Automatische Hinderniserkennung im Fahrenden Kraftfahrzeug

DISSERTATION
zur Erlangung des Doktorgrades
der Naturwissenschaften

vorgelegt am Fachbereich Physik
der Johann Wolfgang Goethe-Universität
zu Frankfurt am Main

von
Dirk Feiden

Frankfurt am Main
2002

Publikationsliste

1. Feiden, D.; Mühlich, M.; Mester, R.: *Robuste Bewegungsschätzung aus monokularen Bildsequenzen von planaren Welten.* In: W. Förstner, W. Buhmann (eds.): Mustererkennung 1999, 21. DAGM-Symposium, Bonn, September 1999, erschienen in der Reihe: Informatik aktuell, Springer Verlag, S. 349-356, 1999.

2. Trautwein, S.; Mühlich, M.; Feiden, D.; Mester, R.: *Estimating Consistent Motion From Three Views: An Alternative To Trifocal Analysis.* Konferenzband zur CAIP 1999 (Computer Analysis of Images and Patterns), Springer Lecture Notes on Computer Science LNCS, Ljubljana, Slowenien, 1999.

3. Feiden, D., Tetzlaff, R.: *Iterative Annealing: A New Efficient Optimization Method for Cellular Neural Networks*, in: Konferenzband zur ICIP 2001 (International Conference on Image Processing), S. 549-552, Thessaloniki, Griechenland, 2001.

4. Feiden, D., Tetzlaff, R.: *Feature Extraction in Motion Estimation with Cellular Neural Networks using Iterative Annealing*, Konferenzband zur ECCTD 2001 (European Conference on Circuit Theory and Design), Band III, S. 413-416, Helsinki, Finnland, 2001.

5. Schönmeyer, R., Feiden, D., Tetzlaff, R.: *Multi-Template Training for Image Processing with Cellular Neural Networks*, Konferenzband zur CNNA 2002 (International Workshop on Cellular Neural Networks and their Applications), Frankfurt, 2002.

6. Feiden, D., Tetzlaff, R.: *Displacement Vector Estimation with Cellular Neural Networks*, Konferenzband zur IJCNN 2002 (International Joint Conference on Neural Networks), Honolulu, USA, 2002.

7. Feiden, D., Tetzlaff, R.: *Obstacle Detection in Planar Worlds using Cellular Neural Networks*, Konferenzband zur CNNA 2002 (International Workshop on Cellular Neural Networks and their Applications), Frankfurt, 2002.

Danksagung

Ich möchte an dieser Stelle allen danken, die in direkter oder indirekter Weise an der Erstellung dieser Arbeit beteiligt waren.

Herrn Prof. Dr. Rudolf Mester danke ich für die Einführung in die Fragestellung.

Herrn HD. Dr. Ronald Tetzlaff gilt mein besonderer Dank für die Betreuung dieser Promotion. Neben seiner sachkundigen und richtungsweisenden Betreuung fand er in schwierigen Zeiten stets ermunternde Worte.

Teile dieser Arbeit entstanden aus einer Kooperation mit der Firma Robert Bosch GmbH, Abteilung FV/SLH in Hildesheim. Hier danke ich insbesondere Herrn Dr. Christoph Stiller und Herrn Dr. Werner Pöchmüller für die fachlichen Diskussionen und die Unterstützung der Forschungsarbeiten. Außerdem danke ich Herrn Dr. Just-Dietrich Büchs für die Bereitstellung der Video-Daten und die Genehmigung, Teile der Arbeitsberichte *XC-RB-A-03*, *XC-RB-A-05* und *XC-RB-A-06* für diese Dissertationsschrift zu verwenden.

Weiterhin danke ich Herrn Prof. Dr. Dietrich Wolf und Herrn Prof. Dr. Arild Lacroix, die mir stets mit Rat und Tat zur Seite standen.

Herrn Ralf Schönmeyer danke ich für viele Anregungen und konstruktive Hinweise.

Weiterer Dank gilt allen Mitgliedern der Arbeitsgruppe für das ausgesprochen angenehme Arbeitsklima.

Ich danke außerdem der Graduiertenförderung der Universität Frankfurt für die finanzielle Unterstützung dieser Arbeit.

Mein besonderer Dank gilt schließlich meiner Frau Veronika. Ihre moralische Unterstützung trug ebenfalls sehr viel zum Gelingen dieser Arbeit bei.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die automatische Erkennung von visuell erfaßbaren Umweltsituationen ist ein wichtiges Teilgebiet der physikalischen Meßtechnik. Dazu zählt beispielsweise die autonome Steuerung von Fahrzeugen, die aufgrund ihrer sicherheitsrelevanten Bedeutung im Zentrum zahlreicher aktueller Forschungsarbeiten steht. Die autonome Navigation von Fahrzeugen und Robotern ist in besonderem Maße auf eine zuverlässige Umgebungsanalyse angewiesen, weshalb die Umgebung üblicherweise mittels technischer Sensoren aufgenommen und anschließend zu einer mathematischen Repräsentation der Umwelt weiterverarbeitet wird. In Anlehnung an die visuellen Fähigkeiten des Menschen, werden in vielen Problemstellungen der physikalischen Meßtechnik, *Videosensoren* für die Abbildung der Umgebung verwendet. Eine sehr wichtige Anwendung von aktuellem Interesse ist es, aus diesen Videodaten potentielle Hindernisse für ein Fahrzeug zu ermitteln. Eine solche Hinderniserkennung würde dann zu unterschiedlichen Reaktionen in Bezug auf die Fahrzeugsteuerung führen, so daß Kollisionen vermieden würden.

Die Erkennung von Hindernissen in Videosequenzen kann beispielsweise über eine Eigenbewegungsschätzung realisiert werden. Die Lösung des Problems der *Bewegungsschätzung* aus Stereo-Aufnahmen ist seit langem bekannt (z. B. [YC90]), allerdings wurde das Problem der Bewegungsschätzung und der Hinderniserkennung in *monokularen Bildsequenzen* bisher noch nicht in allgemeingültiger Weise gelöst. Der Ausdruck „monokulare Bildsequenzen“ bezeichnet Videosequenzen, die von einer *einzigen* Kamera aufgenommen wurden. Im Gegensatz zu monokularen Bildsequenzen werden für die Aufnahme von Stereosequenzen zwei Kameras benötigt, was zwar einerseits die Bewegungsschätzung erleichtert, aber andererseits zu einer Verdopplung des Datenvolumens führt, nach einer aufwändigen Kalibrierung verlangt und für reale Anwendungen praktische Nachteile mit sich bringt. Wäre also unter Verwendung von monokularen Sequenzen eine ebenso genaue Bewegungsschätzung möglich wie unter Verwendung von Stereosequenzen, so würde dies zu einer deutlichen Reduzierung des meßtechnischen Aufwandes bei realen Anwendungen führen und die Realisierung komplexer Verfahren für intelligente Sensoren erlauben. Im Vergleich zu Stereoansätzen führt die Verwendung von monokularen Bildsequenzen jedoch zu einer beträchtlichen Verschärfung des Problems der Bewegungsschätzung, insbesondere dann, wenn die Blickrichtung der Kamera und die Bewegungsrichtung nahezu identisch sind. In diesem Fall schneiden sich die Sehstrahlen, die zu gleichen Objektpunkten gehören, unter einem extrem spitzen Winkel und somit stellt die Gewinnung von Tiefeninformationen ein schlecht konditioniertes Problem dar. Die wissenschaftliche Herausforderung besteht nun darin, zu zeigen, daß die Aufgabenstellungen der Stereosequenzverarbeitung, wie zum Beispiel Eigenbewegungsschätzung oder Hinderniserkennung, in gleichem Maße durch die Verarbeitung von monokularen Bildsequenzen gelöst werden können.

In dieser Arbeit sollte das Problem der Eigenbewegungsschätzung eines Kraftfahrzeugs unter Verwendung von monokularen Bildaufnahmen betrachtet werden. Die Bildsequenz[1] in Bild 1.1 zeigt exemplarisch einige Bilder einer Verkehrsszene, die aus einem Fahrzeug heraus in Fahrtrichtung aufgenommen wurde.

Bild 1.1: Beispielbilder aus einer monokularen Bildsequenz

Das Ziel dieser Arbeit sollte die Implementierung eines Verfahrens sein, welches aus diesem Ausgangsbildmaterial die Schätzung der Eigenbewegung des Fahrzeugs erlaubt, so daß in einer anschließenden Hinderniserkennung potentielle Hindernisse, wie z.B. andere Verkehrsteilnehmer extrahiert werden können. Hierzu sollten im Rahmen dieser Arbeit zwei grundlegend verschiedene Wege zur Hinderniserkennung in monokularen Bildsequenzen untersucht und verglichen werden. Auf der einen Seite sollte ein Ansatz entwickelt werden, welcher auf statistischen Methoden mit einem zweidimensionalen Modell beruht und auf der anderen Seite sollte ein neuer Weg gefunden werden, der die *Künstlichen Neuronalen Netze* zugrunde legt und insbesondere die in den letzten Jahren verstärkt diskutierten *Zellularen Neuronalen Netze* verwendet.

[1] mit freundlicher Genehmigung der Robert Bosch GmbH

Teil I

Bewegungsschätzung mit statistischen Methoden

Kapitel 2

Einführung

In den folgenden Kapiteln wird eine neue Methode zur Schätzung der Eigenbewegung und Position eines bewegten Fahrzeugs auf der Basis von monokularen Bildsequenzen vorgestellt. Die statistische Schätzung von Bewegung aus monokularen Bildsequenzen wurde bisher noch nicht in allgemeingültiger Weise gelöst. Der Hauptgrund hierfür ist, daß bei der monokularen Bildfolgenanalyse mehrere miteinander gekoppelte schlecht konditionierte Probleme zu lösen sind. Insbesondere dann, wenn die Blickrichtung der Kamera und die Bewegungsrichtung nahezu identisch sind, führt dies zu einer beträchtlichen Verschärfung des Problems, da sich dabei die Sehstrahlen, die zu gleichen Objektpunkten gehören, unter einem extrem spitzen Winkel schneiden. Eine mögliche Verbesserung besteht darin, Bilder mit sehr hoher örtlicher Auflösung zu verwenden, was aber in Konflikt zum Ziel der Echtzeitverarbeitung steht. Für eine Echtzeitverarbeitung von $10 - 25$ Bildern/s und einer begrenzten Rechenleistung (Pentium3, 500 MHZ) können nur sehr kleine Sequenzbilder für die Schätzung verwendet werden. Deshalb kamen bei den vorliegenden Untersuchungen Verkehrsaufnahmen mit einer Auflösung von 140×174 Bildpunkten zum Einsatz. Erfordert also die Echtzeitfähigkeit der zu entwickelnden Verfahren die Verwendung von Bildern mit einer niedrigen Ortsauflösung, so kommt der *Robustheit* der verwendeten Schätzmethoden eine besondere Bedeutung zu.

Dem hier beschriebenen Ansatz zur Bewegungsschätzung liegt die Annahme zugrunde, daß sich die meisten Punkte im dreidimensionalen Raum im wesentlichen auf einer Ebene befinden. Insbesondere für Verkehrsszenen ist diese Annahme sinnvoll, da die Straßen, auf denen sich Kraftfahrzeuge bewegen, im wesentlichen planar sind. Im folgenden wird dieses Modell kurz als „*planares Weltmodell*“ bezeichnet. Offensichtlich stellt das „planare Weltmodell“ nur eine Approximation an die Realität dar, da einerseits eine reale Straße keine exakte Ebene darstellt und andererseits dreidimensionale Objekte, wie z.B. Bäume und Fahrzeuge, in Verkehrsszenen auftauchen. Dies tritt als Modellabweichung in Erscheinung, was aber durch den Einsatz robuster Schätzstatistik deutlich reduziert werden kann. Ein besonderer Vorzug des planaren Weltmodells besteht darin, daß zusammen mit den geschätzten Bewegungsparametern eine *Hinderniserkennung* anhand von bewegungskompensierten Differenzbildern realisiert werden kann [CE90]. Eine solche Hinderniserkennung kann anschließend in einem System zur automatischen Führung von Kraftfahrzeugen zum Einsatz kommen.

In Bild 2.1 ist das Verfahren der statistischen Hinderniserkennung aus monokularen Bildsequenzen dargestellt, welches in den folgenden Kapiteln genauer vorgestellt wird. Das Kapitel 3 beschreibt den ersten Schritt der statistischen Hinderniserkennung, welcher die Suche nach markanten Bildbereichen darstellt. Anschließend wird in Kapitel 4 die Verschiebung der er-

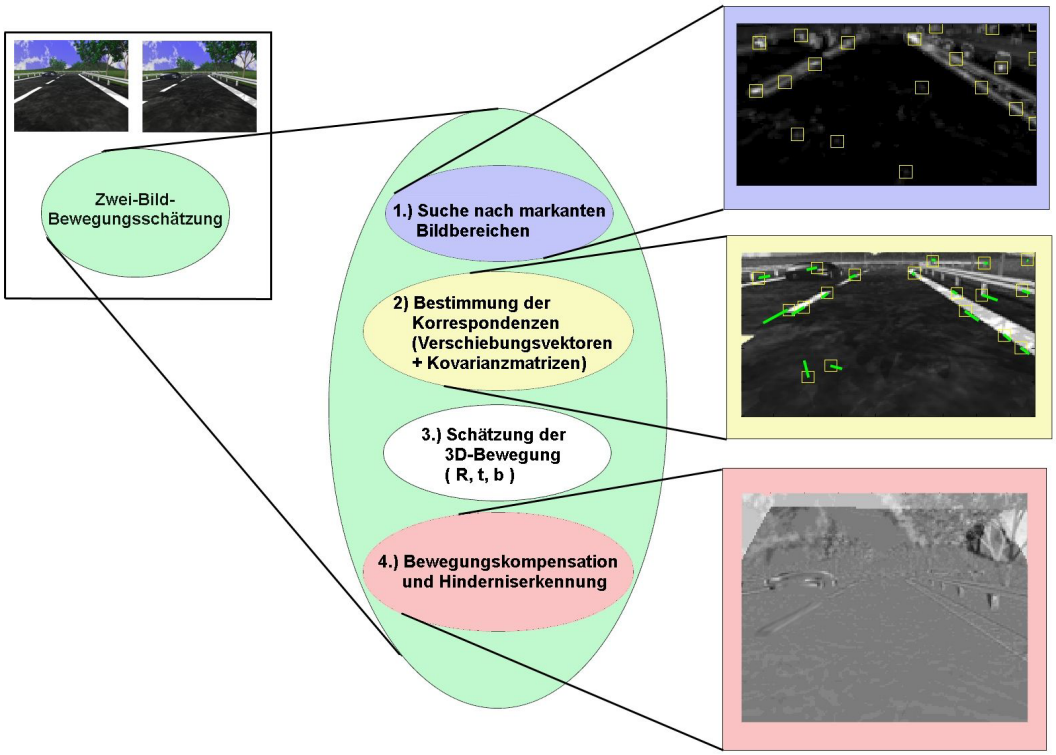


Bild 2.1: Das Verfahren der statistischen Hinderniserkennung aus monokularen Bildsequenzen

mittelten Bildbereiche im zeitlich darauffolgenden Bild der Sequenz mittels einer sogenannten *Verschiebungsvektorschätzung* beschrieben. In den Kapiteln 5 bis 7 wird die so bestimmte Menge an Verschiebungsvektoren anschließend dazu verwendet, unter Einbeziehung des planaren Weltmodells die dreidimensionalen Bewegungsparameter und den Normalenvektor der Ebene zu bestimmen. Schließlich wird in Abschnitt 5.4 anhand der ermittelten Parameterwerte eine Bewegungskompensation durchgeführt, so daß das Resultat ein bewegungskompensiertes Bild darstellt, in dem nur noch dreidimensionale Objekte sichtbar sind.

