei binären Variablen. In diesen Bezeichnungen sind i und j die Nummern der Knoten. Seien x_i bzw. x_j ($x \in \{0,1\}$) die entsprechenden Variablen. In dieser Arbeit wird bewiesen, dass die Aufgabe der Energieminimierung (in unserer Sprache die MinSum Aufgabe) lösbar ist, wenn

$$E^{i,j}(0,0) + E^{i,j}(1,1) \leq E^{i,j}(0,1) + E^{i,j}(1,0)$$

gilt. Man sieht leicht, das diese Forderung zur Definition der Monotonie (3.2) äquivalent ist.

In [**12**] beschreiben die Autoren die folgende lösbare Klasse von MinSum Problemen. Sei die Zustandsmenge $K = \{1,\ldots,|K|\}$ eine Teilmenge der ganzen Zahlen. Sei die Funktion $g : K \times K \to \mathbb{R}$ eine Funktion $F : \mathbb{Z} \to \mathbb{R}$ der Differenz zweier Zustandsnummern, d.h. $g(k,k') = F(k-k')$. Die Autoren beweisen, dass die MinSum Aufgabe lösbar ist, wenn die Funktion F konvex ist, d.h.

$$F(x_2)\cdot(x_3-x_1) \leq F(x_1)\cdot(x_3-x_2) + F(x_3)\cdot(x_2-x_1)$$

für jedes Dreitupel $x_1 \leq x_2 \leq x_3$ gilt.

Es kann gezeigt werden, dass für solche Funktionen die Forderung (3.2) immer erfüllt ist (solche Funktionen sind monoton). Dafür vergleiche man die linke und die rechte Seiten der Formel (3.2) für die oben beschriebene Funktion:

$$g(k,k') + g(k+1,k'+1) \quad \text{und} \quad g(k,k'+1) + g(k+1,k').$$

Bezeichnen wir die Differenz $k-k'$ mit $\triangle k$. Somit ergibt sich:

$$\begin{aligned} F(\triangle k) + F(\triangle k) \quad &\text{und} \quad F(\triangle k-1) + F(\triangle k+1) \\ 2\cdot F(\triangle k) \quad &\text{und} \quad F(\triangle k-1) + F(\triangle k+1). \end{aligned}$$

Da die Funktion F konvex ist, ist die linke Seite der letzten Formel immer kleiner gleich der rechten Seite. Das bedeutet, dass solche Funktionen g monoton sind.

Die umgekehrte Aussage ist offensichtlich falsch. Man kann sich sehr einfach eine monotone Funktion vorstellen, die sich nicht als Funktion der Differenz von Zustandsnummern darstellen lässt. Dafür reicht es, wenn ein Zustandspaar k und k' existiert, so dass $g(k,k') \neq g(k+1,k'+1)$ gilt.

An dieser Stelle möchten wir noch eine Definition angeben, die wir im Weiteren benötigen. Wir betrachten monotone OrAnd Probleme. Für OrAnd Aufgaben wird diese Eigenschaft genauso wie im Fall von MinSum Problem definiert. Man ersetzt nur die Operationen „min“ durch „$\vee$“ bzw. „$+$“ durch „$\wedge$“. Dabei berücksichtige man, dass die betrachteten Funktionen $q_r : K \to \{0,1\}$ und $g_{rr'} : K \times K \to \{0,1\}$ boolesche Funktionen sind. Somit

wird die Monotonie der OrAnd Probleme wie folgt definiert. Seien k_1 und k_2 zwei Zustände im Knoten r, so dass $k_2 \geq k_1$ (bezüglich der eingeführten Ordnung der Zustände) gilt. Ähnlich, seien k'_1 und k'_2 zwei Zustände im Knoten r', so dass $k'_2 \geq k'_1$ gilt. Die Funktion $g_{rr'}$ ist monoton, wenn für jedes wie oben beschriebene Viertupel k_1, k_2, k'_1, k'_2

$$g_{rr'}(k_1,k'_2) \wedge g_{rr'}(k_2,k'_1) \Rightarrow g_{rr'}(k_1,k'_1) \wedge g_{rr'}(k_2,k'_2) \tag{3.4}$$

gilt. Die OrAnd Aufgabe ist monoton, wenn alle Funktionen $g_{rr'}$, $(r,r') \in E$ monoton sind. Dabei sind die Funktionen q_r beliebig. Monotone OrAnd Aufgaben lassen sich durch den Relaxation Labeling Algorithmus [**21**] lösen. Ausführliche Betrachtungen solcher Probleme findet man in [**7, 26**].

Man muss bemerken, dass die für die Stereorekonstruktion oft benutzten Funktionen $g_{rr'}$ („scharfe" Glattheitsforderung (2.4), lineare Metrik (2.5), quadratischer Abstand (2.6)) glücklicherweise monoton sind. Somit werden wir uns im Weiteren auf monotone Aufgaben konzentrieren. In den zwei nachfolgenden Abschnitten betrachten wir Algorithmen zur Lösung solcher Probleme.

3.2. MinSum – Maximieren der scheinbaren Qualität

3.2.1. Äquivalente Transformation der Aufgabe. Eine MinSum Aufgabe $\mathcal{A}$ werden wir durch ihre Komponenten beschreiben, d.h. $\mathcal{A} = (V,K,q,g)$ mit:

$V = (R,E)$	der Basisgraph;
K	die Zustandsmenge;
$q : R \times K \to \mathbb{R}$	die Qualitäten der Zustände;
(oder $q_r : K \to \mathbb{R}$, $\forall r \in R$)	
$g : E \times K \times K \to \mathbb{R}$	die Qualitäten der Zustandspaare.
(oder $g_{rr'} : K \times K \to \mathbb{R}$, $\forall (r,r') \in E$)	

Gegeben seien zwei Aufgaben $\mathcal{A} = (V,K,q,g)$ und $\mathcal{A}' = (V,K,q',g')$. Sei f ein Labeling. Seine Qualitäten in den Aufgaben $\mathcal{A}$ und $\mathcal{A}'$ sind

$$G(f) = \sum_{r \in R} q_r\big(f(r)\big) \; + \sum_{(r,r') \in E} g_{rr'}\big(f(r),f(r')\big)$$

bzw.

$$G'(f) = \sum_{r \in R} q'_r\big(f(r)\big) \; + \sum_{(r,r') \in E} g'_{rr'}\big(f(r),f(r')\big).$$

Man nennt die beiden Aufgaben äquivalent, wenn $G(f) = G'(f)$ für jedes Labeling $f \in K^R$ gilt [**25**]. Alle äquivalenten Aufgaben bilden eine Äquivalenzklasse (bezeichnen wir diese mit $\mathcal{E}(\mathcal{A})$ – die Menge aller zu $\mathcal{A}$ äquivalenten Aufgaben).

Man kann anhand der gegebenen Aufgabe $\mathcal{A}$ eine äquivalente Aufgabe $\mathcal{A}'$ mit Hilfe der in der Abbildung 3.1a) gezeigten Transformation bilden. In dieser Abbildung sind die Knoten mit Quadraten bezeichnet, die Zustände sind Kreise. Betrachten wir einen Knoten und einen Zustand in dem Knoten (das Paar (r^*,k^*)). Weiterhin betrachten wir zwei Kanten des Basisgraphen, die mit r^* inzident sind ((r^*,r') und (r^*,r'')). Wenn man die Werte der Funktion $g_{r^*r'}(k,k')$ für alle Zustandspaare (r^*,r',k^*,k'), $k' \in K$ um eine Zahl φ vergrößert und gleichzeitig die Werte der Funktion $g_{r^*r''}(k,k'')$ für alle Zustandspaare (r^*,r'',k^*,k''), $k'' \in K$ um dieselbe Zahl verkleinert, ändert sich die Qualität keines Labelings. Die Zahlen φ nennen wir Potentiale und die Transformation nennen wir „äquivalente Transformation der Aufgabe". Ähnlich kann man eine äquivalente Transformation für die Qualitäten der

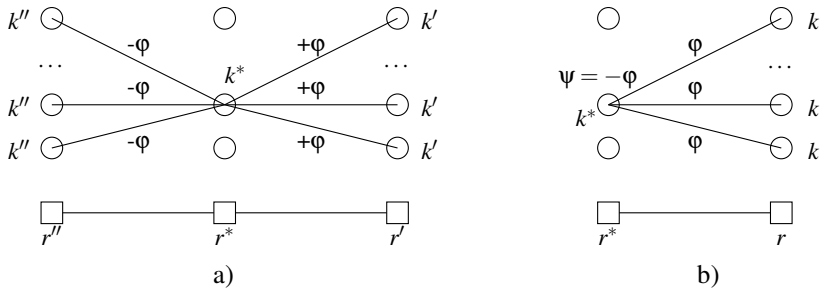


ABBILDUNG 3.1. Äquivalente Transformationen

Zustände $q_r(k)$ definieren. Man betrachte einen Knoten r^*, einen Zustand k^* in dem Knoten und eine Kante (r^*, r). Man vergrößere alle Werte $g_{r^*r}(k^*, k)$, $k \in K$ um eine Zahl φ und gleichzeitig verkleinere man den Wert $q_{r^*}(k^*)$ um dieselbe Zahl. Damit ändert sich die Qualität keines Labelings (siehe Abbildung 3.1b)).

Im Allgemeinen definiert man eine äquivalente Transformation als ein Satz von Potentialen $\Phi = \{\varphi_{rr'}(k), \psi_r(k)\}$, so dass

$$\psi_r(k) + \sum_{r':(r,r')\in E} \varphi_{rr'}(k) = 0 \quad \forall r \in R,\ k \in K \tag{3.5}$$

gilt. Wendet man die Transformation auf eine Aufgabe $\mathcal{A} = (V, K, q, g)$ an, so erhält man eine neue, zu $\mathcal{A}$ äquivalente Aufgabe $\mathcal{A}' = (V, K, q', g')$, in der sich die neuen Werte der Funktionen q' und g' wie folgt ergeben:

$$\begin{aligned} q'_r(k) &= q_r(k) + \psi_r(k) \\ g'_{rr'}(k,k') &= g_{rr'}(k,k') + \varphi_{rr'}(k) + \varphi_{r'r}(k'). \end{aligned} \tag{3.6}$$

Man kann leicht sehen, dass die Aufgabe $\mathcal{A}'$ zu der Aufgabe $\mathcal{A}$ äquivalent ist, wenn die Forderungen (3.5) für den Satz der Potentiale erfüllt sind:

$$\begin{aligned} G'(f) = & \\ & \sum_{r\in R} q'_r\big(f(r)\big) + \sum_{(r,r')\in E} g'_{rr'}\big(f(r), f(r')\big) = \\ & \sum_{r\in R} \big[q_r\big(f(r)\big) + \psi_r\big(f(r)\big)\big] + \\ & \qquad \sum_{(r,r')\in E} \big[g_{rr'}\big(f(r), f(r')\big) + \varphi_{rr'}\big(f(r)\big) + \varphi_{r'r}\big(f(r')\big)\big] = \\ & G(f) + \sum_{r\in R} \left[\psi_r\big(f(r)\big) + \sum_{r':(r,r')\in E} \varphi_{rr'}\big(f(r)\big)\right] = \\ = & \ G(f). \end{aligned}$$

Äquivalenten Transformationen haben einige Eigenschaften, die wir im Weiteren benutzen werden.

Seien Φ_1 und Φ_2 zwei äquivalente Transformationen. Wendet man die Transformationen auf eine Aufgabe $\mathcal{A}$ sequenziell (nacheinander) an, so entspricht das der Anwendung einer Transformation Φ_3, für die $\varphi_{3rr'}(k) = \varphi_{1rr'}(k) + \varphi_{2rr'}(k)$ und $\psi_{3r}(k) = \psi_{1r}(k) + \psi_{2r}(k)$ gilt. Sind die Forderungen (3.5) für Φ_1 und Φ_2 erfüllt, so sind die Forderungen auch für Φ_3 erfüllt. Man kann auch leicht sehen, dass die erzeugte Aufgabe $\mathcal{A}'$ in beiden Fällen

(sequenzielle Anwendung von Φ_1 und Φ_2 und Anwendung von Φ_3) dieselben Werte von q und g hat. Die Transformation Φ_3 nennen wir „Superposition“ der zwei Transformationen Φ_1 und Φ_2 und bezeichnen wir sie im Weiteren als $\Phi_3 = \Phi_1 \oplus \Phi_2$.

Äquivalente Transformationen sind kommutativ. Das bedeutet, dass die sequenzielle Anwendung zweier Transformationen Φ_1 und Φ_2 zur sequenziellen Anwendung von Φ_2 und Φ_1 äquivalent ist (d.h. $\Phi_1 \oplus \Phi_2 = \Phi_2 \oplus \Phi_1$). Äquivalente Transformationen sind auch assoziativ, d.h. $(\Phi_1 \oplus \Phi_2) \oplus \Phi_3 = \Phi_1 \oplus (\Phi_2 \oplus \Phi_3)$. Es existiert eine „identische“ Transformation Φ^0, so dass $\Phi^0(\mathcal{A}) = \mathcal{A}$ gilt (bei der Transformation sind alle φ-s und ψ-s Null). Für jede Transformation Φ existiert eine „inverse“ Transformation Φ^{-1}, so dass $\Phi \oplus \Phi^{-1} = \Phi^0$ gilt (d.h. $\Phi(\Phi^{-1}(\mathcal{A})) = \mathcal{A}$). Alle oben beschriebenen Eigenschaften folgen, eigentlich, aus den entsprechenden Eigenschaften der reellen Zahlen und der Operation „+“. Somit bildet die Menge aller äquivalenten Transformationen mit der entsprechend definierten Operation $\oplus$ eine Gruppe. Alle oben beschriebenen Eigenschaften sind in der Tabelle (3.7) zusammengefasst.

$$
\begin{aligned}
&\Phi = \{\varphi_{rr'}(k), \psi_r(k)\} \\
&\Phi^0 : \varphi_{rr'}(k) = 0, \psi_r(k) = 0 \\
&\Phi_3 = \Phi_1 \oplus \Phi_2 : \varphi_{3rr'}(k) = \varphi_{1rr'}(k) + \varphi_{2rr'}(k), \\
&\qquad\qquad\qquad\qquad \psi_{3r}(k) = \psi_{1r}(k) + \psi_{2r}(k) \\
&\Phi_1 \oplus \Phi_2 = \Phi_2 \oplus \Phi_1 \\
&(\Phi_1 \oplus \Phi_2) \oplus \Phi_3 = \Phi_1 \oplus (\Phi_2 \oplus \Phi_3) \\
&\Phi_1 = \Phi_2^{-1} : \varphi_{1rr'}(k) = -\varphi_{2rr'}(k), \psi_{1r}(k) = -\psi_{2r}(k).
\end{aligned}
\tag{3.7}
$$

An der Stelle möchten wir eine weitere Eigenschaft der Menge aller äquivalenten Transformationen angeben. Für ein beliebiges Paar von äquivalenten Aufgaben $\mathcal{A}$ und $\mathcal{A}'$ existiert eine Transformation Φ, so dass $\mathcal{A}' = \Phi(\mathcal{A})$ gilt:

$$
\mathcal{A}' \in \mathcal{E}(\mathcal{A}) \;\Leftrightarrow\; \exists\, \Phi : \mathcal{A}' = \Phi(\mathcal{A}).
\tag{3.8}
$$

Wir werden sagen, dass die Menge aller äquivalenten Transformationen eine Äquivalenzklasse „vollständig beschreibt“. Streng genommen, gilt das nur, wenn der Basisgraph der Aufgabe zusammenhängend ist. Im Weiteren betrachten wir nur solche Fälle.

Eine wichtige Eigenschaft der äquivalenten Transformationen besteht darin, dass diese Transformationen die Monotonie der Aufgabe nicht ändern. Um diese Behauptung zu beweisen, betrachten wir die im vorigen Abschnitt eingeführten Zahlen

$$
\alpha_{rr'}(k,k') = g_{rr'}(k,k') + g_{rr'}(k+1,k'+1) - g_{rr'}(k+1,k') - g_{rr'}(k,k'+1).
\tag{3.9}
$$

Nach der äquivalenten Transformation $\Phi : \mathcal{A} \to \mathcal{A}'$ werden die Zahlen

$$
\begin{aligned}
&\alpha'_{rr'}(k,k') = \\
&\quad g'_{rr'}(k,k') + g'_{rr'}(k+1,k'+1) - g'_{rr'}(k+1,k') - g'_{rr'}(k,k'+1) = \\
&\quad g_{rr'}(k,k') + \varphi_{rr'}(k) + \varphi_{r'r}(k') + \\
&\qquad\quad g_{rr'}(k+1,k'+1) + \varphi_{rr'}(k+1) + \varphi_{r'r}(k'+1) - \\
&\qquad\quad g_{rr'}(k+1,k') - \varphi_{rr'}(k+1) - \varphi_{r'r}(k') - \\
&\qquad\quad g_{rr'}(k,k'+1) - \varphi_{rr'}(k) - \varphi_{r'r}(k'+1) = \\
&= \alpha_{rr'}(k,k').
\end{aligned}
$$

Somit ändert die äquivalente Transformation diese Zahlen nicht. Das bedeutet, dass entweder beide Aufgaben $\mathcal{A}$ und $\mathcal{A}'$ monoton sind (alle α-s sind nicht positiv), oder beide

Aufgaben nicht monoton sind (es existiert mindestens ein streng positives α). Da die Menge aller äquivalenten Transformationen vollständig eine Äquivalenzklasse beschreibt (3.8), gilt das Folgende. Wenn eine Aufgabe $\mathcal{A}$ monoton ist, sind alle zu dieser Aufgabe äquivalenten Aufgaben auch monoton. (Die Äquivalenzklasse, zu welcher die Aufgabe $\mathcal{A}$ gehört, besteht nur aus monotonen Aufgaben). Die umgekehrte Aussage ist auch richtig. Wenn eine Aufgabe $\mathcal{A}$ nicht monoton ist, sind alle zu dieser Aufgabe äquivalenten Aufgaben auch nicht monoton.

Wir führen noch eine andere Transformation ein, die wir im Weiteren benutzen werden. Ändert man für eine Kante (r,r') alle Werte von $g_{rr'}(k,k')$ um eine und dieselbe Zahl ($g'_{rr'}(k,k') = g_{rr'}(k,k') + c_{rr'} \ \ \forall k \in K, k' \in K$), so ändern sich die Qualitäten aller Labelings um dieselbe Zahl. Somit wird die Lage des besten Labelings (arg min) nicht geändert. Dasselbe gilt auch für die Änderung der q-s ($q'_r(k) = q_r(k) + c_r \ \ \forall k \in K$). Diese Transformationen nennen wir „uniforme Transformationen“ der Aufgabe und bezeichnen sie mit $U = \{c_r, c_{rr'}\}$. Nach der Anwendung einer solchen Transformation ändern sich die Qualitäten aller Labelings um die Zahl $\sum_r c_r + \sum_{rr'} c_{rr'}$. Man kann leicht sehen, dass auch die Menge aller uniformen Transformationen eine Gruppe bildet. Die Operation $\oplus$ ist in diesem Fall (genauso wie im Fall der äquivalenten Transformationen) die einfache komponentenweise Addition. Es ist leicht zu sehen, dass uniforme Transformationen die Monotonie der Aufgabe nicht ändern.

3.2.2. Triviale Aufgabe. Bezeichnen wir mit $K_r \subset K$ die Menge aller Zustände im Knoten r, die die beste Qualität haben:

$$K_r = \{k : \ q_r(k) \leq q_r(k') \ \ \forall k' \in K\}.$$

Sei $SQ(r)$ dieser minimale Wert: $SQ(r) = \min_k q_r(k)$. Ähnlich bezeichnen wir mit $K_{rr'} \subset K \times K$ die Menge aller Zustandspaare in der Kante (r,r'), die die beste Qualität haben:

$$K_{rr'} = \{(k,k') : \ g_{rr'}(k,k') \leq g_{rr'}(k'',k''') \ \ \forall k'' \in K, \ k''' \in K\}.$$

Sei $SQ(r,r')$ dieser minimale Wert: $SQ(r,r') = \min_{kk'} g_{rr'}(k,k')$. Die Größen $SQ(r)$ und $SQ(r,r')$ nennen wir „scheinbare Qualitäten der Kanten bzw. der Knoten“. Die Zahl

$$SQ = \sum_{r \in R} SQ(r) + \sum_{(r,r') \in E} SQ(r,r') = \sum_{r \in R} \min_k q_r(k) + \sum_{(r,r') \in E} \min_{k,k'} g_{rr'}(k,k') \tag{3.10}$$

nennen wir „scheinbare Qualität der Aufgabe“. Offensichtlich ist diese Qualität kleiner gleich der Qualität eines beliebigen Labelings. Somit ist sie auch kleiner gleich der Lösung der MinSum Aufgabe.

Nehmen wir an, dass ein Labeling $\hat{f}$ existiert, so dass dieses Labeling nur durch die Zustände bzw. Zustandspaare durchgeht, die in den oben beschriebenen Mengen K_r bzw. $K_{rr'}$ enthalten sind, d.h. $\hat{f}(r) \in K_r \ \forall r \in R$ und $\left(\hat{f}(r), \hat{f}(r')\right) \in K_{rr'} \ \forall (r,r') \in E$. Es ist offensichtlich, dass das Labeling $\hat{f}$ die Lösung der ursprünglichen MinSum Aufgabe ist, weil seine Qualität gleich der scheinbaren Qualität ist. Aufgaben, für die ein solches Labeling $\hat{f}$ existiert, nennen wir „triviale Aufgaben“. Diese Eigenschaft (wir nennen sie „Trivialität“) kann auch auf eine etwas andere Art definiert werden: eine Aufgabe ist trivial, wenn

$$\sum_{r \in R} \min_k q_r(k) + \sum_{(r,r') \in E} \min_{k,k'} g_{rr'}(k,k') = \min_f \left[\sum_{r \in R} q_r\big(f(r)\big) + \sum_{(r,r') \in E} g_{rr'}\big(f(r), f(r')\big) \right] \tag{3.11}$$

gilt. Beide Definitionen der Trivialität (die Existenz eines wie oben beschrieben erzeugten Labelings $\hat{f}$ und (3.11)) sind äquivalent. Aus der Existenz eines solchen Labelings folgt (3.11) direkt. Man kann leicht sehen, dass mindestens ein solches Labeling $\hat{f}$ existiert,

wenn (3.11) erfüllt ist. Betrachten wir den Fall, dass kein solches Labeling existiert, obwohl (3.11) erfüllt ist. Bezeichnen wir mit f^* die Lösung der Aufgabe ($f^* = \arg\min_f G(f)$). Das Labeling f^* besitzt mindestens einen Zustand oder ein Zustandspaar, welcher (welches) nicht in der entsprechenden Menge K_r bzw. $K_{rr'}$ enthalten ist (sonst würde ein Labeling existieren, welches nur aus Zuständen bzw. Zustandspaaren dieser Mengen besteht). Somit existieren entweder solche Knoten oder solche Kanten, für die $q_r\big(f^*(r)\big) > \min_k\ q_r(k)$ bzw. $g_{rr'}\big(f^*(r), f^*(r')\big) > \min_{kk'}\ g_{rr'}(k,k')$ gilt. Somit wäre die Qualität dieses Labelings streng größer als die scheinbare Qualität ($G(f^*) > SQ$), was der Formel (3.11) widerspricht.

Man muss bemerken, dass die Überprüfung der Trivialität einer gegebenen Aufgabe im allgemeinen Fall selbst NP-vollständig ist. Diese Aufgabe kann wie folgt formuliert werden: „Man stelle fest ob ein Labeling f *existiert*, so dass *für jeden* Knoten *und für jede* Kante die Zugehörigkeitsbedingung $f(r) \in K_r$ bzw. $\big(f(r), f(r')\big) \in K_{r,r'}$ gilt“. Das ist nichts anderes, als ein OrAnd Problem (1.2). Man führe die zwei binären Funktionen $\tilde{q}_r : K \to \{0,1\}$ und $\tilde{g}_{rr'} : K \times K \to \{0,1\}$ wie folgt ein:

$$\tilde{q}_r(k) = \begin{cases} 1 & \text{wenn } k \in K_r \\ 0 & \text{sonst} \end{cases}$$

$$\tilde{g}_{rr'}(k,k') = \begin{cases} 1 & \text{wenn } (k,k') \in K_{r,r'} \\ 0 & \text{sonst.} \end{cases}$$

Dann berechne man

$$\bigvee_f \left[\bigwedge_{r \in R} \tilde{q}_r\big(f(r)\big) \ \wedge \bigwedge_{(r,r') \in E} \tilde{g}_{rr'}\big(f(r), f(r')\big) \right]. \tag{3.12}$$

Es lässt sich zeigen, dass die entstehende OrAnd Aufgabe monoton ist (im Sinn der Monotonie für OrAnd Probleme (3.4)), wenn die ursprüngliche MinSum Aufgabe monoton ist. Wie bereits gesagt, lassen sich monotone OrAnd Probleme durch den Relaxation Labeling Algorithmus lösen. Somit wird es möglich, die Trivialität einer gegebenen monotonen MinSum Aufgabe zu prüfen. Eine ausführliche Analyse derartiger Probleme findet man in [**7, 25, 26**].

Schließlich möchten wir auf eine weitere Eigenschaft der scheinbaren Qualität hinweisen. Das ist der Fakt, dass diese Qualität durch Anwendung einer äquivalenten Transformation geändert werden kann. Somit besteht eine beliebige Äquivalenzklasse aus Aufgaben mit unterschiedlichen scheinbaren Qualitäten.

3.2.3. Maximieren der scheinbaren Qualität. Wie bereits gesagt, ist die scheinbare Qualität einer Aufgabe immer kleiner gleich der Qualität des besten Labelings. Somit kann man die scheinbare Qualität als untere Schranke (eine Approximation) für die Qualität des besten Labelings verstehen. Gegeben sei eine Aufgabe $\mathcal{A}_0$. Die Hauptidee des Verfahrens besteht in der Suche nach der „Aufgabe mit der bestmöglichen Approximation“ innerhalb der Äquivalenzklasse $\mathcal{E}(\mathcal{A}_0)$ – d.h. der Suche nach der Aufgabe mit der maximalen scheinbaren Qualität:

$$\mathcal{A}^* = \arg\max_{\mathcal{A} \in \mathcal{E}(\mathcal{A}_0)} SQ(\mathcal{A}).$$

Drückt man dies mittels äquivalenter Transformationen aus, so ergibt sich zusammen mit den Forderungen für den Satz von Potentialen (3.5) die folgende konvexe Optimierungsaufgabe:

$$\Phi^* = \arg\max_{\Phi} SQ\big(\Phi(\mathcal{A}_0)\big) =$$

$$\arg\max_{\Phi}\Big[\sum_{r\in R}\min_{k}\big(q_r(k)+\psi_r(k)\big)+\sum_{(r,r')\in E}\min_{k,k'}\big(g_{rr'}(k,k')+\varphi_{rr'}(k)+\varphi_{r'r}(k')\big)\Big]. \tag{3.13}$$

Nehmen wir an, dass das Problem (3.13) bereits gelöst ist. Dann ist die Trivialität der gefundenen Aufgabe $\mathcal{A}^*$ zu prüfen. Man bilde die Mengen K_r, $K_{rr'}$ und die Funktionen $\tilde{q}_r(k)$, $\tilde{g}_{rr'}(k,k')$. Dann versuche man das entstehende OrAnd Problem (3.12) zu lösen. Man kann dazu den Relaxation Labeling Algorithmus verwenden. Nach dem Anhalten dieses Algorithmus entsteht eine von drei möglichen Situationen:

a) Das gesuchte Labeling $\hat{f}$ existiert. Das bedeutet, dass die ursprüngliche MinSum Aufgabe bereits gelöst ist. Das gefundene Labeling $\hat{f}$ ist die Lösung f^*.
b) Es ist unbestimmt, ob das Labeling $\hat{f}$ existiert. Somit ist das beste Labeling nicht gefunden. In diesem Fall kann man die erreichte scheinbare Qualität als untere Schranke für den Wert $G(f^*)$ verstehen.
c) Das gesuchte Labeling $\hat{f}$ existiert nicht. In diesem Fall ist die erreichte scheinbare Qualität genauso wie in b) als untere Schranke für den Wert $G(f^*)$ zu verstehen. Man kann weiterhin zeigen, dass die Äquivalenzklasse $\mathcal{E}(\mathcal{A}_0)$ in diesem Fall keine triviale Aufgabe enthält.

Betrachten wir den monotonen Fall, d.h. die Äquivalenzklasse $\mathcal{E}(\mathcal{A}_0)$ besteht nur aus monotonen Aufgaben. Wie bereits gesagt, lässt sich die Trivialität einer monotonen MinSum Aufgabe durch den Relaxation Labeling Algorithmus exakt prüfen. Somit ist der Fall b) ausgeschlossen.

Es kann bewiesen werden, dass jede aus monotonen Aufgaben bestehende Äquivalenzklasse mindestens eine triviale Aufgabe enthält. Somit ist der Fall c) auch ausgeschlossen. Den obigen Satz kann man auf verschiedene Arten beweisen. Eine Möglichkeit ist zum Beispiel:

a) Man formuliert die ursprüngliche MinSum Aufgabe als eine diskrete Optimierungsaufgabe, wobei die Variablen nur Werte von $\{0,1\}$ annehmen können.
b) Die Aufgabe wird relaxiert. Das heißt, dass die Variablen jetzt im Bereich von $[0,1]$ liegen können. Somit wird die Aufgabe stückweise linear. Es kann bewiesen werden, dass diese kontinuierliche Optimierungsaufgabe wenigstens eine Lösung hat (arg min in dem Fall), bei der alle Variablen nur die Werte von 0 oder 1 annehmen.
   Bemerkung: Wir weisen speziell darauf hin, dass diese Aussage nicht nur für monotone sondern für beliebige Aufgaben gilt.
c) Ist die ursprüngliche Aufgabe monoton, so ist die erhaltene kontinuierliche Aufgabe konvex. Man linearisiert sie und bildet die duale Optimierungsaufgabe. Es lässt sich zeigen, dass die entstehende duale Aufgabe genau die Aufgabe der Maximierung der scheinbaren Qualität (3.13) ist (die Potentiale sind die Variablen dieser Aufgabe).

Den kompletten Beweis findet man in [**7**].

Es gibt noch eine andere Möglichkeit den Satz zu beweisen, nämlich einen Algorithmus zu formulieren, der die Aufgabe löst. Einen solchen Algorithmus zu formulieren ist

möglich [**24**]. Der Algorithmus arbeitet iterativ. In jeder Iteration prüft er, ob die aktuelle Aufgabe $\mathcal{A}^{(i)}$ bereits trivial ist. Wenn das der Fall ist, beendet der Algorithmus seine Arbeit. Anderenfalls wird eine äquivalente Transformation Φ gefunden, die die scheinbare Qualität vergrößert. Somit erhält man eine neue Aufgabe $\mathcal{A}^{(i+1)} = \Phi(\mathcal{A}^{(i)})$. Dabei ist garantiert, dass $SQ(\mathcal{A}^{(i+1)}) > SQ(\mathcal{A}^{(i)})$ gilt.

Schließlich möchten wir noch einmal darauf hinweisen, dass der in diesem Abschnitt beschriebene Ansatz (Maximieren der scheinbaren Qualität) auf beliebige MinSum Aufgaben anwendbar ist. Im allgemeinen Fall ist allerdings nicht garantiert, dass dabei das optimale Labeling gefunden wird (siehe (3.13) und die nachfolgende Betrachtung der unterschiedlichen Fälle). Für monotone Aufgaben findet das Verfahren die Lösung exakt.

3.3. MinSum – Überführung in eine MinCut-MaxFlow Aufgabe

In diesem Abschnitt betrachten wir einen anderen Ansatz zur Lösung monotoner MinSum Aufgaben. Die Schlüsselidee ist, dass sich solche Aufgaben in so genannte MinCut Aufgaben überführen lassen. Diese Aussage kann man auf verschiedene Arten beweisen. Eine mögliche Variante ist in [**7**] beschrieben. Die Idee besteht darin, dass nach der Formulierung der ursprünglichen Aufgabe als kontinuierliche Optimierungsaufgabe (siehe vorigen Abschnitt) ein Ausdruck entsteht, der genau mit einer MinCut Aufgabe übereinstimmt.

Einen anderen Weg beschreiben wir in diesem Abschnitt. Wir werden zeigen, dass aus der initialen MinSum Aufgabe eine äquivalente MinCut Aufgabe direkt konstruiert werden kann. Der Beweis besteht aus zwei Teilen:

(1) Zunächst werden wir zeigen, dass eine beliebige Aufgabe mit einer Zustandsmenge aus $|K|$ Zuständen in eine Aufgabe mit einer Zustandsmenge aus nur zwei Zuständen überführt werden kann. Weiterhin wird gezeigt, dass diese Überführung die Monotonie der Aufgabe erhält, d.h. dass die so konstruierte Aufgabe mit zwei Zuständen genau dann monoton ist, wenn die ursprüngliche Aufgabe mit $|K|$ Zuständen monoton ist.

(2) Für Aufgaben mit zwei Zuständen lassen sich die entsprechenden MinCut Aufgaben relativ einfach konstruieren. Wir geben dafür konkrete Regeln und Formeln an. Dabei ist wichtig, dass alle Kantenkosten der entstehenden MinCut Aufgabe nicht negativ sind, wenn die ursprüngliche MinSum Aufgabe mit zwei Zuständen monoton ist. In diesem Fall lässt sich die MinCut Aufgabe mit polynomialer Zeitkomplexität lösen.

Den ersten Teil des Beweises halten wir für besonders wichtig. Die beschriebene Überführung (nennen wir sie „Abbildung von K zu 2“) hat einige interessante Eigenschaften. Zum Beispiel kann man leicht sehen, dass diese Abbildung auf gewisse andere Semiringe übertragbar ist. Das kann ein Schlüssel zur Lösung des SumProd Problems sein. Außerdem, eröffnet diese Abbildung einen Weg zu approximativen Lösungen von nichtmonotonen Problemen.

3.3.1. Abbildung von K zu 2. Gegeben sei eine MinSum Aufgabe $\mathcal{A}_K(V_K, K, q_K, g_K)$ mit ihren Komponenten:

$V_K(R_K, E_K)$ – der Basisgraph,
$K = \{1 \dots |K|\}$ – die Zustandsmenge,
$q_{Kr}(k)$ – die Qualitäten der Zustände und
$g_{Krr'}(k,k')$ – die Qualitäten der Zustandspaare.

Sei f_K ein Labeling $f_K : R_K \to K$. Die Qualität dieses Labelings ergibt sich als

$$G_K(f_K) = \sum_{r \in R_K} q_{Kr}\big(f_K(r)\big) \; + \sum_{(r,r') \in E_K} g_{Krr'}\big(f_K(r), f_K(r')\big).$$

Für die jetzt betrachtete Abbildung spielt die eigentliche Minimierung dieser Zielfunktion keine Rolle. Wir wollen eine neue Aufgabe konstruieren, so dass für jedes Labeling der ersten Aufgabe ein Labeling in der zweiten Aufgabe existiert und die Qualitäten dieser zwei Labelings gleich sind. Dabei können in der neuen Aufgabe Labelings existieren, die keinem Labeling der initialen Aufgabe entsprechen. Es ist allerdings möglich, die neue Aufgabe so zu konstruieren, dass die Qualitäten aller solchen Labelings unendlich sind. Bezeichnen wir die Komponenten der zweiten Aufgabe als: $V_2(R_2, E_2)$, $K_2 = \{0,1\}$, $q_{2r}(k)$ und $g_{2rr'}(k,k')$. Somit ist das Ziel, eine Aufgabe $\mathcal{A}_2(V_2, \{0,1\}, q_2, g_2)$ zu bilden, so dass für jedes Labeling f_K ein Labeling $f_2 : R_2 \to \{0,1\}$ existiert und $G_K(f_K) = G_2(f_2)$ gilt.

Die neue Aufgabe wird wie folgt konstruiert:

(1) Zuerst bilden wir die Menge der Knoten R_2 der neuen Aufgabe. Wir verbinden die Knoten mit bestimmten „speziellen“ Kanten. Für jede solche Kante definieren wir die Funktion g_2, so dass in der neuen Aufgabe genau $|K|^{|R_K|}$ Labelings eine endliche Qualität haben. Somit wird ein eindeutiger Zusammenhang $K^{R_K} \leftrightarrow \{0,1\}^{R_2}$ zwischen den Labelings f_K der ursprünglichen Aufgabe und den Labelings f_2 (mit einer endlichen Qualität) der neuen Aufgabe konstruiert.

(2) Wir definieren die Qualitäten q_2 der Zustände in der neuen Aufgabe, so dass für jedes Paar (f_K, entsprechendes f_2)

$$\sum_{r \in R_K} q_{Kr}\big(f_K(r)\big) = \sum_{r \in R_2} q_{2r}\big(f_2(r)\big)$$

gilt.

(3) Weiterhin definieren wir die Menge der Kanten E_2 und die Funktionen g_2 so dass für jedes Paar (f_K, entsprechendes f_2)

$$\sum_{(r,r') \in E_K} g_{Krr'}\big(f_K(r), f_K(r')\big) = \sum_{(r,r') \in E_2} g_{2rr'}\big(f_2(r), f_2(r')\big)$$

gilt.

(4) Anschließend wird die erhaltene Aufgabe mit Hilfe von äquivalenten Transformationen vereinfacht.

Wir bilden die Menge der Knoten R_2 der zweiten Aufgabe, so dass jedem Paar (r,k) der initialen Aufgabe ein Knoten in der zweiten Aufgabe entspricht: $R_2 = R_K \times K$. Wir bezeichnen die Knoten der zweiten Aufgabe als $r_2 = (r,k)$. Wählen wir einen Knoten r der initialen Aufgabe und betrachten wir die Teilmenge von Knoten r_2, die diesem Knoten entspricht. Bezeichnen wir diese Teilmenge der Knoten mit $R_{2r} = \{(r,k) : k = 1 \dots |K|\}$. Jetzt sind die Knoten mit Hilfe von Kanten so zu verbinden bzw. die Funktionen g_2 für diese Kanten so zu definieren, dass genau $|K|$ „Teillabelings“ mit einer endlichen Qualität entstehen. Unter dem Begriff „Teillabeling“ verstehen wir eine Abbildung $f_{2r} : R_{2r} \to \{0,1\}$ – Einschränkung des Labelings f_2 auf der Menge R_{2r}. Um dies zu realisieren, verbinden wir „benachbarte“ (im Sinn der Ordnung der Zustände in der initialen Aufgabe) Knoten

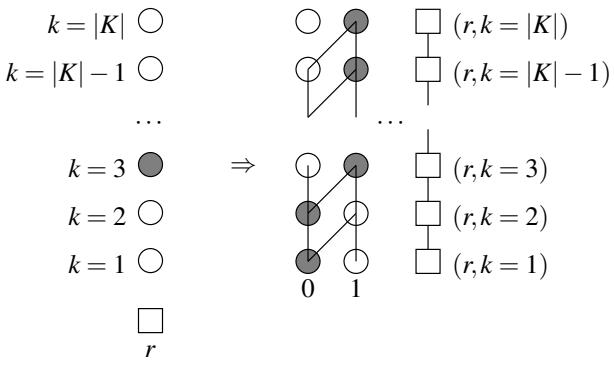


ABBILDUNG 3.2. Aufbau der Menge der Knoten

mit den Kanten. Die Funktionen $g_{2(r,k)(r,k+1)}$, $k = 1 \ldots |K|-1$ definieren wir wie folgt:

$$\begin{aligned} g_{2(r,k)(r,k+1)}(0,0) &= 0 \\ g_{2(r,k)(r,k+1)}(0,1) &= 0 \\ g_{2(r,k)(r,k+1)}(1,0) &= \infty \\ g_{2(r,k)(r,k+1)}(1,1) &= 0. \end{aligned}$$

Speziell für $k = |K|-1$ fordern wir zusätzlich $g_{2(r,k)(r,k+1)}(0,0) = \infty$. Im Weiteren werden wir solche Funktionen in der folgenden Form schreiben: $g_{2rr'} = \begin{array}{|c|c|}\hline g_{2rr'}(1,0) & g_{2rr'}(1,1) \\ \hline g_{2rr'}(0,0) & g_{2rr'}(0,1) \\ \hline \end{array}$

Somit gilt $g_{2(r,k)(r,k+1)} = \begin{array}{|c|c|}\hline \infty & 0 \\ \hline 0 & 0 \\ \hline \end{array}$ für $k \neq |K|-1$ und $\begin{array}{|c|c|}\hline \infty & 0 \\ \hline \infty & 0 \\ \hline \end{array}$ für $k = |K|-1$.

Es ist leicht zu sehen, dass nur $|K|$ Teillabelings der Teilmenge R_{2r} eine endliche Qualität haben. In der Abbildung 3.2 ist der Aufbau der Menge der Knoten (sowie ein Beispiel für $f_K(r) = 3$) schematisch gezeigt. Die „nicht erlaubten“ Zustandspaare (auf denen die Funktion g_2 den Wert ∞ annimmt) sind dabei ausgeblendet.

Die Bedeutung der Zustände in der erhaltenen Aufgabe lässt sich wie folgt interpretieren. In einem Knoten (r,k) der neuen Aufgabe bedeutet $f_2(r,k) = 0$, dass das Labeling f_K der initialen Aufgabe im Knoten r über dem Zustand k durchgeht. Der andere Fall (d.h. $f_2(r,k) = 1$) bedeutet, dass das Labeling f_K im Knoten r unter oder durch den Zustand k durchgeht. (Die Begriffe „über“ und „unter“ sind bezüglich der eingeführten Ordnung auf der Zustandsmenge K zu verstehen). Somit entspricht jedem Paar (r,k^*) der ursprünglichen Aufgabe ein Teillabeling f_{2r} mit $f_{2r}(r,k) = 0$ für $k < k^*$ und $f_{2r}(r,k) = 1$ für $k \geq k^*$.

Der nächste Schritt besteht in der Wahl geeigneter Qualitäten für die Zustände der neuen Aufgabe (nennen wir die Qualitäten $q_{2(r,k)}(0)$ und $q_{2(r,k)}(1)$). Diese Qualitäten sind so zu wählen, dass die Qualität jedes Teillabelings genau den Wert $q_{Kr}(k^*)$ annimmt, wobei

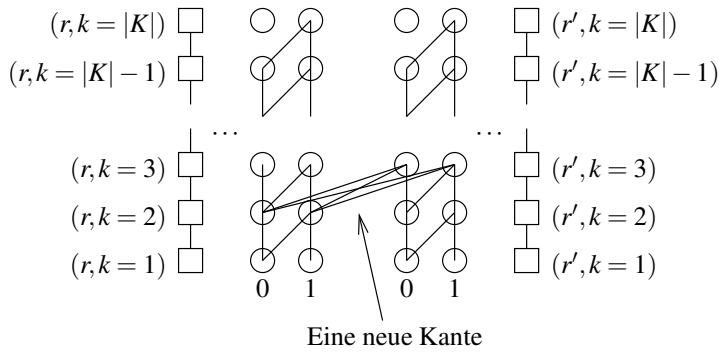


ABBILDUNG 3.3. Aufbau der Menge der Kanten

k^* der entsprechende Zustand der alten Aufgabe ist. Das lässt sich relativ einfach erreichen:

$$\begin{aligned}
&q_{2(r,|K|)}(0) = \infty \\
&q_{2(r,|K|)}(1) = q_{Kr}(|K|) \\
&q_{2(r,k)}(0) = 0,\ \ k < |K| \\
&q_{2(r,k)}(1) = q_{Kr}(k) - q_{Kr}(k{+}1),\ \ k < |K|.
\end{aligned}$$

Man sieht leicht, dass für ein Teillabeling $f_{2r}(r,k) = 0,\ k < k^*$, $f_{2r}(r,k) = 1,\ k \geq k^*$ seine Qualität

$$\sum_k q_{(r,k)}\big(f_{2r}(r,k)\big) = \sum_{k^* \leq k < |K|} \big[q_{Kr}(k) - q_{Kr}(k{+}1)\big] + q_{Kr}(|K|) = q_{Kr}(k^*) \tag{3.14}$$

ist.

Um die Menge der Kanten E_2 bzw. die Funktionen g_2 zu konstruieren (der nächste Schritt im gesamten Schema der Abbildung „von K zu 2“), betrachten wir zunächst zwei Knoten r und r' der initialen Aufgabe, die durch eine Kante verbunden sind $(r,r') \in E_K$. Diesen zwei Knoten entsprechen zwei Knotenmengen der neuen Aufgabe: $\{(r,k)\}$ bzw. $\{(r',k')\}$, $k,k' = 1 \ldots |K|$. Führen wir in der neuen Aufgabe die Kanten $\big((r,k),(r',k')\big)$ ein, d.h. verbinden wir jeden Knoten (r,k) mit jedem Knoten (r',k') durch eine Kante (siehe Abbildung 3.3. Hier ist nur eine von den neu eingeführten Kanten gezeigt). Weiterhin betrachten wir ein Zustandspaar (k^*,k'^*) auf der Kante (r,r') der initialen Aufgabe. Diesem Paar entspricht ein Paar von Teillabelings der neuen Aufgabe. Das Ziel ist nun, die Qualitäten auf den neu eingeführten Kanten so zu definieren, dass die Qualität dieses Paares von Teillabelings (ohne Berücksichtigung des q-Anteils) gleich der Qualität des Zustandpaares $g_{Krr'}(k^*,k'^*)$ ist, d.h.

$$\begin{aligned}
&\sum_{k,k'} g_{2(r,k)(r',k')}\big(f_2(r,k), f_2(r',k')\big) = \\
&\quad \sum_{k<k^*} \sum_{k'<k'^*} g_{2(r,k)(r',k')}(0,0) \ + \sum_{k<k^*} \sum_{k'\geq k'^*} g_{2(r,k)(r',k')}(0,1) + \\
&\quad \sum_{k\geq k^*} \sum_{k'<k'^*} g_{2(r,k)(r',k')}(1,0) \ + \sum_{k\geq k^*} \sum_{k'\geq k'^*} g_{2(r,k)(r',k')}(1,1) = \\
&\quad = g_{Krr'}(k^*,k'^*).
\end{aligned} \tag{3.15}$$

Diese Forderung muss für jedes Paar (k^*,k'^*) erfühlt werden. Solche Qualitäten g_2 auf den Kanten sind wie folgt auszuwählen:

$$\begin{aligned}
&g_{2(r,k)(r',k')}(0,0) = g_{2(r,k)(r',k')}(0,1) = g_{2(r,k)(r',k')}(1,0) = 0\\
&g_{2(r,k)(r',k')}(1,1) = \alpha(k,k') \quad k,k' < |K|\\
&g_{2(r,k)(r',|K|)}(1,1) = g_{Krr'}(k,|K|) - g_{Krr'}(k{+}1,|K|) \quad k < |K|\\
&g_{2(r,|K|)(r',k')}(1,1) = g_{Krr'}(|K|,k') - g_{Krr'}(|K|,k'{+}1) \quad k' < |K|\\
&g_{2(r,|K|)(r',|K|)}(1,1) = g_{Krr'}(|K|,|K|)
\end{aligned}$$

mit

$$\alpha(k,k') = g_{Krr'}(k,k') + g_{Krr'}(k{+}1,k'{+}1) - g_{Krr'}(k{+}1,k') - g_{Krr'}(k,k'{+}1)$$

(siehe (3.9)). Jetzt zeigen wir, dass bei dieser Wahl von g_2 die Forderungen (3.15) erfüllt sind. Setzen wir obige Werte von g_2 in die Formel (3.15) ein:

$$\begin{aligned}
&0+0+0+\sum_{k\ge k^*}\sum_{k'\ge k'^*} g_{2(r,k)(r',k')}(1,1) =\\
&\quad\sum_{k^*\le k<|K|}\ \sum_{k'^*\le k'<|K|} \alpha(k,k') +\\
&\qquad\sum_{k^*\le k<|K|} [g_{Krr'}(k,|K|) - g_{Krr'}(k{+}1,|K|)] +\\
&\qquad\sum_{k'^*\le k'<|K|} [g_{Krr'}(|K|,k') - g_{Krr'}(|K|,k'{+}1)] +\\
&\qquad g_{Krr'}(|K|,|K|) =\\
&\\
&[g_{Krr'}(k^*,k'^*) - g_{Krr'}(k^*,|K|) - g_{Krr'}(|K|,k'^*) + g_{Krr'}(|K|,|K|)] +\\
&\quad [g_{Krr'}(k^*,|K|) - g_{Krr'}(|K|,|K|)] +\\
&\quad [g_{Krr'}(|K|,k'^*) - g_{Krr'}(|K|,|K|)] +\\
&\quad g_{Krr'}(|K|,|K|) =\\
&\\
&= g_{Krr'}(k^*,k'^*). \qquad (3.16)
\end{aligned}$$

Schließlich betrachten wir ein Labeling f_K der initialen Aufgabe. Diesem Labeling entspricht in der neuen Aufgabe das folgende Labeling f_2:

$$f_2(r,k) = \begin{cases} 1 & \text{wenn } k \ge f_K(r)\\ 0 & \text{sonst.}\end{cases}$$

Aus (3.14) und (3.16) sieht man, dass die Qualitäten $G_2(f_2)$ (die Qualität des Labelings f_2 in der neuen Aufgabe) und $G_K(f_K)$ (die Qualität des Labelings f_K in der ursprünglichen Aufgabe) gleich sind.

Die neu erhaltene Aufgabe lässt sich weiter vereinfachen. Man kann sehen, das bestimmte Elemente (Qualitäten auf den Knoten bzw. Kanten) einen konstanten Beitrag für die Zielfunktion liefern. Unter „konstantem Beitrag“ verstehen wir eine Zahl, die nicht vom Labeling abhängt. Solche Elemente sind die „obersten“ Knoten $(r,|K|)$ für alle $r \in R_K$ und die „obersten“ Kanten $\big((r,|K|),(r',|K|)\big)$ für alle $(r,r') \in E_K$. Das gilt, weil in den Knoten $(r,|K|)$ tatsächlich nur ein Zustand erlaubt ist: $f_2(r,|K|) = 1$. Man kann alle Werte

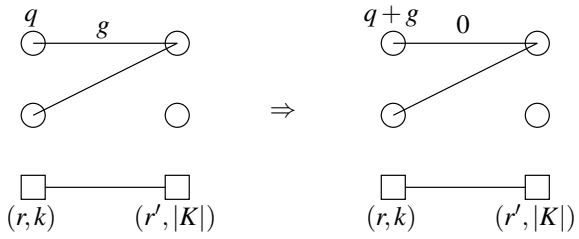


ABBILDUNG 3.4. Äquivalente Transformation

von $q_{(r,|K|)}(1)$ und $g_{(r,|K|)(r',|K|)}(1,1)$ auf Null setzen. Somit erhält man eine neue Aufgabe, in der die Qualität jedes Labelings um ein und dieselbe Zahl C geändert ist. Die Zahl ist die Summe aller Qualitäten, die auf Null gesetzt wurden:

$$C = \sum_{r \in R_K} q_{Kr}(|K|) \; + \sum_{(r,r') \in E_K} g_{Krr'}(|K|,|K|).$$

Die obige Vereinfachung ist tatsächlich nichts anderes, als eine im Abschnitt 3.2 eingeführte uniforme Transformation der Aufgabe.

Eine weitere Vereinfachung besteht darin, das bestimmte Kanten der neuen Aufgabe einen Beitrag für die Zielfunktion liefern, der nur vom Zustand in *einem* Knoten abhängt. Das sind diejenigen Kanten, die mit den „obersten“ Knoten inzident sind: $\big((r,k),(r',|K|)\big)$ für alle $(r,r') \in E_K,\ k < |K|$. Der Beitrag dieser Kanten hängt nur vom Zustand im Knoten (r,k) ab. Dieser Beitrag lässt sich mit Hilfe äquivalenter Transformationen auf Null setzen. In der Abbildung 3.4 ist diese Transformation schematisch gezeigt. Nach dieser äquivalenten Transformation bekommt jeder Zustand $\big((r,k),1\big)$ eine zusätzliche Qualität $\triangle q_{2(r,k)}(1)$. Sie entspricht der Summe aller Qualitäten der Zustandspaare, die mit dem Zustand $\big((r,k),1\big)$ inzident sind und deren Qualität auf Null gesetzt wurde:

$$\triangle q_{2(r,k)}(1) = \sum_{r':(r,r') \in E_K} g_{2(r,k)(r',|K|)}(1,1) = \sum_{r':(r,r') \in E_K} [g_{Krr'}(k,|K|) - g_{Krr'}(k{+}1,|K|)].$$

Nach Anwendung beider Transformationen kann man die obersten Knoten $(r,|K|)$ aus der Knotenmenge R_2 entfernen, ohne die Aufgabe zu ändern.

Fassen wir alles bisher gesagte zusammen. Gegeben sei eine Aufgabe $\mathcal{A}_K = (R_K, E_K, q_K, g_K)$ mit einer Zustandsmenge aus $|K|$ Zuständen: $K = \{1 \ldots |K|\}$. Die entsprechende Aufgabe $\mathcal{A}_2 = (R_2, E_2, q_2, g_2)$ mit einer Zustandsmenge aus zwei Zuständen bildet man wie folgt:

- Die Menge der Knoten:

$$R_2 = \{(r,k) : r \in R_K,\ k = 1 \ldots |K|{-}1\}.$$

- Die Menge der Kanten:

$$E_2 = \{\big((r,k),(r,k{+}1)\big) : r \in R_K,\ k = 1 \ldots |K|{-}2\} \cup$$
$$\{\big((r,k),(r',k')\big) : (r,r') \in E_K,\ k,k' = 1 \ldots |K|{-}1\}.$$

- Die Qualitäten auf den Knoten:

$$
\begin{aligned}
&q_{2(r,k)}(0) = 0\\
&q_{2(r,k)}(1) =\\
&\quad q_{Kr}(k) - q_{Kr}(k{+}1) +\\
&\quad \sum_{r':(r,r')\in E_K} [g_{Krr'}(k,|K|) - g_{Krr'}(k{+}1,|K|)].
\end{aligned}
$$

- Die Qualitäten auf den Kanten:

$$
g_{2(r,k)(r,k+1)} = \begin{array}{|c|c|}\hline \infty & 0\\ \hline 0 & 0\\ \hline\end{array}
$$

$$
g_{2(r,k)(r',k')} = \begin{array}{|c|c|}\hline 0 & \alpha_{rr'}(k,k')\\ \hline 0 & 0\\ \hline\end{array} \text{ mit}
$$

$$
\begin{aligned}
&\alpha_{rr'}(k,k') =\\
&\quad g_{Krr'}(k,k') + g_{Krr'}(k{+}1,k'{+}1) -\\
&\quad g_{Krr'}(k{+}1,k') - g_{Krr'}(k,k'{+}1).
\end{aligned}
$$

- Jedem Labeling $f_K : R_K \to K$ der initialen Aufgabe entspricht ein Labeling $f_2 : R_2 \to \{0,1\}$ der neuen Aufgabe. Dieses Labeling ergibt sich als:

$$
f_2(r,k) = \begin{cases} 1 & \text{wenn } k \geq f_K(r)\\ 0 & \text{sonst.}\end{cases}
$$

  Nur derartige Labelings der neuen Aufgabe haben eine endliche Qualität.
- Die Qualitäten der Labelings f_K und des entsprechenden f_2 unterscheiden sich um eine Konstante, die nicht vom Labeling abhängt:

$$
G_k(f_K) = G_2(f_2) + \sum_{r\in R_K} q_{Kr}(|K|) + \sum_{(r,r')\in E_K} g_{Krr'}(|K|,|K|). \tag{3.17}
$$

Wir wollen noch einmal daran erinnern, dass die oben beschriebene Überführung „von K zu 2“ für beliebige Aufgaben durchgeführt werden kann. Falls die initiale Aufgabe monoton ist, ist auch die erhaltene Aufgabe monoton. Dieser Fakt lässt sich einfach zeigen. In der neuen Aufgabe entstehen nur Kanten des Typs $\begin{array}{|c|c|}\hline \infty & 0\\ \hline 0 & 0\\ \hline\end{array}$ oder des Typs $\begin{array}{|c|c|}\hline 0 & \alpha\\ \hline 0 & 0\\ \hline\end{array}$ (nennen wir diese „Typ 1“ bzw. „Typ 2“). Die Kanten des ersten Typs sind monoton. Ist die ursprüngliche Aufgabe monoton, so sind alle α-s nicht positiv. Somit sind auch alle Kanten des zweiten Typs monoton.

Bevor wir den nächsten Schritt (nennen wir ihn „2 zu MinCut“) betrachten, möchten wir das Folgende bemerken. Die erhaltene Aufgabe mit der Zustandsmenge aus zwei Zuständen ist nicht die einzig mögliche Aufgabe $\mathcal{A}_2$, die der ursprünglichen Aufgabe $\mathcal{A}_K$ entspricht. Wendet man auf die erhaltene Aufgabe eine äquivalente Transformation Φ an, so erhält man eine andere Aufgabe mit zwei Zuständen, die auch der initialen Aufgabe entspricht. So kann man zum Beispiel alle Kanten des Typs $\begin{array}{|c|c|}\hline 0 & \alpha\\ \hline 0 & 0\\ \hline\end{array}$ in Kanten des Typs $\begin{array}{|c|c|}\hline \tilde\alpha & 0\\ \hline 0 & \tilde\alpha\\ \hline\end{array}$ mit $\tilde\alpha = -\alpha/2$ überführen. Das wird mit Hilfe der in der Abbildung 3.5 gezeigten äquivalenten Transformationen erreicht. Mit anderen Worten, die Überführung „von K zu

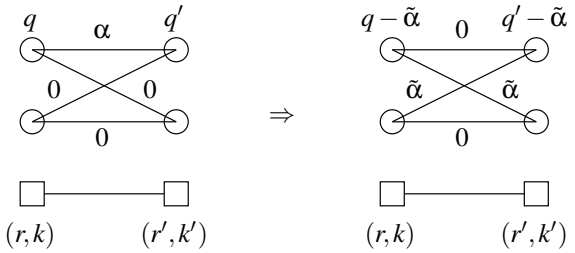


ABBILDUNG 3.5. Äquivalente Transformation

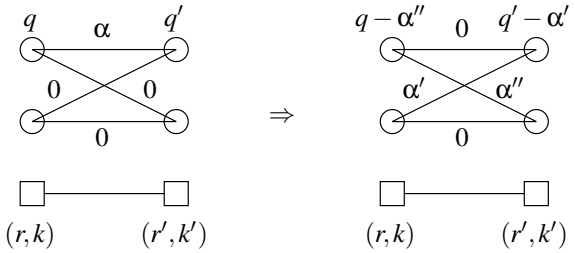


ABBILDUNG 3.6. Äquivalente Transformation

2“ transformiert nicht *eine* Aufgabe $\mathcal{A}_K$ in *eine* andere Aufgabe $\mathcal{A}_2$, sondern vielmehr eine Äquivalenzklasse $\mathcal{E}(\mathcal{A}_K)$ in eine andere Äquivalenzklasse $\mathcal{E}(\mathcal{A}_2)$.

Die oben beschriebene Darstellung von Funktionen g_2 nennen wir kanonische Darstellung. Somit kann man annehmen, das alle Kanten der Aufgabe die Form $\begin{array}{|c|c|}\hline \tilde{\alpha} & 0 \\ \hline 0 & \tilde{\alpha} \\ \hline \end{array}$ statt der allgemeinen Form $\begin{array}{|c|c|}\hline c & d \\ \hline a & b \\ \hline \end{array}$ haben. Für uns hat die Darstellung den Nachteil, dass sich die Kanten des ersten Typs (mit $g_{2\ldots}(1,0)=\infty$) nicht in dieser Form darstellen lassen. Wir werden im Weiteren die folgende Darstellung benutzen: $\begin{array}{|c|c|}\hline \alpha'' & 0 \\ \hline 0 & \alpha' \\ \hline \end{array}$. Dabei müssen die Zahlen α' und α'' nicht unbedingt gleich sein. Man kann leicht sehen, dass alle Kanten (alle Funktionen g_2) der erhaltenen Aufgabe in dieser Form darstellbar sind. Die Kanten des ersten Typs haben bereits diese Form ($\alpha'=0$ und $\alpha''=\infty$). Die Kanten des zweiten Typs können mit Hilfe der in der Abbildung 3.6 gezeigten äquivalenten Transformation in die gewünschte Form überführt werden. Dabei gilt $\alpha'+\alpha''=-\alpha$. Wenn die Aufgabe monoton ist ($\alpha \leq 0$), kann man zusätzlich fordern, dass beide α' und α'' nicht negativ sind.

Eine weitere Möglichkeit die erhaltene Aufgabe zu vereinfachen liegt in der Anwendung einer uniformen Transformation. Wendet man für alle Knoten (r,k) die folgende uniforme Transformation an

$$q_{2(r,k)}(0) = q_{2(r,k)}(0) - c, \;\; q_{2(r,k)}(1) = q_{2(r,k)}(1) - c$$
$$\text{mit} \;\; c = \min[q_{2(r,k)}(0), q_{2(r,k)}(1)],$$

so erhält man eine neue Aufgabe, in der alle q-s nicht negativ sind und für alle Knoten (r,k) wenigstens eines der beiden q-s Null ist.

Zusammenfassend, werden wir uns im Weiteren mit folgenden Aufgaben $\mathcal{A}(V,K,q,g)$ beschäftigen:

- Der Basisgraph $V=(R,E)$ ist beliebig.
- Die Zustandsmenge besteht aus zwei Zuständen $K=\{0,1\}$.
- Die Qualitätsfunktionen q_r besitzen nur nichtnegative Werte.
- Die Qualitätsfunktionen $g_{rr'}$ haben die Form $\begin{array}{|c|c|}\hline \alpha''_{rr'} & 0 \\ \hline 0 & \alpha'_{rr'} \\ \hline \end{array}$, wobei $\alpha'_{rr'}$ und $\alpha''_{rr'}$ nicht negativ sind.

Es genügt, nur derartige Aufgaben zu betrachten, weil beliebige monotone Aufgaben in diese Form überführbar sind.

3.3.2. Überführung der Aufgabe mit zwei Zuständen in eine MinCut Aufgabe. Zunächst möchten wir an die Aufgabestellung einer MinCut Aufgabe erinnern.

Sei $\mathcal{G}(\mathcal{V},\mathcal{E})$ ein gerichteter Graph. In der Menge der Knoten seien zwei spezielle Knoten s (Quelle) und t (Empfänger) ausgezeichnet. Allen gerichteten Kanten $(u,v)\in\mathcal{E}$ sind Kosten $c(u,v)$ zugeordnet. Ein Schnitt $\mathcal{C}$ ist eine Teilmenge von Kanten, so dass im Graphen $\mathcal{G}(\mathcal{V},\mathcal{E}/\mathcal{C})$ kein gerichteter Weg von der Quelle s zu dem Empfänger t existiert. Zusätzlich wird verlangt, dass diese Teilmenge im folgenden Sinn minimal ist. Entfernt man eine beliebige Kante aus dieser Teilmenge, so würde ein gerichteter Weg von s nach t existieren. Die Qualität eines Schnitts $Q(\mathcal{C})$ ist die Summe der Kosten aller Kanten der Teilmenge $\mathcal{C}$. Die Aufgabe besteht in der Suche nach dem Schnitt mit minimaler Qualität.

Betrachten wir einen Knoten v des Graphen $\mathcal{G}(\mathcal{V},\mathcal{E})$. Nehmen wir an, dass sowohl ein gerichteter Weg von diesem Knoten zum Empfänger als auch ein gerichteter Weg von der Quelle zu diesem Knoten existieren (im Weiteren betrachten wir nur Graphen, in denen dies für alle Knoten gilt). Weiterhin betrachten wir einen Schnitt $\mathcal{C}$ dieses Graphen. Es ist offensichtlich, dass im Graphen $\mathcal{G}(\mathcal{V},\mathcal{E}/\mathcal{C})$ entweder kein gerichteter Weg vom Knoten v zum Empfänger oder kein gerichteter Weg von der Quelle zu diesem Knoten existieren. Sollten beide Wege existieren, so existiert auch ein gerichteter Weg von der Quelle zum Empfänger, was der Definition des Schnittes widerspricht. Es ist auch klar, dass im Graphen $\mathcal{G}(\mathcal{V},\mathcal{E}/\mathcal{C})$ entweder ein gerichteter Weg vom Knoten v zum Empfänger oder ein gerichteter Weg von der Quelle zu diesem Knoten existiert. Sonst wäre der Schnitt nicht minimal. Somit kann man den Schnitt $\mathcal{C}$ als Zerlegung der Menge der Knoten in zwei Teilmengen S und T verstehen. Es gilt dabei $S\cup T=\mathcal{V}$, $S\cap T=\{\emptyset\}$, $s\in S, t\in T$. Eine solche Zerlegung nennt man auch „Partitionierung“ der Menge $\mathcal{V}$. Zur Teilmenge S gehören die Knoten, für die ein gerichteter Weg von der Quelle zu diesem Knoten existiert. Zur Teilmenge T gehören die Knoten, für die ein gerichteter Weg von dem Knoten zum Empfänger existiert. Folglich lässt sich die Qualität des Schnitts wie folgt schreiben:

$$Q(S,T)=\sum_{u\in S}\sum_{v\in T}c(u,v) \tag{3.18}$$

(Es sei $c(u,v)=0$ für alle $(u,v)\notin\mathcal{E}$). Somit besteht die Aufgabe in der Suche nach der Partitionierung mit minimaler Qualität:

$$(S,T)^*=\arg\min_{(S,T)}Q(S,T)=\arg\min_{(S,T)}\sum_{u\in S}\sum_{v\in T}c(u,v). \tag{3.19}$$

Die Aufgabe kann für beliebige Kosten $c(u,v)$ formuliert werden. Aber ohne Beschränkungen an die Kantenkosten ist die Aufgabe NP-vollständig. Sind aber alle Kosten nicht negativ, so ist die Aufgabe mit polynomialer Zeitkomplexität lösbar. Die Grundidee besteht darin, dass die MinCut Aufgabe als eine stückweise lineare Optimierungsaufgabe

formuliert werden kann. Wenn alle Kosten nicht negativ sind, ist diese Optimierungsaufgabe konvex. In diesem Fall lässt sich die dazu duale Optimierungsaufgabe formulieren. Letztere duale Optimierungsaufgabe ist unter dem Namen „MaxFlow Aufgabe“ bekannt [**9**]. Es existieren viele Algorithmen, die diese Aufgabe effizient lösen [**1**].

Schreiben wir die Aufgabestellung (3.19) in etwas anderer Form:

$$(S,T)^* = \arg\min_{(S,T)} \left[\sum_{u\in S/s} c(u,t) \; + \sum_{v\in T/t} c(s,v) \; + \sum_{u\in S/s} \sum_{v\in T/t} c(u,v) \right]. \tag{3.20}$$

Jetzt kehren wir zur MinSum Aufgabe mit zwei Zuständen zurück. Jedes Labeling $f : R \to \{0,1\}$ kann man auch als Partitionierung der Menge aller Knoten R in zwei Teilmengen R_0 und R_1 verstehen, wobei $r \in R_0$ wenn $f(r) = 0$ und $r \in R_1$ wenn $f(r) = 1$ gilt. Die Qualität dieses Labelings (dieser Partitionierung) ist:

$$\begin{aligned} Q(R_0,R_1) = & \sum_{r\in R_0} q_r(0) \; + \sum_{r\in R_1} q_r(1) \; + \\ & \sum_{\substack{(r,r')\in E \\ r\in R_0, r'\in R_1}} g_{rr'}(0,1) \; + \sum_{\substack{(r,r')\in E \\ r\in R_1, r'\in R_0}} g_{rr'}(1,0) \; + \\ & \sum_{\substack{(r,r')\in E \\ r\in R_0, r'\in R_0}} g_{rr'}(0,0) \; + \sum_{\substack{(r,r')\in E \\ r\in R_1, r'\in R_1}} g_{rr'}(1,1) \end{aligned}$$

(Man berücksichtige, dass die Menge E der Kanten eine Teilmenge $E \subset V \times V$ ist. Dabei gilt $(r',r) \notin E$, wenn $(r,r') \in E$). Da die Funktionen g die Form $\begin{array}{|c|c|}\hline \alpha''_{rr'} & 0 \\ \hline 0 & \alpha'_{rr'} \\ \hline \end{array}$ haben, kann die obige Formel vereinfacht werden:

$$Q(R_0,R_1) = \sum_{r\in R_0} q_r(0) \; + \sum_{r\in R_1} q_r(1) \; + \sum_{\substack{(r,r')\in E \\ r\in R_0, r'\in R_1}} \alpha'_{rr'} \; + \sum_{\substack{(r,r')\in E \\ r\in R_1, r'\in R_0}} \alpha''_{rr'}. \tag{3.21}$$

Sei $\alpha : R \times R \to \mathbb{R}$ eine Funktion, die wie folgt definiert ist:

$$\alpha(r,r') = \begin{cases} \alpha'_{rr'} & \text{wenn } (r,r') \in E \\ \alpha''_{rr'} & \text{wenn } (r',r) \in E \\ 0 & \text{sonst.} \end{cases}$$

Dann lässt sich die Formel (3.21) in folgender Form schreiben:

$$Q(R_0,R_1) = \sum_{r\in R_0} q_r(0) \; + \sum_{r\in R_1} q_r(1) \; + \sum_{r\in R_0} \sum_{r'\in R_1} \alpha(r,r'). \tag{3.22}$$

Vergleicht man die Formeln (3.20) und (3.22), so sieht man, dass beide Formeln bis auf Bezeichnungen dieselben sind.