Kapitel 3

Statistische Generierung von markanten Bildbereichen

3.1 Einleitung

Wie bereits in Kapitel 2 erläutert, basiert die hier vorgestellte dreidimensionale Bewegungsschätzung auf der Verarbeitung von Verschiebungsvektoren. Die Bestimmung von Verschiebungsvektoren ist jedoch ein rechenaufwendiges Problem. Deshalb ist es im Hinblick auf die Echtzeitfähigkeit erforderlich, den Zuordnungsprozeß nur auf diejenigen Bildregionen zu beschränken, in denen Korrespondenzen mit genügend hoher Konfidenz und Genauigkeit bestimmt werden können. Im folgenden werden solche bevorzugten Bildpositionen mit dem Ausdruck *markante Bildbereiche* bezeichnet. In diesem Kapitel wird eine Methode zur Auffindung von markanten Bildbereichen vorgestellt, die im wesentlichen auf den Prinzipien des FÖRSTNER-Operators [Foe91] basiert, also das Eigensystem der Kovarianzmatrix der Grauwertgradienten in einem lokalen Bereich auswertet. In dem entwickelten Verfahren zur Ermittlung markanter Bildbereiche wird einerseits die Textur verwendet, welche die Varianz des Grauwertgradienten angibt und andererseits das von FÖRSTNER entwickelte Isotropiemaß, welches sich in der Praxis ([Si98], [Ro01], [BA98]) bereits vielfach bewährt hat. Unter Verwendung von diesem Operator können sowohl punkt- als auch linienförmige Strukturen, die für die Verschiebungsvektorschätzung besonders geeignet sind, detektiert werden. Hierbei erhalten Ecken im Gegensatz zu Kanten eine größere „Markantheit“, da sie für die Bewegungsschätzung eine wichtigere Rolle spielen.

3.2 Mathematischer Hintergrund

In diesem Abschnitt wird nun präzisiert, was mathematisch unter einem markanten Bildbereich zu verstehen ist. Die hier vorgestellte Methode basiert auf der Grundidee, den Grauwertgradienten in einer begrenzten Bildregion zu schätzen und anschließend das Eigensystem von dessen Kovarianzmatrix auszuwerten.

Die Grauwertfunktion $s(\vec{x})$ eines Bildes ist eine räumlich quantisierte zweidimensionale Funktion mit dem diskreten Wertebereich $0 \ldots 255$. Werden *beliebige* Bilder betrachtet, so stellt die Grauwertfunktion $s(\vec{x})$ eine Realisation eines zweidimensionalen skalaren Zufallsprozesses dar.

Somit ist der Grauwertgradient an einer Bildstelle $\vec{x}$,

$$\vec{g}(\vec{x}) = \begin{pmatrix} g_x(\vec{x}) \\ g_y(\vec{x}) \end{pmatrix} = \vec{\nabla} s(\vec{x}) = \begin{pmatrix} \partial s(\vec{x})/\partial x \\ \partial s(\vec{x})/\partial y \end{pmatrix}, \tag{3.1}$$

ebenfalls eine Realisation eines vektoriellen Zufallsprozesses. Die Kovarianzmatrix dieses Grauwertgradienten $\vec{g}$ ist dann gegeben durch

$$\mathsf{Cov}\left[\vec{g}\right] = \mathsf{E}\left[(\vec{g} - \mathsf{E}\left[\vec{g}\right]) \cdot (\vec{g} - \mathsf{E}\left[\vec{g}\right])^T\right]. \tag{3.2}$$

Unter der Annahme, daß die Erwartungswerte der partiellen Ableitungen Null sind[1], also

$$\left(\mathsf{E}\left[g_x\right] = \mathsf{E}\left[g_y\right] = 0\right), \tag{3.3}$$

ergibt sich die Kovarianzmatrix von $\vec{g}$ zu

$$\mathsf{Cov}\left[\vec{g}\right] = \mathsf{E}\left[\vec{g} \cdot \vec{g}^T\right] = \begin{pmatrix} \mathsf{E}\left[g_x \cdot g_x\right] & \mathsf{E}\left[g_x \cdot g_y\right] \\ \mathsf{E}\left[g_y \cdot g_x\right] & \mathsf{E}\left[g_y \cdot g_y\right] \end{pmatrix}. \tag{3.4}$$

Da die Grauwertfunktion eines Bildes eine quantisierte Funktion darstellt, ist die Kovarianzmatrix $\mathsf{Cov}\left[\vec{g}\right]$ eine Größe, die hier nicht exakt bestimmt werden kann. Es kann lediglich eine Schätzung der Kovarianzmatrix erfolgen. Sind N Realisierungen der Grauwertgradienten bekannt, so kann die Kovarianzmatrix geschätzt werden zu

$$\widehat{\mathbf{C}}_{\mathbf{g}} := \frac{1}{N} \begin{pmatrix} \sum_{i=1}^{N} g_{x_i} g_{x_i} & \sum_{i=1}^{N} g_{x_i} g_{y_i} \\ \sum_{i=1}^{N} g_{y_i} g_{x_i} & \sum_{i=1}^{N} g_{y_i} g_{y_i} \end{pmatrix}, \tag{3.5}$$

wobei N die Anzahl der Bildpunkte angibt und $g_{x_i} = \frac{\partial s(\vec{x})}{\partial x} \mid_{x_i}$.

Aus der so erhaltenen Kovarianzmatrix $\widehat{\mathbf{C}}_{\mathbf{g}}$ können nun Informationen über die Texturierung eines zu betrachtenden Bildes gewonnen werden. Unter anderem kann ein *Texturmaß* und ein *Anisotropiemaß* damit berechnet werden.

3.2.1 Anisotropiemaß

Das sogenannte Anisotropiemaß gibt an, ob der zu untersuchende Bildbereich eine Vorzugsrichtung besitzt oder ob er isotrop ist. Hier wird ein Isotropiemaß verwendet, welches von Wolfgang Förstner im Rahmen seiner Habilitation [Foe91] entwickelt wurde und im folgenden vorgestellt wird.

Es seien $\lambda_i = \{\lambda_1, \lambda_2\}$ die Eigenwerte der 2×2 - Kovarianzmatrix $\widehat{\mathbf{C}}_{\mathbf{g}}$ mit $\lambda_1 \geq \lambda_2$. Die Anisotropie a des stochastischen Prozesses kann dann über das Verhältnis der Eigenwerte λ_i

$$a := \frac{\lambda_1}{\lambda_2} \tag{3.6}$$

[1] Diese Annahme ist sinnvoll, da die hier entwickelten Formeln für jedes Bildmaterial gültig sein sollen und wenn über alle möglichen Bilder gemittelt wird, so kann erwartet werden, daß im Mittel kein Gradient erhalten wird.

erfaßt werden, da stark unterschiedliche Eigenwerte eine Vorzugsrichtung der Struktur wiederspiegeln[2]. Das Anisotropiemaß a wird dann in die monotone Funktion p der Form

$$p := \left(\frac{a-1}{a+1}\right)^2 = \left(\frac{\frac{\lambda_1-\lambda_2}{\lambda_2}}{\frac{\lambda_1+\lambda_2}{\lambda_2}}\right)^2 = \left(\frac{\lambda_1-\lambda_2}{\lambda_1+\lambda_2}\right)^2 \tag{3.7}$$

überführt. Dieses Maß hat nun den Vorteil einer Normierung, so daß es zwischen 0 (isotrop) und 1 (anisotrop) liegt. Unter Verwendung der Beziehungen

$$\mathsf{Tr}\left\{\widehat{\mathbf{C}}_\mathbf{g}\right\} = \sum_i \lambda_i = \lambda_1 + \lambda_2 \quad \text{und} \tag{3.8}$$

$$\det \widehat{\mathbf{C}}_\mathbf{g} = \prod_i \lambda_i = \lambda_1 \cdot \lambda_2 \tag{3.9}$$

mit

$$\widehat{\mathbf{C}}_\mathbf{g} = \frac{1}{N}\begin{pmatrix} \sum_{i=1}^N g_{x_i} g_{x_i} & \sum_{i=1}^N g_{x_i} g_{y_i} \\ \sum_{i=1}^N g_{y_i} g_{x_i} & \sum_{i=1}^N g_{y_i} g_{y_i} \end{pmatrix} =: \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} \text{ wobei } c_{12} = c_{21}, \tag{3.10}$$

läßt sich das Anisotropiemaß p auch ohne explizite Berechnung der Eigenwerte bestimmen:

$$\begin{aligned}
(\lambda_1-\lambda_2)^2 &= \lambda_1^2 + 2\lambda_1\lambda_2 + \lambda_2^2 - 4\lambda_1\lambda_2 = (\lambda_1+\lambda_2)^2 - 4\lambda_1\lambda_2 \\
&= (\mathsf{Tr}\left\{\widehat{\mathbf{C}}_\mathbf{g}\right\})^2 - 4\det\widehat{\mathbf{C}}_\mathbf{g} \\
\Rightarrow p &= \frac{(\mathsf{Tr}\left\{\widehat{\mathbf{C}}_\mathbf{g}\right\})^2 - 4\det\widehat{\mathbf{C}}_\mathbf{g}}{(\mathsf{Tr}\left\{\widehat{\mathbf{C}}_\mathbf{g}\right\})^2} = 1 - \frac{4\det\widehat{\mathbf{C}}_\mathbf{g}}{(\mathsf{Tr}\left\{\widehat{\mathbf{C}}_\mathbf{g}\right\})^2} \\
&= 1 - \frac{4\cdot(c_{11}\cdot c_{22} - c_{12}^2)}{(c_{11}+c_{22})^2} = \frac{(c_{11}+c_{22})^2 - 4\cdot(c_{11}\cdot c_{22} - c_{12}^2)}{(c_{11}+c_{22})^2} \\
\Rightarrow p &= \frac{(c_{11}-c_{22})^2 + 4c_{12}^2}{(c_{11}+c_{22})^2}
\end{aligned} \tag{3.11}$$

Als Isotropiemaß q kann $q = 1 - p$ verwendet werden. Steckt in einem Bildausschnitt keine Richtungsinformation, so ist die Isotropie dieses Bildausschnittes $q = 1$ und eine stark gerichtete Struktur besitzt entsprechend die Isotropie $q \approx 0$.

3.2.2 Textur-Stärke

Die Texturstärke eines Bildbereichs τ wird üblicherweise [Go94] über den Erwartungswert der quadratischen Norm des Grauwertgradienten

$$\tau := \mathsf{E}\left[|\vec{g}|^2\right]$$

definiert. Da jedoch nur N Realisierungen des Grauwertgradienten bekannt sind, ergibt sich folgender Ansatz zur Schätzung der Texturstärke,

$$\hat{\tau} = \frac{1}{N}\left(\sum_{i=1}^N g_{x_i} g_{x_i} + \sum_{i=1}^N g_{y_i} g_{y_i}\right) = \mathsf{Tr}\left\{\widehat{\mathbf{C}}_\mathbf{g}\right\}.$$

[2]Die Eigenwerte der Kovarianzmatrix sind proportional zu der Länge der Hauptachsen der Kovarianzmatrix, so daß unterschiedliche Eigenwerte auf eine Anisotropie des Bildinhaltes hinweisen.

Die Schätzung der Texturstärke entspricht also der Spur der Kovarianzmatrix $\widehat{\mathbf{C}}_{\mathbf{g}}$. Eine hohe Texturstärke bedeutet folglich, daß der Betrag der Grauwertgradienten im Mittel groß ist.

3.2.3 Definition einer skalaren Maßzahl für die Markantheit von Punkten

Das in Abschnitt 3.2.1 eingeführte Isotropiemaß liefert eine Aussage über den Grad der Ausrichtung eines Bildausschnitts. Kommen in einem Bildbereich viele ähnliche Gradienten vor, so liegt eine *Objektkante* vor, was zu einer sehr niedrigen Isotropie führt. Dagegen besitzen Bildbereiche, die nur einen einzigen Grauwert oder ein weißes Rauschen aufweisen, eine hohe Isotropie. Ein weiteres Beispiel für eine hohe Isotropie bilden *Objektecken*, denn die entsprechenden Bildbereiche enthalten zwei senkrecht zueinander stehende Grauwertgradienten. Mit Hilfe des Isotropiemaßes kann also eine Unterscheidung von Ecken und Kanten erfolgen. Somit ist es sinnvoll, die Isotropie mit der Texturstärke zu verknüpfen, so daß nur solche Bildpunkte als markant angesehen werden, bei denen auch eine offensichtliche Textur vorliegt. Dies kann erreicht werden, indem Isotropie und Texturstärke miteinander multipliziert werden, also

$$\text{Markantheit } \hat{M} := \hat{\tau} \cdot q. \tag{3.12}$$

Linienartige Strukturen, welche ebenso wichtige Informationen enthalten wie Objektecken, bleiben jedoch bei dieser Vorgehensweise unberücksichtigt, denn bei ihnen ist die Isotropie $q \approx 0$. Um Informationen aus linienartigen Strukturen ebenfalls zu nutzen, wird hier die erweiterte Beziehung

$$\begin{aligned} \text{Markantheit} &:= \text{Texturstärke} + \text{Texturstärke} \cdot \text{Isotropie, also} \\ M &:= \hat{\tau} + \hat{\tau} \cdot q \end{aligned} \tag{3.13}$$

zur Bestimmung des Martkantheitsbildes verwendet. Auf diese Weise treten im Markantheitsbild Ecken stärker in Erscheinung als Kanten. Mit der gleichen Gewichtung der beiden Summanden in Gleichung (3.13) ergibt sich eine doppelt so hohe Markantheit für Objektecken im Vergleich zu Objektkanten, da Objektkanten nur einen Beitrag vom zweiten Summanden erhalten. Dies ist von besonderem Vorteil für die anschließende Verschiebungsvektorschätzung, da in erster Linie Objektecken für die Bildkorrespondenzen herangezogen werden sollten; sie weisen, wie in Kapitel 4 beschrieben, eine höhere Konfidenz in der Schätzung auf als dies bei der Zuordnung von Objektkanten der Fall ist. In Tabelle 3.1 ist der Algorithmus zur Ermittlung eines Markantheitsbildes dargestellt.

3.3 Die Extraktion von markanten Bildbereichen aus einem Markantheitsbild

Die Extraktion der markantesten Punkte eines Bildes kann anschließend aus dem Markantheitsbild erfolgen, indem die hellsten Bereiche des Markantheitsbildes mittels eines Suchalgorithmus detektiert und in einer Liste abgespeichert werden. Ein solcher Suchalgorithmus kann beispielsweise so aussehen, daß iterativ die Position mit maximalem Grauwert im Markantheitsbild gesucht wird und anschließend das Markantheitsbild in einem vorgegebenen Radius um den ermittelten Punkt herum gelöscht wird[3], so daß dieser Punkt kein weiteres Mal gefunden wird.

[3] also auf den Grauwert 0 gesetzt wird

1. Das Eingabebild wird zunächst tiefpaßgefiltert (siehe Bild 3.1 und Bild 3.2), da nur bei tiefpaßgefilterten Bilddaten eine verläßliche Schätzung von Grauwertgradienten möglich ist. Diese Filterung geschieht über die $3 \times 3 -$ Maske,

$$\mathbf{G} = \frac{1}{16} \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}, \tag{3.14}$$

mit der das gesamte Bild mit der Grauwertfunktion $s(\vec{x})$ entsprechend

$$\hat{s}(\vec{x}) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} \mathbf{G}_{ij} \cdot s(\vec{x}) \quad \text{gefaltet wird.} \tag{3.15}$$

2. Ein horizontales und ein vertikales Differenzbild (g_x und g_y) werden durch Faltung mit den Masken

$$\mathbf{T}^x = \frac{1}{2} \begin{pmatrix} -1 & 0 & 1 \end{pmatrix} \quad \text{und} \quad \mathbf{T}^y = \frac{1}{2} \begin{pmatrix} -1 & 0 & 1 \end{pmatrix}^T \tag{3.16}$$

entsprechend

$$g_{x/y}(\vec{x}) = \sum_{i=-1}^{1} \mathbf{T}_i^{x/y} \cdot \hat{s}(\vec{x}) \tag{3.17}$$

berechnet (siehe Bild 3.3 und Bild 3.4).

3. Die Produktbilder g_{xx}, g_{xy} und g_{yy} werden pixelweise mit

$$g_{xx} := g_x \cdot g_x, \quad g_{xy} := g_x \cdot g_y \quad \text{und} \quad g_{yy} := g_y \cdot g_y \tag{3.18}$$

berechnet (siehe Bild 3.5, 3.6 und 3.7).

4. In den Produktbildern wird jeweils über kleine Bildausschnitte gemittelt, so daß die Bilder

$$\overline{g_{xx}} := \frac{1}{N} \sum g_{xx}, \quad \overline{g_{xy}} := \frac{1}{N} \sum g_{xy} \quad \text{und} \quad \overline{g_{yy}} := \frac{1}{N} \sum g_{yy} \tag{3.19}$$

entstehen (siehe Bild 3.8, 3.9 und 3.10).

5. Die Texturstärke $\hat{\tau}$ mit

$$\hat{\tau} = \overline{g_{xx}} + \overline{g_{yy}} \tag{3.20}$$

(siehe Bild 3.13), die Isotropie $q = 1 - p$ mit p aus Glg. 3.11, also

$$q = 1 - \frac{(\overline{g_{xx}} - \overline{g_{yy}})^2 + 4\overline{g_{xy}}^2}{(\overline{g_{xx}} + \overline{g_{yy}})^2} \tag{3.21}$$

(siehe Bild 3.12) und das Markantheitsbild

$$M = \hat{\tau} + \hat{\tau} \cdot q \tag{3.22}$$

(siehe Bild 3.14) werden berechnet.

Tabelle 3.1: Algorithmus zur Generierung eines Markantheitsbildes

Die Bilder 3.15 bis 3.17 zeigen ein Bild aus einer Verkehrsszene zusammen mit ihren Markantheitsbildern und den 35 markantesten Stellen. Markante Stellen werden hierbei allerdings nur unterhalb einer waagerechten Linie (50 Punkte unterhalb des oberen Bildrandes) gesucht, damit in erster Linie markante Stellen auf der Straße gefunden werden.

Bild 3.1: Originalbild

Bild 3.2: tiefpaßgefiltertes Bild

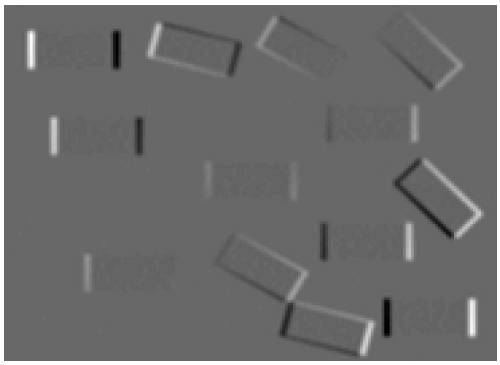


Bild 3.3: horizontales Differenzbild

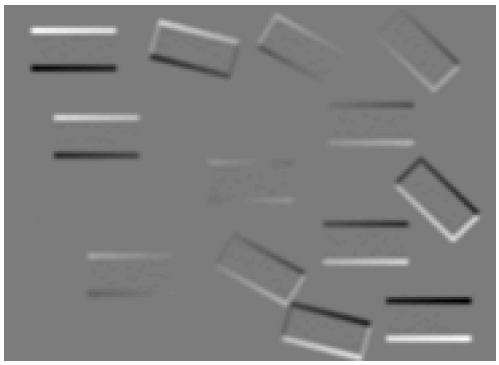


Bild 3.4: vertikales Differenzbild

Bild 3.5: Produktbild g_{xx}

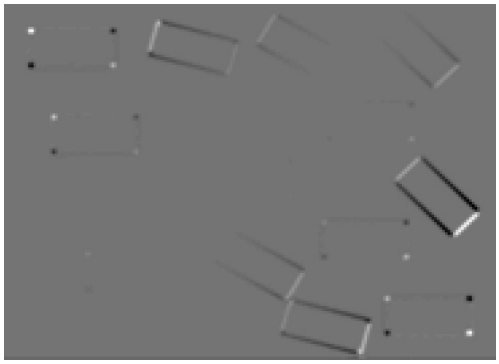


Bild 3.6: Produktbild g_{xy}

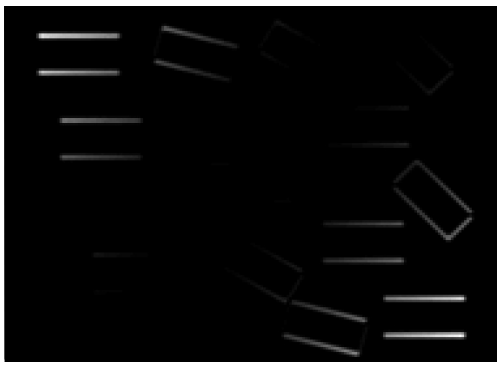


Bild 3.7: Produktbild g_{yy}

Bild 3.8: gemitteltes Produktbild $\overline{g_{xx}}$

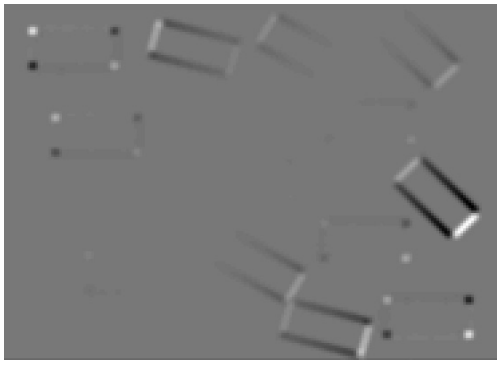


Bild 3.9: gemitteltes Produktbild $\overline{g_{xy}}$

Bild 3.10: gemitteltes Produktbild $\overline{g_{yy}}$

Bild 3.11: Texturstärke

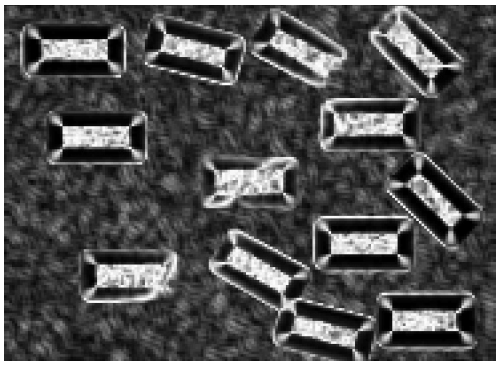


Bild 3.12: Isotropie

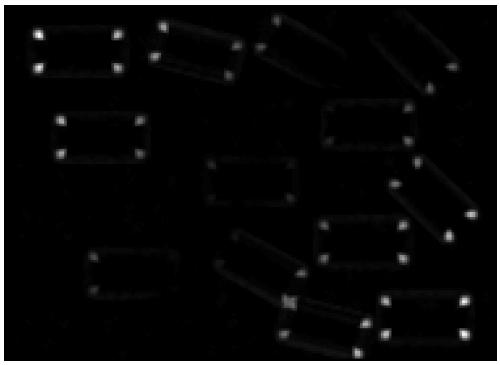


Bild 3.13: Textur × Isotropie

Bild 3.14: Markantheitsbild

Bild 3.15: Bild aus „Hildesheim-Sequenz“, Originalbild

Bild 3.16: Bild aus „Hildesheim-Sequenz“, Markantheitsbild

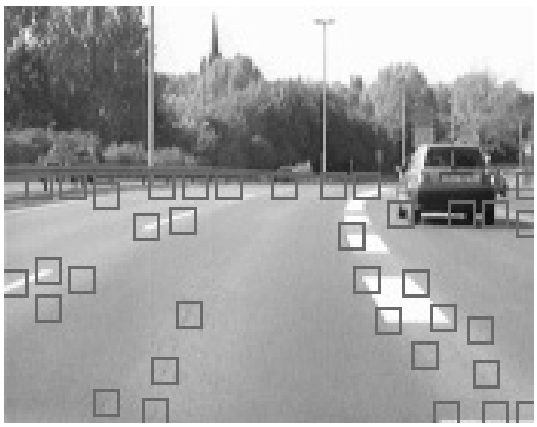


Bild 3.17: Bild aus „Hildesheim-Sequenz“: die 35 markantesten Punkte, oberer Bildteil unberücksichtigt

Kapitel 4

Statistische Verschiebungsvektorschätzung

4.1 Das *Block-Matching* Verfahren

In Kapitel 3 wurde erläutert, wie aus einem Bild ein Markantheitsbild erzeugt werden kann, so daß anschließend daraus eine gewisse Anzahl von besonders markanten Bildbereichen extrahiert werden können. Die Auszeichnung dieser Bildbereiche gegenüber ihrer Nachbarschaft ermöglichen es, die entsprechenden Bildbereiche in dem zeitlich folgenden Bild der Sequenz wiederzufinden. Eine solche Ermittlung von korrespondierenden Bildbereichen, die jeweils von denselben Objektpunkten stammen, wird üblicherweise als *Verschiebungsvektorschätzung* bezeichnet.

Die am häufigsten verwendete Methode zur Schätzung von Verschiebungsvektoren ist das sogenannte *Block-Matching-Verfahren* (engl. *to match*: übereinstimmen, zusammenpassen). Die Methode des Block-Matchings beruht darauf, die Grauwertdifferenzen zwischen einem Referenzblock in Bild $\mathcal{B}1$ und einem korrespondierenden Block in Bild $\mathcal{B}2$ zu bestimmen und zu bewerten. Seien also $\mathcal{B}1$ und $\mathcal{B}2$ die beiden Ausgangsbilder und sei weiterhin $\mathcal{M}1$ ein kleiner quadratischer markanter Bildbereich in Bild $\mathcal{B}1$, welcher dann in Bild $\mathcal{B}2$ wiedergefunden werden soll (siehe Bild 4.1). Da das Block-Matching-Verfahren mit einem hohen Berechnungsaufwand verbunden ist, wenn der Ausschnitt $\mathcal{M}_1$ bei der Suche nach seinem Gegenstück $\mathcal{M}_2$ in Bild $\mathcal{B}_2$ über einen großen Suchbereich in 1-Pixel-Schritten verschoben wird, ist i.a. die Wahl eines kleineren Suchfensters $\mathcal{S}$ sinnvoll, in dem der passende Pixelblock in Bild $\mathcal{B}_2$ vermutet wird. Das Suchfenster kann beispielsweise sehr klein gewählt werden, wenn ein gewisses Vorwissen über Art und Richtung der Bewegung bekannt ist. Leider wird i.a. keine exakte Übereinstimmung der beiden Pixelblöcke erzielt werden können, da sich einerseits die Ansicht der Szene zwischen den beiden Bildern durch die Bewegung der Kamera etwas verändert und andererseits Rauscheffekte[1] auftreten. Das Ziel ist also, einen möglichst *ähnlichen* Block zum Bildausschnitt $\mathcal{M}_1$ in Bild $\mathcal{B}2$ zu finden.

[1] Die Rauscheffekte sind beispielsweise auf das Pixelrauschen der Kamera oder auf Digitalisierungseffekte zurückzuführen.

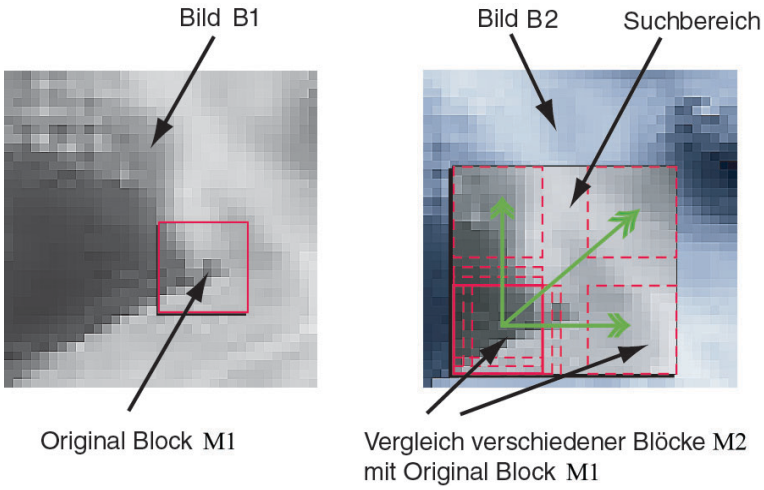


Bild 4.1: Beim Block-Matching wird ein Ausschnitt aus Bild B1 so lange über den Suchbereich von Bild B2 verschoben, bis ein möglichst gut dazu passender Bildausschnitt gefunden ist.

4.2 Das Mester-Hötter-Verfahren

Für das hier vorgestellte Verfahren zur Bewegungsschätzung kommt eine spezielle Variante des Block-Matching Verfahrens zum Einsatz, die von MESTER und HÖTTER in [MH95] und [MH95b] entwickelt wurde. Die Grundidee dieses Verfahrens ist es, zunächst die Grauwertdifferenzen in eine lokale Wahrscheinlichkeitsverteilung für die Verschiebungsvektoren zu transformieren und anschließend daraus den Erwartungswert des gesuchten Verschiebungsvektors und seine Kovarianzmatrix zu berechnen. Im folgenden wird das Verfahren detailliert beschrieben.

Wenn der Koordinatenvektor des Mittelpunktes des Originalblocks aus Bild $\mathcal{B}1$ mit $\vec{x}_1$ und der Koordinatenvektor des Mittelpunktes des Blocks aus $\mathcal{B}2$ mit $\vec{x}_2$ bezeichnet wird, dann ist der Differenzvektor $\vec{d} = \vec{x}_2 - \vec{x}_1$ der Verschiebungsvektor zwischen diesen Blöcken. Der Vektor $\vec{x}_1$ ist unveränderlich, da der entsprechende Block aus dem Markantheitsbild extrahiert wurde. Dahingegen existieren viele mögliche Vektoren $\vec{x}_2$ und somit auch viele potentielle Verschiebungsvektoren, wovon nur einer der „wahre“ sein kann. Als mögliches Ähnlichkeitskriterium r zwischen den beiden Blöcken kann die Summe der quadratischen Abweichungen der einzelnen Pixel-Grauwerte $g_1(\vec{x})$ aus Bild $\mathcal{B}1$ und $g_2(\vec{x})$ aus Bild $\mathcal{B}2$ gewählt werden, also

$$r(\vec{x}_1, \vec{d}) = \sum_{\vec{x} \in \mathcal{K}_1} (g_1(\vec{x}) - g_2(\vec{x} + \vec{d}))^2 \,, \tag{4.1}$$

wobei $\mathcal{K}_1$ den Suchbereich in Bild $\mathcal{B}2$ angibt. Dieses Ähnlichkeitskriterium r wird auch als *Residuum* bezeichnet. Anschließend wird jedem potentiellen Verschiebungsvektor eine Wahrscheinlichkeitsdichte zugeordnet. Die zweidimensionale Anordnung der Wahrscheinlichkeitsdichte für die Verschiebungsvektoren an allen möglichen Positionen innerhalb des Suchbereichs wird im folgenden als Verteilung der Wahrscheinlichkeitsdichte bezeichnet. Der optimale Schätzwert des Verschiebungsvektors stellt dann derjenige Verschiebungsvektor dar, der dem Erwartungswert der Verteilung der Wahrscheinlichkeitsdichte entspricht.