Somit ergeben sich folgende Regeln für die Überführung einer Aufgabe mit zwei Zuständen in eine entsprechende MinCut Aufgabe:

- Die Menge der Knoten der MinCut Aufgabe ist die Menge der Knoten der MinSum Aufgabe mit den zwei zusätzlichen Knoten s und t („Quelle“ und „Empfänger“):

$$\mathcal{V} = R \cup \{s,t\}.$$

- Die Menge der Kanten der MinCut Aufgabe ist die „verdoppelte“ Menge der Kanten der MinSum Aufgabe mit zusätzlichen Kanten, die von der Quelle zu allen Knoten r führen bzw. von r zum Empfänger:

$$\begin{aligned}\mathcal{E} = &\{(r,r') : (r,r') \in E\} \cup \{(r',r) : (r,r') \in E\} \cup \\ &\{(s,r) : r \in R\} \cup \{(r,t) : r \in R\}.\end{aligned}$$

- Die Kantenkosten sind:

$$\begin{aligned} c(s,r) &= q_r(1) \\ c(r,t) &= q_r(0) \\ c(r,r') &= \alpha'_{rr'} \\ c(r',r) &= \alpha''_{rr'}. \end{aligned} \tag{3.23}$$

Fassen wir alle oben beschriebenen Schritte zusammen, nämlich: die Überführung „von K zu 2“, die Überführung der Funktionen q und g zu der benötigten Form und die Bildung der MinCut Aufgabe.

Gegeben sei eine Aufgabe $\mathcal{A} = (V,K,q,g)$ mit den folgenden Komponenten:

Der Basisgraph $V = (R,E)$ ist beliebig.
Die Zustandsmenge $K = \{1 \dots |K|\}$ ist eine geordnete Menge.
Die Qualitätsfunktionen q_r sind beliebig.
Die Qualitätsfunktionen $g_{rr'}$ sind monoton.

Die entsprechende MinCut Aufgabe $(\mathcal{G} = (\mathcal{V},\mathcal{E}), s \in \mathcal{V}, t \in \mathcal{V}, c)$ wird wie folgt gebildet:

- Die Menge der Knoten:

$$\mathcal{V} = \{(r,k) : r \in R,\ k = 1 \dots |K|-1\} \cup \{s\} \cup \{t\}.$$

- Die Menge der Kanten:

$$\begin{aligned}\mathcal{E} = &\big\{\big((r,k),(r,k+1)\big) : r \in R,\ k = 1 \dots |K|-2\big\} \cup \\ &\big\{\big((r,k+1),(r,k)\big) : r \in R,\ k = 1 \dots |K|-2\big\} \cup \\ &\big\{\big((r,k),(r',k')\big) : (r,r') \in E,\ k,k' = 1 \dots |K|-1\big\} \cup \\ &\big\{\big((r',k'),(r,k)\big) : (r,r') \in E,\ k,k' = 1 \dots |K|-1\big\} \cup \\ &\big\{\big(s,(r,k)\big) : r \in R,\ k = 1 \dots |K|-1\big\} \cup \\ &\big\{\big((r,k),t\big) : r \in R,\ k = 1 \dots |K|-1\big\}.\end{aligned}$$

- Die Kantenkosten:

$$\begin{aligned} &c\big((r,k),(r,k+1)\big) = 0,\ \ r \in R,\ k = 1 \dots |K|-2 \\ &c\big((r,k+1),(r,k)\big) = \infty,\ \ r \in R,\ k = 1 \dots |K|-2 \\ &c\big((r,k),(r',k')\big) = -\alpha_{rr'}(k,k')/2, \\ &\qquad (r,r') \in E,\ k = 1 \dots |K|-1,\ k' = 1 \dots |K|-1 \\ &c\big((r',k'),(r,k)\big) = -\alpha_{rr'}(k,k')/2, \\ &\qquad (r,r') \in E,\ k = 1 \dots |K|-1,\ k' = 1 \dots |K|-1 \\ &c\big(s,(r,k)\big) = \max[q'_{rk},0],\ r \in R,\ k = 1 \dots |K|-1 \\ &c\big((r,k),t\big) = \max[-q'_{rk},0],\ r \in R,\ k = 1 \dots |K|-1 \end{aligned}$$

mit

$$\begin{aligned}
&\alpha_{rr'}(k,k') = \\
&\quad g_{rr'}(k,k') + g_{rr'}(k+1,k'+1) - \\
&\quad g_{rr'}(k+1,k') - g_{rr'}(k,k'+1) \\
&q'_{rk} = q_r(k) - q_r(k+1) + \\
&\quad 1/2 \sum_{(r,r')\in E} [g_{rr'}(k,1) + g_{rr'}(k,|K|) - \\
&\quad g_{rr'}(k+1,1) - g_{rr'}(k+1,|K|)] .
\end{aligned} \tag{3.24}$$

Da die ursprüngliche Aufgabe $\mathcal{A}$ monoton ist, haben alle Kanten der entstehenden MinCut Aufgabe nicht negative Kosten.

Betrachten wir einen einfachen Algorithmus, der eine MinCut Aufgabe mit nicht negativen Kantenkosten löst. In [**9**] wurde gezeigt, das die Lösung einer solchen MinCut Aufgabe durch die Lösung einer entsprechenden MaxFlow Aufgabe erhalten werden kann. Eine MaxFlow Aufgabe formuliert man auf folgende Art. Gegeben sei ein gerichteter Graph $\mathcal{G}(\mathcal{V},\mathcal{E})$. In der Knotenmenge $\mathcal{V}$ sind zwei besondere Knoten s (Quelle) und t (Empfänger) ausgezeichnet. Jede gerichtete Kante $(v,v') \in \mathcal{E}$ hat ihre *Kapazität* $c(v,v')$. Ein Fluss entlang der Kante (v,v') (vom Knoten v zum Knoten v') ist eine nichtnegative Zahl $x(v,v')$. Man sagt auch: „Der Fluss $x(v,v')$ *fließt* entlang der Kante (v,v')“. Der Fluss entlang einer Kante darf dabei deren Kapazität nicht überschreiten:

$$x(v,v') \le c(v,v'), \quad \forall (v,v') \in \mathcal{E} . \tag{3.25}$$

Für alle Knoten außer den zwei besonderen (Quelle und Empfänger) muss zusätzlich gelten, dass der summarische *in* diesen Knoten eingehende Fluss gleich dem summarischen *aus* diesem Knoten ausgehenden Fluss sein soll:

$$\sum_{v':(v,v')\in\mathcal{E}} x(v,v') \;=\; \sum_{v':(v',v)\in\mathcal{E}} x(v',v), \quad \forall v \in \mathcal{V}/\{s,t\}. \tag{3.26}$$

Man benutzt auch die folgende Terminologie. Einen zusätzlichen Fluss $\triangle x$ entlang der Kante (v,v') *zu schicken* heißt den aktuellen Fluss entlang dieser Kante zu vergrößern: $x_{neu}(v,v') = x_{alt}(v,v') + \triangle x$. Die Kante heißt *gesättigt*, wenn kein zusätzlicher Fluss entlang dieser Kante geschickt werden kann, d.h. $x(v,v') = c(v,v')$.

Die Aufgabe besteht in der Suche nach dem maximalen Fluss $X = \big(x(v,v'),\ (v,v') \in \mathcal{E}\big)$, der (unter Berücksichtigung der Nebenbedingungen (3.25) und (3.26)) von der Quelle zum Empfänger geschickt werden kann:

$$X^* = \arg\max_X \sum_{v:(v,t)\in\mathcal{E}} x(v,t). \tag{3.27}$$

Eine MinCut Aufgabe und eine MaxFlow Aufgabe sind zueinander dual, wenn die Kantenkapazitäten der MaxFlow Aufgabe gleich den Kantenkosten der MinCut Aufgabe sind. In dem Fall ist der Wert des maximalen Flusses gleich dem Wert des minimalen Schnitts.

Der einfachste Algorithmus zur Lösung der MaxFlow Aufgabe basiert auf der iterativen Verbesserung des aktuellen Flusses. Sei $X^{(t)}$ ein aktueller Fluss (am Anfang ist der Fluss Null). Wenn der Fluss noch nicht maximal ist, lässt sich ein gerichteter Pfad (Augmenting Path) finden, entlang dessen der Fluss erhöht werden kann. Der Pfad ist eine Folge der Kanten $P = \{(s,v_1),(v_1,v_2)\dots(v_l,t)\}$, so dass alle Kanten (v_i,v_{i+1}) noch nicht gesättigt sind, d.h. $c(v_i,v_{i+1}) - x^{(t)}(v_i,v_{i+1}) > 0$. Diese Differenz zwischen der Kantenkapazität

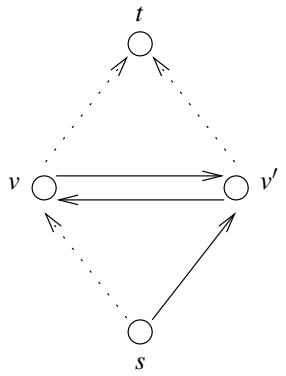


ABBILDUNG 3.7. Finden des minimalen Schnitts

und dem Fluss nennt man „Restkapazität“ der Kante und bezeichnet sie mit $r(v,v')$. Diese Zahl zeigt, wieviel Fluss die Kante (v,v') noch durchlassen kann. Entlang des gefundenen Pfades kann ein zusätzlicher Fluss geschickt werden. Der Wert dieses Flusses ist die minimale Restkapazität entlang des Pfades. Da die Restkapazitäten aller Kanten entlang des Pfades streng größer Null sind, ist der zusätzliche Fluss $\triangle x$ auch größer Null. Somit wird die Zielfunktion (3.27) nach der Anwendung dieses Flusses (d.h. $x(v_i,v_{i+1}) = x(v_i,v_{i+1}) + \triangle x$) verbessert. Die Restkapazitäten der Kanten entlang des Pfades müssen entsprechend korrigiert werden.

Der Algorithmus besteht aus den folgenden Schritten:

(1) Man initialisiere die Restkapazitäten mit den initialen Kapazitäten:
$r(v,v') = c(v,v'),\ \ \forall (v,v') \in \mathcal{E}$.

(2) Man finde einen gerichteten Pfad $P = \{(s,v_1),(v_1,v_2)\dots(v_l,t)\}$, so dass jede Kante entlang dieses Pfades eine streng positive Restkapazität hat:
$r(v_i,v_{i+1}) > 0,\ \ \forall (v_i,v_{i+1}) \in P$.

(3) Falls ein solcher Pfad nicht existiert, endet der Algorithmus. Den maximalen Fluss X findet man als
$x(v,v') = c(v,v') - r(v,v'),\ \ \forall (v,v') \in \mathcal{E}$.

(4) Man korrigiere die Restkapazitäten entlang des gefundenen Pfades:
$\triangle x = \min_{(v_i,v_{i+1})\in P} r(v_i,v_{i+1})$,
$r(v_i,v_{i+1}) = r(v_i,v_{i+1}) - \triangle x$,
$r(v_{i+1},v_i) = r(v_{i+1},v_i) + \triangle x,\ \ \forall (v_i,v_{i+1}) \in P$,
gehe zu (2).

Nach dem Anhalten des Algorithmus erhält man den maximalen Fluss. Die ursprüngliche Aufgabe ist allerdings, den minimalen Schnitt zu finden. Folglich entsteht die Frage, wie sich der minimale Schnitt der ursprünglichen MinCut Aufgabe aus dem maximalen Fluss ergibt.

Es lässt sich zeigen, dass der minimale Schnitt nur Kanten enthält, deren Restkapazitäten Null sind. Es ist aber auch offensichtlich, dass *nicht jede* Kante, deren Restkapazität Null ist, zum minimalen Schnitt gehört. Zum Beispiel entsteht dieser Fall in der Situation, die in der Abbildung 3.7 angegeben ist. Die Restkapazitäten sind schematisch mit durchgezogenen Linien (streng positive Restkapazität) und mit punktierten Linien (Restkapazität ist Null) gezeigt. Hier ist die Restkapazität der Kante (s,v) Null. Aber kein minimaler Schnitt enthält diese Kante (der minimale Schnitt ist in diesem Fall $\mathcal{C} = \{(v,t),(v',t)\}$).

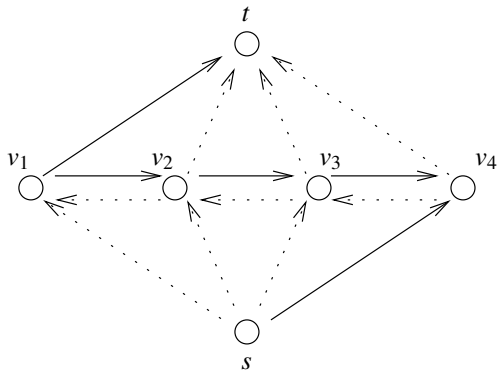


ABBILDUNG 3.8. Finden des minimalen Schnitts

Ein möglicher Ausweg ist, den Schnitt nicht als die Menge der Kanten zu betrachten, sondern als Partitionierung der Menge der Knoten in zwei Teilmengen S und T (siehe (3.19)). Diese Partitionierung kann auf folgende Art definiert werden. Ein Knoten v^* gehört zu der Teilmenge S wenn ein gerichteter Pfad $P(s,v^*) = \{(s,v_1),(v_1,v_2)\dots(v_l,v^*)\}$ existiert, so dass alle Restkapazitäten entlang dieses Pfades streng positiv sind. In dem Fall werden wir sagen, dass der Knoten v^* mit der Quelle *verbunden* ist. Ähnlich kann man die Existenz der Verbindung zwischen einem Knoten v^* und dem Empfänger definieren. Ein Knoten v^* gehört zu der Teilmenge T wenn ein gerichteter Pfad $P(v^*,t) = \{(v^*,v_1),(v_1,v_2)\dots(v_l,t)\}$ existiert, so dass alle Restkapazitäten entlang dieses Pfades streng positiv sind. Wenn man mit Hilfe dieser Definition für das Beispiel in der Abbildung 3.7 die Partitionierung findet, stellt es sich heraus, dass der Knoten v zu der Teilmenge S gehört, weil eine Verbindung zwischen diesem Knoten und der Quelle existiert: $P(s,v) = \{(s,v'),(v',v)\}$.

Es können allerdings Situationen eintreten, in denen die obige Definition der Partitionierung nicht hilft, den minimalen Schnitt zu finden. Nach dem Anhalten des MaxFlow Algorithmus, können Knoten entstehen, die weder mit der Quelle noch mit dem Empfänger verbunden sind. Dabei kann die Entscheidung, welcher der beiden Teilmengen ein solcher Knoten zuzuordnen ist, im Algemeinen nicht unabhängig getroffen werden. Betrachten wir dazu das Beispiel in der Abbildung 3.8. Nach der Partitionierung entsprechend der obigen Definition werden nur die Knoten v_1 und v_4 eindeutig zugeordnet: $v_1 \in T$, $v_4 \in S$. Die Knoten v_2 und v_3 bleiben unbestimmt (sie sind weder mit der Quelle noch mit dem Empfänger verbunden). Ordnet man die Knoten $v_2 \in S$, $v_3 \in T$ zu, so erhält man eine Partitionierung, die keinem minimalen Schnitt entspricht.

Um dieses Problem zu lösen kann man das Finden der Partitionierung als Aufgabe der Suche nach einem Labeling formulieren. Die erhaltenen Restkapazitäten lassen sich auf folgende Art interpretieren. Ist eine Restkapazität $r(v,v')$ streng größer Null, so bedeutet das, dass der minimale Schnitt diese Kante nicht enthält. Somit ist die Zuordnungsvariante $v \in S$, $v' \in T$ verboten. Analog kann man die Situation $r(v',v) > 0$ interpretieren – die Zuordnungsvariante $v' \in S$, $v \in T$ ist verboten. Wenn die beiden Restkapazitäten $r(v,v')$ und $r(v',v)$ Null sind, heißt das, dass keine der vier möglichen Zuordnungsvarianten verboten ist. Die oben beschriebenen Interpretationen kann man benutzen, um die Aufgabe der Suche nach der Partitionierung als eine OrAnd Aufgabe zu formulieren.

- Der Basisgraph der OrAnd Aufgabe $V = (R,E)$ ist dem Graphen der MaxFlow Aufgabe isomorph, d.h. jedem Knoten v der MaxFlow Aufgabe entspricht ein Knoten r der OrAnd Aufgabe.
- Die Zustandsmenge besteht aus zwei Zuständen: $K = \{0,1\}$. Die Zustände haben die folgenden Bedeutungen. In einem Knoten r heißt $k = 0$ „Der entsprechende Knoten v der MaxFlow Aufgabe gehört zur Teilmenge S“, $k = 1$ heißt „Dieser Knoten gehört zur Teilmenge T“.
- Die Qualitätsfunktionen $\tilde{q}_r : K \to \{0,1\}$ sind binäre Funktionen, die angeben, zu welcher Teilmenge (S oder T) der entsprechende Knoten v gehören kann. Ist die Restkapazität $r(s,v)$ streng größer Null, so bedeutet das, dass der Knoten v zur Teilmenge T nicht gehört, d.h. $\tilde{q}_r(1) = 0$. Anderenfalls gilt $\tilde{q}_r(1) = 1$. Ähnlich interpretiert man $r(v,t) > 0$ als $\tilde{q}_r(0) = 0$ und $r(v,t) = 0$ als $\tilde{q}_r(0) = 1$. Es ist klar das die beiden Restkapazitäten $r(s,v)$ und $r(v,t)$ nicht zugleich streng positiv sein können, weil in dem Fall der MaxFlow Algorithmus nicht anhalten würde.
- Die Qualitätsfunktionen $\tilde{g}_{rr'} : K \times K \to \{0,1\}$ sind binäre Funktionen, die angeben, welche Zuordnungskombinationen auf dem entsprechenden Paar (v,v') erlaubt bzw. verboten sind. Die Kombinationen $(v \in S, v' \in S)$ und $(v \in T, v' \in T)$ sind immer erlaubt, d.h. $\tilde{g}_{rr'}(0,0) = \tilde{g}_{rr'}(1,1) = 1$. Ist die Restkapazität entlang dieser Kante streng größer Null $r(v,v') > 0$, so gilt für die entsprechende Kante $\tilde{g}_{rr'}(0,1) = 0$, anderenfalls $\tilde{g}_{rr'}(0,1) = 1$. Ähnlich interpretiert man $r(v',v) > 0$ als $\tilde{g}_{rr'}(1,0) = 0$ und $r(v',v) = 0$ als $\tilde{g}_{rr'}(1,0) = 1$.

(In der Abbildung 3.9 sind obige Regeln schematisch gezeigt). Somit kann man die Aufgabe des Finden des minimalen Schnitts wie folgt formulieren. Man finde ein Labeling $f : R \to \{0,1\}$, so dass für alle Knoten und für alle Kanten des Basisgraphen die ausgewählten Zustände bzw. Zustandspaare erlaubt sind. Das ist eine OrAnd Aufgabe (3.12), in der ein Labeling gesucht wird, für welches

$$\bigwedge_{r \in R} \tilde{q}_r\big(f(r)\big) \quad \wedge \bigwedge_{(r,r') \in E} \tilde{g}_{rr'}\big(f(r), f(r')\big) = 1$$

gilt. Man kann leicht sehen, dass diese OrAnd Aufgabe monoton ist (im Sinn der OrAnd Monotonie). Da die Werte $g_{rr'}(1,1)$ und $g_{rr'}(0,0)$ Eins sind, ist die Forderung der Monotonie für den OrAnd Fall

$$g_{rr'}(0,1) \wedge g_{rr'}(1,0) \Rightarrow g_{rr'}(1,1) \wedge g_{rr'}(0,0)$$

immer erfüllt. Wie bereits gesagt, lassen sich monotone OrAnd Aufgaben durch den Relaxation Labeling Algorithmus lösen.

Man kann sehen, dass der oben beschriebene MaxFlow Algorithmus nicht besonders effizient ist, weil die Anzahl der Iterationen (2)–(4) vom Wert des maximalen Flusses abhängt. Wir haben den Algorithmus nur angegeben, um den Begriff „Restkapazität“ einzuführen. Die meisten bezüglich der Zeitkomplexität besseren Algorithmen (auch der in [**1**] beschriebene Preflow-Push Algorithmus, den wir in der Praxis benutzen) basieren auch auf diesem Begriff. Anhand der Analyse der Restkapazitäten ist es möglich, den minimalen Schnitt (und somit die Lösung der ursprünglichen MinSum Aufgabe) zu finden.

3.4. Schätzung der marginalen Wahrscheinlichkeitsverteilungen

In diesem Abschnitt werden wir das Problem diskutieren, welches bei der Formulierung der Aufgabe der Stereorekonstruktion als Aufgabe der Bayesschen Entscheidung mit

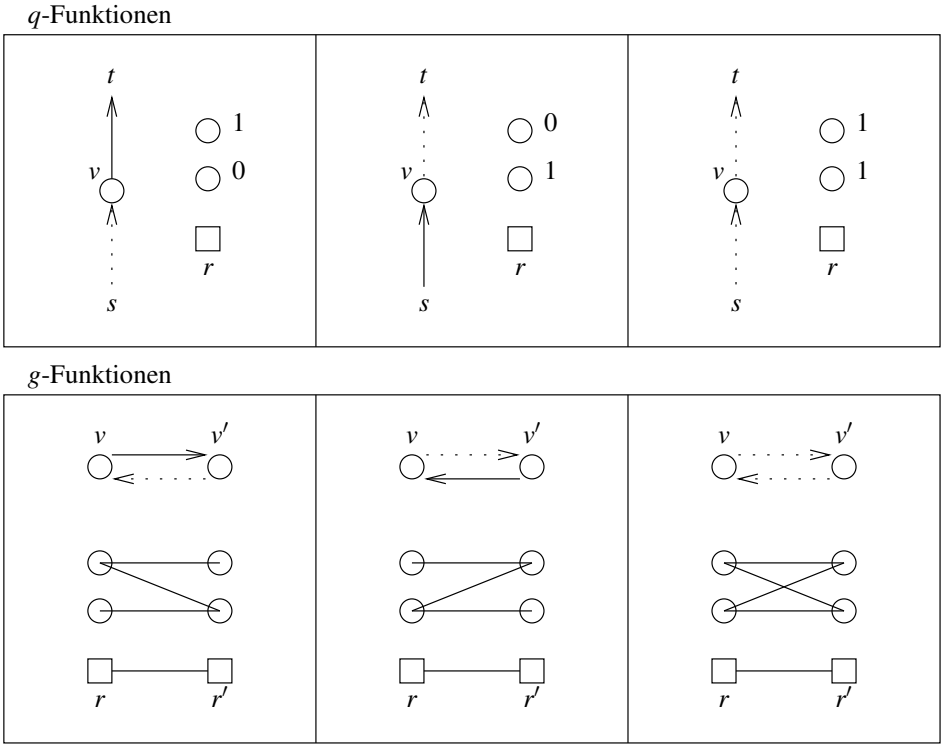


ABBILDUNG 3.9. Überführung in eine OrAnd Aufgabe

additiven Kostenfunktionen entsteht. Das Ziel ist, für jedes Paar (r^*,k) die Wahrscheinlichkeit zu berechnen, dass das Labeling f im Knoten r^* durch den Zustand k geht. Die benötigten Größen lassen sich wie folgt ausdrücken:

$$P\big(X, f(r^*)=k\big) = \frac{1}{Z} \sum_{f:f(r^*)=k} \left[\prod_{(r,r')\in E} g_{rr'}\big(f(r), f(r')\big) \cdot \prod_{r\in R} q_r\big(f(r)\big) \right]. \tag{3.28}$$

Zur Zeit ist kein Algorithmus bekannt, der den Wert der obigen Formel exakt berechnet (außer für einige sehr eingeschränkte Spezialfälle). Es ist allerdings möglich, die benötigten Wahrscheinlichkeiten $P\big(f(r^*)=k \mid X\big)$ zu schätzen, ohne die Formel (3.28) explizit zu verwenden. Wir werden im Weiteren eigentlich nicht die SumProd Aufgabe (3.28), sondern die Aufgabe „Man schätze die Wahrscheinlichkeitswerte $P\big(f(r^*)=k \mid X\big)$“ diskutieren.

Dafür kann man den so genannten Gibbs Sampler benutzen [**10**]. Die Idee besteht darin, dass Labelings mit Hilfe einer bestimmten Methode *generiert* werden können. Betrachten wir diese ausführlicher.

Sei $f : R \to K$ ein Labeling des Graphen $V = (R,E)$. Sei $P(f)$ eine (beliebige) Wahrscheinlichkeitsverteilung der Labelings. Die Generierungsprozess fängt von einem initialen Labeling $f^{(0)}$ an und besteht aus Wiederholung der folgenden Generierungsschritte:

(1) Man wähle zufällig einen Knoten r und berechne die Wahrscheinlichkeitsverteilung der Zustände in diesem Knoten unter der Bedingung der fixierten Zustände

im Rest des Graphen:

$$P\big(f(r) = k \mid f(r') = f^{(t)}(r'),\ r' \in R/\{r\}\big). \tag{3.29}$$

(2) Man generiere den neuen Zustand k^* im Knoten r entsprechend der Wahrscheinlichkeitsverteilung (3.29). Somit entsteht ein neues Labeling $f^{(t+1)}$, in dem nur der Zustand im Knoten r geändert wird, d.h. $f^{(t+1)}(r) = k^*$ und $f^{(t+1)}(r') = f^{(t)}(r')$, $r' \in R/\{r\}$.

Nach einer bestimmten Anzahl der Generierungsschritte (diese Anzahl diskutieren wir später) geht die Wahrscheinlichkeit, dass ein Labeling f durch den Prozess generiert wird, gegen $P(f)$. Streng genommen, gilt das nur, wenn die Anzahl der Generierungsschritte gegen Unendlich strebt. Ersetzt man die Unendlichkeit durch eine sehr große Zahl, so kann man die relative Häufigkeit des Auftretens eines Labelings f als Schätzung für $P(f)$ verstehen.

Eine weitere Eigenschaft muss erfüllt werden, damit die relativen Häufigkeiten der Labelings zu den Wahrscheinlichkeiten der Labelings konvergieren – der Prozess muss ergodisch sein. Es ist leicht zu sehen, dass der Generierungsprozess ergodisch ist, wenn alle Werte der Funktion g ungleich Null sind (alle Labelings haben streng positive Wahrscheinlichkeit). Es kann einfach gezeigt werden, dass diese Eigenschaft auch für die scharfe Glattheitsforderung (2.11) erfüllt ist, wenn der Parameter δ größer Null ist.

Da die Wahrscheinlichkeitsverteilung $P(f)$ in unserem Fall eine Gibbssche Wahrscheinlichkeitsverteilung zweiter Ordnung ist

$$P(f) \ \sim \prod_{(r,r')\in E} g_{rr'}\big(f(r), f(r')\big) \cdot \prod_{r\in R} q_r\big(f(r)\big), \tag{3.30}$$

lässt sich die bedingte Wahrscheinlichkeitsverteilung (3.29) wie folgt ausdrücken:

$$P\big(f(r) = k \mid f(r') = f^{(t)}(r'),\ (r,r') \in E\big) = \frac{h(k)}{\sum\limits_{k'} h(k')} \tag{3.31}$$

mit

$$h(k) = q_r(k) \cdot \prod_{(r,r')\in E} g_{rr'}\big(k, f^{(t)}(r')\big).$$

Somit ist es möglich, diesen Prozess einfach zu realisieren (die Wahrscheinlichkeitsverteilungen (3.29) zu berechnen).

Bezeichnen wir mit $H(\cdot)$ die Häufigkeit eines Ereignisses während des Generierungsprozesses. Zum Beispiel ist $H(f)$ die Häufigkeit des Labelings f (das ist in unserem Fall ein elementares Ereignis) – wie oft das Labeling f während des Prozess generiert wurde. Die Zahlen $H(f)/T$, wobei T die Anzahl der Generierungsschritte ist, kann man als Schätzung für $P(f)$ annehmen. Unsere Ziel ist es, die marginalen Wahrscheinlichkeiten der Zustände

$$P\big(f(r) = k\big) = \sum_{f:f(r)=k} P(f)$$

zu berechnen. Ersetzt man in der obigen Formel $P(f)$ durch $H(f)/T$, so erhält man

$$P\big(f(r) = k\big) \ \approx \sum_{f:f(r)=k} H(f)/T \ = \ \frac{H\big(f(r) = k\big)}{T}.$$

Das ist die relative Häufigkeit des Zustandes k im Knoten r während des Generierungsprozesses.

Eine weitere Frage, die bei der Realisierung des Generierungsprozesses entsteht, ist, wie groß die Anzahl der Generierungsschritte sein soll, um eine mehr oder weniger sichere Statistik mit Hilfe des Generierungsprozesses zu sammeln. Diese Frage lässt sich in folgende zwei Fragen zerlegen: a) Wieviel entsprechend der Wahrscheinlichkeitsverteilung $P(f)$ unabhängig generierte Labelings werden benötigt, um eine sichere Statistik für die Schätzung der marginalen Wahrscheinlichkeitsverteilungen zu sammeln (die Größe der Stichprobe der Labelings) und b) Wieviel Generierungsschritte sollen durchgeführt werden, um ein Labeling entsprechend $P(f)$ zu generieren. Sei die oben genannte Anzahl der Labelings in der Stichprobe bekannt. Dann sollte der Algorithmus für die Schätzung der marginalen Wahrscheinlichkeitsverteilungen wie folgt aussehen:

(1) Man wähle zufällig ein Labeling.
(2) Man generiere Labelings solange, bis er sicher ist, dass das zur Zeit generierte Labeling entsprechend der ursprünglichen Gibbsschen Wahrscheinlichkeitsverteilung erzeugt wurde.
(3) Man nehme das zuletzt generierte Labeling in die Stichprobe auf.

Die Schritte (1)–(3) werden so oft wiederholt, wieviel Labelings in der Stichprobe benötigt werden.

Die Hauptschwierigkeit liegt darin, dass die Anzahl der Generierungsschritte in (2) bei praktischen Anwendungen in der Regel riesig groß ist. In [**20**] wurde eine Methode vorgestellt, die es erlaubt, diese Anzahl zu schätzen. Allerdings zeigen entsprechende Experimente, dass diese Anzahl aus praktischer Sicht nicht verwendbar ist (es werden Millionen von Generierungsschritten benötigt, nur um ein Labeling zu erzeugen).

Eine andere Vorgehensweise wäre, den Generierungsprozess mit einem Labeling $f^{(0)}$ zu initialisieren, die Labelings entsprechend der Regel (3.31) zu generieren und dabei *jedes* generierte Labeling in die Stichprobe aufzunehmen. Es ist offensichtlich, dass die erzeugte Stichprobe der Labelings in diesem Fall nicht unabhängig ist. Folglich entsteht die Frage, wie groß die Anzahl der Labelings in einer solchen nicht unabhängigen Stichprobe sein soll, um eine sichere Statistik zu erhalten. Es ist dabei klar, dass diese Anzahl viel größer ist, als die Anzahl der Generierungsschritte für die Generierung nur eines Labelings entsprechend der Wahrscheinlichkeitsverteilung $P(f)$ ausgehend von einem zufälligen Labeling. Somit scheint diese Vorgehensweise aus praktischer Sicht auch nicht verwendbar zu sein.

Andererseits kann man bei praktischen Anwendungen sehr oft durch verschiedene Heuristiken die Anzahl der Generierungsschritte deutlich reduzieren und somit die Laufzeit des Programms in einen vernünftigen Rahmen bringen. Präziser gesagt, entstehen subjektiv gute Ergebnisse schon nach der Anwendung einer Anzahl von Generierungsschritten, die viel kleiner ist, als es theoretisch korrekt wäre. Im Kapitel 4 betrachten wir solche für die Stereorekonstruktion nützliche Heuristiken ausführlich. Dabei werden wir die Vorgehensweise benutzen, in der jedes generierte Labeling in die Stichprobe aufgenommen wird. Die benötigte Anzahl der Generierungsschritte bestimmen wir experimentell.

3.5. Wahl der Parameter

In diesem Abschnitt diskutieren wir das Problem der geeigneten Wahl der Parameter. Die oben beschriebenen Wahrscheinlichkeitsmodelle enthalten viele Parameter – die Funktionen $g_{rr'}$, die lokalen Funktionen q_r usw. In diesem Zusammenhang entsteht die Frage, ob man die Wahl dieser Parameter (oder einiger davon) automatisieren kann. Im Weiteren

geben wir an, wie sich diese Bestimmung für die Aufgabe der Stereorekonstruktion durchführen lässt. Einige zu den unten beschriebenen Ansätzen ähnlichen Beispiele findet man in [**8, 14**].

Für die Ermittelung von Parametern verwendet man oft das so genannte *Maximum Likelihood* Prinzip. Dieser Ansatz lässt sich wie folgt formulieren. Gegeben sei eine Klasse von Wahrscheinlichkeitsverteilungen $P(x,k;\theta)$. Hier ist $x \in X$ die observable Größe, $k \in K$ ist der verborgene Zustand und $\theta \in \Theta$ ist ein Parameter. Weiterhin sei eine Stichprobe L gegeben. Beim überwachten Lernen besteht die Stichprobe aus Paaren (x,k), beim nicht überwachten Lernen besteht die Stichprobe nur aus Werten von x. Im Weiteren betrachten wir den nicht überwachten Fall, d.h. $L = (x_1, x_2, \ldots, x_{|L|})$. Man sucht den Wert des Parameters θ, bei dem die Wahrscheinlichkeit der Stichprobe maximal wird. Dabei wird angenommen, dass die Muster x_l der Stichprobe voneinander unabhängig generiert wurden:

$$\theta^* = \arg\max_{\theta\in\Theta} \prod_{l=1}^{|L|} \sum_{k\in K} P(x_l,k;\theta). \tag{3.32}$$

Das anschließende Ziel ist, eine Bayessche Entscheidungsstrategie $d : X \to D$ zu ermitteln, die jedem Wert von x einen Wert $d(x)$ aus der Menge der Entscheidungen D zuordnet. Dabei wird das Risiko einer Kostenfunktion $C(d,k)$ minimiert:

$$d(x) = \arg\min_{d\in D} \sum_k P(x,k;\theta^*)C(d,k). \tag{3.33}$$

Für die Spezifizierung der „wahren“ Wahrscheinlichkeitsverteilung benutzt man den aus (3.32) erhaltenen Wert θ^*.

Im Fall der Stereorekonstruktion ist die observable Größe x das Bildpaar X (die Stichprobe besteht in dem Fall aus einem einzigen Element), die verborgenen Zustände sind die Labelings f, die Wahrscheinlichkeitsverteilung $P(x,k;\theta)$ ist die Gibbssche Wahrscheinlichkeitsverteilung. Somit kann man die Formel (3.32) wie folgt ausdrücken:

$$\theta^* = \arg\max_{\theta\in\Theta} \sum_{f\in|K|^{|R|}} P(f;\theta)\cdot P(X|f;\theta).$$

Im Weiteren betrachten wir den Fall, in dem nur die Funktion q unbekannt ist (das a-priori Wahrscheinlichkeitsmodell ist fixiert):

$$q^* = \arg\max_{q} \sum_{f\in|K|^{|R|}} P(f)\cdot P(X|f;q). \tag{3.34}$$

Für die Lösung der Aufgabe (3.32) benutzen wir den EM-Algorithmus ([**6, 19, 27**]). Dieser Algorithmus basiert auf der iterativen Maximierung des Funktionals

$$\sum_k \alpha(k)\ln P(x,k;\theta) - \sum_k \alpha(k)\ln\frac{P(x,k;\theta)}{\sum_{k'} P(x,k';\theta)}, \tag{3.35}$$

wobei $\alpha(k) \geq 0$ und $\sum_k \alpha(k) = 1$ gilt. Jede Iteration besteht aus den zwei folgenden Schritten. Im ersten Schritt – man nennt ihn E-Schritt – werden die Zahlen α so gewählt, dass der zweite Summand bezüglich $P(x,k;\theta)$ maximal wird:

$$\alpha^{(n)}(k) = P^{(n)}(k|x;\theta^{(n)}). \tag{3.36}$$

Im zweiten Schritt – man nennt ihn M-Schritt – wird der erste Summand bezüglich des Parameters maximiert:

$$\theta^{(n+1)} = \arg\max_{\theta} \sum_k \alpha^{(n)}(k)\ln P(x,k;\theta). \tag{3.37}$$

Setzen wir die Formel (3.36) und die Formel für die Gibbssche Wahrscheinlichkeitsverteilung (2.16) in die Formel für den M-Schritt (3.37) (unter der Annahme (3.34)) ein. Dabei berücksichtigen wir, dass die a-priori Wahrscheinlichkeit $P(f)$ für die Optimierung irrelevant ist. Somit erhalten wir

$$\begin{aligned} q^{(n+1)} &= \arg\max_q \sum_f P^{(n)}\big(f|X;q^{(n)}\big)\cdot\sum_r \ln q_r\big(f(r)\big) = \\ &\arg\max_q \sum_r\sum_f P^{(n)}\big(f|X;q^{(n)}\big)\cdot\ln q_r\big(f(r)\big) = \\ &\arg\max_q \sum_r\sum_k\sum_{f:f(r)=k} P^{(n)}\big(f|X;q^{(n)}\big)\cdot\ln q_r(k) = \\ &\arg\max_q \sum_r\sum_k P^{(n)}\big(f(r)=k\ |X;q^{(n)}\big)\cdot\ln q_r(k). \end{aligned} \tag{3.38}$$

Um die obige Aufgabe (3.38) zu lösen, erinnere man sich daran, dass die Funktion q eine Funktion des Ähnlichkeitsmaßes ist: $q_r(k) = Q\big(A(r,k)\big)$ mit $Q(a) \geq 0$ und $\sum_{a=0}^{\infty} Q(a) = 1$ (siehe Abschnitt 2.4). Betrachten wir zunächst den Fall, dass die Funktion Q eine beliebige Wahrscheinlichkeitsverteilung $Q : \mathcal{A} \to \mathbb{R}^+$ ist. Die Lösung der Optimierungsaufgabe (3.38) ergibt sich dann auf folgende Art. Bezeichnen wir mit Z_a die Menge aller Paare (r,k), in denen das Ähnlichkeitsmaß $A(r,k)$ den Wert a annimmt: $Z_a = \{(r,k) : A(r,k) = a\}$. Somit ist die Lösung

$$Q^{(n+1)}(a) = \frac{1}{|R|}\sum_{(r,k)\in Z_a} P^{(n)}\big(f(r)=k\ |X;q^{(n)}\big). \tag{3.39}$$

In der Praxis schränken wir die Menge der Funktionen Q auf folgende Art ein:

$$Q(a) = C\frac{1}{\sigma^l}\exp\left[-\frac{a}{\sigma^2}\right]. \tag{3.40}$$

Hier ist σ der Parameter dieser Funktion. Somit kann die Formel (3.38) wie folgt modifiziert werden:

$$\begin{aligned} \sigma^{(n+1)} &= \arg\max_\sigma \sum_r\sum_k \left[P^{(n)}\big(f(r)=k\ |X;\sigma^{(n)}\big)\cdot\big(-l\cdot\ln\sigma - \frac{A(r,k)}{\sigma^2}\big)\right] = \\ &\arg\min_\sigma \left[|R|\cdot l\cdot\ln\sigma + \frac{1}{\sigma^2}\sum_r\sum_k A(r,k)\cdot P^{(n)}\big(f(r)=k\ |X;\sigma^{(n)}\big)\right]. \end{aligned} \tag{3.41}$$

Die Lösung dieser Aufgabe ergibt sich als:

$$\sigma^{(n+1)2} = \frac{2}{|R|\cdot l}\sum_r\sum_k A(r,k)\cdot P^{(n)}\big(f(r)=k\ |X;\sigma^{(n)}\big). \tag{3.42}$$

Um diesen Wert zu berechnen, werden die marginalen Wahrscheinlichkeiten $P^{(n)}\big(f(r)=k\ |X;\sigma^{(n)}\big)$ benötigt. Sie können mit Hilfe des Gibbs Samplers geschätzt werden (siehe vorigen Abschnitt). Somit kann man den Parameter *während* des Generierungsprozesses (der benutzt wird, um die Bayessche Entscheidung zu ermitteln) anlernen.

An der Stelle möchten wir das Maximum Likelihood Prinzip kritisieren. Bei diesem Ansatz geht man davon aus, dass die Beobachtung eine möglichst große Wahrscheinlichkeit hat. In manchen Fällen (insbesondere im Fall der Stereorekonstruktion) ist diese Annahme nicht begründet.

Bevor wir einen anderen Ansatz zur Wahl der Parameter formulieren, möchten wir das Maximum Likelihood Prinzip aus einem etwas anderen Gesichtspunkt betrachten. Sei

$P(x,k,\theta)$ eine Wahrscheinlichkeitsverteilung mit der observable Größe $x \in X$ und mit dem verborgenen Zustand $k \in K$. Die Variable $\theta \in \Theta$ nennen wir nach wie vor den Parameter. Im Unterschied zum vorigen Modell ist der Parameter hier auch wie x und k eine Zufallsvariable. Weiterhin betrachten wir eine „gewöhnliche" Erkennungsaufgabe: zu ermitteln ist eine Bayessche Entscheidungsstrategie, die jedem Wert x eine Entscheidung $d(x)$ zuordnet. Sei die Menge der Entscheidungen die Menge aller Paare (θ,d), $\theta \in \Theta$, $d \in D$. Somit ist die Aufgabe, für eine gegebene Beobachtung x die Entscheidung $d(x)$ zu bestimmen, die das Risiko minimiert:

$$d(x) = (\theta^*, d^*) = \arg\min_{\theta,d} \sum_{\theta',k} P(x,k,\theta') \cdot C(\theta,d,\theta',k). \tag{3.43}$$

Betrachten wir die folgende Kostenfunktion C:

$$C(\theta,d,\theta',k) = \begin{cases} G & \text{wenn } \theta \neq \theta' \\ C(d,k) & \text{sonst.} \end{cases} \tag{3.44}$$

Hier ist G eine sehr große Zahl. Setzt man die obige Formel für die Kostenfunktion in (3.43) ein, so erhält man folgende Aufgabe:

$$\min_{\theta} \left[G \cdot (1 - P(x,\theta)) + \min_{d} \sum_{k} P(x,k,\theta) \cdot C(d,k) \right].$$

Wenn die Zahl G genügend groß ist, ergibt sich die Lösung durch

$$\theta^* = \arg\max_{\theta} P(x,\theta)$$
$$d^* = \arg\min_{d} \sum_{k} P(x,k,\theta^*) \cdot C(d,k).$$

Nehmen wir weiterhin an, dass die a-priori Wahrscheinlichkeit $P(\theta)$ des Parameters konstant ist. Dann lassen sich die obigen Formeln wie folgt schreiben:

$$\theta^* = \arg\max_{\theta} P(x|\theta) \tag{3.45}$$

$$d^* = \arg\min_{d} \sum_{k} P(x,k|\theta^*) \cdot C(d,k). \tag{3.46}$$

Fasst man die *bedingte* Wahrscheinlichkeitsverteilung $P(x,k|\theta)$ als eine *parametrisierte* Wahrscheinlichkeitsverteilung $P(x,k;\theta)$ auf, so entspricht (3.45) dem Maximum Likelihood Prinzip. Die Formel (3.46) ist eine Aufgabe der Bayesschen Entscheidung, wobei die Wahrscheinlichkeitsverteilung durch den angelernten Parameter θ^* spezifiziert ist. Somit ist das Maximum Likelihood Prinzip zusammen mit der nachfolgenden Ermittelung der Bayesschen Entscheidung nichts anderes, als die Lösung der Erkennungsaufgabe (3.43) mit der Kostenfunktion (3.44) unter Annahme, dass die a-priori Wahrscheinlichkeit $P(\theta)$ des Parameters konstant ist.

Man kann leicht sehen, dass die Kostenfunktion (3.44) nicht die bestmögliche ist. Bei praktischen Anwendungen ist der Wert des Parameters oft überhaupt irrelevant. Das bedeutet, dass die Kostenfunktion nicht vom Parameter abhängen darf, d.h. $C(\theta,d,\theta',k) = C(d,k)$. Auch die Menge der Entscheidungenden hängt nicht vom Wert des Parameter ab. Setzt man diese (vom Parameter nicht abhängige) Kostenfunktion in die Formel (3.43) ein, so entsteht die folgende Optimierungsaufgabe:

$$d^* = \arg\min_{d} \sum_{\theta} \sum_{k} [C(d,k) \cdot P(x,k|\theta) \cdot P(\theta)] =$$
$$\arg\min_{d} \sum_{k} C(d,k) \cdot P(x,k). \tag{3.47}$$

Mit anderen Worten, muss man bei diesem Ansatz über alle Werte des Parameters *absummieren* statt bezüglich des Parameters *minimieren*.

Anschließend betrachten wir, wie der obige Ansatz für die Aufgabe der Stereorekonstruktion formuliert werden kann. Wir ändern das Wahrscheinlichkeitsmodell (2.16), indem wir die Wahrscheinlichkeitsverteilung auf Dreitupeln (X, f, θ) formulieren:

$$P(X,f,\theta) = P(\theta)\cdot P(X,f|\theta) = \\ P(\theta)\cdot\frac{1}{Z}\prod_{rr'} g_{rr'}\big(f(r),f(r')\big)\cdot\prod_{r} q_r\big(f(r),\theta\big).$$

In dem Fall, wenn die Funktion Q die Form (3.40) hat, ist diese Wahrscheinlichkeitverteilung

$$P(X,f,\sigma) = P(\sigma)\cdot\frac{1}{Z}\cdot C^{|R|}\cdot\prod_{rr'} g_{rr'}\big(f(r),f(r')\big)\cdot\prod_{r}\frac{1}{\sigma^l}\exp\left[-\frac{A\big(r,f(r)\big)}{\sigma^2}\right].$$

Dieses Modell kann man als eine Gibbssche Wahrscheinlichkeitsverteilung dritter Ordnung verstehen. Es ist auch leicht zu sehen, dass bei der fixierten Beobachtung X (d.h. bei der Fixierung von $A(r,k)$) die Wahrscheinlichkeitsverteilung $P(f,\sigma|X)$ eine Gibbssche Wahrscheinlichkeitsverteilung zweiter Ordnung ist. Führen wir in den Basisgraphen der Aufgabe einen zusätzlichen Knoten r_σ ein, der dem Parameter σ entspricht. Die Zustandsmenge in diesem Knoten ist der Wertebereich von σ (bezeichnen wir diesen mit Σ). Die Menge der Labelings ist somit $K^R\times\Sigma$. Weiterhin führen wir neue Kanten (r,r_σ) ein, die den Knoten r_σ mit allen anderen Knoten verbinden. Die Funktion g_{rr_σ} entlang dieser neuen Kanten definieren wir als

$$g_{rr_\sigma}(k,\sigma) = \frac{1}{\sigma^l}\exp\left[-\frac{A(r,k)}{\sigma^2}\right].$$

Die Funktion q ändern wir entsprechend wie folgt:

$$q_r(k) = 1$$
$$q_{r_\sigma}(\sigma) = P(\sigma).$$

Somit ist die für die Ermittelung der Bayesschen Entscheidung benötigte Wahrscheinlichkeitsverteilung

$$P(f|X)\sim q_{r_\sigma}\big(f(r_\sigma)\big)\cdot\prod_{rr'} g_{rr'}\big(f(r),f(r')\big)$$

mit

$$f(r)\in K \qquad \text{wenn } r\neq r_\sigma$$
$$f(r_\sigma)\in\Sigma.$$

Um die Bayessche Entscheidung für eine additive Kostenfunktion zu ermitteln, werden die marginale Wahrscheinlichkeiten $P\big(f(r)=k\ |X\big)$ benötigt (siehe Abschnitt 2.3). Sie können genauso wie für das vorherige Modell (2.16) (in dem der Parameter σ fixiert ist) mit Hilfe des Gibbs Samplers geschätzt werden. Der Unterschied ist nur, dass man im Knoten r_σ den Zustand (einen Wert von σ) auch wie in allen anderen Knoten entsprechend der Wahrscheinlichkeitsverteilung $P(\sigma|X,f)$ *generieren* muss.

Obwohl der obige Ansatz vielversprechend aussieht, unterscheidet er sich bei der praktischen Anwendung (insbesondere bei der Stereorekonstruktion) kaum vom Maximum Likelihood Prinzip. Die Ursache ist, dass die Wahrscheinlichkeitsverteilung $P(\sigma|X,f)$ nur in einem sehr kleinen Bereich von σ praktisch ungleich Null ist. Betrachten wir dies etwas

ausführlicher. Die für die Generierung benötigte Verteilung lässt sich wie folgt schreiben:

$$P(\sigma|X,f) \sim P(\sigma) \cdot \prod_r \frac{1}{\sigma^l} \exp\left[-\frac{A\big(r,f(r)\big)}{\sigma^2}\right] =$$

$$P(\sigma) \cdot \frac{1}{\sigma^{l|R|}} \exp\left[-\frac{\sum_r A\big(r,f(r)\big)}{\sigma^2}\right] = P(\sigma) \cdot \left[\frac{1}{\sigma^l} \exp\left(-\frac{\bar{A}}{\sigma^2}\right)\right]^{|R|}$$

mit

$$\bar{A} = \frac{1}{|R|} \sum_r A\big(r,f(r)\big).$$

Die Besonderheit unserer Aufgabe besteht darin, dass die Zahl $|R|$ riesig groß ist. Aus der obigen Formel sieht man, dass bei einer sehr großen Zahl $|R|$ diese Verteilung einen sehr ausgeprägten Pik hat. Außerhalb dieses Piks sind die Wahrscheinlichkeitswerte der benötigten Verteilung praktisch Null.

Man kann hoffen, dass der oben beschriebene Ansatz (nennen wir ihn „Sum Likelihood“) für andere Anwendungen bessere Resultate liefert, als das Maximum Likelihood Prinzip. Für die Stereorekonstruktion ist die Benutzung dieses Verfahrens aus den oben genannten Gründen leider sinnlos. Deswegen verbleiben wir bei dem Maximum Likelihood Prinzip. Für das Lernen des Parameters σ benutzen wir den EM-Algorithmus. Die Details betrachten wir im nachfolgenden Kapitel.

KAPITEL 4

Implementierung und Experimente

In diesem Kapitel beschreiben wir zunächst die Implementierung der oben beschriebenen Ansätze im Kontext der Aufgabe der Stereorekonstruktion. Danach werden die experimentellen Ergebnisse präsentiert.

4.1. Modell

Wie bereits gesagt, wird als Modell der Oberfläche eine Gibbssche Wahrscheinlichkeitsverteilung benutzt. Der Basisgraph der Aufgabe $V = (R,E)$ ist ein Gitter. Jeder Knoten r dieses Graphen repräsentiert ein Koordinatenpaar (X,Y) im dreidimensionalen Raum. Die Zustände $k \in K$ im Knoten r repräsentieren die Z-Koordinaten am Ort r. Somit entspricht jedem Weltpunkt (X,Y,Z) ein Paar (r,k) – ein Knoten des Basisgraphen und ein Zustand in diesem Knoten. Die benachbarten (im Sinn der 4-Nachbarschaft) Knoten sind mit Kanten verbunden. Somit ergibt sich die a-priori Wahrscheinlichkeit eines Labelings $f : R \to K$ durch

$$P(f) = \frac{1}{Z} \prod_{(r,r')\in E} g_{rr'}(f(r), f(r')). \tag{4.1}$$

Im Weiteren betrachten wir nur den regulären Fall, nämlich wenn die Funktion $g_{rr'} : K \times K \to \mathbb{R}$ ortsunabhängig ist. Bei der Spezifizierung dieser Funktion ist es auch zu beachten, dass die Zustandsmenge in unserem Fall geordnet ist: $K = \{1,\ldots,|K|\}$. Für die Experimente benutzen wir die folgenden (unserer Meinung nach für die Stereorekonstruktion geeigneten) Funktionen:

– scharfe Glattheitsforderung:

$$g(k,k') = \begin{cases} 1 & \text{wenn } |k-k'| \le \delta \\ 0 & \text{sonst,} \end{cases} \tag{4.2}$$

– Gaussche Glattheitsforderung:

$$g(k,k') = \exp\left[-\frac{(k-k')^2}{\sigma_g}\right], \tag{4.3}$$

– Potts Modell:

$$g(k,k') = \begin{cases} a & \text{wenn } k = k' \\ 1 & \text{sonst.} \end{cases} \tag{4.4}$$

Die bedingte Wahrscheinlichkeit der Beobachtung (X_l, X_r) (das linke bzw. das rechte Bild) unter Bedingung eines gegebenen Labelings f ist

$$P(X_l, X_r | f) = \prod_{r\in R} q_r\big(f(r)\big). \tag{4.5}$$

Die Werte der Funktionen q_r werden auf folgende Art erzeugt (siehe auch Abschnitt 2.4). Zuerst wird für jedes Paar (r,k) das Ähnlichkeitsmaß $A(r,k) = A\big(T_l(X,Y,Z), T_r(X,Y,Z)\big)$

berechnet. Dabei benutzen wir die im Abschnitt 2.1 betrachteten Ähnlichkeitsmaße, nämlich:

- die quadratische Differenz von Grauwerten (2.1),
- die quadratische Differenz von Grauwerten mit der „besten Grauwertverschiebung“ (2.2),
- den Korrelationskoeffizient (2.3).

Die Werte der Qualitätsfunktionen q_r ergeben sich dann durch

$$q_r(k) \sim \frac{1}{\sigma_q^l} \cdot \exp\left[-\frac{A(r,k)}{\sigma_q^2}\right], \tag{4.6}$$

wobei σ_q der Parameter ist.

Man suche nach der Bayesschen Entscheidung, die wie folgt definiert wird. Die Menge der Entscheidungen am Ort r ist entweder ein reeller Wert im Bereich von $[1,|K|]$ (das entspricht einem Wert der Z-Koordinate) oder die spezielle Antwort „Rückweisung“ (im Weiteren mit *RW* bezeichnet). Die Kostenfunktion für Fehlklassifikation ist die additive quadratische mit dem Zustand „Rückweisung“ erweiterte Kostenfunktion $C(d,r) = \sum_r c\big(d(r),f(r)\big)$ mit

$$c\big(d(r),f(r)\big) = \begin{cases} \big(d(r)-f(r)\big)^2 & \text{wenn } d(r) \in [1,|K|] \\ \varepsilon & \text{wenn } d(r) = RW. \end{cases} \tag{4.7}$$

4.2. Algorithmus

Zunächst betrachten wir einen einfachen Algorithmus, in dem der Parameter σ_q in (4.6) fixiert (vom Benutzer angegeben) ist. In diesem Fall ist der Vorfaktor $1/\sigma^l$ (und somit die Konstante l) für die Ermittelung der Bayesschen Entscheidung irrelevant. Die Eingangsdaten für diesen Algorithmus sind zwei Bilder X_l und X_r (das Stereopaar). Zusätzlich müssen die folgende Parameter angegeben werden:

- Der Typ der Berechnung des Ähnlichkeitsmaßes (*a_mode*). Es sind drei Varianten: die quadratische Differenz von Grauwerten, dasselbe mit der Grauwertverschiebung, der Korrelationskoeffizient.
- Die Fenstergröße (d). Dieser Parameter wird für Berechnung des Ähnlichkeitsmaßes benötigt (siehe (2.1), (2.2), (2.3)).
- σ_q. Dieser Parameter wird für die Berechnung von Werten der Funktionen q_r anhand der Ähnlichkeitsmaße $A(r,k)$ benutzt (4.6).
- Der Typ der Modellfunktion g (*g_mode*). Es gibt hier drei Varianten: die scharfe Glattheitsforderung (4.2), die Gaussche Glattheitsforderung (4.3), das Potts Modell (4.4).
- Der Parameter für die Modellfunktion. Entsprechend dem ausgewählten Typ der Modellfunktion soll entweder δ oder σ_g oder a angegeben werden.
- Die Anzahl der Generierungsschritte (*num*). Diesen Parameter diskutieren wir später.
- ε. Die Strafe für die Antwort „Rückweisung“ (siehe (4.7))

Der Algorithmus besteht aus den folgenden Schritten:

(1) Anhand der Bilder X_l und X_r wird für jedes Paar (r,k) das Ähnlichkeitsmaß $A(r,k)$ entsprechend den angegebenen Parametern *a_mode* und d berechnet.
(2) Anhand der berechneten Ähnlichkeitsmaße werden die Werte $q_r(k)$ entsprechend dem Parameter σ_q berechnet.

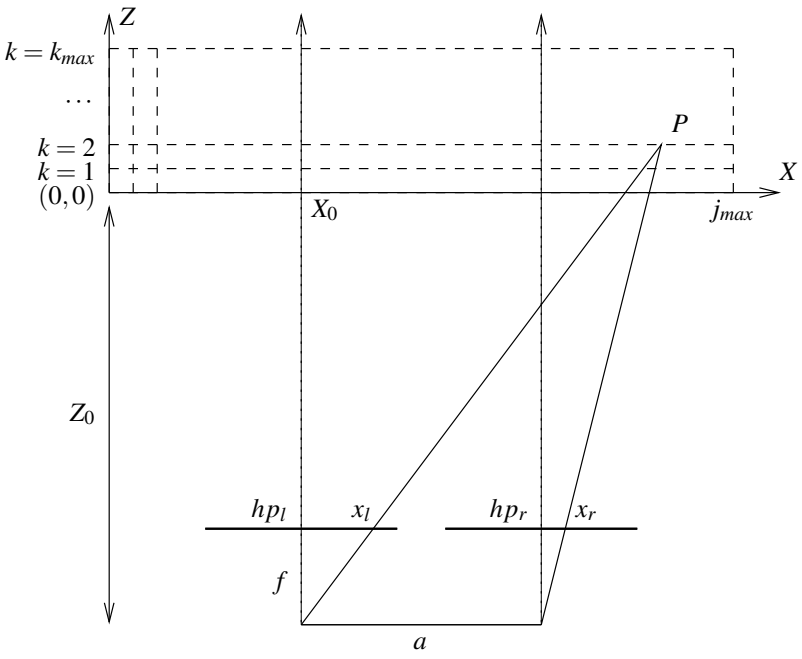


ABBILDUNG 4.1. Das Schema für die Berechnung des Ähnlichkeitsmaßes

(3) Anhand der berechneten Werte $q_r(k)$ wird das Histogramm $H_r(k)$ (die Häufigkeit des Auftretens des Zustandes k im Knoten r) mit Hilfe des Generierungsprozesses erzeugt. Dabei sind die Parameter *g_mode*, δ oder σ_g oder a, *num* zu beachten.
(4) Anhand des erzeugten Histogramms und des angegebenen Parameters ε ist die Entscheidung für jeden Knoten r zu treffen.

Der Schritt (2) ist offensichtlich. Betrachten wir die anderen Schritte dieses Algorithmus ausführlicher.

4.2.1. Berechnung des Ähnlichkeitsmaßes. Wie bereits gesagt, betrachten wir in dieser Arbeit den rektifizierten Fall. Das heißt, dass korrespondierende Pixel gleiche y-Koordinaten in den Bildern haben. Deswegen reicht es, die Berechnung des Ähnlichkeitsmaßes nur für eine Zeile des Bildes zu betrachten. In der Abbildung 4.1 ist das Berechnungsschema skizziert. In dem Schema werden die folgende Hilfsgrößen verwendet:

a	Der Basisabstand – die Länge der Strecke, die die Projektionszentren der Kameras verbindet. Diese Strecke nennen wir Basislinie.
f	Die Brennweite.
hp_l, hp_r	Die Hauptpunkte – die Punkte der Bilder, in denen die jeweilige optische Achse den Bildschirm kreuzt.
X, Z	Die Weltkoordinaten.
P	Der betrachtete Weltpunkt. Wir bezeichnen seine Koordinaten mit X_P und Z_P.
$(0,0)$	Der Koordinatenursprung im Weltkoordinatensystem.
Z_0	Der Abstand von der Basislinie zu dem Koordinatenursprung entlang der Z-Achse.
X_0	Die Koordinate X der Weltpunkte, die sich auf hp_l abbilden.
j_{max}, k_{max}	Die Anzahl der Diskretisierungsschritte entlang der X- bzw. Z-Achse.

Aus der einfachen geometrischen Betrachtung ergibt sich das folgende Gleichungssystem:

$$\begin{cases} \dfrac{x_l - hp_l}{f} = \dfrac{X_P - X_0}{Z_P + Z_0} \\ \dfrac{x_r - hp_r}{f} = \dfrac{X_P - X_0 - a}{Z_P + Z_0}. \end{cases} \tag{4.8}$$

Sei der betrachtete dreidimensionale Raum diskret. Das heißt, dass die Weltkoordinaten X und Z die Form $X = j \cdot \triangle X$ bzw. $Z = k \cdot \triangle Z$ haben, wobei j und k ganze Zahlen sind (die Indizes in einer Tabelle) und $\triangle X$ und $\triangle Z$ die Diskretisierungsschritte sind. Dann lässt sich das obige Gleichungssystem wie folgt schreiben:

$$\begin{cases} x_l = \dfrac{j \cdot \triangle X - X_0}{k \cdot \triangle Z + Z_0} \cdot f + hp_l \\ x_r = \dfrac{j \cdot \triangle X - X_0 - a}{k \cdot \triangle Z + Z_0} \cdot f + hp_r. \end{cases} \tag{4.9}$$

Wenn alle benötigten Koeffizienten (a, f, Hauptpunkte usw.) bekannt sind, kann man diese Formeln direkt benutzen, um die Bildkoordinaten x_l und x_r anhand der Weltposition (X,Z) zu ermitteln. Meistens ist es leider nicht der Fall. In der Regel ist für ein gegebenes Stereopaar nur bekannt, dass die Disparitäten ($d = x_l - x_r$) in einem bestimmten Bereich $[d_{min}, d_{max}]$ liegen. Diese Information reicht nicht um alle Parameter wiederherzustellen. Im Weiteren nehmen wir zusätzlich an, dass auch die Hauptpunkte hp_l und hp_r bekannt sind. Dann kann man das obige Gleichungssystem auf folgende Art ausdrücken:

$$\begin{cases} x_l = \dfrac{j \cdot C_1 - C_2}{k + C_4} + hp_l \\ x_r = \dfrac{j \cdot C_1 - C_2 - C_3}{k + C_4} + hp_r. \end{cases} \tag{4.10}$$

Um die Koeffizienten C_1 bis C_4 zu bestimmen, benutzen wir folgende Nebenbedingungen. Zunächst kann man bemerken, dass die Formel für Berechnung der Disparität nur die Koeffizienten C_3 und C_4 umfasst:

$$d = x_l - x_r = \frac{C_3}{k + C_4} + \triangle hp \tag{4.11}$$

(die Differenz $hp_l - hp_r$ ist hier mit $\triangle hp$ bezeichnet). Die maximale Disparität wird bei dem minimalen Wert der Koordinate Z erreicht, d.h. bei $k = 0$. Ähnlich wird die minimale

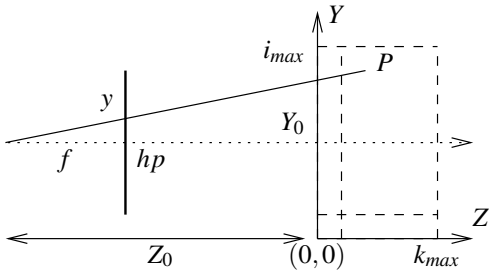


ABBILDUNG 4.2. Das Schema für die Berechnung der y-Bildkoordinate

Disparität bei $k = k_{max}$ erreicht. Aus diesen zwei Bedingungen ergeben sich die Koeffizienten C_3 und C_4:

$$C_3 = \frac{k_{max} \cdot (d_{min} - \triangle hp) \cdot (d_{max} - \triangle hp)}{d_{max} - d_{min}}$$

$$C_4 = \frac{k_{max} \cdot (d_{min} - \triangle hp)}{d_{max} - d_{min}}. \tag{4.12}$$

Für die Bestimmung der beiden anderen Koeffizienten benutzen wir folgende Nebenbedingungen. Wir verlangen, dass alle betrachteten Weltpunkte (j,k) von beiden Kameras beobachtet werden können. Das heißt, dass alle Projektionspunkte p_l (sowie p_r) ihre x-Koordinaten im Bereich von $[0, x_{max}]$ haben, wobei x_{max} die maximale x-Koordinate im Bild ist. Das entspricht den folgenden Bedingungen:

$$x_l(j = 0, k = 0) = \frac{-C_2}{C_4} + hp_l \geq 0 \tag{4.13}$$

$$x_r(j = 0, k = 0) = \frac{-C_2 - C_3}{C_4} + hp_r \geq 0 \tag{4.14}$$

$$x_l(j = j_{max}, k = 0) = \frac{j_{max} \cdot C_1 - C_2}{C_4} + hp_l \leq x_{max} \tag{4.15}$$

$$x_r(j = j_{max}, k = 0) = \frac{j_{max} \cdot C_1 - C_2 - C_3}{C_4} + hp_r \leq x_{max}. \tag{4.16}$$

Außerdem möchten wir den Bereich der Welt X-Koordinaten so breit wie möglich wählen. Aus der Betrachtung der Bedingungen (4.13) und (4.14) ergibt sich der Wert von C_2:

$$C_2 = \min\left[C_4 \cdot hp_l,\ C_4 \cdot hp_r - C_3\right]. \tag{4.17}$$

Aus der Betrachtung der Bedingungen (4.15) und (4.16) ergibt sich der Wert von C_1:

$$C_1 = \min\left[\frac{(x_{max} - hp_l) \cdot C_4 + C_2}{j_{max}},\ \frac{(x_{max} - hp_r) \cdot C_4 + C_2 + C_3}{j_{max}}\right]. \tag{4.18}$$

Somit sind die Abbildungen $x_l = T_l(j,k)$ und $x_r = T_r(j,k)$ beschrieben. Anhand der bekannten Parameter d_{min}, d_{max}, hp_l, hp_r, j_{max}, k_{max} und x_{max} werden mit Hilfe von (4.12), (4.17) und (4.18) die Koeffizienten C_1 bis C_4 ermittelt. Dann benutzt man (4.10) um die Bildkoordinaten x_l bzw. x_r zu bestimmen.

Ähnlich lässt sich die Formel für die Berechnung der y-Bildkoordinate ableiten (siehe Abbildung 4.2):

$$\frac{y - hp}{f} = \frac{Y - Y_0}{Z + Z_0}. \tag{4.19}$$

Input: d_{min}, d_{max}, hp_l, hp_r, j_{max}, k_{max},
die Bilder $I_l[height][width]$ und $I_r[height][width]$
Output: Der Massiv der Qualitäten $A[height][j_{max}+1][k_{max}+1]$

(1) Berechne die Koeffizienten C_1 bis C_4 mit Hilfe der (4.12), (4.17) und (4.18)
(2) **Für** i **von** 0 **bis** y_{max}
Für j **von** 0 **bis** j_{max}
Für k **von** 0 **bis** k_{max}
(3) Bestimme x_l und x_r mit Hilfe der (4.10)
$A[i][j][k] = 0$
(4) **Für** i' **von** $-d$ **bis** d
Für j' **von** $-d$ **bis** d
Bestimme x'_l und x'_r für $x_l + i'$ bzw. $x_r + i'$
$\gamma_l = x_l - x'_l$, $\gamma_r = x_r - x'_r$
$I_l = I_l[i+i'][x'_l] \cdot (1-\gamma_l) + I_l[i+i'][x'_l+1] \cdot \gamma_l$,
$I_r = I_r[i+i'][x'_r] \cdot (1-\gamma_r) + I_r[i+i'][x'_r+1] \cdot \gamma_r$
$A[i][j][k] = A[i][j][k] + (I_l - I_r)^2$

ABBILDUNG 4.3. Einfacher Algorithmus für die Berechnung des Ähnlichkeitsmaßes

Unter der Annahme, dass die zu rekonstruierende Szene von den Kameras relativ weit entfernt ist ($Z_0 \gg Z_{max}$), ist die Bildkoordinate y zu der Weltkoordinate i näherungsweise proportional. Somit kann die Zeilennummer des Bildes als Index i benutzt werden.

Für die Berechnung des Ähnlichkeitsmaßes werden die Grauwerte (Farbwerte) in den korrespondierenden Punkten benötigt. Aber die anhand der Formel (4.10) berechneten Koordinaten x_l und x_r sind reell. Für die Bestimmung der Grauwerte in den nicht ganzzahligen Positionen benutzen wir die lineare Interpolation:

$$I(y,x) = I(y,x') \cdot (1-\gamma) + I(y,x'+1) \cdot \gamma. \tag{4.20}$$

Mit x' ist hier die nächste ganzzahlige Koordinate bezeichnet, die kleiner x ist. Der Koeffizient γ ergibt sich als $\gamma = x - x'$.