Werden die Grauwertdifferenzen der beiden zu vergleichenden Pixelblöcke in einem Vektor $\vec{e}$ mit der Länge N, dem sogenannten Fehlervektor, zusammengefaßt und unter der Annahme, daß

für die Berechnung der Wahrscheinlichkeitsdichte, die Grauwerte der einzelnen Pixel von einem Gaußschen Rauschen der Varianz σ^2 überlagert sind, dann sei die Verbundverteilungsdichte des Differenzvektors $\vec{e}$ unter der Vorgabe des Verschiebungsvektors $\vec{d}$ bestimmt durch

$$p(\vec{e}) = \left(1/\sqrt{2\pi\sigma_e^2}\right)^N \exp\left[-\sum_{i=1}^{N} e_i^2/2\sigma_e^2\right] . \tag{4.2}$$

Wie in Bild 4.2 dargestellt, wird die Verteilung des Residuenfeldes in diesem Ansatz also durch eine Gaußsche Verteilung beschrieben. Der Verschiebungsvektor $\hat{\vec{d}}$, welcher die Verbundverteilungsdichte $p(\vec{e})$ maximiert, ist der *Maximum-Likelihood-Schätzwert* für den unbekannten Verschiebungsvektor. Es ergibt sich [MH95] der Erwartungswert für den Betrag des Verschiebungsvektors zu

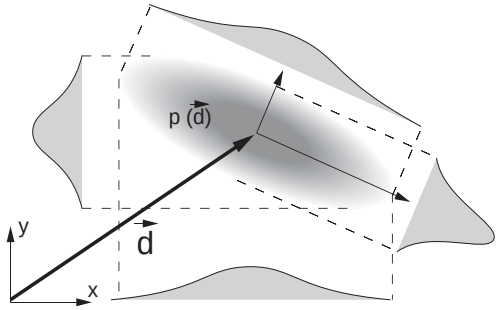


Bild 4.2: Modellhafte örtliche Wahrscheinlichkeitsdichte für einen Verschiebungsvektor $\vec{d}$ mit dem Hauptachsensystem, das sich aus der Analyse der Kovarianzmatrizen ergibt. [Tr99]

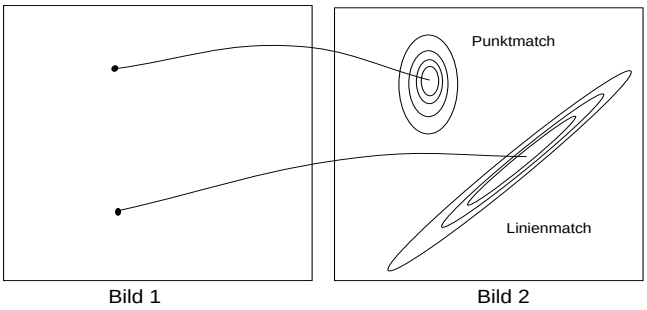


Bild 4.3: Punktmatch und Linienmatch [Tr99]

$$\left|\hat{\vec{d}}\right| = \sum_{x_1,x_2\in\mathcal{S}} \left[(x_1 - x_{1m})^2 + (x_2 - x_{2m})^2\right] P(\vec{x}) , \tag{4.3}$$

und die Kovarianzkoeffizienten der Wahrscheinlichkeitsverteilung zu

$$\sigma_{x_1}^2 = \sum_{x_1,x_2\in\mathcal{S}} (x_1 - x_{1m})^2 P(\vec{x}) , \tag{4.4}$$

$$\sigma_{x_2}^2 = \sum_{x_1,x_2\in\mathcal{S}} (x_2 - x_{2m})^2 P(\vec{x}) \text{ und} \tag{4.5}$$

$$\sigma_{x_1x_2} = \sum_{x_1,x_2\in\mathcal{S}} (x_1 - x_{1m})(x_2 - x_{2m}) P(\vec{x}) . \tag{4.6}$$

Eine Eigenwertanalyse der Kovarianzmatrix

$$\mathbf{C} = \begin{pmatrix} \sigma_{x_1}^2 & \sigma_{x_1x_2} \\ \sigma_{x_1x_2} & \sigma_{x_2}^2 \end{pmatrix} \tag{4.7}$$

liefert anschließend die Eigenwerte λ_1 und λ_2 mit $\lambda_1 \geq \lambda_2$. Dies eröffnet die Möglichkeit der graphischen Repräsentation der Kovarianzmatrix durch Höhenlinien in Form von Ellipsen in der Wahrscheinlichkeitsdichteverteilung. Somit kann die Qualität einer Verschiebungsvektormessung sowohl numerisch als auch graphisch beurteilt werden. Sind nämlich beide Eigenwerte in etwa gleich groß und kleiner als eine gewisse Schwelle ϵ, so handelt es sich bei dem Match um einen

sogenannten *Punktmatch*. Die Fehlerellipse hat dann die Form von Kreisen, wie in Bild 4.3 dargestellt. Falls jedoch der Eigenwert λ_1 viel größer als λ_2 ist, so ist die Fehlerellipse sehr lang gezogen und es handelt sich um einen sogenannten *Linienmatch* (siehe Bild 4.3). Dies unterstreicht die Leistungsfähigkeit des Verfahrens, da eine Unterscheidung von Punkt- und Linienmatches nicht länger notwendig ist. Abschließend seien hier noch beispielhaft ein Punktmatch in den Bildern 4.4 bis 4.7 und ein Linienmatch in den Bilden 4.8 bis 4.11 dargestellt. In den Residuenbildern 4.6 und 4.10 repräsentiert „schwarz“ eine völlige Übereinstimmung der beiden Blöcke mit dem Residuenwert $r(\vec{x}_1, \vec{d}) = 0$ und „weiß“ maximal unterschiedliche Blöcke. In den Bildern 4.7 und 4.11 sind die aus einer Analyse der Kovarianzmatrizen ermittelten Höhenlinien der Wahrscheinlichkeitsdichteverteilung dargestellt. Die kreisförmigen Höhenlinien in Bild 4.7 deutet auf einen Punktmatch hin und die langgezogenen Ellipsen in Bild 4.11 visualisieren einen Linienmatch.

Bild 4.4: Bildausschnitt mit einer Objektecke

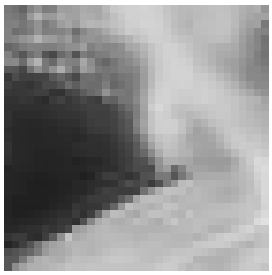


Bild 4.5: Suchbereich für Abb. 4.4

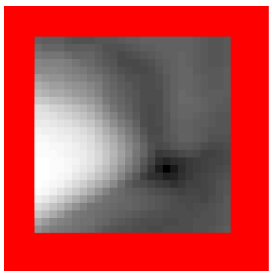


Bild 4.6: Das Residuenbild

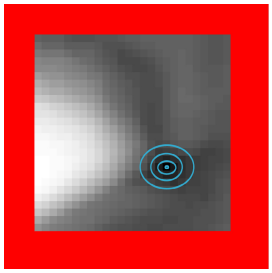


Bild 4.7: Visualisierung der ermittelten Verschiebungsvektorposition und der Höhenlinien der Wahrscheinlichkeitsdichteverteilung

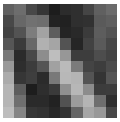


Bild 4.8: Bildausschnitt mit einer linienartigen Struktur

Bild 4.9: Suchbereich für Bildausschnitt 4.8

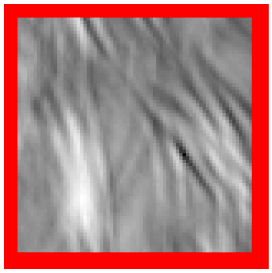


Bild 4.10: Das Residuenbild

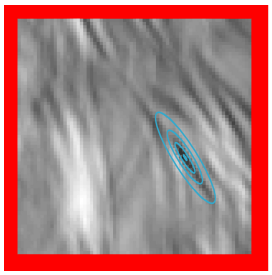


Bild 4.11: Visualisierung der ermittelten Verschiebungsvektorposition und der Höhenlinien der Wahrscheinlichkeitsdichteverteilung

Als Resultat dieses Verarbeitungssschrittes wird eine Liste von Verschiebungsvektoren zusammen mit ihren Kovarianzmatrizen erhalten. Diese bilden im folgenden Schritt die Datenbasis, mit der die Schätzung der dreidimensionalen Bewegungsparameter durchgeführt werden kann.

Kapitel 5

Bewegungsschätzung in planaren Szenen

5.1 Überblick

Nach der Bestimmung eines Verschiebungsvektorfeldes $\{\vec{d}_i\}$ wird im nächsten Schritt des hier vorgestellten Verfahrens die Bewegung der Kamera im dreidimensionalen Raum geschätzt. Diese Bewegungsschätzung basiert einerseits auf den ermittelten Verschiebungsvektoren mit ihren Unsicherheiten in Form der Kovarianzmatrix und andererseits auf einem *Ebenenmodell*. Die Annahme, die Welt bestehe aus einer Ebene, ist für bestimmte Anwendungen durchaus realistisch. Die Verwendung von Verkehrsszenen, die aus einem sich bewegenden Fahrzeug aufgenommen wurden, ist ein Beispiel dafür, daß die Welt in der unteren Hälfte des aufgenommenen Bildes dem Modell einer Ebene sehr gut entspricht. Natürlich gibt es in realem Bildmaterial Abweichungen von diesem Modell, jedoch können die Auswirkungen dieser Abweichungen auf die Bewegungsschätzung durch die Verwendung von robusten Schätzmethoden deutlich reduziert werden. Bevor jedoch die robuste Bewegungsschätzung aus monokularen Bildsequenzen in Kapitel 6 behandelt wird, muß zuerst über die Einführung einer geeigneten Parametrisierung der Bewegung die grundlegende „Least-Squares"-Prozedur [PTVF92] zur Bewegungsschätzung vorgestellt werden.

5.2 Parametrisierung der dreidimensionalen Bewegung

Eine Bewegung im dreidimensionalen Raum läßt sich ganz allgemein über eine Rotation mit der Rotationsmatrix $\mathbf{R}$ und eine Translation mit dem Translationsvektor $\vec{t}$ beschreiben. Die mathematische Beziehung,

$$\vec{q} = \mathbf{R} \cdot \vec{p} + \vec{t} \tag{5.1}$$

$$\begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \quad , \tag{5.2}$$

beschreibt die Lage eines Punktes im dreidimensionalen Raum vor ($\vec{p}$) und nach ($\vec{q}$) der Bewegung. Da in dieser Arbeit von einem planaren Weltmodell ausgegangen wird, ist zur Parametrisierung der Bewegung neben diesen eigentlichen Bewegungsgrößen $\mathbf{R}$ und $\vec{t}$ auch eine

Parametrisierung der Lage der Ebene im Raum notwendig. Hier wird üblicherweise der Normalenvektor der Ebene mit der Bezeichnung $\vec{b}$ verwendet. Bild 5.1 visualisiert noch einmal die zu schätzenden Bewegungsgrößen.

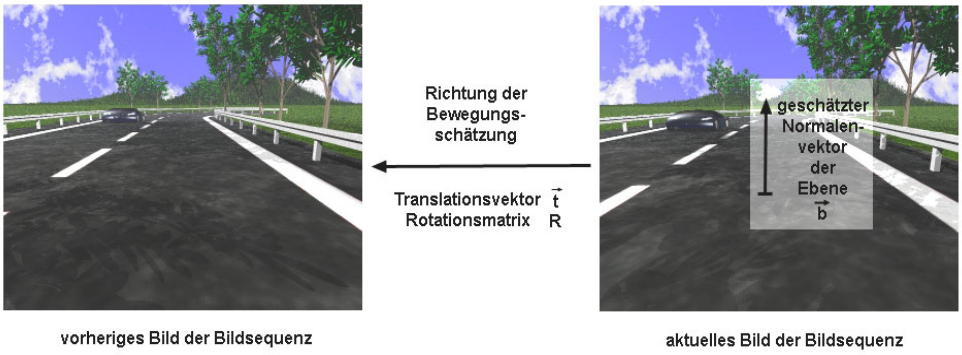


Bild 5.1: Visualisierung der dreidimensionalen Bewegungsparameter

Rotationen im dreidimensionalen Raum werden in den meisten Fällen durch *Euler-Winkel* dargestellt, was jedoch zu folgenden Problemen führt:

1. Die Angabe eines Satzes von Eulerwinkeln ist in einigen Fällen nicht eindeutig. Wird beispielsweise keine Drehung um die y-Achse durchgeführt, ist also $\theta = 0$, beschreibt jeder Satz von Eulerwinkeln mit $\phi = \phi$, $\theta = 0$ und $\psi = -\phi$ dieselbe Rotation.

2. In der Literatur existieren unterschiedliche Definitionen von Eulerwinkeln. Vor ihrer Benutzung ist eine Festlegung der Drehachsen notwendig und weiterhin muß eine Festlegung der Reihenfolge der Achsen, um die gedreht wird, erfolgen.

3. Als problematisch bei der Verwendung von Eulerwinkeln kann sich weiterhin herausstellen, daß unter gewissen Voraussetzungen ein Freiheitsgrad in der Rotationsmatrix verschwinden kann. Als Beispiel sei hier $\cos\theta = 0$ gegeben, denn die Rotationsmatrix hängt dann nur noch von der Differenz $(\phi - \psi)$ ab.

Um diese Probleme mit Euler-Winkeln zu umgehen, wird im folgenden eine Parametrisierung der Rotationsmatrix über sogenannte *Einheitsquaternionen* eingesetzt. Anhang B gibt eine kurze Einführung in Quaternionen und die Verwendung von Einheitsquaternionen als eine mögliche Parametrisierung von Rotationen im dreidimensionalen Raum. An dieser Stelle sei nur auf den Zusammenhang des Einheitsquaternions $\epsilon = (\epsilon_0, \epsilon_1, \epsilon_2, \epsilon_3)$ mit $\sum_{i=0}^{3} \epsilon_i^2 = 1$ und die damit dargestellte Rotationsmatrix $\mathbf{R}$ hingewiesen. Die Komponenten der Rotationsmatrix ergeben sich aus dem Einheitsquaternion gemäß der Formel

$$\mathbf{R} = \begin{pmatrix} \epsilon_0^2 + \epsilon_1^2 - \epsilon_2^2 - \epsilon_3^3 & 2(\epsilon_2\epsilon_1 - \epsilon_0\epsilon_3) & 2(\epsilon_3\epsilon_1 + \epsilon_0\epsilon_2) \\ 2(\epsilon_1\epsilon_2 + \epsilon_0\epsilon_3) & \epsilon_0^2 - \epsilon_1^2 + \epsilon_2^2 - \epsilon_3^2 & 2(\epsilon_2\epsilon_3 - \epsilon_0\epsilon_1) \\ 2(\epsilon_1\epsilon_3 - \epsilon_0\epsilon_2) & 2(\epsilon_2\epsilon_3 + \epsilon_0\epsilon_1) & \epsilon_0^2 - \epsilon_1^2 - \epsilon_2^2 + \epsilon_3^2 \end{pmatrix} .$$

Es kann gezeigt werden [Po95], daß die Parametrisierung einer Rotation durch ein Einheitsquaternion im wesentlichen äquivalent ist zu einer Darstellung der Rotation mittels einer Rotationsachse und eines Rotationswinkels um diese Achse.

Neben der Rotation sind auch die beiden Vektoren der Translation $\vec{t}$ und der Ebene $\vec{b}$ zu parametrisieren. Es ist bereits allgemein bekannt [Bi77], daß für den hier behandelten Fall planarer Welten nur fünf von den sechs Parametern des Translationsvektors und des Ebenen-Normalenvektors bestimmt werden können. Der Grund hierfür liegt in der Tatsache, daß aus einem einzigen Bild keine Aussagen über absolute Längen der Objekte im Bild gemacht werden können. Beispielsweise haben ein bestimmtes Objekt und ein baugleiches Objekt das doppelt so weit entfernt und doppelt so groß ist, im Bild identische Ausmaße und können somit nicht voneinander unterschieden werden. In Bild 5.2 ist dieser Sachverhalt visualisiert; es kann keine Aussage über die absolute Größe der dreidimensionalen Objekte gemacht werden, wenn ausschließlich *ein* zweidimensionales Bild betrachtet wird. Um dieser Unbestimmtheit Rechnung zu tragen, wird eine

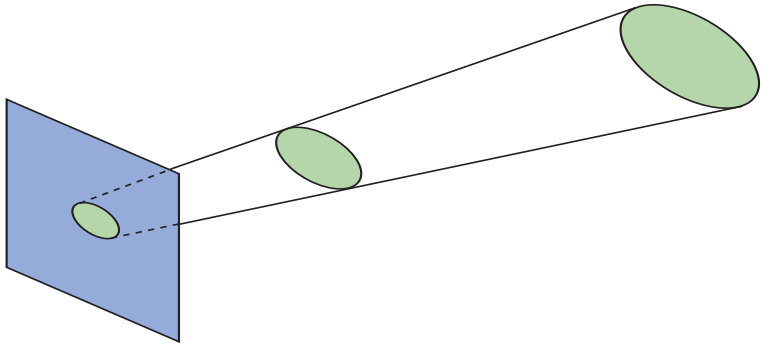


Bild 5.2: Unbestimmtheit der Objektgröße in der dreidimensionalen Welt bei Betrachtung ihres zweidimensionalen Bildes

der sechs Komponenten dieser beiden Vektoren auf einen Wert festgelegt, z.B. ($b_2 := -1$). Als Ergebnis dieser Überlegungen verbleibt ein Parametervektor $\vec{p}$ mit neun Elementen

$$\vec{p} = (p_1 \dots p_9) := (\epsilon_0, \epsilon_1, \epsilon_2, \epsilon_3, t_1, t_2, t_3, b_1, b_3) \quad , \tag{5.3}$$

wobei die Nebenbedingung $\sum_{i=0}^{3} \epsilon_i^2 = 1$ erfüllt sein muß.