Schließlich wollen wir bei der Berechnung des Ähnlichkeitsmaßes nicht nur die Grauwerte in den entsprechenden Pixeln, sondern auch in deren kleinen Umgebungen betrachten – in den Fenster $F = \{(i,j) : d \le i \le d,\ d \le j \le d\}$.

In der Abbildung 4.3 ist ein einfacher Algorithmus dargestellt, der alles oben gesagte zusammenfasst. Die Höhe und die Breite des Bildes sind hier mit *height* bzw. *width* bezeichnet. Somit sind die maximalen Werte der x- und y-Koordinate $y_{max} = height - 1$ bzw. $x_{max} = width - 1$. Dabei haben wir nur den Fall betrachtet, in dem die quadratische Differenz der Grauwerte als Qualität benutzt wird.

Die Zeitkomplexität dieses Algorithmus ist $O(height \cdot j_{max} \cdot k_{max} \cdot d^2)$. Mit Hilfe einer einfachen Methode kann die Zeitkomplexität deutlich verbessert werden. Betrachten wir

die während der Schleife (4) berechnete Summe ausführlicher:

$$\sum_{(i,j)\in F}\big(I_l(y+i,x_l+j)-I_r(y+i,x_r+j)\big)^2 =$$
$$\sum_{(i,j)\in F}\big[I_l(y+i,x'_l+j)\cdot(1-\gamma_l)+I_l(y+i,x'_l+j+1)\cdot\gamma_l-$$
$$I_r(y+i,x'_r+j)\cdot(1-\gamma_r)+I_r(y+i,x'_r+j+1)\cdot\gamma_r\big]^2.$$

Führen wir die neuen Bezeichnungen ein:

$$I_l(i,j) = I_l(y+i,x'_l+j)$$
$$I_l^-(i,j) = I_l(y+i,x'_l+j+1)-I_l(y+i,x'_l+j)$$
$$I_r(i,j) = I_r(y+i,x'_r+j)$$
$$I_r^-(i,j) = I_r(y+i,x'_r+j+1)-I_r(y+i,x'_r+j).$$

Dann kann die obige Formel äquivalent wie folgt geschrieben werden:

$$\sum_{(i,j)\in F}\big[I_l(i,j)+I_l^-(i,j)\cdot\gamma_l-I_r(i,j)-I_r^-(i,j)\cdot\gamma_r\big]^2 =$$
$$\sum_{(i,j)\in F}\big(I_l(i,j)-I_r(i,j)\big)^2+$$
$$2\gamma_l\cdot\sum_{(i,j)\in F}I_l^-(i,j)\cdot\big(I_l(i,j)-I_r(i,j)\big)+$$
$$2\gamma_r\cdot\sum_{(i,j)\in F}I_r^-(i,j)\cdot\big(I_r(i,j)-I_l(i,j)\big)-$$
$$2\gamma_l\cdot\gamma_r\cdot\sum_{(i,j)\in F}I_l^-(i,j)\cdot I_r^-(i,j)+$$
$$\gamma_l^2\cdot\sum_{(i,j)\in F}I_l^-(i,j)^2+\gamma_r^2\cdot\sum_{(i,j)\in F}I_r^-(i,j)^2. \tag{4.21}$$

Nehmen wir an, dass die Werte aller benötigten Summen (zum Beispiel der Wert der Summe $\sum_{(i,j)\in F}[I_l(i,j)-I_r(i,j)]^2$) im Voraus bereits berechnet und gespeichert sind. Dann ist es nicht mehr notwendig, die Schleife (4) des Algorithmus durchzuführen. Man kann die Qualität unmittelbar mit Hilfe der obigen Formel (4.21) berechnen. In diesem Fall wäre die Zeitkomplexität nicht von d abhängig.

Die benötigten Summen lassen sich im Voraus mit der Zeitkomplexität $O\big(height\cdot width\cdot(d_{max}-d_{min})\big)$ berechnen. Die Werte dieser Summen (für alle Positionen (i,j)) ergeben sich als die Faltungen der entsprechenden Funktionen der zwei Variablen. Sei $f:\mathbb{N}\times\mathbb{N}\to\mathbb{R}$ eine solche Funktion (zum Beispiel $f(i,j)=\big(I_l(i,j)-I_r(i,j)\big)^2$). Die Aufgabe besteht darin, die Summe der Werte dieser Funktion innerhalb eines Fensters F für jede Position (i,j) zu berechnen:

$$S(i,j)=\sum_{(i',j')\in F}f(i+i',j+j')\quad\forall i,j. \tag{4.22}$$

Führen wir die Hilfsfunktion $S':\mathbb{N}\times\mathbb{N}\to\mathbb{R}$ auf folgende Art ein:

$$S'(i,j)=\sum_{i'=0}^{i'\le i}\sum_{j'=0}^{j'\le j}f(i',j'). \tag{4.23}$$

Es ist leicht zu sehen, dass sich die Werte der Funktion S durch die Werte der Funktion S' wie folgt ausdrücken lassen:

$$\begin{aligned} S(i,j) = S'(i+d, j+d) - S'(i+d, j-d-1) - \\ S'(i-d-1, j+d) + S'(i-d-1, j-d-1). \end{aligned} \tag{4.24}$$

Andererseits können die Werte der Funktion S' effizient mit Hilfe der folgenden rekursiven Formel berechnet werden:

$$S'(i,j) = f(i,j) - S'(i-1,j) - S'(i,j-1) + S'(i-1,j-1). \tag{4.25}$$

Somit besteht der Algorithmus für die Berechnung der Faltung S aus den folgenden drei Schritten:

(1) Berechne die Werte von $f(i,j)$ für alle (i,j).
(2) Berechne die Werte von $S'(i,j)$ für alle (i,j) mit Hilfe der (4.25).
(3) Berechne die Werte von $S(i,j)$ für alle (i,j) mit Hilfe der (4.24).

Jeder Schritt hat die Zeitkomplexität $O(height \cdot width)$. Somit hat das ganze Verfahren dieselbe Zeitkomplexität.

Man muss bemerken, dass bei dem in der Abbildung 4.3 dargestellten Algorithmus die Werte der benötigten Summen nicht nur von Koordinaten x' und i abhängen, sondern auch von der „Verschiebung“ $\triangle x = x'_l - x'_r$. Diese Verschiebung ist aber ganzzahlig und begrenzt (die Anzahl solcher Verschiebungen ist die Anzahl der im Bildpaar vorkommenden Disparitäten). Deswegen kann man die benötigten Summen für alle Werte dieser Verschiebung im Voraus berechnen.

Zusammenfassend, kann man den in der Abbildung 4.4 angegebenen Algorithmus zur Berechnung des Ähnlichkeitsmaßes konstruieren. Während des ersten Teils des Algorithmus (die Schleife (1)) werden die benötigten Summen S_1 bis S_6 berechnet. Dieses Teil hat die Zeitkomplexität $O\big((d_{max} - d_{min}) \cdot height \cdot width\big)$. Während des zweiten Teils (die Schleife (3)) werden die Qualitäten $A(r,k)$ mit Hilfe der (4.21) (der Schritt (5)) ermittelt. Die Koeffizienten C_1 bis C_4 (der Schritt (2)) sind mit Hilfe der (4.12), (4.17) und (4.18) zu berechnen. Für die Bestimmung der Koordinaten x_l und x_r (Schritt (4) des Algorithmus) wird (4.10) benutzt. Die Zeitkomplexität des zweiten Teiles ist $O(k_{max} \cdot height \cdot j_{max})$.

Alle obigen Betrachtungen wurden für den quadratischen Abstand als Qualitätsmaß durchgeführt. Für die anderen Ähnlichkeitsmaße lassen sich entsprechende effiziente Algorithmen in ähnlicher Weise konstruieren.

4.2.2. Generierung und Treffen der Entscheidung. Betrachten wir zunächst das allgemeine Schema des Generierungsprozesses und der nachfolgenden Ermittelung der Entscheidung. Gegeben seien:

$V = (R,E)$	der Basisgraph mit der Menge der Knoten R und der Menge der Kanten E;
$g : K \times K \to \mathbb{R}$	die Modellfunktion;
$q_r : K \to \mathbb{R}$	die lokalen Bewertungsfunktionen;
ε	die Strafe für die Entscheidung „Rückweisung“.

Gesucht ist die Entscheidung $d : R \to [0 \ldots k_{max}] \cup RW$, die den Erwartungswert des Risikos minimiert. Dabei wird als Kostenfunktion die additive quadratische mit dem Zustand „Rückweisung“ erweiterte Kostenfunktion (4.7) benutzt.

Das allgemeine Schema des Generierungsprozesses ist in der Abbildung 4.5 angegeben. Die Schritte (3)–(6) bezeichnen wir als Generierungsschritt. Die Reihenfolge, in der die Knoten für den Generierungsschritt ausgewählt werden (der Schritt (3)), ist irrelevant.

Input: d_{min}, d_{max}, hp_l, hp_r, j_{max}, k_{max},
$I_l[height][width]$, $I_r[height][width]$
Output: $A[height][j_{max}+1][k_{max}+1]$
Local: $f_1, \ldots, f_6$, $S'_1[height][width], \ldots, S'_6[height][width]$,
$S_1[d_{min}-\triangle hp, \ldots, d_{max}-\triangle hp][height][width], \ldots,$
$\ldots, S_6[d_{min}-\triangle hp, \ldots, d_{max}-\triangle hp][height][width]$

(1) **Für** $\triangle x$ **von** $d_{min}-\triangle hp$ **bis** $d_{max}-\triangle hp$

Für i **von** 0 **bis** y_{max}
Für j **von** 0 **bis** x_{max}
$f_1 = (I_l[i][j]-I_r[i][j+\triangle x])^2$
$f_2 = (I_l[i][j+1]-I_l[i][j])\cdot(I_l[i][j]-I_r[i][j+\triangle x])$
$f_3 = (I_r[i][j+\triangle x+1]-I_r[i][j+\triangle x])\cdot(I_r[i][j+\triangle x]-I_l[i][j])$
$f_4 = (I_l[i][j+1]-I_l[i][j])\cdot(I_r[i][j+\triangle x+1]-I_r[i][j+\triangle x])$
$f_5 = (I_l[i][j+1]-I_l[i][j])^2$
$f_6 = (I_r[i][j+\triangle x+1]-I_r[i][j+\triangle x])^2$
Für l **von** 1 **bis** 6
$S'_l[i][j] = f_l - S'_l[i-1][j]-$
$S'_l[i][j-1]+S'_l[i-1][j-1]$
Für i **von** 0 **bis** y_{max}
Für j **von** 0 **bis** x_{max}
Für l **von** 1 **bis** 6
$S_l[\triangle x][i][j] = S'_l[i+d][j+d]-S'_l[i-d-1][j+d]-$
$S'_l[i+d][j-d-1]+S'_l[i-d-1][j-d-1]$

(2) Berechne die Koeffizienten C_1 bis C_4

(3) **Für** i **von** 0 **bis** y_{max}
Für j **von** 0 **bis** j_{max}
Für k **von** 0 **bis** k_{max}
(4) Bestimme x_l und x_r
Bestimme x'_l, x'_r, γ_l, γ_r, $\triangle x = x'_l - x'_r$
(5) $A[i][j][k] =$
$S_1[\triangle x][i][x'_l]+$
$2\cdot\gamma_l\cdot S_2[\triangle x][i][x'_l]+$
$2\cdot\gamma_r\cdot S_3[\triangle x][i][x'_l]-$
$2\cdot\gamma_l\cdot\gamma_r\cdot S_4[\triangle x][i][x'_l]+$
$\gamma_l^2\cdot S_5[\triangle x][i][x'_l]+$
$\gamma_r^2\cdot S_6[\triangle x][i][x'_l]$

ABBILDUNG 4.4. Effizienter Algorithmus für die Berechnung des Ähnlichkeitsmaßes

Input: $V(R,E)$, q, g, ε
Output: d
Local: Das „aktuelle" Labeling $f^* : R \to K$,
die Histogramme $H_r : K \to \mathbb{N}$ (am Anfang sind alle Elemente Null)

(1) Wähle das initiale Labeling f^*
(2) **Wiederhole** sehr oft:
(3) Wähle einen Knoten r
(4) Ermittele die Wahrscheinlichkeitsverteilung

$$P\big(f(r) = k | f(r') = f^*(r')\big), \quad r' \neq r$$

(5) Würfele einen neuen Zustand k^* im Knoten r entsprechend der obigen Verteilung
(6) Aktualisiere das Histogramm: $H_r(k^*) = H_r(k^*)+1$
Aktualisiere das Labeling: $f^*(r) = k^*$

(7) **Für jeden Knoten** r:
(8) Berechne den Mittelwert

$$\bar{k} = \frac{\sum_k k \cdot H_r(k)}{\sum_k H_r(k)}$$

(9) Berechne die Streuung

$$\sigma = \frac{\sum_k (k-\bar{k})^2 \cdot H_r(k)}{\sum_k H_r(k)}$$

(10)

$$d(r) = \begin{cases} \bar{k} & \text{wenn } \sigma < \varepsilon \\ RW & \text{sonst} \end{cases}$$

ABBILDUNG 4.5. Allgemeines Schema für die Ermittelung der Entscheidung

Wichtig ist nur, dass die Knoten gleichwahrscheinlich gewählt werden. Die einfachste Variante ist, alle Knoten einfach nacheinander zu wählen. Einen solchen Durchlauf über alle Knoten nennen wir eine Iteration.

Bezeichnen wir mit N die Anzahl der Iterationen. Man kann leicht sehen, dass bei der gewählten Reihenfolge der Generierungen $\sum_k H_r(k) = N$ für jeden Knoten r gilt. Somit braucht man am Ende des Generierungsprozesses für jeden Knoten r nur die folgenden Werte: $\sum_k k \cdot H_r(k)$ und $\sum_k (k-\bar{k})^2 \cdot H_r(k)$. Betrachten wir zunächst die erste Größe. Bezeichnen wir mit $\tilde{k}^{(t)}(r)$ ihren Wert im Knoten r nach t Iterationen. Um den Mittelwert $\bar{k}(r)$ zu berechnen, wird der Wert $\tilde{k}^{(N)}(r)$ nach dem Generierungsprozess benötigt. Nehmen wir an, dass in der $(t+1)$-ten Iteration in diesem Knoten der Zustand k^* generiert wird. Es ist leicht zu sehen, dass $\tilde{k}^{(t+1)}(r) = \tilde{k}^{(t)}(r) + k^*$ gilt. Dass heißt, dass das Histogramm nicht benötigt wird, um den Mittelwert am Ende zu berechnen. Man kann unmittelbar die Größen $\tilde{k}^{(t)}(r)$ während des Prozesses aktualisieren. Ähnlich lässt sich die Berechnung von $\sigma(r)$ modifizieren. Vor allem schreiben wir die im Schritt (9) berechnete Größe in etwas anderer Form:

$$\sigma(r) = \frac{\sum_k (k-\bar{k})^2 \cdot H_r(k)}{\sum_k H_r(k)} = \frac{\sum_k k^2 \cdot H_r(k) - [\tilde{k}^{(N)}(r)]^2/N}{N}.$$

Bezeichnen wir mit $\tilde{\sigma}^{(t)}(r)$ die Summe $\sum_k k^2 \cdot H_r(k)$ nach den t Iterationen. Um die Streuungen $\sigma(r)$ zu Berechnen werden am Ende die Werte $\tilde{\sigma}^{(N)}(r)$ für jeden Knoten benötigt. Man kann leicht sehen, dass auch diese Größen während des Prozesses aktualisiert werden können: $\tilde{\sigma}^{(t+1)}(r) = \tilde{\sigma}^{(t)}(r) + k^{*2}$.

Somit wird das Histogramm für die Berechnung von $\bar{k}(r)$ und $\sigma(r)$ nicht benötigt. Man kann stattdessen unmittelbar die Größen $\tilde{k}^t(r)$ und $\tilde{\sigma}^t(r)$ während des Prozesses aktualisieren. Dieser Ansatz hat den Vorteil, dass die Speicherkomplexität des Algorithmus in diesem Fall $O(height \cdot j_{max})$ statt $O(height \cdot j_{max} \cdot k_{max})$ ist.

An der Stelle wollen wir daran erinnern, dass es eine Möglichkeit gibt, den Parameter σ_q der bedingten Wahrscheinlichkeitsverteilung während des Prozesses zu lernen. Wir realisieren diese Möglichkeit auf folgende Art. Wir aktualisieren den Wert von σ_q in jeder N'-ten Iteration ($N' \ll N$). Während dieser N' Iterationen wird die Statistik zur Schätzung der Wahrscheinlichkeitswerte $P\big(f(r) = k \mid X; \sigma^{(n)}\big)$ gesammelt. Nach den N' Iterationen wird der Wert von σ_q mit Hilfe der (3.42) aktualisiert (siehe Abschnitt 3.5).

Man kann leicht sehen, dass sich die für die Aktualisierung von σ_q benötigten Größen genauso wie $\tilde{k}^t(r)$ und $\tilde{\sigma}^t(r)$ während des Prozesses aktualisieren lassen. Betrachten wir die Formel für die Aktualisierung (3.42) ausführlicher:

$$\sigma^{(n+1)2} = \frac{2}{|R| \cdot l} \sum_r \sum_k A(r,k) \cdot P^{(n)}\big(f(r) = k | X; \sigma^{(n)}\big) \approx \frac{2 \cdot \sum_r \sum_k A(r,k) \cdot H_r(k)}{|R| \cdot l \cdot N'}.$$

Bezeichnen wir mit $A^{(t)}$ den Wert der doppelten Summe $\sum_r \sum_k A(r,k) \cdot H_r(k)$ nach t Generierungsschritten. Sei im t+1-ten Generierungsschritt im Knoten r der Zustand k^* generiert. Der Wert von $A^{(t+1)}$ ergibt sich dann als $A^{(t+1)} = A^{(t)} + A(r,k^*)$. Somit wird das Histogramm auch für das Lernen von σ_q nicht benötigt.

Betrachten wir nochmals die Regeln für die Ermittelung der Entscheidung am Ende des Generierungsprozesses. Zu diesem Zeitpunkt sind die Streuungen $\sigma(r)$ für alle Knoten bekannt. Den Parameter ε kann man sich als einen Schwellwert vorstellen. Für einen gegebenen Schwellwert ist es sehr einfach zu zählen, wieviel Knoten mit der Entscheidung „Rückweisung“ bewertet werden. Die umgekehrte Bestimmung ist auch möglich. Anhand der gegebenen Anzahl der Pixel, für die die Entscheidung „Rückweisung“ getroffen werden soll, kann man den Schwellwert bestimmen, bei dem dieses passiert. Sei n_R die Anzahl solcher Pixel. Man *ordne* die Pixel nach fallenden Werten von $\sigma(r)$. Der gesuchte Schwellwert ist dann die Streuung des n_R-ten Pixels in dieser Ordnung. Somit kann man statt des Parameters ε den Parameter n_R angeben. Mann kann auch die zwei Forderungen für die „Sicherheit“ der Entscheidung zum Beispiel wie folgt mischen. Man sucht nach der Entscheidung mit der Strafe für die Antwort „Rückweisung“ gleich ε. Zusätzlich wird verlangt, dass die Anzahl der unbestimmten Pixel eine vorgegebene Zahl n_R nicht überschrittet.

Alle obigen Änderungen sind im Algorithmus in der Abbildung 4.6 zusammengefasst.

4.3. Verwendung der Maximum a-posteriori Entscheidung

Weitere Änderungen des Algorithmus basieren auf verschiedenen Heuristiken. Die in diesem Abschnitt beschriebenen Ansätze sind (streng genommen) theoretisch nicht genügend gut begründet. Die Experimente zeigen allerdings, dass man Heuristiken benutzen kann, ohne die Ergebnisse zu verschlechtern. Andererseits führt dieses zu einer deutlichen Verbesserung des gesamten Zeitaufwands.

Input: $V(R,E)$, $A(r,k)$, l, g, N, N', n_R, ε
Output: d
Local: Das „aktuelle" Labeling $f^* : R \to K$,
das Massiv von $\tilde{k}(r)$ und $\tilde{\sigma}(r)$ (am Anfang sind alle Elemente Null), $A = 0$, σ_q,
Das Massiv von $\bar{k}(r)$, $\sigma(r)$

Wähle das initiale Labeling f^*
Wähle das initiale σ_q^2, berechne $q_r(k) \;\; \forall\, r,k$ mit

$$q_r(k) = \exp\left(-A(r,k)/\sigma_q^2\right)$$

Für t **von** 0 **bis** N:
Für jeden Knoten r:
Würfele einen neuen Zustand k^* im Knoten r entsprechend der Wahrscheinlichkeitsverteilung

$$P\left(f(r) = k | f(r') = f^*(r')\right), \;\; r' \neq r$$

Aktualisiere die Größen:

$$\tilde{k}(r) = \tilde{k}(r) + k^*$$
$$\tilde{\sigma}(r) = \tilde{\sigma}(r) + k^{*2}$$
$$A = A + A(r,k)$$

Aktualisiere das Labeling: $f^*(r) = k^*$
Wenn t **mod** $N' = N' - 1$:
Aktualisiere den Wert von σ_q^2

$$\sigma_q^2 = (2 \cdot A)/(|R| \cdot l \cdot N')$$

Aktualisiere $q_r(k) \;\; \forall\, r,k$ mit

$$q_r(k) = \exp\left(-A(r,k)/\sigma_q^2\right)$$

Setze $A = 0$

Für jeden Knoten r:
Berechne den Mittelwert und die Streuung

$$\bar{k}(r) = \tilde{k}(r)/N$$
$$\sigma(r) = \left[\tilde{\sigma}(r) - [\tilde{k}(r)]^2/N\right]/N$$

Ordne die Knoten nach fallenden Werten von $\sigma(r)$. Wähle den Schwellwert ε' als die Streuung des n_R-ten Knotens in dieser Ordnung. Wähle den neuen Schwellwert: $\varepsilon = \max[\varepsilon, \varepsilon']$

Für jeden Knoten r:

$$d(r) = \begin{cases} \bar{k}(r) & \text{wenn } \sigma(r) < \varepsilon \\ RW & \text{sonst} \end{cases}$$

ABBILDUNG 4.6. Modifiziertes Schema für die Ermittelung der Entscheidung

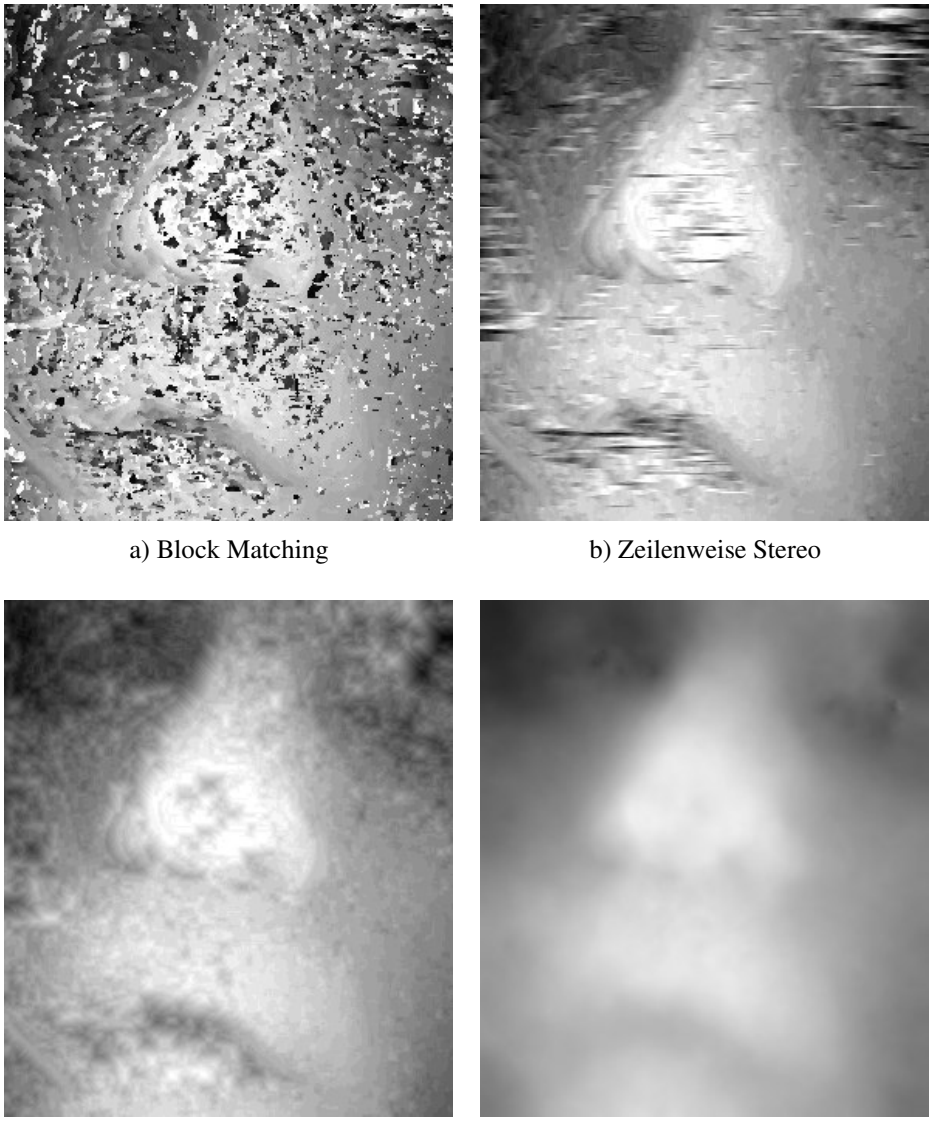
a) Das linke Bild b) Das rechte Bild

ABBILDUNG 4.7. Beispiel „Raucher“

In der Abbildung 4.7 ist ein Stereopaar gezeigt. In der Abbildung 4.8 sind die Ergebnisse von verschiedenen Algorithmen dargestellt. Die Tiefenkarten sind mit den Grauwerten kodiert, so dass hellere Grauwerte kleineren Werten der Z-Koordinate entsprechen. Für alle Methoden wurde dasselbe Qualitätsmaß benutzt. Als Glattheitsforderung für b) (nur entlang der Zeilen), c) und d) wurde die scharfe Glattheitsforderung benutzt.

Vor allem sieht man eine deutliche Verbesserung beim Übergang von den einfachen Algorithmen zu den komplizierteren. Andererseits kann man auch bemerken, dass die gefundenen Werte der Z-Koordinaten für viele der Pixel fast übereinstimmen. Am besten ist dies beim Übergang von der MAP-Entscheidung zu der Entscheidung mit der additiven Kostenfunktion zu sehen. In dem Zusammenhang entsteht die Frage, ob man während des Generierungsprozesses (oder vor dem Prozess) die Information benutzen kann, die die MAP-Entscheidung liefert. In diesem Abschnitt diskutieren wir diese Frage.

4.3.1. Reduzierung der Anzahl der Generierungsschritte. Als erstes möchten wir die folgende Beobachtung diskutieren. Während der Generierung werden ab einer bestimmten Zeit (Anzahl der Iterationen) fast nur noch die Labelings generiert, die (im räumlichen Sinn) *nahe* zu der endgültigen Lösung liegen. In der Abbildung 4.9 ist die Karte der Streuungen gezeigt, die diesen Fakt illustriert. In dieser Karte entsprechen die hellen Grauwerte größeren Werten der Streuung. Die maximale vorkommende Streuung ist hier ungefähr 4.05 (bei der Anzahl der k-Koordinaten $k_{max} = 30$). Die mittlere Streuung beträgt 0.81. Das heißt, dass die Labelings, die sehr weit von der endgültigen Lösung entfernt sind, während des Prozesses praktisch nie generiert wurden. Folglich liegen die Labelings, die große Wahrscheinlichkeitswerte haben, meistens nahe zu der endgültigen Entscheidung. Andererseits kann man vermuten, dass es eigentlich nicht nötig ist, Labelings mit kleinen Wahrscheinlichkeitswerten zu berücksichtigen, weil solche Labelings einen sehr kleinen Beitrag für die Histogramme H_r liefern. Außerdem kann man bemerken (das haben wir bereits erwähnt), dass die endgültige Lösung nahe zu der MAP-Entscheidung liegt.



a) Block Matching

b) Zeilenweise Stereo

c) MAP-Entscheidung

d) Entscheidung für eine additive Kostenfunktion

ABBILDUNG 4.8. Die Ergebnisse von verschiedenen Algorithmen

Aufgrund der obigen Überlegungen scheint es sinnvoll zu sein, den Generierungsprozess mit dem wahrscheinlichsten Labeling zu initialisieren statt mit einem zufälligen Labeling zu beginnen. Bei der Initialisierung mit einem zufälligen Labeling benötigt der Prozess eine bestimmte Zeit bevor er beginnt, Labelings zu generieren, die große Wahrscheinlichkeitswerte haben. Beginnt man mit dem wahrscheinlichsten Labeling, so ist diese Zeit nicht mehr erforderlich.

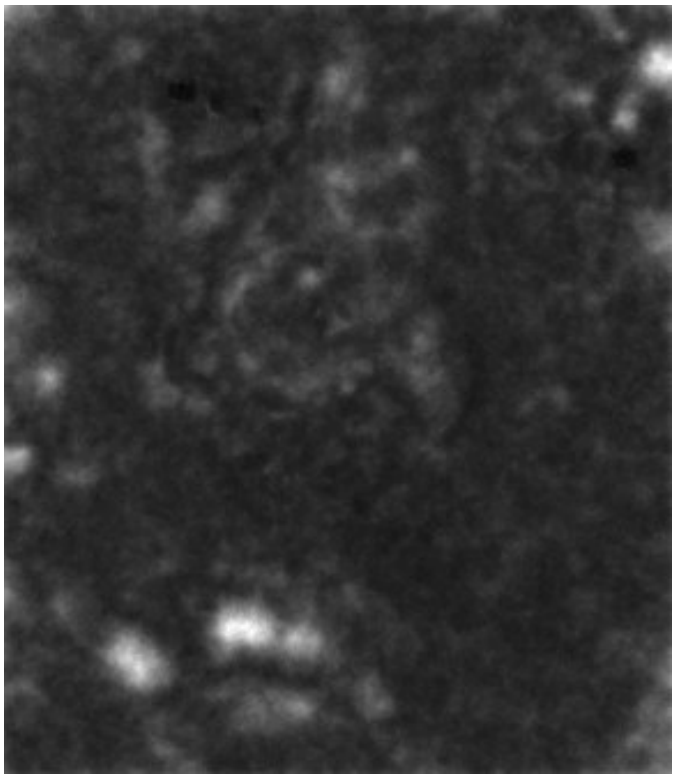


ABBILDUNG 4.9. Karte der Streuungen

Um das Verhalten des Algorithmus zu untersuchen, definieren wir zunächst, was wir unter dem Begriff „Abstand zwischen zwei Entscheidungen“ verstehen werden. Sei $d : R \to [0 \dots k_{max}]$ eine Funktion, die jedem Knoten des Basisgraphen einen Wert zuordnet. Im Allgemeinen ist der Wert eine reelle Zahl im Bereich von $[0 \dots k_{max}]$ (dabei verbieten wir die Antwort „Rückweisung“). Ist d ein Labeling, so ist der Wert ganzzahlig. Wir werden im Weiteren die folgenden zwei Abstände benutzen – den maximalen Abstand

$$D_{max}(d,d') = \max_{r \in R} |d(r) - d'(r)| \tag{4.26}$$

und den mittleren Abstand

$$D_{middle}(d,d') = \frac{1}{|R|} \sum_{r \in R} |d(r) - d'(r)|. \tag{4.27}$$

In der Abbildung 4.10 ist zu sehen, wie der Algorithmus zu der „wahren“ Entscheidung konvergiert. Als „wahre“ Entscheidung d^* wird hier diejenige Entscheidung bezeichnet, die nach einer sehr großen Anzahl von Iterationen erzeugt wird. Dabei ist die Antwort „Rückweisung“ verboten (die Strafe ε dafür ist auf eine große Zahl gesetzt). Sei $d(t)$ die Entscheidung, die nach t Iterationen getroffen werden sollte, falls der Prozess in diesem Zeitpunkt beendet wird. Die Grafiken zeigen die Abhängigkeiten $D_{middle}\big(d(t), d^*\big)$ bzw. $D_{max}\big(d(t), d^*\big)$. Die oberen Kurven entsprechen der Initialisierung mit dem zufälligen Labeling. Die unteren Kurven zeigen die Abhängigkeiten für den Fall, dass der Generierungsprozess mit dem wahrscheinlichsten Labeling startet.

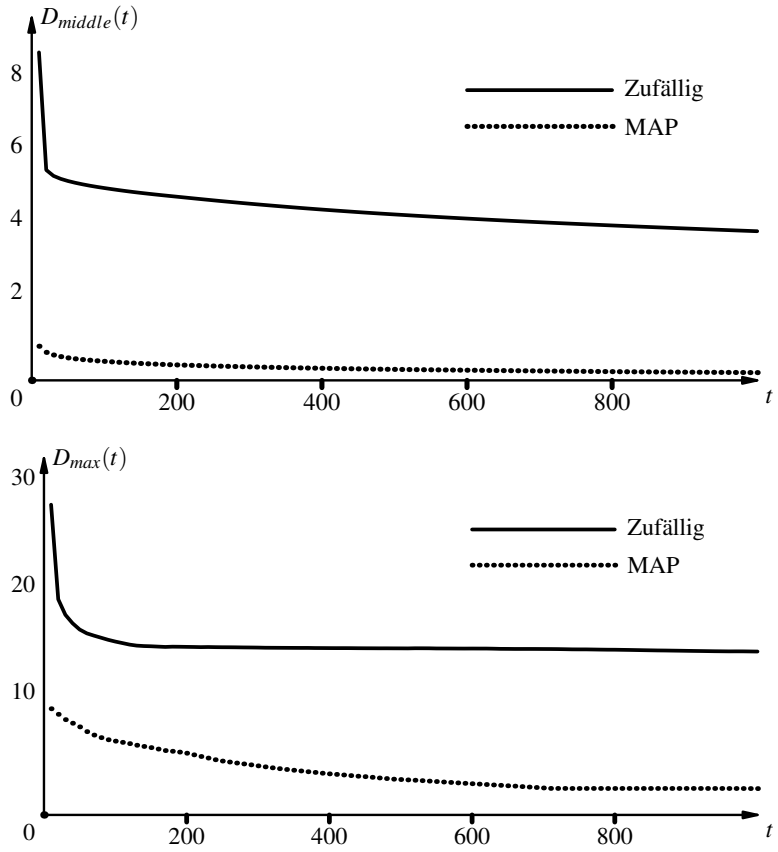
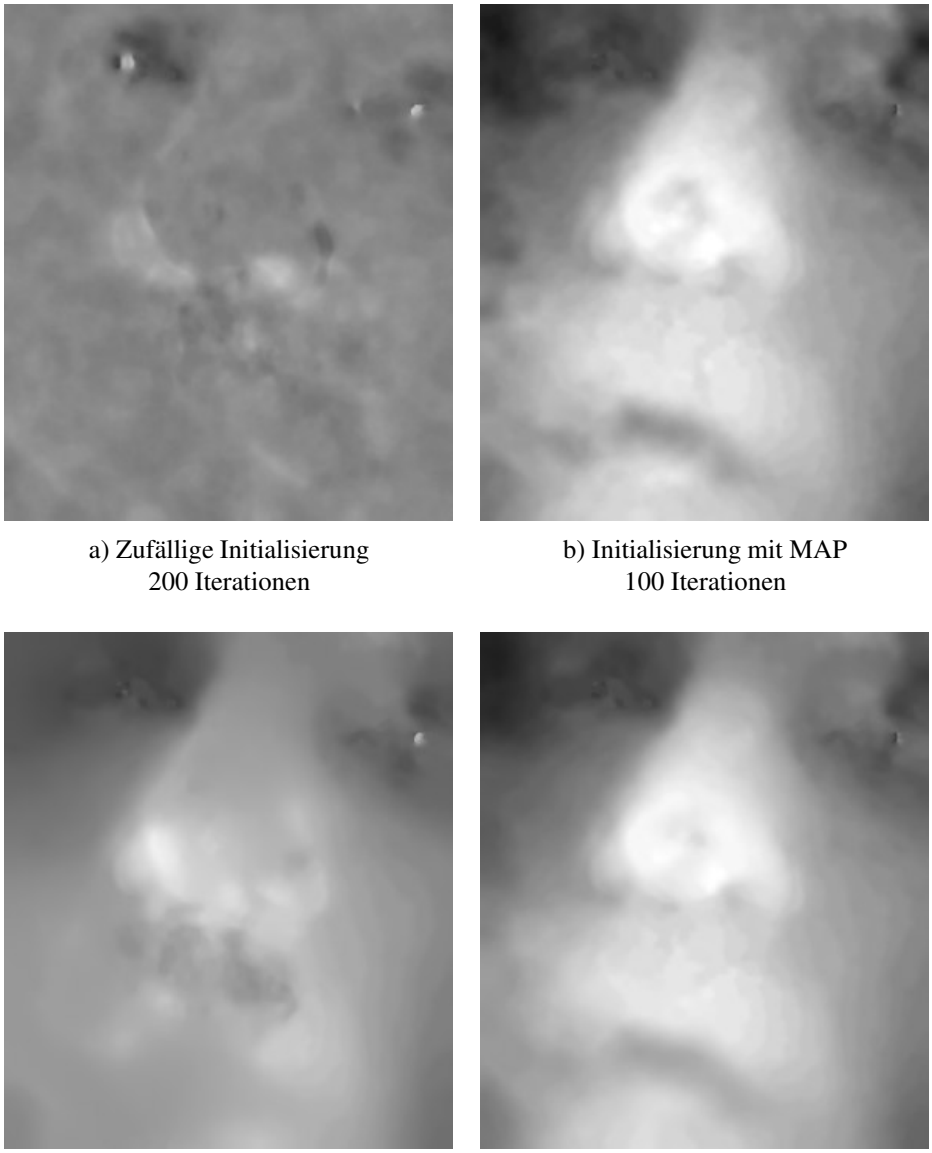


ABBILDUNG 4.10. Konvergenz des Generierungsprozesses

In der Abbildung 4.11 sind die Tiefenkarten angegeben, die bei der Benutzung der verschiedenen Initialisierungen entstehen. Anhand dieser Abbildung sieht man, dass bei der Initialisierung mit dem wahrscheinlichsten Labeling eine subjektiv gute Tiefenkarte nach einer kleineren Anzahl der Iterationen entsteht, als beim Starten mit einem zufälligen Labeling. Somit scheint es sinnvoll zu sein, den Generierungsprozess mit dem wahrscheinlichsten Labeling zu initialisieren.

Bevor wir weitere Änderungen des Algorithmus betrachten, möchten wir Folgendes bemerken. Da wir die MAP-Entscheidung nur für die Initialisierung benutzen, genügt es, diese *näherungsweise* zu ermitteln. Bei der Verwendung des Potts Modells ist es eigentlich gar nicht möglich, die MAP-Entscheidung exakt zu bestimmen, weil in diesem Fall die entsprechende Aufgabe NP-vollständig ist [**5**]. Deswegen benutzen wir für die Initialisierung des Generierungsprozesses die MAP-Entscheidung für die scharfe Glattheitsforderung (4.2) (unabhängig davon, welches Modell während der Generierung benutzt wird).

4.3.2. Schätzung des Parameters σ_q. Das gesamte Schema enthält mehrere Parameter: den Typ der Berechnung des Ähnlichkeitsmaßes, die Art der benutzten Glattheitsforderung usw. Es ist leider sehr schwer, die Wahl *aller* Parameter zu automatisieren. Die



a) Zufällige Initialisierung
200 Iterationen

b) Initialisierung mit MAP
100 Iterationen

c) Zufällige Initialisierung
10000 Iterationen

d) Initialisierung mit MAP
400 Iterationen

ABBILDUNG 4.11. Evaluierung der Tiefenkarte

Hauptschwierigkeit liegt darin, dass viele Parameter gewissen Annahmen über das benutzte Modell entsprechen (zum Beispiel die Art der benutzten Glattheitsforderung). Einen solchen Parametersatz (bei einem gegebenen Bildpaar) auszuwählen heißt eine Entscheidung zu treffen, welches Modell für dieses Bildpaar am besten geeignet ist. Für die Automatisierung dieser Entscheidung wird ein Maß benötigt, bezüglich dessen die Wahl eines aus

mehreren Modellen durchgeführt werden soll. Zur Zeit scheint es kaum möglich zu sein, ein solches Maß vernünftig zu definieren.

Bei der manuellen Auswahl der Parameter spielt Folgendes eine wichtige Rolle. Manche Parameter haben eine offensichtliche physikalische Bedeutung. Vor allem sind das die Parameter, die das benutzte Modell definieren, nämlich: der Typ der Berechnung des Ähnlichkeitsmaßes und der Typ der Modellfunktion g. Fast alle anderen „numerischen“ Parameter können vom Benutzer relativ einfach gesetzt werden, weil es zumindest bekannt ist, in welchem Bereich die Werte dieser Parameter liegen. Betrachten wir als Beispiel den Parameter δ – d.h. den maximalen Sprung der Z-Koordinate bei der scharfen Glattheitsforderung. Es ist einfach zu sehen, dass der Wert dieses Parameters nur dann einen Sinn hat, wenn er viel kleiner als die Anzahl der Diskretisierungsschritte entlang der Z-Achse k_{max} ist.

Bei der Auswahl des Parameters σ_q kommt es allerdings zu Schwierigkeiten. Sein Wert wird benötigt, um die bedingte Wahrscheinlichkeitsverteilung q zu spezifizieren. Falls der Parameter während des Generierungsprozesses gelernt wird, muss man einen initialen Wert dieses Parameters angeben (siehe Algorithmus in der Abbildung 4.6). Im Gegensatz zu den anderen Parametern lässt sich für diesen Parameter im Voraus leider kein „vernünftiger“ Wertebereich angeben. Die angemessenen Werte dieses Parameters hängen sehr stark sowohl von anderen Parametern als auch vom gegebenen Stereopaar ab. Genauer gesagt, hängen sie meistens vom Wertebereich ab, in dem sich die Werte des Ähnlichkeitsmaßes befinden.

Andererseits bemerkt man, dass die MAP-Entscheidung für die scharfe Glattheitsforderung (die wir benutzen um den Generierungsprozess zu initialisieren) nicht vom Parameter σ_q abhängt. Bei diesem Modell haben alle Labelings die a-priori Wahrscheinlichkeit entweder $1/Z$ oder 0. Bezeichnen wir mit $\mathcal{F} \subset K^R$ die Menge aller Labelings, die die a-priori Wahrscheinlichkeit $1/Z$ haben. Somit lässt sich die Formel für die MAP-Entscheidung wie folgt ausdrücken:

$$f^* = \arg\max_{f\in\mathcal{F}} \prod_r q_r\big(f(r)\big) = \arg\max_{f\in\mathcal{F}} \prod_r \frac{1}{\sigma_q^l} \exp\left[-\frac{A\big(r,f(r)\big)}{\sigma_q^2}\right] = \arg\min_{f\in\mathcal{F}} \sum_r A\big(r,f(r)\big).$$

Das heißt, dass man die MAP-Entscheidung für die scharfe Glattheitsforderung ermitteln kann ohne den Wert des Parameters σ_q zu kennen. Somit entsteht eine Möglichkeit, die erhaltene MAP-Entscheidung zu benutzen, um diesen Wert zu schätzen.

Wir benutzen dafür die folgende Heuristik. Wir gehen davon aus, dass die erhaltene MAP-Entscheidung die endgültige Lösung bereits sehr gut approximiert. Somit wählen wir den Wert von $\sigma_q^{(0)}$, so dass die Wahrscheinlichkeit der MAP-Entscheidung maximal wird. Mit anderen Worten, ersetzen wir für initiale Schätzung von $\sigma_q^{(0)}$ die ursprüngliche Aufgabe $P(X;\sigma_q) \to \max_{\sigma_q}$ durch die Aufgabe $P(X,f^*;\sigma_q) \to \max_{\sigma_q}$, d.h.

$$\sigma_q^{(0)} = \arg\max_{\sigma_q} \prod_{rr'} g_{rr'}\big(f^*(r),f^*(r')\big) \prod_r q_r\big(f^*(r')\big) = \arg\max_{\sigma_q} \prod_r \frac{1}{\sigma_q^l} \exp\Big[-\frac{A\big(r,f^*(r)\big)}{\sigma_q^2}\Big].$$

Der optimale Wert von $\sigma_q^{(0)}$ ergibt sich aus

$$\sigma_q^{(0)2} = \frac{2}{|R|\cdot l} \sum_r A\big(r,f^*(r)\big). \tag{4.28}$$

Die Experimente zeigen, dass der damit geschätzte Wert von $\sigma_q^{(0)}$ mit dem während des Prozesses gelernten Wert sehr gut übereinstimmt.

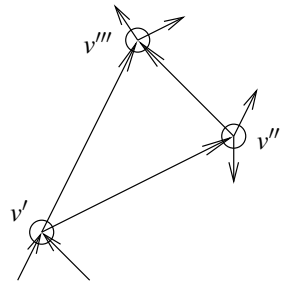


ABBILDUNG 4.12. Zyklus aus Kanten mit unendlichen Kapazitäten

4.3.3. Ermittelung der MAP-Entscheidung. In diesem Abschnitt betrachten wir die Ermittelung der MAP-Entscheidung für den Fall, dass die scharfe Glattheitsforderung als Modellfunktion g benutzt wird. Für die Ermittelung der MAP-Entscheidung überführen wir die entstehende MinSum Aufgabe in die entsprechende MaxFlow Aufgabe (siehe Abschnitt 3.3). Letztere lösen wir mit Hilfe des in [**1**] beschriebenen Preflow-Push Algorithmus.

Bei der Konstruktion des Graphen für die MaxFlow Aufgabe entsteht das folgende Problem. Wenn einige Werte der Funktion g unendlich groß sind, lassen sich die Zahlen α (siehe (3.24)) nicht immer definieren. Um dieses Problem zu umgehen, ersetzen wir die ursprüngliche Funktion g durch eine andere Funktion wie folgt:

$$g(k,k') = \begin{cases} 0 & \text{wenn } |k-k'| \le \delta \\ \infty & \text{sonst} \end{cases} \Rightarrow$$

$$\Rightarrow g(k,k') = \begin{cases} 0 & \text{wenn } |k-k'| \le \delta \\ G \cdot (|k-k'| - \delta) & \text{sonst.} \end{cases}$$

Die Konstante G ist hier eine sehr große Zahl. Nach dieser Änderung lässt sich die Aufgabe in eine Aufgabe mit zwei Zuständen überführen, in der nur noch Kanten der folgenden Typen auftreten: $\begin{array}{|c|c|}\hline \infty & 0 \\ \hline 0 & 0 \\ \hline \end{array}$ und $\begin{array}{|c|c|}\hline 0 & -G \\ \hline 0 & 0 \\ \hline \end{array}$. Mit Hilfe von äquivalenten Transformationen lassen sich die Kanten des zweiten Typs in die Form $\begin{array}{|c|c|}\hline G & 0 \\ \hline 0 & 0 \\ \hline \end{array}$ überführen. Ersetzt man G durch ∞, so erhält man eine Aufgabe mit zwei Zuständen, in der nur Kanten des Typs $\begin{array}{|c|c|}\hline \infty & 0 \\ \hline 0 & 0 \\ \hline \end{array}$ auftreten. Die erhaltene Aufgabe lässt sich weiter in eine MinCut (MaxFlow) Aufgabe überführen.

Der erhaltene Graph der MaxFlow Aufgabe kann auf folgende Art vereinfacht werden. Nach der Überführung enthält dieser Graph ungerichtete Zyklen aus Kanten mit unendlichen Kapazitäten. Betrachten wir einen solchen Zyklus (siehe Abbildung 4.12). In dem Beispiel haben die mit den Pfeilen gezeichneten Kanten unendliche Kapazitäten: $c(v',v'') = c(v'',v''') = c(v',v''') = \infty$. Nehmen wir an, das der maximale Fluss eine streng positive Komponente auf die Kante (v',v''') hat:

$$X^* = \big(\ldots, x(v',v''), x(v'',v'''), x(v',v''') > 0, \ldots\big).$$

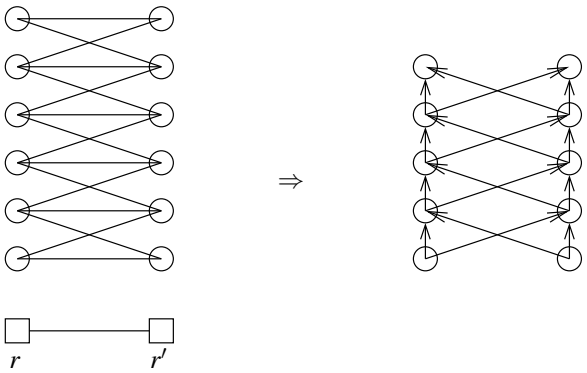


ABBILDUNG 4.13. Überführung der MinSum Aufgabe in die entsprechende MinCut Aufgabe

Da alle Kanten unendlichen Kapazitäten haben, kann man einen anderen Fluss konstruieren, der offensichtlich auch maximal ist:

$$X'^{*} = \big(\ldots, x(v',v'') + x(v',v'''), x(v'',v''') + x(v',v'''), 0, \ldots\big).$$

Dabei bleiben die Nebenbedingungen (3.25) und (3.26) für den Fluss erfüllt. Das heißt, dass unter allen maximalen Flüssen solche existieren, in denen die gewissen Komponenten (wie $x(v',v''')$ im obigen Beispiel) Null sind. Somit kann man die Kapazitäten solcher Kanten auf Null setzen, d.h. diese Kanten aus der Kantenmenge entfernen. Das führt zu einer deutlichen Verbesserung der Speicherkomplexität des Algorithmus. Im allgemeinen Fall (siehe (3.24)) ist die Anzahl der Kanten in der erhaltenen MaxFlow Aufgabe $O(|E| \cdot |K|^2)$ ($|E|$ ist die Anzahl der Kanten im Basisgraphen der MinSum Aufgabe, $|K|$ ist die Anzahl der Zustände). Ist der Basisgraph ein Gitter, so ist die Speicherkapazität $O(|R| \cdot |K|^2)$. In dem Fall der scharfen Glattheitsforderung wird die Speicherkapazität auf $O(|R| \cdot |K|)$ reduziert.

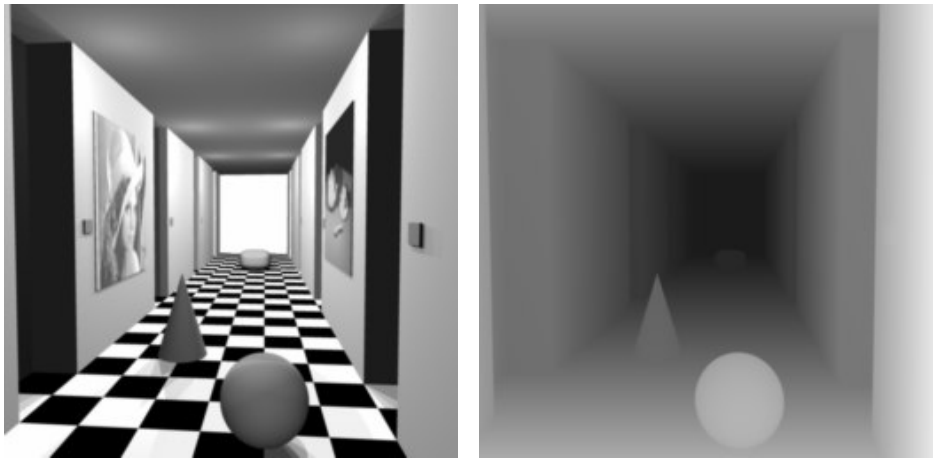
Die gesamte Überführung (von der MinSum Aufgabe mit der scharfen Glattheitsforderung zu der entsprechenden MaxFlow Aufgabe) ist schematisch in der Abbildung 4.13 gezeigt. In der MinSum Aufgabe (links) sind nur die „erlaubten“ Zustandspaare (mit $g(k,k') = 0$) gezeigt. In der MaxFlow Aufgabe (rechts) repräsentieren die Pfeile die Kanten mit unendlichen Kapazitäten. Das Beispiel entspricht dem Wert des Modellparameters $\delta = 1$. Unten sind die Regeln für die Überführung zusammengefasst:

- Die Menge der Knoten:

$$\mathcal{V} = \{(r,k) : r \in R,\ k = 1 \ldots |K|-1\} \cup \{s\} \cup \{t\}.$$

- Die Menge der Kanten:

$$\begin{aligned}\mathcal{E} = &\big\{\big((r,k),(r,k+1)\big) : r \in R,\ k = 1 \ldots |K|-2\big\} \cup \\ &\big\{\big((r,k),(r',k+\delta)\big) : (r,r') \in E,\ k = 1 \ldots |K|-1-\delta\big\} \cup \\ &\big\{\big((r',k),(r,k+\delta)\big) : (r,r') \in E,\ k = 1 \ldots |K|-1-\delta\big\} \cup \\ &\big\{\big(s,(r,k)\big) : r \in R,\ k = 1 \ldots |K|-1\big\} \cup \\ &\big\{\big((r,k),t\big) : r \in R,\ k = 1 \ldots |K|-1\big\}.\end{aligned}$$



a) Linkes Bild b) Disparitätskarte

ABBILDUNG 4.14. Beispiel „Korridor“

- Die Kantenkapazitäten:

$$c(v,v') = \infty,\ (v,v') \in \mathcal{E},\ v \neq s, v' \neq t$$
$$c\big(s,(r,k)\big) = \max[q'_{rk},0],\ r \in R,\ k = 1 \ldots |K|-1$$
$$c\big((r,k),t\big) = \max[-q'_{rk},0],\ r \in R,\ k = 1 \ldots |K|-1$$

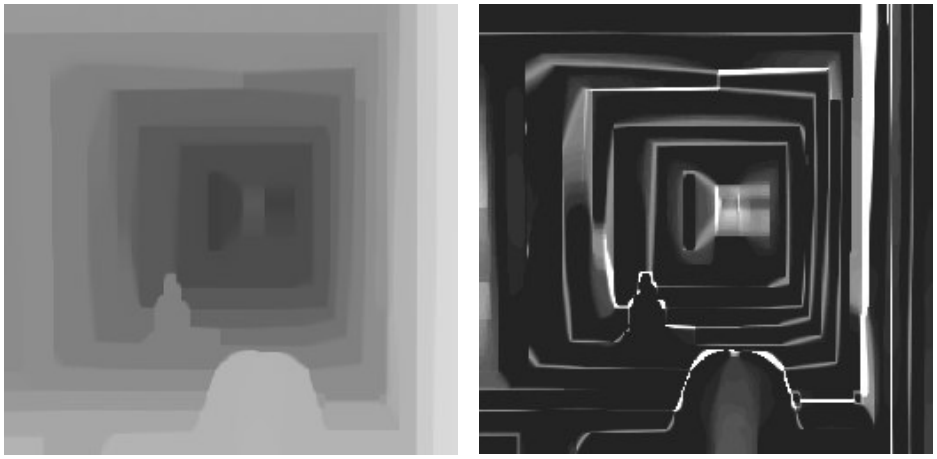
mit

$$q'_{rk} = q_r(k) - q_r(k+1). \tag{4.29}$$

4.4. Experimente

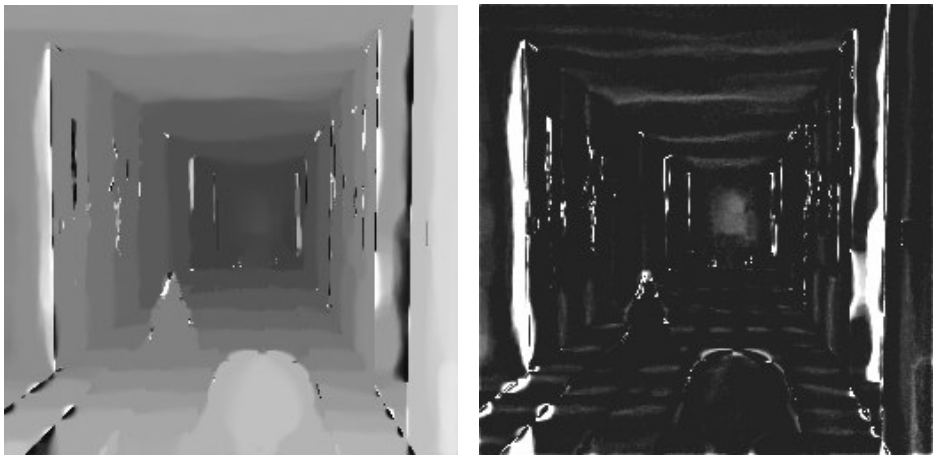
Vor allem möchten wir die Reihenfolge angeben, in welcher wir die Ergebnisse präsentieren werden. Wie bereits gesagt, enthält das gesamte Schema viele Parameter. Somit scheint es kaum möglich zu sein, die Experimente für alle Sätze der Parameter durchzuführen. Deswegen diskutieren wir als erstes nur die Frage, welche Modelle (d.h. welche Funktionen g) für welche Fälle am besten geeignet sind. Dann untersuchen wir die Einflüsse verschiedener Parameter (zum Beispiel der Typ der Berechnung des Ähnlichkeitsmaßes, die Größe des Fensters für diese Berechnung usw.) anhand nur eines Modells. Anschließend geben wir die „besten“ Ergebnisse für verschiedene Beispiele (Stereopaare) mit den Angaben von Werten der benutzten Parameter an.

4.4.1. Eigenschaften der Modelle und Einflüsse der Parameter auf die Ergebnisse. Um zu untersuchen, welche Modelle für welche Fälle am besten geeignet sind, benutzen wir das in der Abbildung 4.14 gezeigte künstlich erzeugte Bildpaar. Dabei betrachten wir für jedes Modell nur zwei „Grenzfälle“ nämlich wenn die Glattheitsbedingungen entweder sehr streng oder sehr schwach sind. Unter den Glattheitsbedingungen verstehen wir die Werte der Modellparameter, die in einem gewissen Sinn einen Kompromiss zwischen den Einflüssen der a-priori Annahme (eine glatte Oberfläche) und der Beobachtung darstellen. Die weiter betrachteten Fälle und die entsprechenden Glattheitsbedingungen sind in der folgenden Tabelle zusammengefasst:



a) Tiefenkarte b) Karte der Streuungen

ABBILDUNG 4.15. Potts Modell – strenge Glattheitsbedingungen

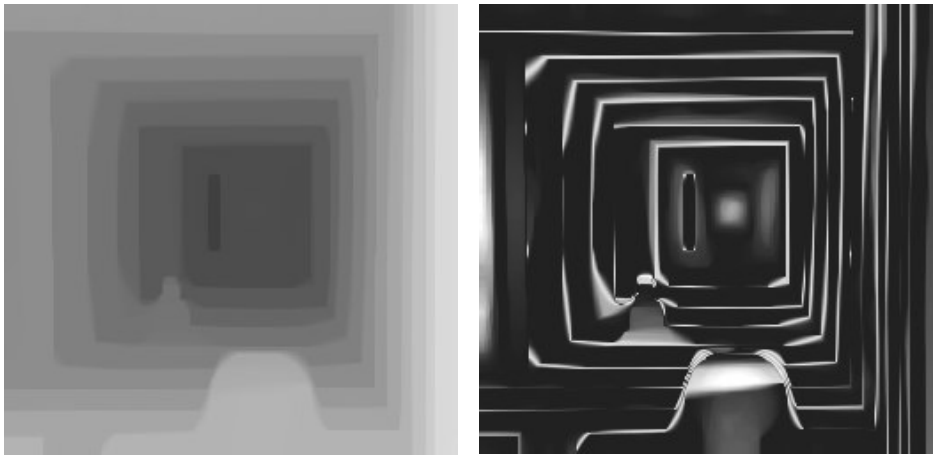


a) Tiefenkarte b) Karte der Streuungen

ABBILDUNG 4.16. Potts Modell – schwache Glattheitsbedingungen

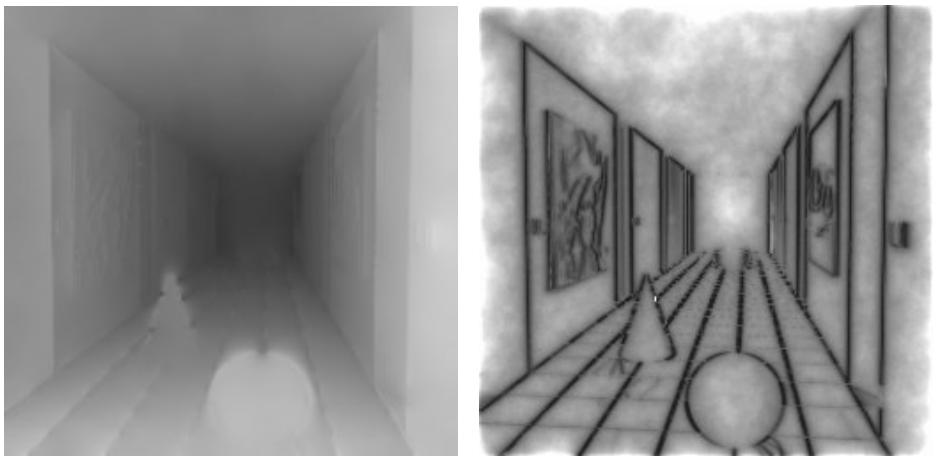
	strenge Bedingungen		schwache Bedingungen	
	a-priori Modell	Beobachtung	a-priori Modell	Beobachtung
Potts Modell	a – groß		a – klein	
Gaussches Modell	σ_g – klein	l – klein	σ_g – groß	l – groß
Scharfes Modell	δ – klein		δ – groß	

Die Ergebnisse für das Potts Modell sind in den Abbildungen 4.15 und 4.16 angegeben. Aus diesen zwei Ergebnissen sieht man bestimmte Nachteile des Potts Modell für die Stereorekonstruktion. Die Besonderheit dieses Modells besteht in der Annahme, dass die zu rekonstruierende Oberfläche eine stückweise konstante Tiefe hat. Die Wirkung dieser Eigenschaft ist besonders gut bei der Benutzung der strengen Glattheitsbedingungen zu sehen. Bei dem Beispiel „Korridor“ trifft die Annahme offensichtlich nicht zu. Einen anderen



a) Tiefenkarte b) Karte der Streuungen

ABBILDUNG 4.17. Gaussches Modell – strenge Glattheitsbedingungen



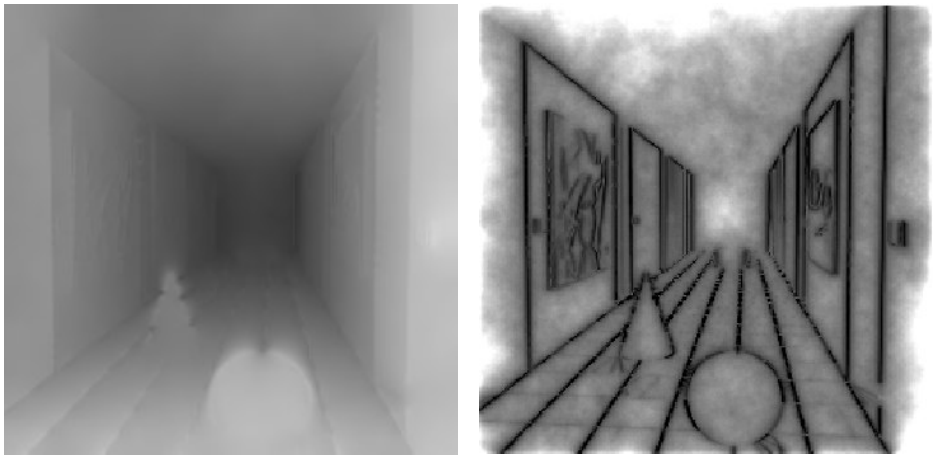
a) Tiefenkarte b) Karte der Streuungen

ABBILDUNG 4.18. Gaussches Modell – schwache Glattheitsbedingungen

Nachteil des Potts Modells ist bei der Benutzung der schwachen Glattheitsbedingungen zu sehen. In diesem Fall entstehen Fehler, die durch Folgendes verursacht werden. Die Strafe für Diskontinuitäten hängt nicht davon ab, wie groß diese Diskontinuitäten sind. Deswegen entstehen Fehler, die manchmal sehr weit (entlang der Z-Koordinate) von der wahren Oberfläche entfernt sind. Derartige Fehler kann man in der Abbildung 4.16 deutlich sehen.

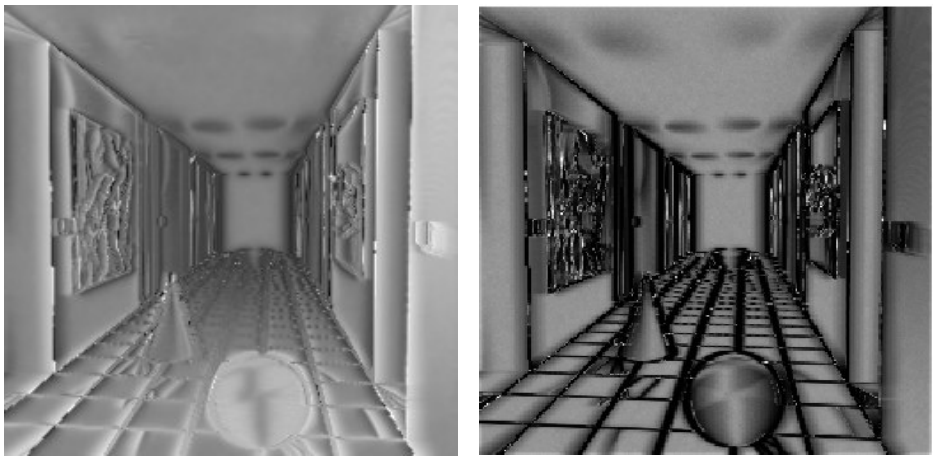
Das Potts Modell hat allerdings auch gewisse Vorteile. Die Diskontinuitäten, die in der Szene tatsächlich vorhanden sind, werden relativ gut erkannt. Außerdem möchten wir bemerken, dass sich die oben beschriebenen Fehler anhand der Karte der Streuungen im Prinzip detektieren lassen. Dafür werden allerdings zusätzliche Heuristiken benötigt.

Betrachten wir die Ergebnisse, die bei der Benutzung des Gausschen Modells entstehen (siehe Abbildungen 4.17 und 4.18). Für den Fall der strengen Glattheitsbedingungen



a) Tiefenkarte b) Karte der Streuungen

ABBILDUNG 4.19. Scharfes Modell – strenge Glattheitsbedingungen



a) Tiefenkarte b) Karte der Streuungen

ABBILDUNG 4.20. Scharfes Modell – schwache Glattheitsbedingungen

sind die Ergebnisse kaum vom Potts Modell zu unterscheiden. Die „stufenartige“ Oberfläche, die auch in diesem Fall erzeugt wird, kann man wie folgt erklären. Bei der Verwendung der strengen Glattheitsbedingungen werden große Sprünge der Z-Koordinate (größer als um einen Diskretisierungsschritt) so stark bestraft, dass sie praktisch nie auftreten. Betrachtet man die Menge nur derartiger Oberflächen (ohne der Sprünge größer als um Eins), so sind das Potts Modell und das Gaussche Modell zu einander äquivalent.

Schließlich, betrachten wir die Ergebnisse für das scharfe Modell (siehe Abbildungen und 4.19 und 4.20). Die Besonderheit dieses Modells besteht darin, dass sogar bei der Benutzung der strengen Glattheitsbedingungen keine „stufenartige“ Oberfläche entsteht. Diese Eigenschaft folgt daraus, dass kleine Sprünge der Z-Koordinate überhaupt nicht bestraft werden. Die schlechten Ergebnisse, die bei der Benutzung der schwachen Glattheitsbedingungen entstehen, kann man wie folgt erklären. Die Szene im Beispiel „Korridor“

enthält viele homogen gefärbte Gebiete. In diesem Fall gibt es viele Knoten r des Basisgraphen (viele Orte (X,Y) im 3D-Raum), für die die Werte des Ähnlichkeitsmaßes $A(r,k)$ für alle Zustände k gleich gut sind. Zusammen mit den schwachen Glattheitsbedingungen führt das dazu, dass die Werte der marginalen Wahrscheinlichkeitsverteilungen in solchen Knoten ungefähr gleich sind. Somit tendiert der Mittelwert der Zustände zu $k_{max}/2$, was offensichtlich falsch ist.

Die Ergebnisse für das scharfe Modell bei der Benutzung der strengen Glattheitsbedingungen (Abbildung 4.19) sind zu den Ergebnissen für das Gaussche Modell bei der Benutzung der schwachen Glattheitsbedingungen (Abbildung 4.18) sehr ähnlich. Einen Unterschied ist nur bei starker Vergrößerung der Bilder zu sehen. Man kann bemerken, dass die Diskontinuitäten bei dem Gausschen Modell ein bisschen besser behandelt werden (schärfer aussehen). Außerdem haben die Pixel, die zu den homogen gefärbten Flächen gehören, etwas kleineren Streuungswerte. Das heißt, dass solche Gebiete in einem gewissen Sinn „sicherer“ erkannt werden. Allerdings wird in diesem Fall eine deutlich größere Anzahl von Generierungsschritten benötigt, um zu den endgültigen Ergebnissen zu kommen, als bei der Benutzung des scharfen Modells.