5.3 Bewegungsschätzung bei planaren Szenen

Es stellt sich nun die Frage, wie aus den Verschiebungsvektoren, ihren Korvarianzmatrizen und dem Ebenenmodell die dreidimensionalen Bewegungsparameter geschätzt werden können. Bild 5.3 verdeutlicht noch einmal diese Fragestellung. Dieses Flußdiagramm wird in den folgenden Kapiteln sukzessive aufgebaut und repräsentiert schließlich das gesamte hier vorgestellte statistische Verfahren zur Bewegungsschätzung.

Es ist allgemein bekannt [Te95], wie aus den

- **Abbildungsgleichungen einer Lochkamera**

$$u_1 = \frac{f}{p_3} \cdot p_1 \tag{5.4}$$

$$u_2 = \frac{f}{p_3} \cdot p_2 \tag{5.5}$$

(Die Gleichungen ergeben sich aus dem Strahlensatz, wenn der Punkt P mit dem Ortsvektor $\vec{p} = (p_1, p_2, p_3)^T$ in Kamerakoordinaten auf den Punkt U in der Bildfläche mit dem Ortsvektor $\vec{u} = (u_1, u_2)^T$ abgebildet wird. Es ergibt sich eine Skalierung der ersten beiden

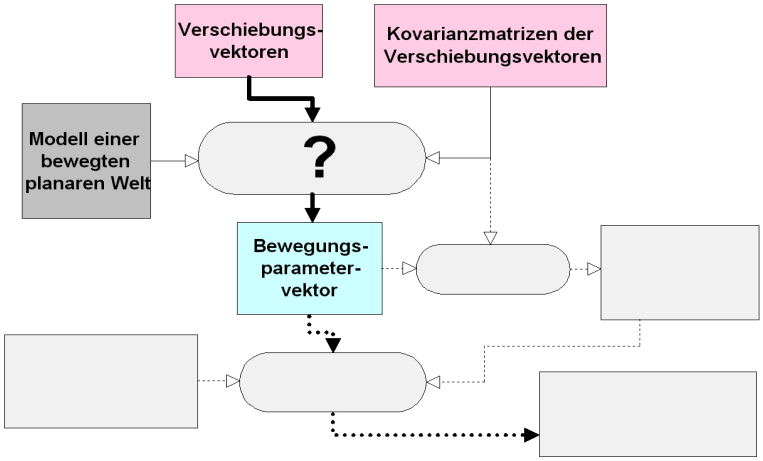


Bild 5.3: Das Verfahren der dreidimensionalen Bewegungsschätzung (Teil 1)

Koordinaten des Punktes P um den Faktor f/p_3, wobei f die Brennweite des Kamerasystems darstellt.)

- **der Ebenengleichung**

$$b_1 p_1 + b_2 p_2 + b_3 p_3 = c \qquad \text{bzw.} \qquad \vec{b}^T \vec{p} = c \tag{5.6}$$

und

- **der 3-D-Bewegungsgleichung**

$$\vec{q} = \mathbf{R} \cdot \vec{p} + \vec{t} \tag{5.7}$$

$$\begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \tag{5.8}$$

der Zusammenhang zwischen den korrespondierenden Punkten $\vec{u}$ und $\vec{v}$ aus zwei Bildern einer Bildsequenz in Abhängigkeit der Bewegungsgrößen gewonnen werden kann. Es seien r_{ij} die Elemente der Rotationsmatrix $\mathbf{R}$ und f die Brennweite, so ergibt sich [Te95]

$$v_{1p} = \frac{u_1(r_{11} + t_1 b_1) + u_2(r_{12} + t_1 b_2) + f(r_{13} + t_1 b_3)}{\frac{u_1}{f}(r_{31} + t_3 b_1) + \frac{u_2}{f}(r_{32} + t_3 b_2) + (r_{33} + t_3 b_3)} \tag{5.9}$$

$$v_{2p} = \frac{u_1(r_{21} + t_2 b_1) + u_2(r_{22} + t_2 b_2) + f(r_{23} + t_2 b_3)}{\frac{u_1}{f}(r_{31} + t_3 b_1) + \frac{u_2}{f}(r_{32} + t_3 b_2) + (r_{33} + t_3 b_3)} \,. \tag{5.10}$$

Die Gleichungen 5.9 und 5.10 geben also das hier verwendete mathematische Modell einer Bewegung auf einer planaren Welt in kompakter Form wieder.

Bedauerlicherweise ergeben sich diese Beziehungen nur, wenn bei der Messung der Korrespondenzen keinerlei Ungenauigkeiten auftreten. Bei realen Messungen ist hingegen der Vektor $\vec{v}_p$ mit additiven Störungen behaftet, so daß sich für gemessene Verschiebungsvektoren $v_1 = v_{1p} + z_1$ und $v_2 = v_{2p} + z_2$ ergeben. Der Fehlervektor $\vec{z}$ ist mittelwertfrei und hat eine Kovarianzmatrix $\mathbf{C}$, von der erst einmal angenommen wird, daß sie die Form $\mathbf{C} = \sigma \cdot \mathbf{I}$, mit der Einheitsmatrix $\mathbf{I}$ habe. Gegeben sei nun ein Ensemble von N Bildkorrespondenzen $\vec{u}_i \leftrightarrow \vec{v}_i$, welche aus den Verschiebungsvektor-Messungen erhalten wurden und im folgenden durch das Subskript i

unterschieden werden. Unter der Annahme, daß die Meßfehlervektoren $\vec{z}_i$ mit einer Kovarianzmatrix $\mathbf{C}_i$ normalverteilt sind, ergibt sich die Verlustfunktion $Q(\vec{p})$, die durch die Variation des Parametervektors $\vec{p}$ aus Gleichung 5.3 minimiert werden soll, zu

$$Q(\vec{p}) = \sum_{i=1}^{N} \left|\vec{z}_i(\vec{p})\right|^2 \quad . \tag{5.11}$$

Das Ziel ist also, unter Verwendung einer *Optimierungsmethode*, diejenigen Bewegungsgrößen zu finden, die bei einer Bewegung auf einer Ebene die gemessenen Verschiebungsvektoren zur Folge haben. Im Rahmen dieser Arbeit wurde hierzu das Levenberg-Marquardt-Verfahren [PTVF92] verwendet, da es mit wenigen Rechenoperationen[1] gute Optimierungsergebnisse liefert.

Es kann gezeigt werden ([SW94],[Me95]), daß das Ergebnis dieser Least-Squares-Schätzung statistisch optimal ist. Das bedeutet, daß unter der Voraussetzung daß keine Korrelationen in den einzelnen Komponenten des Fehlervektors auftreten, also wenn $\mathbf{C} = \sigma \cdot \mathbf{I}$ gilt, das Ergebnis der Schätzung des Parametervektors *erwartungstreu* und diejenige Schätzung mit *minimaler Varianz* ist. Einen solchen Schätzer bezeichnet man in der Estimationstheorie auch als *Best Linear Unbiased Estimator* oder einfach als *BLUE*.

Wie im Anhang A gezeigt wird, muß für den Fall allgemeiner Kovarianzmatrizen die Verlustfunktion $Q(\vec{p})$ abgeändert werden zu

$$\tilde{Q}(\vec{p}) = \sum_{i=1}^{N} \left|\mathbf{W}_i^T \vec{z}_i(\vec{p})\right|^2 \quad , \tag{5.12}$$

wobei $\mathbf{W}_i$ die Wurzel der inversen Kovarianzmatrix $\mathbf{C}_i^{-1}$ des Meßfehlervektors $\vec{z}_i$ ist, so daß $\mathbf{C}_i^{-1} = \mathbf{W}_i \mathbf{W}_i^T$ gilt. Gleichung (5.12) legt fest, daß die Abweichung zwischen dem Modell (v_{ip}) und der Messung (v_i) mit einer Metrik bewertet wird, die durch die Kovarianzmatrix $\mathbf{C}_i$ gegeben ist. Auch für den Fall allgemeiner Kovarianzmatrizen läuft die Bestimmung des Parametervektors $\vec{p}$ auf die Minimierung der Verlustfunktion Q mit einem Optimierungsverfahren hinaus. Allerdings ist hierbei zu beachten, daß eine Minimierung unter einer nichtlinearen Randbedingung erfolgen muß, da das gesuchte Einheitsquaternion den Betrag 1 besitzen muß.

5.4 Bewegungskompensation und Hinderniserkennung

Testet man die in diesem Kapitel beschriebene Prozedur mit synthetischen Bildsequenzen, die exakt dem planaren Weltmodell entsprechen, so erhält man ausgezeichnete Resultate (siehe dazu Abschnitt 6.5). Diese Methode der Least-Squares-Schätzung der Bewegungsparameter weist jedoch große Defizite bei der Verarbeitung von realen Verkehrsszenen auf. Da die wahren Bewegungsgrößen von diesen Verkehrsszenen nicht bekannt sind, kann nur eine visuelle Beurteilung der Schätzergebnisse erfolgen. Hierfür eignet sich hervorragend die Verwendung von *bewegungskompensierten Differenzbildern*. Dieses von Carlsson und Eklundh entwickelte Verfahren [CE90] stammt aus dem Jahr 1990 und gehört mittlerweile zu den Standardverfahren der sog. *structure-from-motion*. Die grundlegende Idee dieses Verfahrens ist es, jeden Pixel des aktuellen Bildes unter Verwendung der geschätzten Bewegungs- und Ebenen-Parameter um eine Zeiteinheit zurück zu transformieren und anschließend ein Differenzbild von dem transformierten

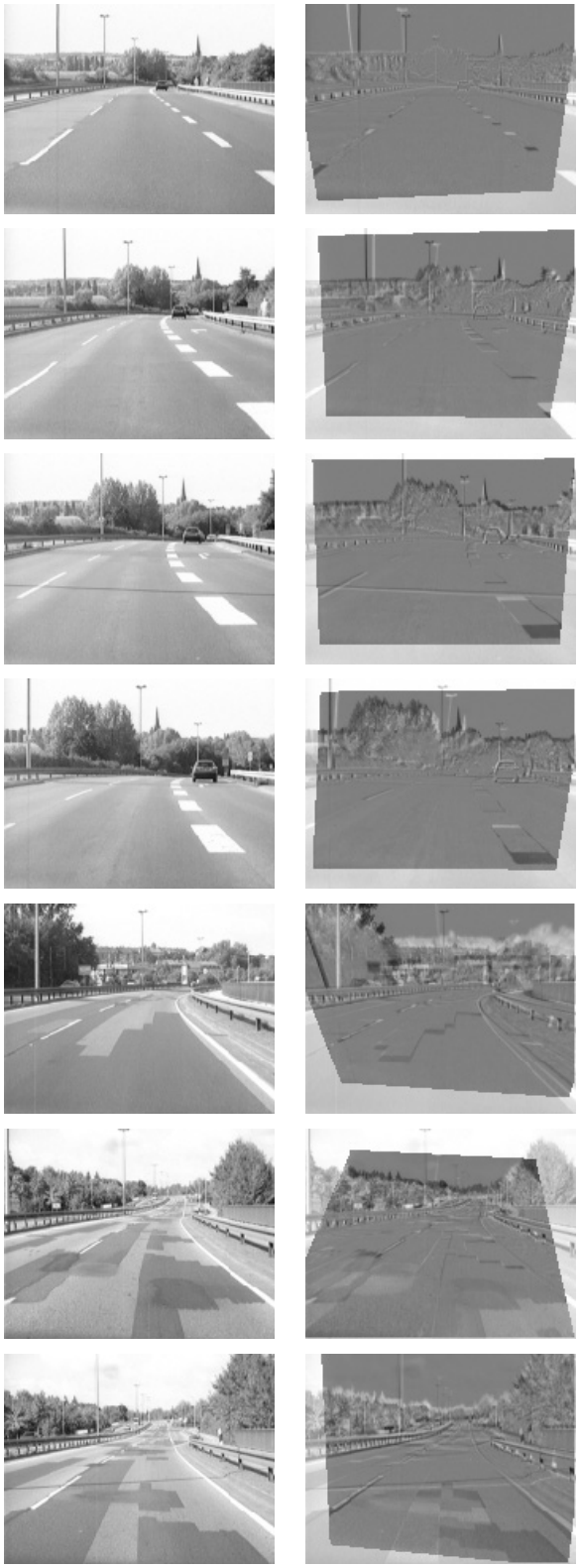
[1] Die Verarbeitungsgeschwindigkeit ist von besonderer Bedeutung, da für praktische Anwendungen eine *Echtzeit*-Bewegungsschätzung wünschenswert ist.

und dem wahren, aufgenommenen Bild zu errechnen. Falls alle Parameter richtig geschätzt wurden, und der Bildinhalt tatsächlich einer bewegten Ebene entspricht, so sollte das bewegungskompensierte Differenzbild in einem homogenen Grau[2] erscheinen. Da allerdings das Modell einer bewegten Ebene für reale Verkehrsszenen durch die Anwesenheit von dreidimensionalen Objekten, wie z.B. Bäumen, Fahrzeugen oder Mästen, verletzt ist, sollte in dem Ergebnisbild nur die Straße in einem homogenen Grau erscheinen und die dreidimensionalen Objekte deutlich hervortreten. Dies eröffnet sodann die Möglichkeit, eine Hinderniserkennung zu realisieren.

5.5 Ergebnisse

In Bild 5.4 sind einige Bilder der sog. Hildesheim-Sequenz zusammen mit ihren bewegungskompensierten Differenzbildern unter Verwendung der Least-Squares-Schätzung der Parameter dargestellt. Die Straße erscheint in den Differenzbildern keineswegs in einem homogenen Grau, was bedeutet, daß die Bewegungsgrößen bei der Verwendung des Least-Squares-Verfahrens grob falsch geschätzt wurden. Besonders bei den letzten beiden Bildern in Bild 5.4, die eine starke Texturierung der Straßenebene aufweisen, wurden die Bewegungsgrößen völlig falsch geschätzt. Wie im folgenden Kapitel detailliert beschrieben wird, sind Fehlmessungen in den Verschiebungsvektoren, sogenannte *Ausreißer*, für dieses Verhalten verantwortlich.

[2]Zur besseren Visualisierung des Differenzbildes entspricht der Grauwert 128 einer Differenz von Null. Die Grauwertbereiche 0 bis 127 bzw. 129 bis 255 visualisieren sowohl negative als auch positive Differenzen in dem Differenzbild.



a b

Bild 5.4: Einige Bilder der sog. Hildesheim-Sequenz (a) zusammen mit den bewegungskompensierten Differenzbildern bei der Least-Squares-Schätzung (b)

Kapitel 6

Robuste Bewegungsschätzung in planaren Szenen

6.1 Ausreißer in den Verschiebungsvektoren

Das in Kapitel 5 vorgestellte Schätzverfahren, das nach dem Least-Squares-Prinzip[1] abläuft, stellt einen optimalen Schätzer dar, wenn die Störungen in den Messungen normalverteilt sind. Leider ist diese Annahme jedoch bei vielen Aufgabenstellungen der Bewegtbildverarbeitung verletzt. Der praktische Einsatz von Algorithmen zur Bildsequenzverarbeitung führt immer wieder zu dem Problem, daß in der Praxis nicht von einem Modell mit normalverteilten Störungen ausgegangen werden kann. Als Konsequenz daraus ergibt sich, daß die Modellparameter nur ungenau bestimmt werden können. Ein noch größeres Problem liegt jedoch in dem Auftreten von sogenannten *Ausreißern* in den Messungen, wie sie beispielsweise von Fehlmessungen hervorgerufen werden können. Besonders im Falle der Bewegungsschätzung in planaren Szenen ist das Auftreten von Ausreißern ein schwerwiegendes Problem, was im folgenden begründet wird.

Wie bereits erläutert, ist der Ausgangspunkt der Bewegungsschätzung eine Liste von Verschiebungsvektoren. Bisher wurde angenommen, daß die Störungen, denen die Verschiebungsvektoren unterliegen, normalverteilt sind. Aus den folgenden beiden Gründen ist dies jedoch nicht der Fall:

1. Bei der Bestimmung der Verschiebungsvektoren kommt es oftmals vor, daß sie für markante Punkte geschätzt werden, die außerhalb der Ebene liegen. Bei Fahrbahnszenen liegt dies meist an dem Vorhandensein von Leitplanken, Bäumen oder weiteren Verkehrsteilnehmern. Für solche Korrespondenzen — die möglicherweise sogar richtig geschätzt wurden — ist die Ebenenannahme verletzt, so daß sie als Ausreißer zu werten sind. In Bild 6.1 sind solche Ausreißer mit einer „1“ zwischen den beiden Bildern der Sequenz gekennzeichnet.