Fassen wir die Eigenschaften der obigen Modelle zusammen.

- Das Potts Modell ist unserer Meinung nach für die Stereorekonstruktion nicht gut geeignet. Dieses Modell sollte nur benutzt werden, wenn die zu rekonstruierende Szene tatsächlich stückweise konstante Tiefe hat. Trifft diese Annahme nicht zu, so werden zusätzliche Kenntnisse über die Szene benötigt, um die für dieses Modell charakteristischen Fehler zu beseitigen. Das Potts Modell kann allerdings benutzt werden, wenn es für konkrete Anwendung besonders wichtig ist, die Positionen der Diskontinuitäten möglichst präzise zu bestimmen.
- Das Gaussche Modell ist besonders gut für die Rekonstruktion von glatten Oberflächen geeignet, die „fast senkrecht“ zu den optischen Achsen der Kameras sind. Die Diskontinuitäten werden dabei auch relativ gut behandelt.
- Das scharfe Modell ist unserer Meinung nach für die Stereorekonstruktion am besten geeignet, wenn keine zusätzliche Information über die zu rekonstruierende Szene zur Verfügung steht. Besonders gut werden Szenen rekonstruiert, die keine große Diskontinuitäten enthalten. Die genauen Positionen der Diskontinuitäten lassen sich bei diesem Modell allerdings nicht bestimmen.

Im Weiteren werden wir die Einflüsse verschiedener Parameter anhand des scharfen Modells untersuchen.

Als nächstes diskutieren wir, wie sich die Ergebnisse bei der Anwendung verschiedener Ähnlichkeitsmaße unterscheiden. Wie bereits gesagt, dienen die „komplizierteren“ Ähnlichkeitsmaße dazu, gewisse Effekte der Beleuchtung zu beseitigen (siehe Abschnitt 2.1). Somit scheint es sinnvoll zu sein, eine derartige Analyse anhand eines realen Stereopaares durchzuführen, weil künstliche Beispiele (meistens) unter der Annahme des Lambertschen Reflexionsmodells erzeugt werden und somit die oben genannten Effekte der Beleuchtung nicht enthalten. Deswegen benutzen wir das in der Abbildung 4.7 gezeigte Beispiel „Raucher“.

In der Abbildung 4.21 sind die Ergebnisse gezeigt, die bei der Benutzung verschiedener Ähnlichkeitsmaße entstehen. Wir geben zusätzlich für alle Ähnlichkeitsmaße die MAP-Entscheidungen an, weil anhand dieser Ergebnisse bestimmte Unterschiede besser

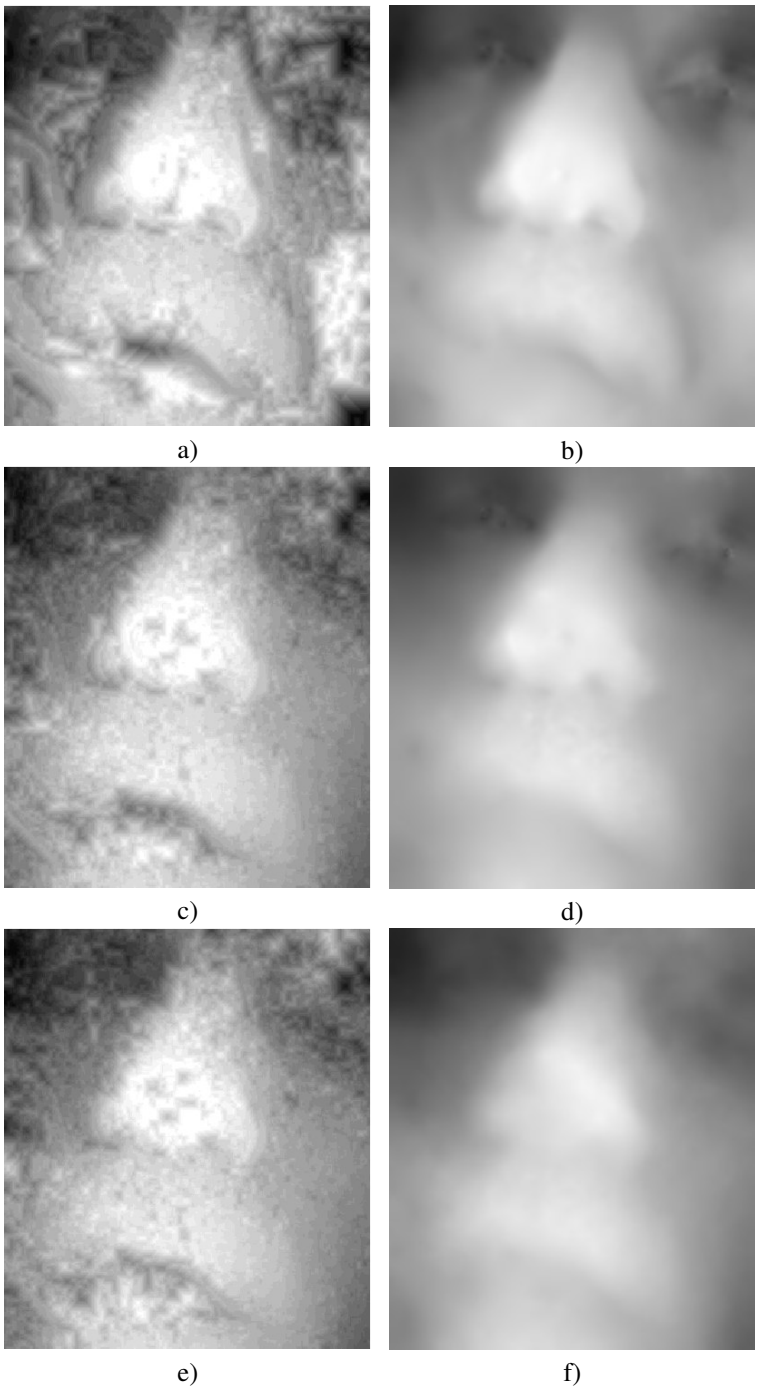


ABBILDUNG 4.21. Beispiel „Raucher“. Vergleich verschiedener Ähnlichkeitsmaße. a) und b) – Quadratische Differenz. c) und d) – Quadratische Differenz mit Verschiebung. e) und f) – Korrelationskoeffizient. a), c) und e) – MAP. b), d) und f) – Ergebnisse des Generierungsprozesses

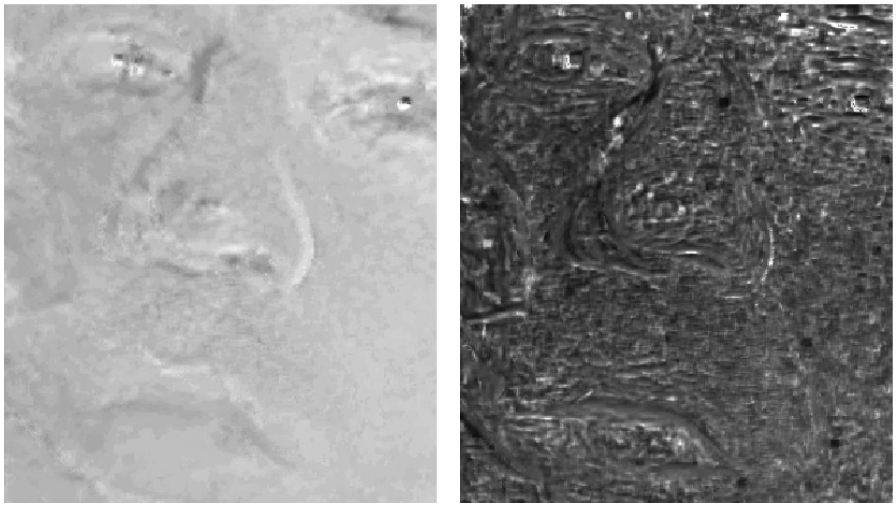
zu sehen sind. Vor allem sieht man, dass durch die Benutzung komplizierterer Ähnlichkeitsmaße tatsächlich gewisse „Fehler“ beseitigt werden. Andererseits ist es leicht zu sehen, dass die Ergebnisse für die komplizierteren Ähnlichkeitsmaße im Vergleich zu den Ergebnissen für die quadratische Differenz etwas glatter aussehen. Mit anderen Worten, werden große Sprünge der Z-Koordinate nicht genügend gut behandelt. Man kann auch bemerken, dass dieser Effekt in den Ergebnissen der Generierung deutlicher zu sehen ist, als in den MAP-Entscheidungen. Der Fakt lässt sich wie folgt erklären. Bei der Berechnung der komplizierteren Ähnlichkeitsmaße werden zusätzlichen Farbtransformationen zugelassen – eine Grauwertverschiebung C_v bei (2.2) und zusätzlich eine Grauwertskalierung C_s beim Korrelationskoeffizient (2.3). Das heißt, dass die Qualitäten (Werte des Ähnlichkeitsmaßes) auch für „falsche“ Korrespondenzpaare gute Werte annehmen können. Dabei ist offensichtlich, dass sich die Werte des Ähnlichkeitsmaßes desto weniger unterscheiden, je breiter die Klasse der zugelassenen Farbtransformationen ist. Dies entspricht einer gewissen „Abschwächung“ des Einflusses der Beobachtung auf die Ergebnisse. In diesem Fall werden kleine Details der Oberfläche sowie Diskontinuitäten in der Regel etwas schlechter rekonstruiert.

Ein möglicher Ausweg könnte darin bestehen, bestimmte zusätzliche Einschränkungen für C_v und C_s zu formulieren. Bezeichnen wir mit $C_v(r,k)$ und $C_s(r,k)$ die Werte dieser Größen, die bei der Berechnung des Ähnlichkeitsmaßes $A(r,k)$ entstehen. Die Größen C_v und C_s entsprechen gewissen physikalischen Eigenschaften der Beleuchtung. Das bedeutet dass sich diese Größen im 3D-Raum $R \times K$ nur „langsam“ ändern können. Dieser Fakt wird aber bei der Berechnung des Ähnlichkeitsmaßes nicht berücksichtigt: diese Größen werden für alle Paare (r,k) von einander unabhängig bestimmt. In der Abbildung 4.22 sind die Verläufe $C_v\big(r, f^*(r)\big)$ und $C_s\big(r, f^*(r)\big)$ für die MAP-Entscheidung f^* und den Korrelationskoeffizient gezeigt. Der Verlauf der Größe C_v bei der quadratischen Differenz mit Verschiebung unterscheidet sich kaum von dem beim Korrelationskoeffizient erhaltenen Verlauf. In dieser Abbildung sieht man, dass sich die Werte von C_v und C_s (für benachbarte Pixel) sehr stark unterscheiden, was offensichtlich nicht der Realität entspricht.

Zusammenfassend, kann man die folgenden Eigenschaften der oben betrachteten Ähnlichkeitsmaße formulieren:

- Je breiter die Klasse der zugelassenen Farbtransformationen ist, desto besser werden die Fehler beseitigt, die durch verschiedene Beleuchtungs- bzw. Reflexionseffekte verursacht werden.
- Je breiter die Klasse der zugelassenen Farbtransformationen ist, desto schlechter werden kleine Details sowie große Sprünge der Z-Koordinate (d.h. Diskontinuitäten) behandelt.

Als nächstes betrachten wir den Einfluss der für die Berechnung des Ähnlichkeitsmaßes verwendeten Fenstergröße d auf die Ergebnisse (siehe (2.1) – (2.3)). Einerseits ist offensichtlich, dass das Ähnlichkeitsmaß desto robuster wird, je größer das Fenster ist. Andererseits entsteht bei der Benutzung großer Fenster das folgende Problem (siehe Abbildung 4.23). Einem Gebiet auf der zu rekonstruierenden 3D-Oberfläche (in der Abbildung mit G bezeichnet) entsprechen Gebiete auf dem linken bzw. auf dem rechten Bild (G_l bzw. G_r), die unterschiedlich sein können. Streng genommen, sind diese korrespondierenden Gebiete nur dann gleich, wenn das entsprechende Gebiet der Oberfläche durch eine Ebene approximiert werden kann und der Normalenvektor dieser Ebene zu den optischen Achsen der Kameras parallel ist. Bei der Benutzung gleicher Gebiete (quadratischer Fenster einer gleichen fest definierten Größe) für die Berechnung des Ähnlichkeitsmaßes wird dies offensichtlich nicht berücksichtigt. Der Unterschied zwischen den korrespon-



a) $C_v\big(r, f^*(r)\big)$ b) $C_s\big(r, f^*(r)\big)$

ABBILDUNG 4.22. Verläufe der Größen C_v und C_s für die MAP-Entscheidung f^* und den Korrelationskoeffizient

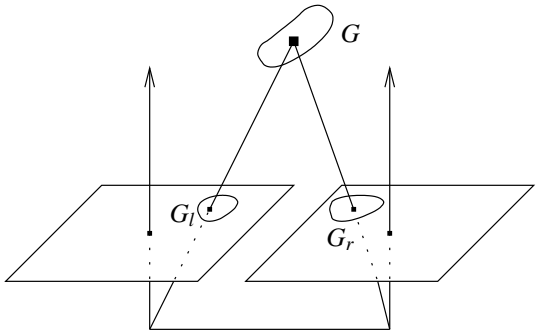
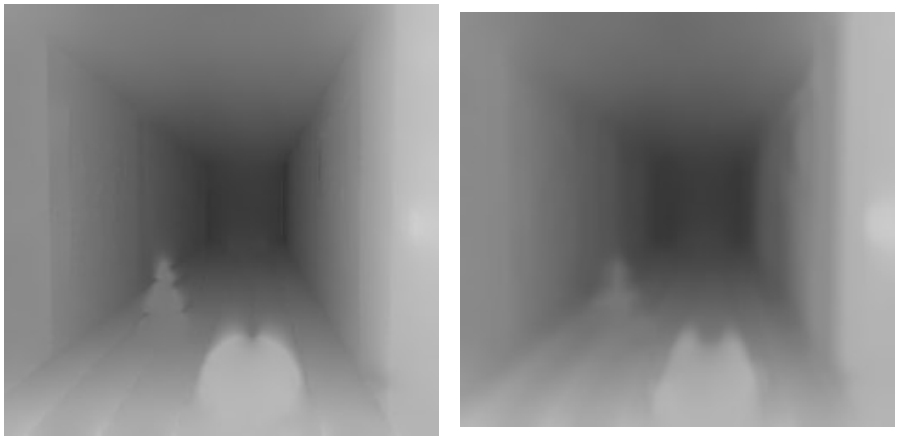


ABBILDUNG 4.23. Korrespondierende Gebiete verschiedener Formen

dierenden Gebieten G_l und G_r ist dabei umso deutlicher, je größer die Gebiete sind. Der Einfluss dieses Effektes ist in der Abbildung 4.24 leicht zu sehen. Um dies zu illustrieren, benutzen wir wieder das Beispiel „Korridor“. Als Ähnlichkeitsmaß wird die quadratische Differenz (2.1) benutzt. Man kann leicht sehen, dass insbesondere Diskontinuitäten bei der Benutzung großer Fenster (11×11 in 4.24,b) viel schlechter rekonstruiert werden, als bei der Benutzung kleiner Fenster (1×1 in 4.24,a – d.h. die Umgebungen der korrespondierenden Pixel werden überhaupt nicht benutzt). Somit scheint es sinnvoll zu sein, nur dann relativ große Fenster für die Berechnung des Ähnlichkeitsmaßes zu benutzen, wenn die zu rekonstruierende Oberfläche hauptsächlich senkrecht zu den optischen Achsen steht und die Bilder stark gestört sind.

Die Wirkungen anderer Modellparameter auf die Ergebnisse haben wir bereits erwähnt (siehe Abbildungen 4.14 – 4.20). In einem gewissen Sinn stellen die Parameter einen Kompromiss zwischen den Einflüssen der a-priori Kenntnisse (eine glatte Oberfläche) und der



a) Fenstergröße 1×1 b) Fenstergröße 11×11

ABBILDUNG 4.24. Ergebnisse für verschiedene Fenstergrößen

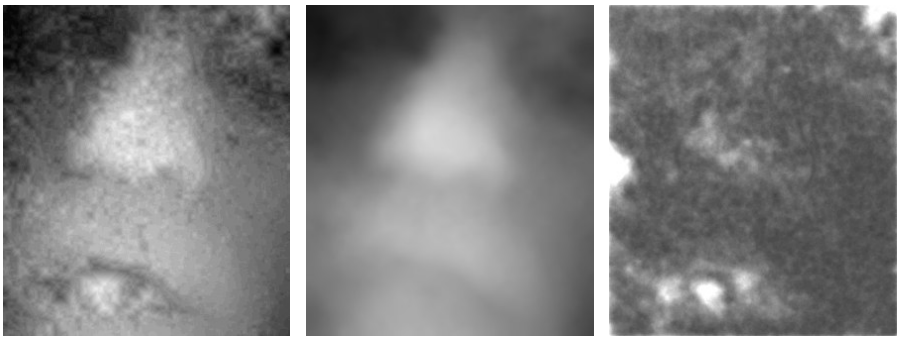
Beobachtung (Stereopaar) dar. Diese Einflüsse werden durch die Parameter δ, σ_g, a (für a-priori Kenntnisse) und durch den Parameter l (für Beobachtung) gesteuert. Die Wirkungen der Modellparameter auf die Ergebnisse kann man (qualitativ) wie folgt zusammenfassen:
a-priori: stark, Beobachtung: stark
Die Ergebnisse der Generierung sind den MAP-Entscheidungen sehr ähnlich. Die Glattheit der Oberfläche hängt dabei vom Verhältnis der Modellparameter ab.
a-priori: stark, Beobachtung: schwach
Diese Kombination von Parametern entspricht den oben betrachteten „strengen Glattheitsbedingungen“. Die dabei erhaltenen Oberflächen sind sehr oft übergeglättet. Diskontinuitäten sind in der Regel schlecht rekonstruiert. Manchmal entstehen dabei „stufenartige“ Ergebnisse. Diese Kombination von Modellparametern kann in dem Fall benutzt werden, wenn die Bilder stark verrauscht sind und die zu rekonstruierende Szene den a-priori Annahmen sehr gut entspricht.
a-priori: schwach, Beobachtung: stark
Diese Kombination von Parametern entspricht den oben betrachteten „schwachen Glattheitsbedingungen“. Die erhaltenen Oberflächen sind den Ergebnissen von Block Matching Verfahren ähnlich. Diese Kombination von Modellparametern kann benutzt werden, wenn die Bilder von einer sehr guten Qualität sind. Dies ermöglicht manchmal eine relativ gute Rekonstruktion kleiner Details und Diskontinuitäten.
a-priori: schwach, Beobachtung: schwach
Eine solche Kombination bedeutet praktisch, dass die erhaltenen Ergebnisse von der Beobachtung (sowie von anderen Parametern und von a-priori Annahmen) kaum abhängen. Die dabei erhaltenen Ergebnisse sind sehr oft nicht befriedigend.
Die konkreten Werte der Modellparameter sind von Beispiel zu Beispiel unterschiedlich. Sie hängen hauptsächlich von verschiedenen Eigenschaften der zu rekonstruierenden Szene (wie zum Beispiel dem Vorhanden sein von Diskontinuitäten und Verdeckungen, der Orientierung der Oberfläche usw.) und von der Qualität der Bilder (Kontrast, Helligkeit, Störungen usw.) ab. Für die weiter betrachteten Beispiele geben wir die jeweils verwendeten Werte der Parameter an.

4.4.2. Beispiele. Bevor wir weitere Ergebnisse präsentieren, möchten wir Folgendes bemerken. Zunächst scheint es kaum möglich zu sein, die Algorithmen für die Stereorekonstruktion quantitativ zu bewerten (und somit auch zu vergleichen), wenn die „wahre“ Lage der Oberfläche (ground truth) nicht bekannt ist. Betrachten wir den Fall, dass die wahre Oberfläche zur Verfügung steht. Ein oft für die Bewertung benutztes Maß ist die Anzahl der falsch klassifizierten Pixel – die Anzahl der Pixel, in denen sich der gefundene Wert der Z-Koordinate (oder die gefundene Disparität) vom wahren Wert unterscheidet. Dabei wird die *Differenz* zwischen diesen Werten ignoriert. Dieses Maß ist tatsächlich nichts anderes, als die Strafe (2.19) für falsche Entscheidungen mit der additiven Kostenfunktion (2.24) (d.h. der additiven Deltafunktion). Sollte die Ergebnisse eines Algorithmus entsprechend dieses Maßes bewertet werden, so soll der Algorithmus das Risiko für diese additive *Deltafunktion* minimieren. In dem Zusammenhang scheint es nicht berechtigt zu sein, die Anzahl der falsch klassifizierten Pixel für die Bewertung von Algorithmen heranzuziehen, die eine andere Zielfunktion optimieren (zum Beispiel von Algorithmen, die auf einer Energieminimierung basieren).

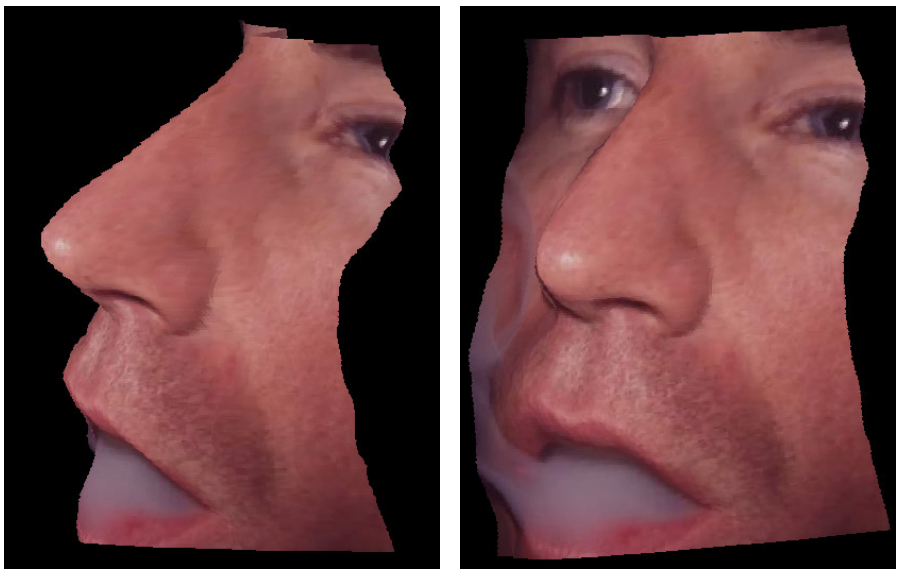
In dieser Arbeit möchten wir gerade zeigen, dass die Anwendung der additiven *quadratischen* Kostenfunktion (2.26) oder (2.30) bessere Resultate liefert, als die Benutzung anderer (im Abschnitt 2.3 betrachteten) Kostenfunktionen. In dem Fall, dass die wahre Lage der Oberfläche bekannt ist, wäre es somit natürlich, die Strafe für falsche Entscheidungen mit der *quadratischen* Kostenfunktion als Bewertung der Ergebnisse zu benutzen. Wenn die wahre Oberfläche nicht bekannt ist, kann der erreichte Wert des Risikos als Bewertung der Ergebnisse benutzt werden. Streng genommen, ist eine Bewertung anhand der Anzahl der falsch klassifizierten Pixel in unserem Fall überhaupt nicht möglich, weil die Ergebnisse reelle Zahlen sind (reelle Werte der Z-Koordinate für jedes Pixel).

Zusammenfassend behaupten wir, dass für die Bewertung der Ergebnisse nur das Maß genutzt werden sollte, welches für die Formulierung der zu optimierenden Zielfunktion verwendet wurde. Somit sind die Methoden, die verschiedene Kriterien optimieren, quantitativ nicht vergleichbar.

Die weiteren Beispiele werden wir in folgender Form präsentieren. Für jedes Stereopaar geben wir zuerst seine „Besonderheiten“ an – die konkret für dieses Beispiel charakteristischen Eigenschaften, die einen deutlichen Einfluss auf die Wahl der Parameter haben. Dann werden die technischen Daten angegeben – die Größe der Bilder, die Werte der Parameter usw. Anschließend werden die Tiefenkarten und damit erzeugte Ansichten von verschiedenen Sichtpunkten gezeigt. Der Anschaulichkeit halber verbieten wir dabei die Entscheidung „Rückweisung“. Stattdessen geben wir die Karten der Streuungen an, anhand derer die Pixel markiert werden können, die mit der Entscheidung „Rückweisung“ zu bewerten sind (siehe (2.31)). Zusätzlich zeigen wir für jedes Beispiel auch die MAP-Entscheidung, um die Vorteile der Entscheidung mir der additiven quadratischen Kostenfunktion (als „Generierung“ bezeichnet) besser zu illustrieren.



a) MAP-Entscheidung b) Generierung c) Karte der Streuungen



d) Ansicht e) Ansicht

ABBILDUNG 4.25. Ergebnisse für das Beispiel „Raucher“

Als erstes möchten wir Beispiele präsentieren, die dem benutzten a-priori Modell der Oberfläche mehr oder weniger gut entsprechen – genau eine Oberfläche ohne Diskontinuitäten und Verdeckungen. Die Ergebnisse für das Beispiel „Raucher“ sind in der Abbildung 4.25 gezeigt. Die Besonderheiten dieses Beispiels sind:

- Die Bilder sind stark verrauscht.
- Die Effekte der Beleuchtung sind ziemlich stark. Vor allem sind das Reflexe auf den Augen und auf der Nase. Desweiteren weisen die Bilder eine deutliche Grauwertverschiebung auf.
- Das ursprüngliche Bildpaar war nicht rektifiziert. Da die geometrischen Parameter (die Lagen der Kameras, inneren Parameter der Kameras usw.) nicht bekannt waren, wurde die Rektifikation nur grob durchgeführt. Deswegen sind kleine y-Verschiebungen der „echten“ korrespondierenden Pixel (um 1-2 Zeilen) vorhanden.

– Der Rauch ist eigentlich keine Oberfläche (halbdurchsichtig). An der Stelle entspricht das benutzte Modell offensichtlich nicht der Realität. Das führt zu relativ großen Werten in den entsprechenden Positionen der Karte der Streuungen.

Die Parameter der Bilder und der Rekonstruktion sind:

Größe der Bilder:	340×295
Disparitätsbereich:	$[-23 \dots 9]$
Typ des Ähnlichkeitsmaßes:	Korrelationskoeffizient
Fenstergröße:	5×5
Anzahl der Diskretisierungsschritte:	
entlang Z-Achse:	33
entlang X-Achse:	290
Modell:	Scharfe Glattheitsforderung
Modellparameter:	$\delta = 1, l = 0.125$

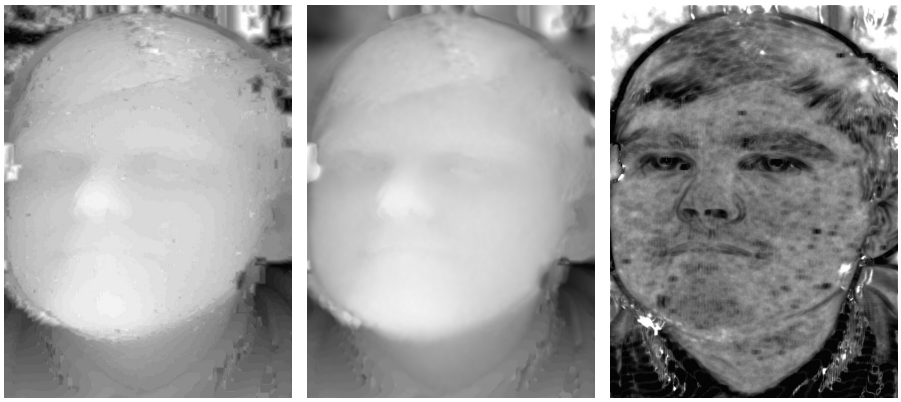
a) Linkes Bild b) Rechtes Bild

ABBILDUNG 4.26. Beispiel „Ivan“

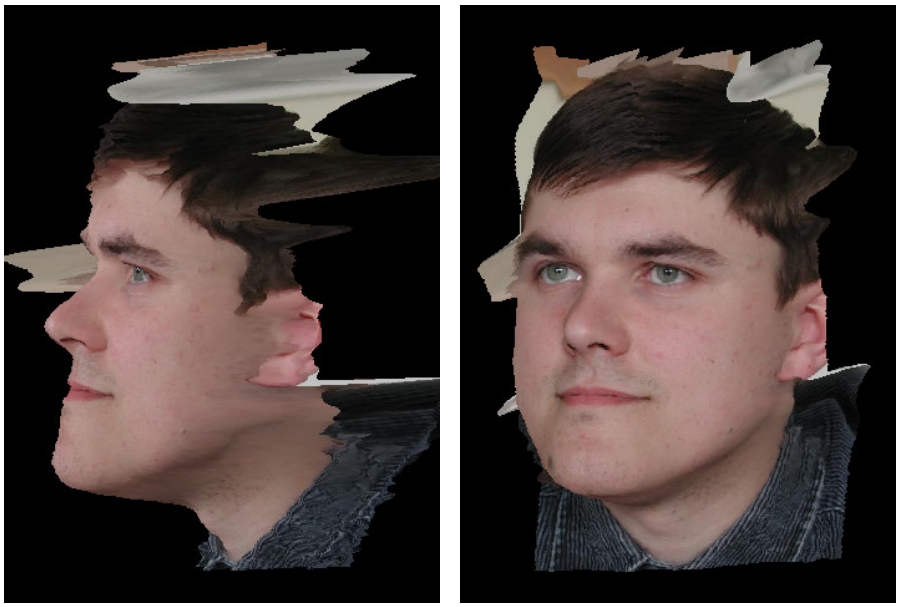
In der Abbildung 4.26 ist ein weiteres Stereopaar gezeigt. Die Ergebnisse sind in der Abbildung 4.27 angegeben. Die Besonderheiten dieses Beispiels sind:

- Der Hintergrund liegt sehr weit vom Objekt entfernt. Eine genaue gleichzeitige Rekonstruktion sowohl des Objektes als auch des Hintergrundes ist deswegen kaum möglich.
- Das rechte Ohr ist auf dem rechten Bild nicht zu sehen. An der Stelle entsteht ein für solche Situationen charakteristischer Fehler.
- Die Beleuchtungseffekte sind vorhanden, allerdings nicht sehr stark. Eine kleine Grauwertverschiebung ist zu bemerken. Die Reflexionen auf den Augen haben wegen ihrer Kleinheit keine bedeutende Wirkung auf die Ergebnisse.
- Die Bilder haben eine sehr gute Qualität. In solchen Fällen sind die MAP-Entscheidungen in der Regel auch sehr gut. Dies ermöglicht eine Wahl der Modellparameter, mit der sich relativ kleine Details der Oberfläche genau rekonstruieren lassen. Da die Bilder sehr fein aufgelöst sind, können dabei relativ große Fenster für die Berechnung des Ähnlichkeitsmaßes verwendet werden.

Die Parameter der Bilder und der Rekonstruktion sind:



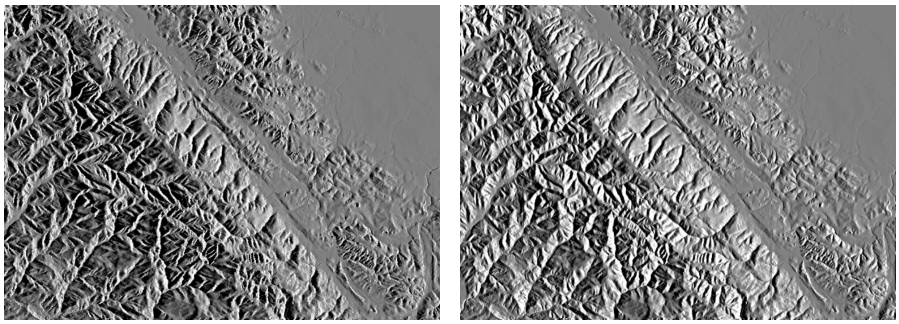
a) MAP-Entscheidung b) Generierung c) Karte der Streuungen



d) Ansicht e) Ansicht

ABBILDUNG 4.27. Ergebnisse für das Beispiel „Ivan“

Größe der Bilder:	745×498
Disparitätsbereich:	$[-30 \ldots 44]$
Typ des Ähnlichkeitsmaßes:	Quadratische Differenz mit Verschiebung
Fenstergröße:	7×7
Anzahl der Diskretisierungsschritte:	
entlang Z-Achse:	40
entlang X-Achse:	540
Modell:	Scharfe Glattheitsforderung
Modellparameter:	$\delta = 2, l = 1$



a) Linkes Bild b) Rechtes Bild

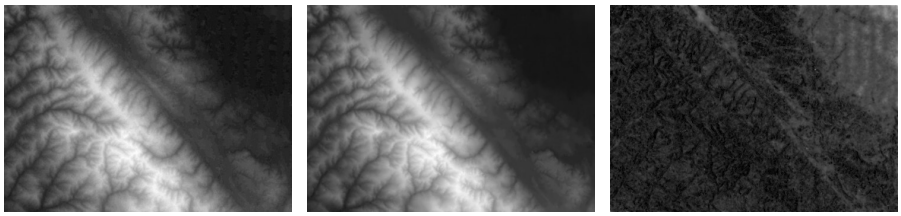
ABBILDUNG 4.28. Beispiel „Gebirge“

In der Abbildung 4.28 ist das nächste Stereopaar „Gebirge“ gezeigt (eine Luftaufnahme). Die Ergebnisse sind in der Abbildung 4.29 angegeben. Die Besonderheiten dieses Beispiels sind:

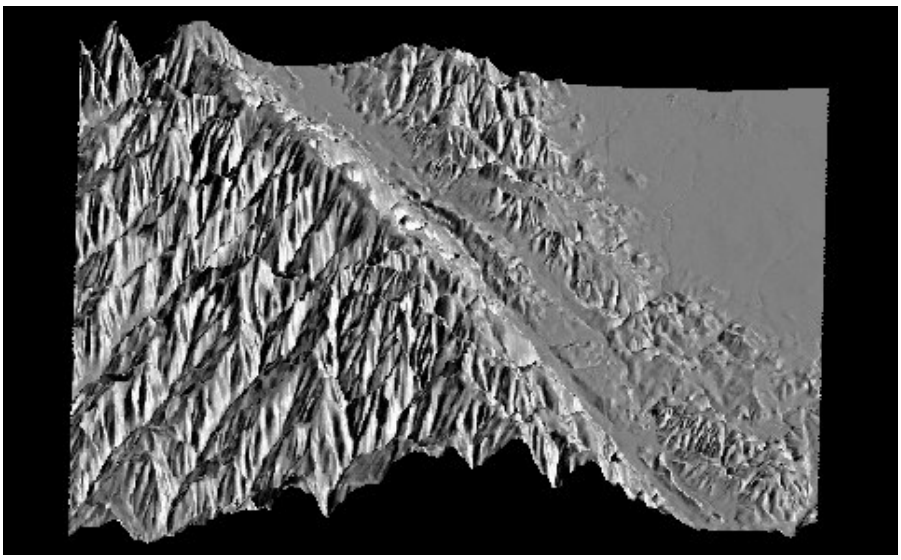
- Die Oberfläche entspricht dem benutzten Modell sehr gut, d.h. es sind keine Verdeckungen bzw. Diskontinuitäten vorhanden.
- Die Bilder weisen eine deutliche Grauwertsverschiebung auf.
- Die Bilder haben eine sehr gute Qualität (keine Störungen, keine Reflexe). Wie bereits erwähnt, sind die MAP-Entscheidungen in solchen Fällen in der Regel sehr gut. Für dieses Beispiel ist einen Unterschied zwischen der MAP-Entscheidung und den Ergebnissen der Generierung kaum zu sehen.