2. Ein weiterer Grund für Ausreißer in den Verschiebungsvektoren sind periodische Strukturen oder Texturen auf der Ebene. Obwohl solche Verschiebungsvektoren innerhalb der Ebene bzw. Fahrbahn liegen, ist die betreffende Korrespondenz unter der Ebenen-Annahme fehlerhaft. In Bild 6.1 sind solche Ausreißer mit einer „2“ gekennzeichnet.

[1] Least-Squares-Schätzer basieren auf dem Prinzip, daß sie die Summe des Quadratfehlers zwischen Messung und Modell minimieren, um daraus einen „optimalen“ Parametersatz zu erhalten.

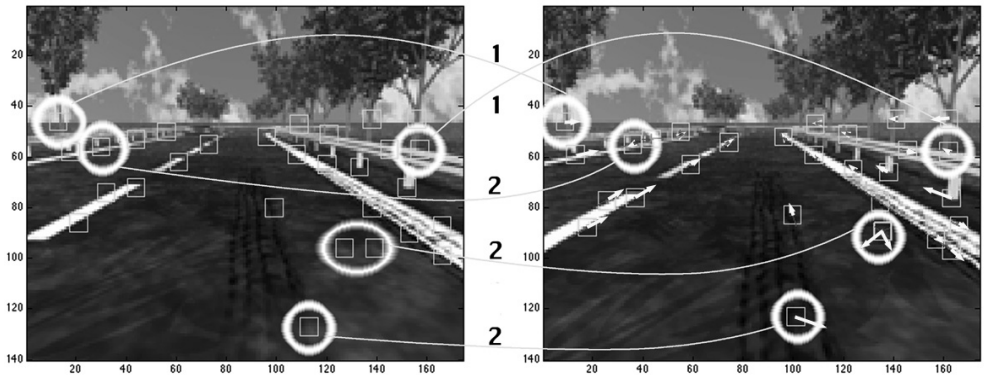


Bild 6.1: Zwei aufeinanderfolgende Bilder einer synthetischen Straßensequenz mit typischen Ausreißern: (1) Bildkorrespondenzen außerhalb der Ebene und (2) falsche Bildkorrespondenzen innerhalb der Ebene

Für das Problem der Bewegungsschätzung haben solche Punkte im Falle einer Least-Squares-Schätzung gravierende Auswirkungen auf das Schätzergebnis. Die Bilder 6.2 und 6.3 sollen die Auswirkungen von Ausreißern für den einfachen Fall einer Least-Squares-Schätzung von einem Polynom 2. Grades demonstrieren. Aus der Gleichung

$$y = a_1 \cdot x^2 + a_2 \cdot x + a_3 + \eta \quad \text{mit} \quad a_1 = 1\,, a_2 = 0\,, a_3 = 0 \tag{6.1}$$

wurde für $x_i = 1 \ldots 26$ ein Datensatz generiert. Der additive Rauschanteil η stellt eine normalverteilte Zufallszahl mit der Varianz $\sigma^2 = 2 \cdot x$ dar. In Bild 6.2 sind die generierten Werte des Testdatensatzes in „blau“ in einen Graphen eingetragen. In „grün“ wurde das Ergebnis einer Least-Squares-Schätzung in den selben Graphen eingetragen. Die quadratische Regression lieferte die Parameter $a_1 = 0.92\,, a_2 = 1.62$ und $a3 = -4.66$, welche recht nahe an den Parametern des zugrundeliegenden Modells liegen. Befinden sich jedoch Ausreißer in dem Datenmaterial, wie in Bild 6.3 zu sehen, so schätzt das Least-Squares-Verfahren die völlig falschen Parameter $a_1 = 0.55\,, a_2 = 3.95$ und $a3 = 47.8\,$.

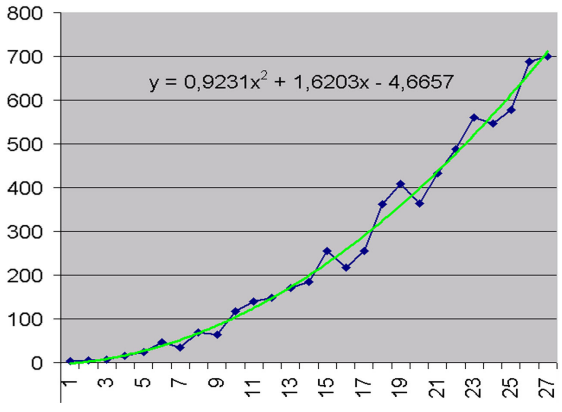


Bild 6.2: Least-Squares-Schätzung ohne Ausreißer

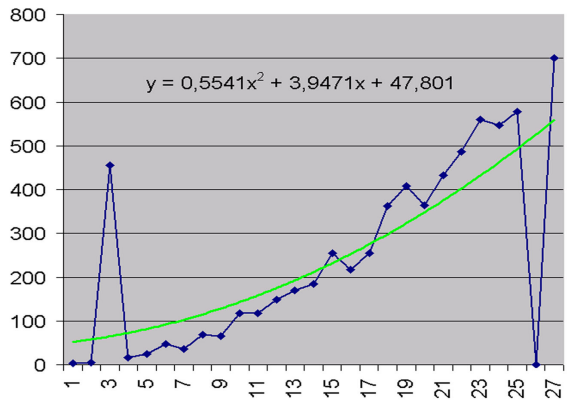


Bild 6.3: Least-Squares-Schätzung mit zwei Ausreißern

Allerdings existieren neben den Least-Squares-Verfahren noch andere Schätzmethoden, mit deren Hilfe die Auswirkung von Ausreißern auf das Schätzergebnis deutlich reduziert werden kann.

Solche Schätzer werden *robuste Schätzer* genannt. Eine Möglichkeit der Realisierung von robusten Schätzern bilden die sogenannten *M-Estimatoren* [Hu81], die im folgenden Abschnitt näher beschrieben werden.

6.2 Robuste Schätzung mit M-Estimatoren

Das grundlegende Prinzip eines M-Estimators besteht darin, das kovarianzgewichtete Residuenquadrat der Least-Squares-Schätzung durch eine geeignetere Funktion ρ zu ersetzen. Dabei wird die Differenz zwischen Messung und Modell $\vec{r}_i = \vec{y}_i - \vec{f}(\vec{x}_i, \vec{a})$ als *Residuum* $\vec{r}_i$ bezeichnet. Sei weiterhin σ_i die Varianz der Messung, dann tritt an die Stelle des Least-Squares-Kriteriums

$$\sum_i \left(\frac{r_i}{\sigma_i}\right)^2 \longrightarrow min \tag{6.2}$$

das M-Estimator-Kriterium

$$\sum_i \rho\left(\frac{r_i}{\sigma_i}\right) \longrightarrow min \qquad (\rho : \text{M-Estimator-Funktion}) \; . \tag{6.3}$$

Der Verlauf der M-Estimator-Funktion bestimmt wie groß der Einfluß von Ausreißern auf das Schätzergebnis ist. Die Ableitung der $\rho-$Funktion stellt ein Maß für den Einfluß eines Residuums auf die Schätzung dar und wird deshalb als *Einflußfunktion* bezeichnet. Die Einflußfunktion wird im folgenden mit ψ bezeichnet. Es gilt

$$\psi(r_i) := \frac{\partial \rho(r_i)}{\partial r_i} \; . \tag{6.4}$$

Auch die zweite Ableitung der $\rho-$Funktion nach dem Residuum ist für die Beurteilung des Einflusses von Ausreißern auf die Schätzung wichtig; sie wird mit Θ bezeichnet. Somit gilt

$$\Theta(r_i) := \frac{\partial^2}{\partial r_i^2} \rho(r_i) \; . \tag{6.5}$$

Die Einflußfunktion von Least-Squares-Schätzern besitzt die Form einer Gerade, was verdeutlicht, daß sehr große Residuen — wie bei Ausreißern — auch einen sehr großen Einfluß auf das Schätzergebnis ausüben. Die Einflußfunktionen der robusten M-Estimatoren hingegen sollten für große Residuen gegen Null laufen, so daß der Einfluß dieser Residuen auf das Ergebnis der Schätzung vernachlässigbar ist. Es wurden verschiedene $\rho-$Funktionen bezüglich ihrer Eignung für das Problem der Bewegungsschätzung untersucht. In der Tabelle 6.1 und in Bild 6.4 sind für (a) den Least-Squares-Schätzer, (b) die „Fair"-Funktion [Zh96] und (c) die Welsh-Funktion [Zh96] die entsprechende $\rho-$Funktion, die zugehörige Einflußfunktion ψ und die zweite Ableitung Θ dargestellt.

Das Ergebnis der Untersuchungen war, daß die „Fair"-Funktion für den vorliegenden Anwendungsfall den am besten geeigneten M-Estimator darstellt.

	Least-Squares	Fair	Welsh
ρ	r_i^2	$c^2\left(\frac{\lvert r_i\rvert}{c} - \ln\left(1+\frac{\lvert r_i\rvert}{c}\right)\right)$	$\frac{c^2}{2}\left(1-e^{-\frac{2r_i^2}{c^2}}\right)$
ψ	$2r_i$	$\frac{r_i}{1+\frac{\lvert r_i\rvert}{c}}$	$2r_i e^{-\frac{2r_i^2}{c^2}}$
Θ	2	$\frac{c^2}{(c+\lvert r_i\rvert)^2}$	$\frac{2(c^2-4r_i^2)}{c^2}e^{-\frac{2r_i^2}{c^2}}$
		mit $c = 1,4$	mit $c = 2,985$

Tabelle 6.1: Vergleich der Verlustfunktion des LS-Schätzers und ihrer Ableitungen mit der Fair- und der Welsh-Funktion

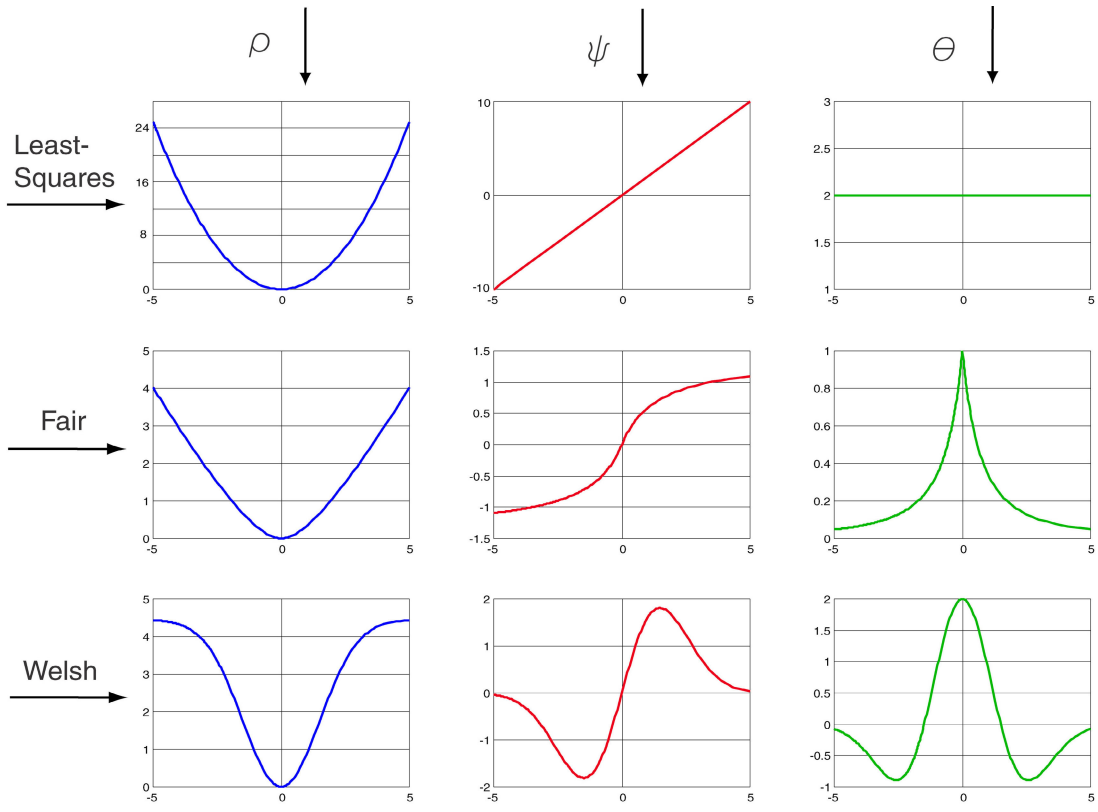


Bild 6.4: Verlustfunktion zusammen mit ihren ersten beiden Ableitungen im Falle des Least-Squares-Schätzers, der „Fair“-Funktion und der Welsh-Funktion

6.3 Bewegungsschätzung mit M-Estimatoren

In diesem Abschnitt wird gezeigt, wie man die dreidimensionale Bewegungsschätzung aus Kapitel 5 unter Verwendung der M-Estimatoren robuster gegenüber Ausreißern machen kann.

Wird das Betragsquadrat des Vektors $\mathbf{W}_i^T \vec{z}_i(\vec{p})$ in Gleichung (5.12) zu einer Quadratsumme der einzelnen Vektorkomponenten aufgespalten und anschließend die Quadrate durch das M-Estimator-Kriterium ersetzt, so ergibt sich die neue Verlustfunktion

$$\begin{aligned} Q^*(\vec{p}) = &\sum_{i=1}^{N} \left[\rho\left(w_{i11}(v_{i1} - v_{i1p}) + w_{i12}(v_{i2} - v_{i2p})\right)\right] \\ &+ \sum_{i=1}^{N} \left[\rho\left(w_{i21}(v_{i1} - v_{i1p}) + w_{i22}(v_{i2} - v_{i2p})\right)\right]. \end{aligned} \tag{6.6}$$

Hierbei entspricht $\vec{v}_i = \begin{pmatrix} v_{i1} \\ v_{i2} \end{pmatrix}$ der i-ten Verschiebungsvektor-Messung und $\vec{v}_{ip} = \begin{pmatrix} v_{i1p} \\ v_{i2p} \end{pmatrix}$ dem mathematischen Modell aus den Gleichungen 5.9 und 5.10. In der Beziehung 6.6 ist die Kovarianzmatrix $\mathbf{C}_i$ des Meßfehlervektors $\vec{z}_i$ und die 2×2 Matrix $\mathbf{W}_i = \{w_{ijk}\}$ über die Beziehung $\mathbf{W}_i \mathbf{W}_i^T := (\mathbf{C}_i)^{-1}$ miteinander verbunden. Die Bestimmung der Bewegungsparameter erfolgt anschließend mittels einer nichtlinearen Optimierung der neuen Verlustfunktion $Q^*(\vec{p})$, wobei die Normierung der Einheitsquaternionen eingehalten werden muß.

6.4 Kovarianzpropagation

Ein statistisches Parameterschätzverfahren ist nur dann für die praktische Anwendung geeignet, wenn neben den Parameterschätzwerten auch deren Genauigkeit angegeben werden kann. Das bedeutet, daß die Unsicherheiten der Meßdaten — hier also die Kovarianzmatrizen der Fehler der Verschiebungsvektor-Schätzungen — durch das *gesamte nichtlineare Optimierungsverfahren* hindurch verfolgt werden müssen, um eine Kovarianzmatrix für die Fehler der resultierenden Parameterwerte zu erhalten. Die grundsätzliche Vorgehensweise der hierzu verwendeten sogenannten *Kovarianz-Propagation* wird beispielsweise in [Ha94] beschrieben.

Bei der Verwendung folgender Definitionen:

- die Nebenbedingung, daß das Quaternion zur Parametrisierung der Rotation den Betrag 1 haben muß, wird durch die Gleichung $c = \epsilon_0^2 + \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2 - 1 = 0$ dargestellt;
- der Lagrangemultiplikator, welcher sich aus der Minimierung unter einer Nebenbedingung ergibt, wird mit μ bezeichnet (siehe hierzu Anhang C);
- der Meßwertvektor $\vec{x}$ habe die Länge $2 \times N$ und setze sich aus N Verschiebungsvektoren zusammen entsprechend

$$\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_{2\times N} \end{pmatrix} := \begin{pmatrix} v_{11} \\ v_{21} \\ \vdots \\ v_{1N} \\ v_{2N} \end{pmatrix} \text{ und}$$

- der Gradient der Verlustfunktion Q^* nach den Parametern lautet

$$\vec{g}(\vec{x},\vec{p}) = \left(\frac{\partial Q^*}{\partial p_i}\right)$$

ergibt sich die Kovarianzmatrix der geschätzten Parameter entsprechend [Ha94] zu

$$\mathsf{Cov}\left[\begin{pmatrix}\vec{p}\\ \mu\end{pmatrix}\right] = \mathbf{A}^{-1}\ \mathbf{B}\ \mathsf{Cov}\left[\vec{x}\right]\ \mathbf{B}^T(\mathbf{A}^T)^{-1} \tag{6.7}$$

mit

$$\mathbf{A} = \begin{pmatrix} \dfrac{\partial g(\vec{x},\vec{p})}{\partial \vec{p}} & \left(\dfrac{\partial \vec{c}}{\partial p}\right)^T \\ \dfrac{\partial \vec{c}}{\partial p} & \mathbf{0} \end{pmatrix}, \tag{6.8}$$

$$\mathbf{B} = \begin{pmatrix} -\dfrac{\partial g(\vec{x},\vec{p})}{\partial \vec{x}} \\ \mathbf{0} \end{pmatrix} \text{ und} \tag{6.9}$$

$$\mathsf{Cov}\left[\vec{x}\right] = \begin{pmatrix} \mathsf{Cov}\left[\vec{v}_1\right] & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathsf{Cov}\left[\vec{v}_2\right] & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathsf{Cov}\left[\vec{v}_N\right] \end{pmatrix}, \tag{6.10}$$

wobei $\mathsf{Cov}\left[\vec{v}_i\right]$ die empirische Kovarianz zum Meßwert $\vec{v}_i$ darstellt, die in unserem Fall von dem Block-Matching-Verfahren geliefert wurde.

Durch dieses Verfahren ist es nun möglich geworden, neben der Schätzung einer Bewegung aus zwei Bildern auch die Güte dieser Schätzung in Form einer Kovarianzmatrix anzugeben. Bild 6.5 zeigt die Weiterentwicklung des in Kapitel 5 begonnenen Flußdiagramms. Mittels der robusten Schätzung und der Kovarianz-Propagation erhält man bereits eine gute Bewegungsschätzung, die aber in Kapitel 7 durch Integration von Vorwissen noch weiter verbessert wird.

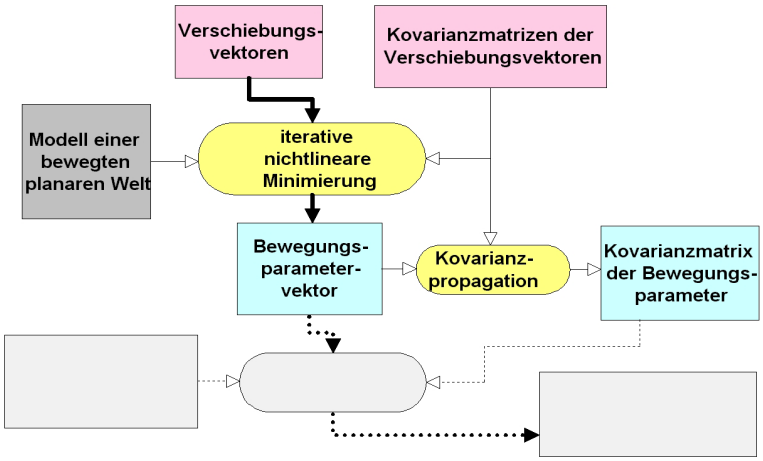


Bild 6.5: Das Verfahren der dreidimensionalen Bewegungsschätzung (Teil 2)

6.5 Ergebnisse

Das vorgestellte Verfahren der robusten Bewegungsschätzung wurde an einer Vielzahl von realen, aber auch synthetischen Sequenzen getestet. Hier werden die Ergebnisse aus drei grundlegend unterschiedlichen Testsituationen vorgestellt. Im einzelnen sind dies

1. die Fahrt über eine synthetisch generierte Ebene mit Struktur,
2. synthetisch generierte Verkehrsbildsequenzen, die neben einer flachen und ebenen Straßenoberfläche ebenfalls dreidimensionale Elemente enthalten und
3. mit einer Videokamera aufgezeichnete Verkehrsbildsequenzen[2].

Die beiden synthetischen Szenen haben den besonderen Vorteil, daß die Bewegungs- und Geometriedaten exakt bekannt sind.

6.5.1 Testsituation 1

Bild 6.6 zeigt im oberen Teil zwei aufeinanderfolgende Bilder einer synthetischen Bildsequenz. Diese Sequenz zeichnet sich dadurch aus, daß einerseits das mathematische Modell einer bewegten Ebene voll erfüllt ist, so daß keine dreidimensionalen Objekte vorkommen, und andererseits, daß keine Ausreißer in den Verschiebungsvektoren auftreten. Die vollkommen flache Welt, durch die sich ein virtuelles Fahrzeug bewegt, ist ein guter Test, ob das vorgestellte Verfahren korrekt implementiert wurde. Neben einer Translation führt das virtuelle Fahrzeug auch eine Rotation von einem Grad pro Bildpaar aus. Die Texturierung der generierten Ebene wurde derart gewählt, daß *keine* Ausreißer in den Verschiebungsvektoren vorkommen.

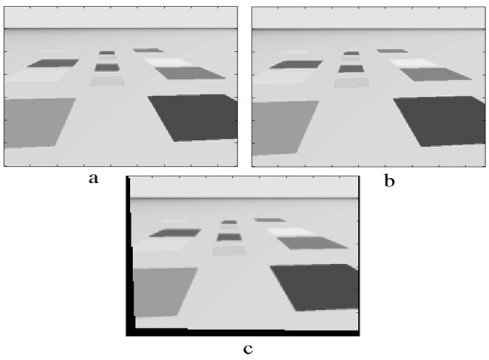


Bild 6.6: Testsituation 1

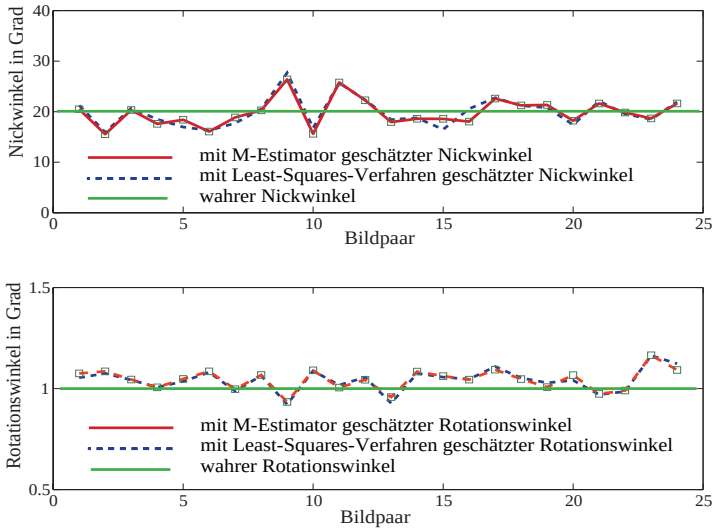


Bild 6.7: geschätzter Rotations- und Nickwinkel für Testsituation 1

Bild 6.7 zeigt eine Darstellung des geschätzten Rotationswinkels (um die geschätzte Rotationsachse) und des geschätzten Nickwinkels der Kamera für 24 Bildpaare bei Verwendung des Least-Squares-Verfahrens bzw. des M-Estimators. Wie bei normalverteilten Fehlern der Eingangsdaten

[2]mit freundlicher Genehmigung der Robert Bosch GmbH

nicht anders zu erwarten, sind die geschätzten Größen der beiden Verfahren im wesentlichen identisch. Im Falle der Least-Squares-Schätzung ergibt sich für den Nickwinkel ein Mittelwert von $\widehat{\psi_{ls}} = 19.945°$ und eine Standardabweichung von $\sigma[\psi_{ls}] = 2.882°$. Der Mittelwert des Rotationswinkels beträgt $\widehat{\alpha_{ls}} = 1.042°$ und seine Standardabweichung $\sigma[\alpha_{ls}] = 0.056°$. Dies stellt eine recht gute Schätzung der wahren Werte $\psi = 20°$ und $\alpha = 1°$ dar. Da sich der Nickwinkel der Kamera direkt aus dem Normalenvektor der Ebene berechnet, bedeutet obiges Ergebnis jedoch auch, daß der Normalenvektor der Ebene aus den Bilddaten deutlich schwieriger zu schätzen ist als beispielsweise eine Rotation der Kamera beziehungsweise des Fahrzeugs.

6.5.2 Testsituation 2

Die Bilder 6.8a und 6.8b zeigen zwei Bilder aus einer synthetisch generierten Verkehrssequenz. Diese Sequenz besteht aus 80 Bildern, von denen die ersten 50 Bilder eine Geradeausfahrt darstellen, während die darauffolgenden 30 Bilder eine Rechtskurve mit einer Rotation von genau einem Grad pro Bildpaar darstellen. Die Sequenz enthält außerdem dreidimensionale Strukturen und führt daher bewußt zu Ausreißern in den Verschiebungsvektoren, bezogen auf das planare Weltmodell. Wie in Testsituation 1 wurden nun die Bewegungsparameter und der Normalenvektor der Straße für diese virtuelle Verkehrssituation geschätzt, wobei einerseits eine gewöhnliche Least-Squares-Schätzung erfolgte und andererseits die robuste Schätzmethode der M-Estimation mit der „Fair-“Funktion verwendet wurde. Wie in Bild 6.9 deutlich zu erkennen ist, führen die Ausreißer beim einfachen Least-Squares-Verfahren zu großen Fehlern in der Schätzung der Bewegungsparameter. Die durchgezogene Linie, die das Resultat der M-Schätzung wiedergibt, zeigt eindrucksvoll, daß eine sehr viel bessere Bewegungsschätzung mit den Methoden der robusten Statistik erzielt werden kann.

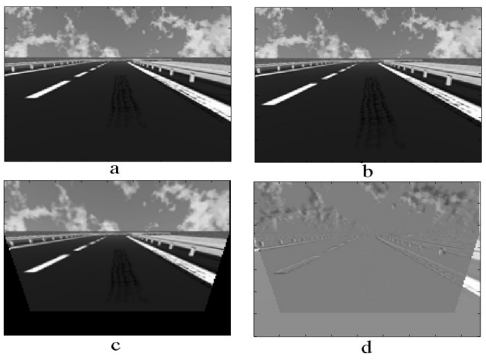


Bild 6.8: Testdatensatz 2: (a,b) Zwei Bilder einer synthetischen Verkehrssequenz, (c) bewegungskompensiertes Bild unter Verwendung der geschätzten Parameter und (d) Differenzbild zwischen den Bildern (c) und (a) (siehe hierzu auch Abschnitt 5.4)

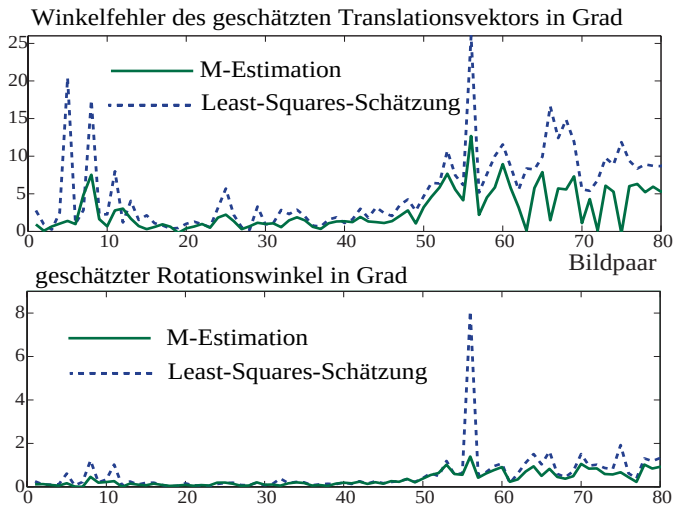


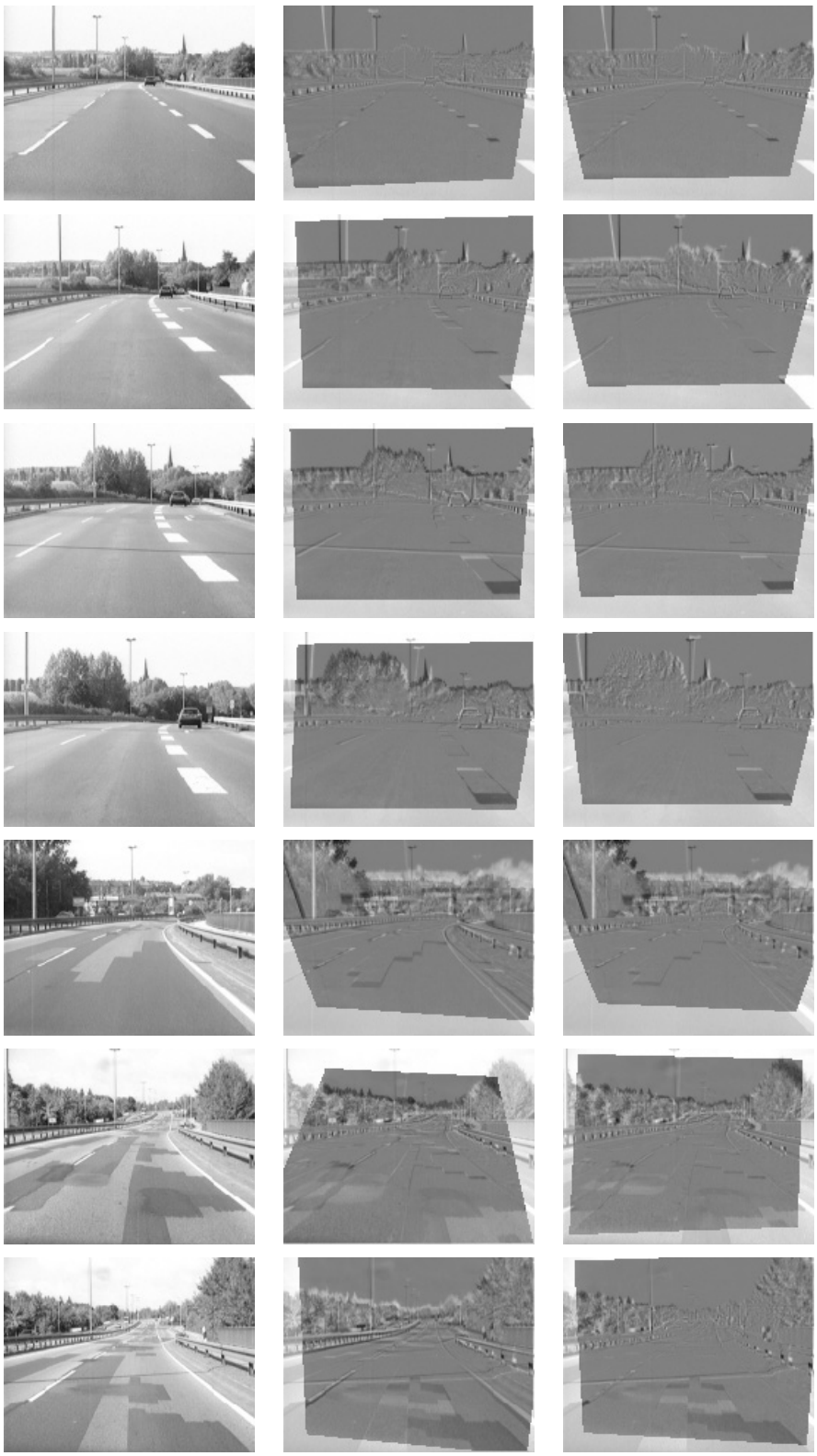
Bild 6.9: Geschätzter Rotationswinkel und Fehler des geschätzten Nickwinkels für Testdatensatz 2

Der Winkelfehler des geschätzten Translationsvektors liegt beim M-Estimator bei maximal 11°, im Vergleich zu maximal 26° beim Least-Squares-Schätzer. Weiterhin wird der Rotationswinkel bei der Kurvenfahrt vom M-Estimator deutlich robuster geschätzt als beim Least-Squares-Schätzer. Beim Bildpaar $55 \longleftrightarrow 56$ führen Ausreißer sogar zu der völligen Fehlschätzung des

Rotationswinkels von 8° beim Least-Squares-Verfahren, wohingegen der Fair-Schätzer sehr nah um den wahren Wert von 1° schwankt.