Die Parameter der Bilder und der Rekonstruktion sind:

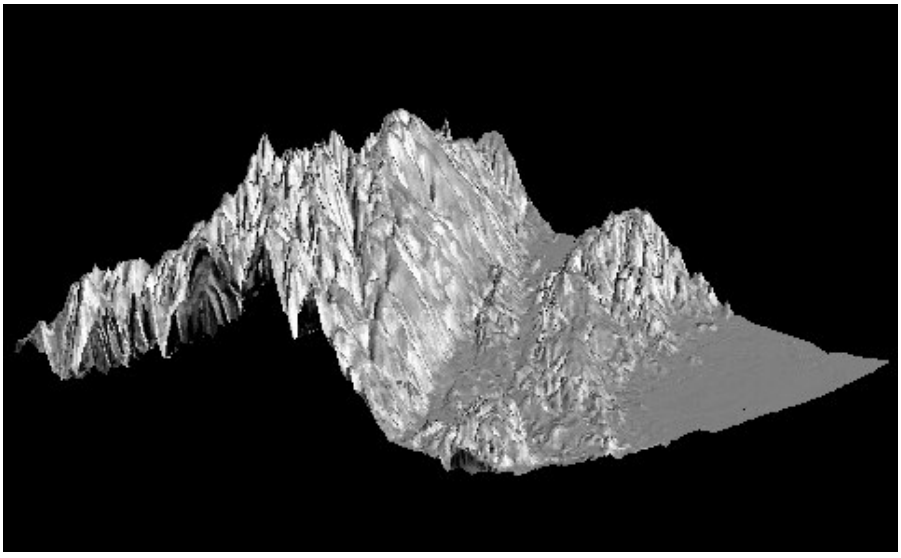
Größe der Bilder:	702×974
Disparitätsbereich:	$[-24 \dots 20]$
Typ des Ähnlichkeitsmaßes:	Quadratische Differenz mit Verschiebung
Fenstergröße:	3×3
Anzahl der Diskretisierungsschritte:	
entlang Z-Achse:	45
entlang X-Achse:	970
Modell:	Scharfe Glattheitsforderung
Modellparameter:	$\delta = 1, l = 1$



a) MAP-Entscheidung b) Generierung c) Karte der Streuungen

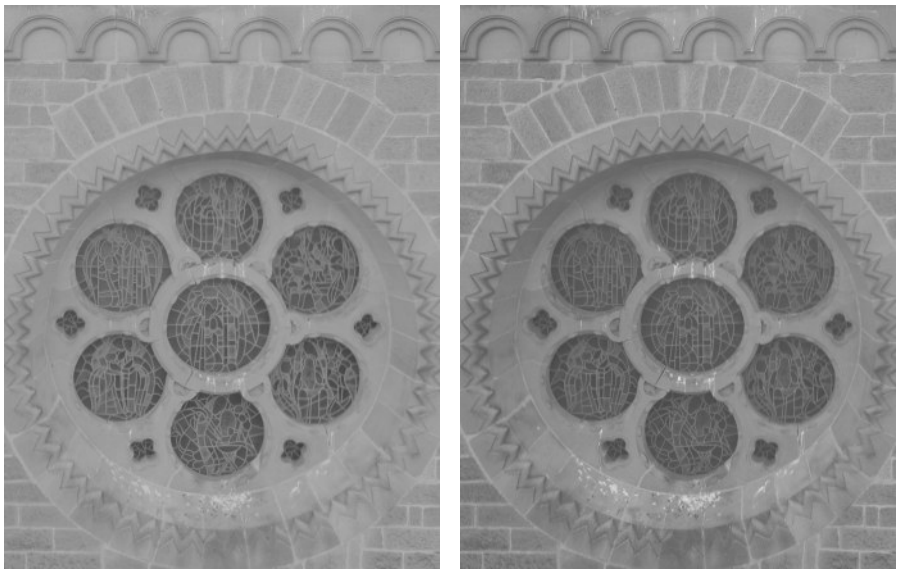


d) Ansicht



e) Ansicht

ABBILDUNG 4.29. Ergebnisse für das Beispiel „Gebirge“



a) Linkes Bild b) Rechtes Bild

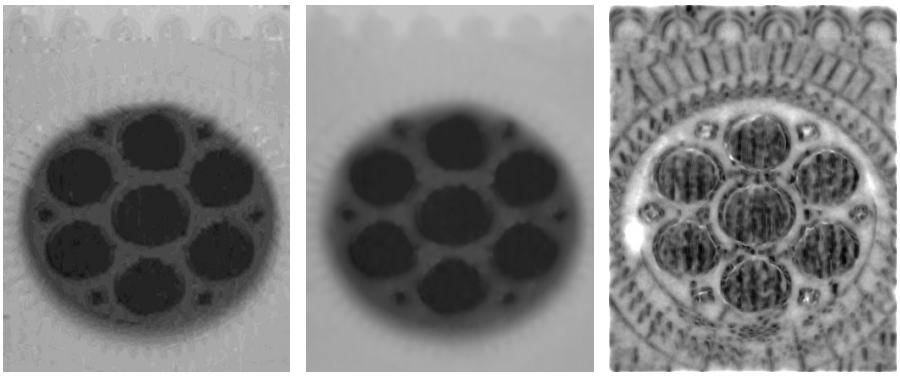
ABBILDUNG 4.30. Beispiel „Kirche“

In der Abbildung 4.30 ist das nächste Stereopaar „Kirche“ gezeigt. Die Ergebnisse sind in der Abbildung 4.31 angegeben. Die Besonderheiten dieses Beispiels sind:

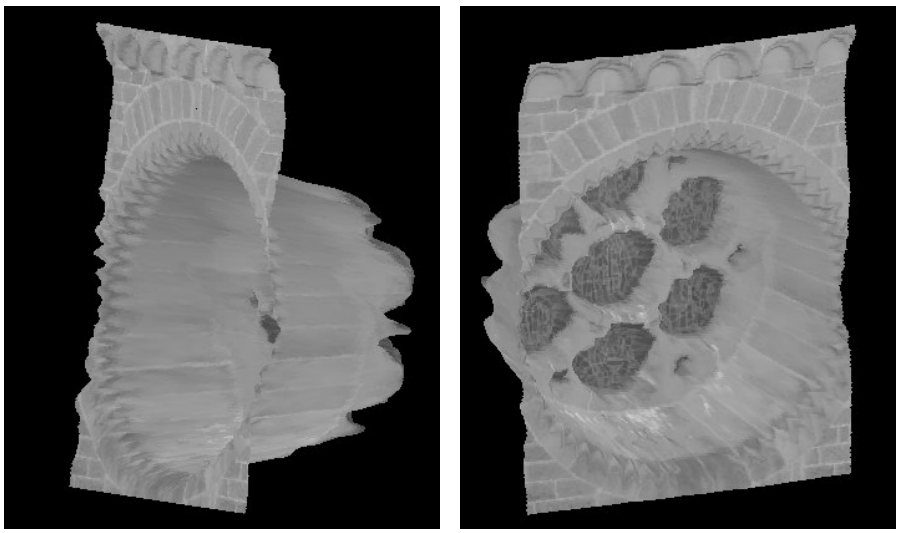
- Eine deutliche Grauwertverschiebung ist vorhanden.
- Die Bilder sind von ziemlich guter Qualität. Dadurch wird die MAP-Entscheidung auch gut.
- Die Szene enthält große Diskontinuitäten, allerdings (fast) keine Verdeckungen. Das verursacht die Wahl der Gausschen Glattheitsforderung als Modell.
- Die Szene ist gut texturiert, d.h. sie enthält keine homogen gefärbten Gebiete. Das ermöglicht eine sehr gute Rekonstruktion der zu den optischen Achsen der Kameras senkrechten Teile der Oberfläche.

Die Parameter der Bilder und der Rekonstruktion sind:

Größe der Bilder:	340×330
Disparitätsbereich:	$[-18 \dots 10]$
Typ des Ähnlichkeitsmaßes:	Quadratische Differenz mit Verschiebung
Fenstergröße:	5×5
Anzahl der Diskretisierungsschritte:	
entlang Z-Achse:	29
entlang X-Achse:	340
Modell:	Gaussche Glattheitsforderung
Modellparameter:	$\sigma_g = 2, l = 0.25$



a) MAP-Entscheidung b) Generierung c) Karte der Streuungen



d) Ansicht e) Ansicht

ABBILDUNG 4.31. Ergebnisse für das Beispiel „Kirche“

KAPITEL 5

Zusammenfassung und Ausblick

In diesem Kapitel fassen wir die Ergebnisse der Arbeit zusammen und diskutieren offene Fragen. Viele dieser Fragen entstehen als Folge der erhaltenen Resultate. Deswegen möchten wir die Zusammenfassung und die Diskussion parallel durchführen, indem wir die Ergebnisse gemeinsam mit den daraus entstehenden Fragen diskutieren. Zunächst betrachten wir die Fragen, die eher als „theoretische“ Fragen bezeichnet werden können. Anschließend diskutieren wir die Anwendbarkeit der beschriebenen Ansätze für praktische Anwendungen.

5.1. Theoretische Fragen

5.1.1. MinSum Probleme. Wie bereits gesagt, sind diese Probleme im allgemeinen Fall NP-vollständig. Andererseits, sind zur Zeit viele Spezialfälle bekannt, die exakt mit polynomialer Zeitkomplexität lösbar sind. Vor allem sind das die Fälle, in denen die Struktur des Basisgraphen einfach ist: Ketten, Zyklen oder, allgemeiner – partielle k-Bäume. Alle bekannten Algorithmen zur Lösung solcher Probleme basieren auf der dynamischen Optimierung. Diese Algorithmen lösen MinSum Probleme exakt für *bestimmte* Graphen und *beliebige* Funktionen g auf den Kanten des Graphen. Andererseits ist eine Klasse von Funktionen g bekannt (monotone Funktionen), für die MinSum Probleme exakt mit polynomialer Zeitkomplexität ohne Einschränkungen an den Basisgraphen lösbar sind. Die Algorithmen zur Lösung solcher Probleme basieren auf MinCut-MaxFlow Verfahren und lösen diese Probleme exakt für *bestimmte* Funktionen g und *beliebige* Basisgraphen. Der Zusammenhang zwischen diesen beiden Gruppen der Algorithmen ist zur Zeit noch nicht vollständig klar. Es ist somit interessant, ob sich ein Verfahren formulieren lässt, das Einschränkungen sowohl an Basisgraphen als auch an Funktionen g berücksichtigt.

Eine weitere interessante Frage ist die Folgende. Wie wir weiter oben gezeigt haben, ist eine *beliebige* MinSum Aufgabe in eine MinCut Aufgabe überführbar. Wenn die ursprüngliche MinSum Aufgabe nicht monoton ist, enthält die entstehende MinCut Aufgabe Kanten mit negativen Kosten. Wenn aber die ursprüngliche MinSum Aufgabe mit polynomialer Zeitkomplexität lösbar ist (zum Beispiel wenn der Basisgraph der Aufgabe eine Kette ist), ist die entstehende MinCut Aufgabe auch mit polynomialer Zeitkomplexität lösbar. Das heißt, dass eine Klasse von MinCut Aufgaben mit negativen Kantenkosten existiert, die mit polynomialer Zeitkomplexität lösbar sind. Es wäre interessant, diese Klasse explizit zu definieren.

Außerdem könnte man versuchen, die Klasse der lösbaren MinSum Probleme zu erweitern. Ein Schritt in diese Richtung ist in [**16**] gemacht. Hier betrachten die Autoren unter anderem MinSum Aufgaben, in denen die Funktionen g nicht nur für Paare von Knoten sondern auch für Tripel von Knoten definiert sind. In diesem Fall spricht man nicht mehr über Labelings eines Graphen, sondern über Labelings eines simplizialen Komplexes. In Termen der Wahrscheinlichkeitsmodelle entspricht dies der Suche nach der Maximum a-posteriori Entscheidung für Gibbs Wahrscheinlichkeitsverteilungen höherer Ordnungen,

als zwei. In dem erwähnten Artikel wird allerdings nur ein ziemlich eingeschränkter Fall betrachtet, nämlich wenn die Ordnung des Komplexes genau drei ist und die Anzahl der Zustände genau zwei ist.

Es ist relativ einfach zu zeigen, dass sich die in dieser Arbeit beschriebene „Abbildung von K zu 2“ auf simpliziale Komplexe beliebiger Ordnung verallgemeinern lässt. Somit ist es möglich, für beliebige Aufgabe der Ordnung L mit K Zuständen eine äquivalente Aufgabe derselber Ordnung L mit nur zwei Zuständen zu konstruieren. Somit bleibt nur die Frage offen, für welche Funktionen g MinSum Aufgaben mit zwei Zuständen auf simplizialen Komplexen beliebiger Ordnung mit polynomialer Zeitkomplexität lösbar sind.

5.1.2. SumProd Probleme. Im Gegensatz zu MinSum Problemen, ist zur Zeit keine Klasse von SumProd Aufgaben bekannt, die mit polynomialer Zeitkomplexität lösbar sind und (unserer Meinung nach) für praktische Anwendungen relevant sind. Die zur Zeit bekannten lösbaren Fälle sind:

- Aufgaben, in denen der Basisgraph einfach ist (k-Baum).
- Das Ising Modell. Bei diesem Modell ist der Basisgraph der Aufgabe beliebig. Die Anzahl der Zustände ist zwei. Die Funktionen q sind „nicht anwesend“, d.h. $q(\cdot) = 1$. Die Funktionen g haben die Form $\begin{array}{|c|c|}\hline \alpha & 1 \\ \hline 1 & \alpha \\ \hline \end{array}$ und sind für alle Kanten des Basisgraphen dieselbe.
- Der Basisgraph ist vollständig verbunden. Die Anzahl der Zustände ist beliebig. Die Funktionen q_r sind beliebig. Die Funktionen g sind für alle Kanten des Basisgraphen (für alle Knotenpaare) dieselbe.

In den für Bilderkennung geeigneten Modellen (zum Beispiel in dem in dieser Arbeit beschriebenen Modell oder auch in den für die Segmentierung geeigneten Modellen [**8, 14, 18**]) entspricht der Basisgraph in der Regel der Nachbarschaftstruktur der Pixel – zum Beispiel einem 2D- oder 3D-Gitter. Die Anzahl der Zustände kann dabei oft beliebig sein (zum Beispiel die Anzahl der Segmente bei Segmentierungsaufgaben). Meistens lassen sich die für solche Modelle entstehenden SumProd Aufgaben nicht auf die oben aufgelisteten lösbaren Probleme reduzieren.

Es ist leicht zu sehen, dass die Überführung „von K zu 2“ auch für SumProd Probleme definiert werden kann. Dabei entstehen Kanten der Typen $\begin{array}{|c|c|}\hline 0 & 1 \\ \hline 1 & 1 \\ \hline \end{array}$ oder $\begin{array}{|c|c|}\hline 1 & \alpha \\ \hline 1 & 1 \\ \hline \end{array}$. Somit kann man die Suche nach Klassen der lösbaren SumProd Aufgaben einschränken, so dass nur Aufgaben mit zwei Zuständen betrachtet werden, deren Funktionen g die oben angegebene Form haben.

Man soll bemerken, dass es für praktische Anwendungen manchmal gar nicht nötig ist, eine SumProd Aufgabe explizit zu lösen. Für gewisse Erkennungsaufgaben genügt es, nur die marginalen Wahrscheinlichkeitsverteilungen zu berechnen. Es kann sein, dass sich diese Verteilungen (für bestimmte Modelle) exakt berechnen lassen, ohne die entsprechenden SumProd Aufgaben explizit zu lösen.

Wie bereits gesagt, ist zur Zeit kein Verfahren bekannt, das es erlaubt, die benötigten marginalen Wahrscheinlichkeitsverteilungen (für relevante Modelle) exakt zu berechnen. In dieser Arbeit haben wir ein Algorithmus vorgeschlagen, der diese Verteilungen näherungsweise mit Hilfe des Gibbs Samplers ermittelt. Dabei bleibt die Frage der Konvergenz des Gibbs Samplers offen (siehe Abschnitt 3.4).

5.1.3. Nicht Bayessche strukturelle Ansätze. In dieser Arbeit haben wir eine praktische Anwendung als Aufgabe der Bayesschen Entscheidung formuliert. Dabei haben

wir als a-priori Modell Gibbssche Wahrscheinlichkeitsverteilungen benutzt, d.h. ein statistisches Modell. Es existieren allerdings auch andere Vorgehensweisen zur Suche einer Entscheidungsstrategie, nämlich nicht Bayessche Formulierungen. Dabei wird angenommen, dass das a-priori Modell des zu erkennenden Objektes in einem gewissen Sinn *nicht statistisch* formuliert ist. Einen ausführlichen Überblick solcher Probleme findet man in [**19**]. Hier wird der allgemeine Fall betrachtet, nämlich wenn die zu erkennenden Klassen eine allgemeine (nicht strukturierte) Menge bilden. Es wäre interessant, diese Methoden für Modelle zu verwenden, in denen die Menge der verborgenen Klassen eine zusätzliche Struktur besitzt, zum Beispiel wenn diese Menge eine (Teil-) Menge der Labelings eines Graphen ist. Obwohl sich nicht Bayessche Ansätze auch für diesen Fall formulieren lassen, ist es nicht klar, wie die entstehenden Optimierungsprobleme zu lösen sind.

5.1.4. Lernen. In dieser Arbeit wurde ein Verfahren zur Schätzung der Parameter des Wahrscheinlichkeitsmodells vorgeschlagen. Das Verfahren basiert auf dem Maximum Likelihood Prinzip. Zur Lösung des Maximum Likelihood Problems benutzen wir den EM-Algorithmus. Dabei entsteht die Frage, ob der EM-Algorithmus zum globalen Optimum konvergiert. Diese Frage kann etwas allgemeiner gestellt werden, nämlich, in welchen Fällen der EM-Algorithmus zum globalen Optimum konvergiert.

Andererseits, haben wir im Abschnitt 3.5 gezeigt, dass sich das gesamte Schema (das Maximum Likelihood Prinzip zusammen mit der nachfolgenden Erkennung) als eine einheitliche Erkennungsaufgabe formulieren lässt. Somit ist das Maximum Likelihood Prinzip ein Speziallfall der Aufgabe der Bayesschen Entscheidung (nämlich die Maximum a-posteriori Entscheidung bezüglich der Parameter). Im selben Abschnitt wurde ein „erweitertes“ Wahrscheinlichkeitsmodell konstruiert, in dem die gesuchten Parameter als zufällige Größen betrachtet werden. Ein weiterer Schritt würde darin bestehen, die Aufgabe des Lernens als Aufgabe der Bayesschen Entscheidung mit anderen Kostenfunktionen zu formulieren, die nicht zu der einfachsten MAP-Entscheidung führen. In diesem Zusammenhang haben wir bereits diskutiert, wie die Bayessche Strategie aussehen sollte, wenn die Kostenfunktion nicht von den Parametern abhängt. Es wäre interessant, auch andere Fälle zu untersuchen, nämlich Kostenfunktionen, die zwar von Parametern abhängen, aber nicht die einfachste Deltafunktion sind.

Ein deutlicher Nachteil der oben beschriebenen „erweiterten“ Modelle besteht darin, dass in der Regel keine vernünftige Annahme über die a-priori Wahrscheinlichkeitsverteilung der Parameter getroffen werden kann. Somit scheint es sinnvoll zu sein, die Aufgabe des Lernens als eine nicht Bayessche Aufgabe zu formulieren.

5.2. Praktische Anwendungen

In diesem Abschnitt diskutieren wir die Anwendbarkeit der beschriebenen Ansätze für praktischen Anwendungen.

5.2.1. Erweiterung des Modells für Stereorekonstruktion. In dieser Arbeit wurde ein sehr einfaches Modell zur Beschreibung der gesuchten Oberfläche formuliert. Eine mögliche Erweiterung kann in der Berücksichtigung der Verdeckungen bestehen (der Gebiete auf der Oberfläche, die nur von einer Kamera beobachtet werden). In [**15**] wurde ein solches Verfahren zur Modellierung von Verdeckungen vorgeschlagen. Die Aufgabe der Stereorekonstruktion wird dabei als Aufgabe der Energieminimierung formuliert. Aufgaben der Energieminimierung sind ihrerseits als MAP-Entscheidungen in entsprechenden Wahrscheinlichkeitsmodellen interpretierbar. Somit wäre es interessant, die Aufgabe der

Stereorekonstruktion als Aufgabe der Bayesschen Entscheidung für das (zum Beispiel) in [**15**] vorgeschlagene Modell mit einer additiven Kostenfunktion zu formulieren.

Eine weitere Verbesserung des Modells kann in der Berücksichtigung der Normalenvektoren der Oberfläche bestehen. Es kann gezeigt werden, dass sich solche Modelle mit Hilfe von Gibbsschen Wahrscheinlichkeitsverteilungen vierter Ordnung beschreiben lassen. Somit ist es möglich, eine entsprechende Aufgabe der Bayesschen Entscheidung zu formulieren. Zur näherungsweise Lösung dieser Aufgabe kann zum Beispiel der Gibbs Sampler benutzt werden. Allerdings entsteht dabei das folgende Problem. Um die Arbeit des Gibbs Samplers zu beschleunigen, wird ein „gutes“ initiales Labeling benötigt. Im Fall der Gibbsschen Modelle zweiter Ordnung benutzen wir dafür die MAP-Entscheidung. Für die Modelle höherer Ordnung ist jedoch nicht bekannt, wie die MAP-Entscheidung zu bestimmen ist. Ein möglicher Ausweg kann in der näherungsweisen Ermittelung der MAP-Entscheidung bestehen. Für solche Zwecke erscheinen uns die in [**17**] beschriebenen Ansätze vielversprechend.

5.2.2. Andere Anwendungen. Unserer Meinung nach lässt sich die strukturelle Erkennung im Allgemeinen dadurch charakterisieren, dass sich die Beschreibung eines komplizierten Objektes durch die Beschreibungen seiner Einzelteile und deren Zusammenhang darstellen lässt. Somit bestehen strukturellen Erkennungsaufgaben in der Suche nach einer komplexen *Interpretation* aller Bestandteile des zu erkennenden Objektes (einschließlich des gesamten Objektes selbst). Einige klassische Beispiele solcher Probleme sind Aufgaben der Theorie der formalen Sprachen (Grammatiken). In diesem Fall sind alle Teilmengen von Positionen (und somit auch das ganze Wort) zu interpretieren. Die Interpretation jedes solchen Teiles ist eine Name aus dem Alphabet der nicht terminalen Symbole. Dabei wird der Zusammenhang zwischen den Interpretationen verschiedener Teile durch die Regelmenge bestimmt. Streng genommen, können alle klassischen Constraint Satisfaction Probleme (wie zum Beispiel das Färbungsproblem, die SAT Probleme aus der Logik usw.) als Probleme der strukturellen Erkennung bezeichnet werden.

Manchmal wird zusätzlich gefordert, dass nicht irgendeine, sondern in einem gewissen Sinn beste Interpretation gesucht wird. Dann spricht man über strukturelle Optimierungsaufgaben. Einige Beispiele dafür sind Gewichtete Grammatiken, MaxSAT Probleme, MultiCut Probleme aus der Graphentheorie usw. Ist dabei ein Wahrscheinlichkeitsmodell auf der Menge der Interpretationen definiert, so entstehen Aufgaben der statistischen strukturellen Erkennung.

Die Labeling Probleme bilden eine wichtige Teilklasse aller strukturellen Probleme. Diese Probleme werden dadurch charakterisiert, dass in diesem Fall nur „elementare“ Bestandteile (zum Beispiel alle Pixel eines Bildes) des zu erkennenden komplizierten Objektes interpretiert werden sollen. Obwohl dies nur ein Speziallfall der strukturellen Erkennung ist, lassen sich viele Probleme als Labeling Aufgaben ausdrücken. Einige Beispiele davon sind Reguläre Sprachen, SAT Probleme, manche Aufgaben der Graphentheorie (zum Beispiel Graph- und Subgraph Isomorphismus, Finden der maximalen Clique), Traveling Salesman Problem und viele andere.

Viele angewandte Probleme der Bilderkennung lassen sich ebenfalls als Labeling Probleme formulieren. Ein Beispiel dafür sind Aufgaben, die sich zum Korrespondenzproblem reduzieren lassen (wie zum Beispiel die Stereorekonstruktion, Bewegungsanalyse und Verfolgung). In diesem Fall ist die Menge der zu interpretierenden elementaren Objekte die Menge der Pixel eines Bildes. Die Menge der Interpretationen eines Pixels (Zustandmenge oder Labelmenge) ist die Menge aller möglichen korrespondierenden Pixel im zweiten Bild. Im Fall der Stereo Rekonstruktion ist diese Menge geordnet – das ist die Menge

aller möglichen Disparitäten. Im Fall der Bewegungsanalyse ist diese Menge die Menge aller möglichen zweidimensionalen Verschiebungen. Somit entsteht eine teilgeordnete Zustandsmenge. Eine solche Formulierung findet man zum Beispiel in [**5**].

Ein weiteres Beispiel zur Formulierung eines angewandten Problems als Labeling Problem sind Aufgaben der Segmentierung. In diesem Fall ist jedes Pixel einem Segment zuzuordnen. Somit ist die Menge der Interpretationen die Menge aller Segmentnamen. Eine solche Formulierung im Kontext der Aufgabe der Textur Segmentierung findet man in [**18**].

An der Stelle möchten wir bemerken, dass in allen oben erwähnten Arbeiten die angewandten Aufgaben als MinSum (bzw. MaxSum) Probleme formuliert werden (als Aufgaben der Energieminimierung oder als Suche nach der Maximum a-posteriori Entscheidung). Somit entsteht die Frage, ob der Übergang zu einer „besseren“ (für jeweilige Anwendung am besten geeigneten) Kostenfunktion eine Verbesserung für die Ergebnisse bringt.

5.2.3. Einheitliche Modelle. In Literatur findet man viele Vorschläge zur Einbeziehung zusätzlicher so genannter „contextual“ Information für die Verbesserung der Endergebnisse. Allerdings reduzieren sich diese Ansätze sehr oft auf eine Vorverarbeitung der Ausgangsdaten. Zum Beispiel schlagen die Autoren in [**5, 15**] vor, die Ergebnisse eines Kantendetektors zu benutzen, um die Parameter des Modells (in dem Fall die Strafe für Diskontinuitäten) einzustellen. Das Modell dieser Einstellung ist aber nicht explizit definiert. Die meisten solcher Ansätze finden wir nicht genügend gut begründet.

Es scheint besser zu sein, solche zusätzliche Information als a-priori Kenntnisse über das zu rekonstruierende Objekt zu betrachten, d.h. diese Information bei der Formulierung des a-priori Modells zu berücksichtigen. Betrachten wir diese Idee anhand eines Beispiels. Nehmen wir an, dass eine Aufgabe der 3D-Rekonstruktion zu lösen ist. Dabei sei bekannt, dass die zu rekonstruierende Szene aus mehreren Objekten besteht und dass die Oberfläche jedes Objektes in einem gewissen Sinn stetig und glatt ist. Somit können Diskontinuitäten nur an den Objektgrenzen auftreten. Sei zusätzlich eine perfekte fehlerfreie Segmentierung der zu rekonstruierenden Szene gegeben. Dies ermöglicht offensichtlich eine bessere 3D-Rekonstruktion im Vergleich zum Fall, wenn keine Segmentierung zur Verfügung steht. Stellen wir uns jetzt die umgekehrte Situation vor. Die eigentliche Aufgabe ist die Aufgabe der Segmentierung. Zur Verfügung steht zusätzlich die fehlerfreie 3D-Rekonstruktion. Es ist anzunehmen, dass in diesem Fall möglich ist, die Ergebnisse der Segmentierung durch Einbeziehung der 3D-Daten zu verbessern. Somit entsteht die Frage, ob es möglich ist, beide Aufgaben im Rahmen eines gesamten Modells zu vereinigen, so dass sich die Aufgaben „gegenseitig unterstützen“. Offensichtlich ist dies oft möglich, insbesondere in dem Fall, wenn sich beide Aufgaben als Labeling Aufgaben mit derselben Mengen der „elementaren“ Objekte formulieren lassen. In diesem Fall kann die Menge der Interpretationen (Zustandsmenge) für ein elementares Objekt so konstruiert werden, dass sie alle möglichen Kombinationen der Interpretationen der einzelnen Modelle umfasst. Zum Beispiel ist diese Menge für die Vereinigung der 3D-Rekonstruktion und Segmentierung das Kreuzprodukt der Menge aller Disparitäten und der Menge aller Segmentnummern.

Unserer Meinung nach ermöglichen derartige Ansätze eine angemessene Modellierung, die es erlaubt, verschiedene Aspekte bei der Erkennung und Interpretation komplexer Objekte zu berücksichtigen.

Literaturverzeichnis

[1] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin, *Network flows: theory, algorithms, and applications*, Prentice-Hall, 1993.

[2] R.E. Bellman and S.E. Dreyfus, *Applied dynamic programming*, Princeton University Press, Princeton, New Jersey, 1962.

[3] J. Besag, *On the statistical analysis of dirty pictures (with discussion)*, Journal of the Royal Statistical Society, Series B **48(3)** (1986), 259–302.

[4] Stan Birchfield and Carlo Tomasi, *A pixel dissimilarity measure that is insensitive to image sampling*, IEEE Transactions on Pattern Analysis and Machine Intelligence **20** (1998), no. 4, 401–406.

[5] Y. Boykov, O. Veksler, and R. Zabih, *Fast approximate energy minimization via graph cuts*, ICCV, 1999, pp. 377–384.

[6] A. P. Dempster, N. M. Laird, and D. B. Durbin, *Maximum likelihood from incomplete data via the EM algorithm*, Journal of the Royal Statistical Society **39** (1977), 185–197.

[7] B. Flach, *Strukturelle Bilderkennung: Habilitationsschrift*, Technische Universität Dresden, 2003.

[8] B. Flach, D. Schlesinger, E. Kask, and A. Skulisch, *Unifying registration and segmentation for multi-sensor images*, DAGM 2002 (Luc Van Gool, ed.), LNCS 2449, Springer, 2002, pp. 190–197.

[9] L. Ford and D. Fulkerson, *Flows in networks*, Princeton University Press, 1962.

[10] Stuart Geman and Donald Geman, *Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images*, IEEE Transactions on Pattern Analysis and Machine Intelligence **6** (1984), no. 6, 721–741.

[11] G. Gimel'farb, *Stereo terrain reconstruction by dynamic programming*, Handbook of Computer Vision and Applications (B. Jaehne, H. Haussecker, and P. Giesser, eds.), vol. 2. Signal Processing and Pattern Recognition, Academic Press, 1999, pp. 505–530.

[12] Hiroshi Ishikawa and Davi Geiger, *Segmentation by grouping junctions*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1998.

[13] Xiaoyi Jiang and Horst Bunke, *Dreidimensionales computersehen*, Springer, 1997.

[14] E. Kask and S. Fuchs, *Ct-basierte 3d-analyse von baustoffen*, 5. Anwendungsbezogener Workshop zur Erfassung, Verarbeitung, Modellierung und Auswertung von 3D-Daten, Dezember 2002, pp. 35–42.

[15] Vladimir Kolmogorov and Ramin Zabih, *Computing visual correspondence with occlusions via graph cuts*, International Conference on Computer Vision, 2001, pp. 508–515.

[16] ———, *What energy functions can be minimized via graph cuts?*, ECCV 2002 (Berlin Heidelberg) (A. Heyden et al., ed.), LNCS, no. 2352, Springer-Verlag, 2002, pp. 65–81.

[17] I. Kovtun, *Partial optimal labeling search for a np-hard subclass of (max,+) problems*, Pattern Recognition (Gerald Krell Bernd Michaelis, ed.), LNCS, vol. 2781, Springer, September 2003, pp. 402–409.

[18] ———, *Texture segmentation of images on the basis of markov random fields*, Tech. report, TUD-FI03, May 2003.

[19] Schlesinger M.I. and Hlavác V., *Ten lectures on statistical and structural pattern recognition*, Kluwer Academic Publishers, Dordrecht, May 2002.

[20] James Gary Propp and David Bruce Wilson, *Exact sampling with coupled markov chains and applications to statistical mechanics*, Random Structures Algorithms **9** (1996), no. 1, 223–252.

[21] A. Rosenfeld, R.A. Hummel, and S.W. Zucker, *Scene labeling by relaxation operations*, IEEE Trans. SMC **6(6)** (1976), 420–433.

[22] S. Roy and I. Cox, *A maximum-flow formulation of the n-camera stereo correspondence problem*, International Conference on Computer Vision, 1998.

[23] R. Sara, *The class of stable matchings for computational stereo*, Tech. Report CTU-CMP-1999-22, Czech Technical University, 1999.

[24] M.I. Schlesinger, *Privates Gespräch*.

[25] ———, *Mathematical methods of image processing*, Naukova Dumka, Kiev, 1989.

[26] M.I. Schlesinger and B. Flach, *Some solvable subclasses of structural recognition problems*, Czech Pattern Recognition Workshop 2000 (Tomáš Svoboda, ed.), 2000, pp. 55–62.

[27] Michail I. Schlesinger, *Connection between unsupervised and supervised learning in pattern recognition*, Kibernetika **2** (1968), 81–88, In Russian.

HSSS AdminTools (c) 2001, last visited: Wed Oct 12 13:37:08 CEST 2005