6.5.3 Testsituation 3

Testsituation 3 stellt die bereits vorgestellte Verkehrssequenz dar, die aus einem fahrenden Auto in der Umgebung von Hildesheim aufgenommen wurde. Bild 6.10 zeigt neben den bewegungskompensierten Differenzbildern der Least-Squares-Schätzung auch die entsprechenden Ergebnisbilder der M-Estimation bei Verwendung der Fair-Funktion. Man erkennt eine deutliche Verbesserung der Bewegungsschätzung, die allerdings bei diesen echten Verkehrsszenen immer noch nicht als „gut“ bezeichnet werden kann. Wie bereits erläutert, sollte bei einer perfekten Bewegungsschätzung die gesamte Fahrbahn in einem mittleren Grau erscheinen, so daß anschließend dreidimensionale Objekte detektiert werden können. Deshalb wird im nächsten Kapitel eine Erweiterung des Verfahrens mit statischem Vorwissen vorgeschlagen.



a b c

Bild 6.10: Einige Bilder der sog. Hildesheim-Sequenz (a) zusammen mit den bewegungskompensierten Differenzbildern bei der Least-Squares-Schätzung (b) und bei der robusten Fair-Schätzung (c)

Kapitel 7

Integration von Vorwissen in die Bewegungsschätzung

7.1 Überblick

Die robuste Bewegungsschätzung mittels M-Estimatoren liefert deutlich bessere Ergebnisse als die gewöhnliche Least-Squares-Schätzung. Wie allerdings in Bild 6.10 deutlich zu erkennen ist, kann die Bewegungschätzung auch bei Verwendung von M-Estimatoren mangelhaft sein, und zwar insbesondere dann, wenn viele Ausreißer in den Verschiebungsvektoren auftreten. Eine solch ungenaue Bewegungsschätzung äußert sich dadurch, daß die geschätzte Kovarianzmatrix des Parametervektors große Eigenwerte besitzt. In diesem Kapitel wird eine Methode vorgestellt, die Bewegungsschätzung dadurch zu stabilisieren, daß Vorwissen über die Art der Bewegung in die Bewegungsschätzung integriert wird. Das Vorwissen sollte allerdings nur für diejenigen Komponenten des Parametervektors zur Verfügung gestellt werden, die durch die Zwei-Bild-Schätzung nur ungenau bestimmt werden konnten[1]. Es ist also bedeutsam, daß die Berücksichtigung von Vorwissen nicht dazu führt, daß die geschätzten Bewegungsparameter stets in Richtung des Vorwissens „gezogen“ werden. Wurde also eine sehr *genaue* Zwei-Bild-Schätzung erreicht, so sollte das Vorwissen kaum einen Einfluß auf das Schätzergebnis ausüben.

7.2 Die Kombination zweier Parameterschätzungen

Der Schlüssel zu besseren Schätzwerten liegt also in der Ausnutzung von Vorwissen über die Verteilung der Parameter im Parameterraum, was beispielsweise durch einen Erwartungswert und eine zugehörige Kovarianzmatrix spezifiziert werden kann. Hierzu muß allerdings vorher geklärt werden, wie dieses Vorwissen und die aktuelle Zwei-Bild-Schätzung in statistisch optimaler Weise miteinander verknüpft werden können.

Es wird angenommen, daß die *Wahrscheinlichkeitsverteilung* der Bewegungsparameter gegeben sei, oder zumindest ihre wichtigsten Momente (siehe hierzu Anhang F), wie der Erwartungswert $\tilde{\vec{x}}$ und die Kovarianzmatrix $\tilde{\mathbf{C}}_x$. Die dazu entsprechenden Größen, die aus den aktuellen *Messungen* stammen, werden im folgenden mit $\widehat{\vec{x}}$ und $\widehat{\mathbf{C}}_x$ bezeichnet.

[1] also eine entsprechende Anzahl von großen Eigenwerten besitzt

In [Bi77] auf Seite 16 finden sich die Beziehungen zur statistisch optimalen Kombination[2] zweier unabhängiger Schätzwerte zusammen mit ihren Unsicherheiten in Form einer Kovarianzmatrix,

$$\vec{x}_{best} = \left(\widehat{\mathbf{C}}_x^{-1} + \tilde{\mathbf{C}}_x^{-1}\right)^{-1} \cdot \left(\widehat{\mathbf{C}}_x^{-1}\widehat{\vec{x}} + \tilde{\mathbf{C}}_x^{-1}\tilde{\vec{x}}\right) \tag{7.1}$$

$$\mathbf{C}_{best} = \left(\widehat{\mathbf{C}}_x^{-1} + \tilde{\mathbf{C}}_x^{-1}\right)^{-1} , \tag{7.2}$$

welche aus dem *Gauß-Markov-Theorem* der Schätztheorie [SW94] abgeleitet werden können (Herleitung siehe Anhang E).

Wenn man die Meßwerte und das Vorwissen in der beschriebenen Weise kombiniert, ist es überaus wichtig, die Kovarianzmatrix zum Vorwissen realistisch abzuschätzen. Falls nämlich die Einträge der Kovarianzmatrix systematisch zu klein geschätzt werden, übt das Vorwissen zu wenig Einfluß auf das Endresultat aus. Werden andererseits die Einträge der Kovarianzmatrix ständig zu groß geschätzt, so bewirkt dies, daß das Endresultat zu stark in die Richtung des Erwartungswertes aus dem Vorwissen „gezogen“ wird. Es ist also erforderlich, eine auf physikalischen Überlegungen basierende Kovarianzmatrix der Bewegungsparameter abzuschätzen.

Die Anwendung der Formeln 7.1 und 7.2 für das hier betrachtete Problem der monokularen Bewegungsschätzung führt jedoch auf ein zunächst unerwartetes Problem. Die Kovarianzmatrizen der aus den Messungen bestimmten Größen weisen nämlich einen *Rangabfall* auf. Der Grund hierfür liegt in der Parametrisierung der Rotation mittels Quaternionen, denn diese unterliegen, wie bereits erläutert, einer Nebenbedingung. Aufgrund dieser Randbedingung sind die einzelnen Komponenten des Einheitsquaternions nicht unabhängig voneinander, was sich folglich in einem Rangabfall der Kovarianzmatrix äußert. Dieser Rangabfall hat zur Folge, daß die Inversion der Kovarianzmatrix $\widehat{\mathbf{C}}_x$ gemäß den Gleichungen 7.1 und 7.2 zunächst nicht möglich ist. Im folgenden Abschnitt wird gezeigt, wie man dieses Problem umgehen und dennoch eine Kombination von Messung und Vorwissen realisieren kann.

7.3 Kombination von Meßwerten und Vorwissen im Falle von Kovarianzmatrizen mit Rangabfall

Ein Rangabfall in einer Matrix bedeutet, daß bestimmte Spalten der Kovarianzmatrix linear abhängig sind, oder, mit anderen Worten, daß bestimmte Linearkombinationen der gesuchten Parameter exakt bestimmt sind. Es ist nun naheliegend, durch eine geeignete Drehung die Matrix mit Rangabfall so zu transformieren, daß man die exakt bestimmten Richtungen im Parameterraum von den anderen Richtungen trennen kann. Im folgenden wird angenommen, daß nur eine der beiden zu kombinierenden Matrizen einen Rangabfall hat, und zwar $\hat{\mathbf{C}}_x$. Dagegen habe die Matrix $\tilde{\mathbf{C}}_x$ vollen Rang.

Eine Hauptachsentransformation wird dazu eingesetzt, die exakt bestimmten Richtungen im Parameterraum von den anderen Richtungen zu trennen[3],

$$\hat{\mathbf{C}}_x = \mathbf{U}\hat{\mathbf{S}}_x\mathbf{U}^T ,$$

wobei $\hat{\mathbf{S}} = \mathrm{diag}(s_1, \ldots, s_p, 0, \ldots, 0)$ und $\mathbf{U}$ orthogonal sind. Hierbei werden alle Diagonalelemente s_i, die kleiner als eine bestimmte Schranke ϵ sind, auf Null gesetzt, so daß der Rang von

[2] im Sinne von „Least-Squares“

[3] $\hat{\mathbf{C}}_x$ ist symmetrisch

$\hat{\mathbf{C}}_x$ auf p festgelegt wird. Mit der gleichen orthogonalen Matrix $\mathbf{U}$ wird auch $\tilde{\mathbf{C}}_x$ transformiert,

$$\tilde{\mathbf{C}}_x = \mathbf{U}\tilde{\mathbf{S}}_x\mathbf{U}^T,$$

wobei $\tilde{\mathbf{S}}_x$ nicht notwendigerweise diagonal sein wird. Die beiden Matrizen $\hat{\mathbf{S}}_x = \mathbf{U}^T\hat{\mathbf{C}}_x\mathbf{U}$ und $\tilde{\mathbf{S}}_x = \mathbf{U}^T\tilde{\mathbf{C}}_x\mathbf{U}$ lassen sich dann folgendermaßen partitionieren,

$$\hat{\mathbf{S}}_x = \begin{pmatrix} \hat{\mathbf{S}}_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \quad \hat{\mathbf{S}}_{11} = \mathrm{diag}(s_1 \ \ldots, \ s_p) \tag{7.3}$$

$$\tilde{\mathbf{S}}_x = \begin{pmatrix} \tilde{\mathbf{S}}_{11} & \tilde{\mathbf{S}}_{12} \\ \tilde{\mathbf{S}}_{12}^T & \tilde{\mathbf{S}}_{22} \end{pmatrix}. \tag{7.4}$$

Es wird nun gezeigt, daß diese Transformation der Kovarianzmatrizen einer Transformation der beiden Schätzwerte $\hat{\vec{x}}$ und $\tilde{\vec{x}}$ entspricht, mit

$$\hat{\vec{x}}' = \mathbf{U}^T\hat{\vec{x}} \ \text{ und } \ \tilde{\vec{x}}' = \mathbf{U}^T\tilde{\vec{x}}\,, \tag{7.5}$$

denn es gilt

$$\mathsf{Cov}\left[\hat{\vec{x}}'\right] = \mathsf{E}\left[\hat{\vec{x}}'\hat{\vec{x}}'^T\right] = \mathsf{E}\left[\mathbf{U}^T\hat{\vec{x}}\hat{\vec{x}}^T\mathbf{U}\right] \tag{7.6}$$

$$= \mathbf{U}^T\mathsf{E}\left[\hat{\vec{x}}\hat{\vec{x}}^T\right]\mathbf{U} = \mathbf{U}^T\mathsf{Cov}\left[\hat{\vec{x}}\right]\mathbf{U} \tag{7.7}$$

$$= \mathbf{U}^T\hat{\mathbf{C}}_x\mathbf{U}\,. \tag{7.8}$$

Folglich sind die Matrizen $\hat{\mathbf{S}}_x$ und $\tilde{\mathbf{S}}_x$ die Kovarianzmatrizen der mit (7.5) transformierten Größen. An dieser Stelle wird der Nutzen der Transformation (7.5) deutlich, denn die letzten $N-p$ Spalten (bzw. Zeilen) von $\hat{\mathbf{S}}_x$ sind Null, d.h. die letzten $N-p$ Elemente von $\hat{\vec{x}}'$ sind exakt bekannt. Für die ersten p Elemente ergibt sich dagegen die Kovarianzmatrix $\mathbf{S}_{11}$ mit vollem Rang p. Die Kombination der beiden Schätzungen $\hat{\vec{x}}$ und $\tilde{\vec{x}}$, die wegen des Rangabfalls von $\hat{\mathbf{C}}_x$ nicht mit den Formeln (7.1) und (7.2) möglich war, läßt sich also im transformierten Raum der gestrichenen Größen durchführen.

Durch diese Überlegungen wurde also das ursprüngliche Problem

kombiniere $\hat{\vec{x}}$ und $\tilde{\vec{x}}$ zu $\vec{x}_{best}$

überführt in das Problem

kombiniere $\hat{\vec{x}}'$ und $\tilde{\vec{x}}'$ zu $\vec{x}'_{best}$.

Die zweite Formulierung des Problems hat den Vorteil, daß die letzten $N-p$ Komponenten von $\hat{\vec{x}}'$ exakt bekannt sind, d.h. diese Komponenten von $\vec{x}'_{best}$ werden direkt aus $\hat{\vec{x}}'$ übernommen und die entsprechenden Elemente von $\tilde{\vec{x}}$ können einfach ignoriert werden.

Da die Matrizen $\hat{\mathbf{S}}_{11}$ und $\tilde{\mathbf{S}}_{11}$ vollen Rang haben und folglich ihre Inversen existieren, können die ersten p Komponenten von $\vec{x}'_{best}$ mit den Formeln (7.1) und (7.2) bestimmt werden. Hierzu werden die Vektoren $\hat{\vec{x}}'_p$ und $\tilde{\vec{x}}'_p$ definiert, die jeweils die ersten p Komponenten von $\hat{\vec{x}}'$ und $\tilde{\vec{x}}'$ enthalten. Entsprechend erhalten die Vektoren $\hat{\vec{x}}'_{N-p}$ und $\tilde{\vec{x}}'_{N-p}$ die letzten $N-p$ Komponenten. Die Partitionierung der Vektoren $\hat{\vec{x}}'$ und $\tilde{\vec{x}}'$ resultiert dann in der Form

$$\hat{\vec{x}}' =: \begin{pmatrix} \hat{\vec{x}}'_p \\ \hat{\vec{x}}'_{N-p} \end{pmatrix} \text{ und } \tilde{\vec{x}}' =: \begin{pmatrix} \tilde{\vec{x}}'_p \\ \tilde{\vec{x}}'_{N-p} \end{pmatrix}.$$

Die Anwendung der Formeln (7.1) und (7.2) ergibt folglich

$$\boxed{\begin{array}{rcl} \mathbf{S}_{11,best} & = & (\hat{\mathbf{S}}_{11}^{-1} + \tilde{\mathbf{S}}_{11}^{-1})^{-1} \\ \vec{x}'_{p,best} & = & (\hat{\mathbf{S}}_{11}^{-1} + \tilde{\mathbf{S}}_{11}^{-1})^{-1}(\hat{\mathbf{S}}_{11}^{-1}\hat{\vec{x}}'_p + \tilde{\mathbf{S}}_{11}^{-1}\tilde{\vec{x}}'_p)\,. \end{array}}$$

Anschließend können $\vec{x}_{best}$ und $\mathbf{S}_{best}$ zu

$$\vec{x}'_{best} = \begin{pmatrix} \vec{x}'_{p,best} \\ \hat{\vec{x}}'_{N-p} \end{pmatrix} \tag{7.9}$$

$$\mathbf{S}_{best} = \begin{pmatrix} \mathbf{S}_{11,best} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{7.10}$$

zusammengesetzt werden, so daß abschließend nur noch die Rücktransformation entsprechend

$$\vec{x}_{best} = \mathbf{U}\vec{x}'_{best} \tag{7.11}$$

$$\mathbf{C}_{best} = \mathbf{U}\mathbf{S}_{best}\mathbf{U}^T \tag{7.12}$$

durchgeführt werden muß. Damit ist es gelungen, eine Meßwertkombination bei rangdefizitären Kovarianzmatrizen zu realisieren.

Im Abschnitt 7.5 wird gezeigt, daß die Anwendung dieses Verfahrens für das hier betrachtete Problem der monokularen Bewegungsschätzung zu einer deutlichen Stabilisierung der Schätzwerte führt.

7.4 Das Gesamtverfahren zur Bewegungsschätzung aus monokularen Bildsequenzen

In Bild 7.1 ist nun das Gesamtverfahren zur statistischen Bewegungsschätzung aus monokularen Bildsequenzen als Blockdiagramm dargestellt. Als abschließender Teil wurde im Vergleich zu Abb. 6.5 die Kombination der Zwei-Bild-Schätzung mit dem Vorwissen eingefügt. Mit diesem Verfahren ist es möglich geworden, eine sehr gute Bewegungsschätzung aus monokularen Verkehrssequenzen zu erhalten.

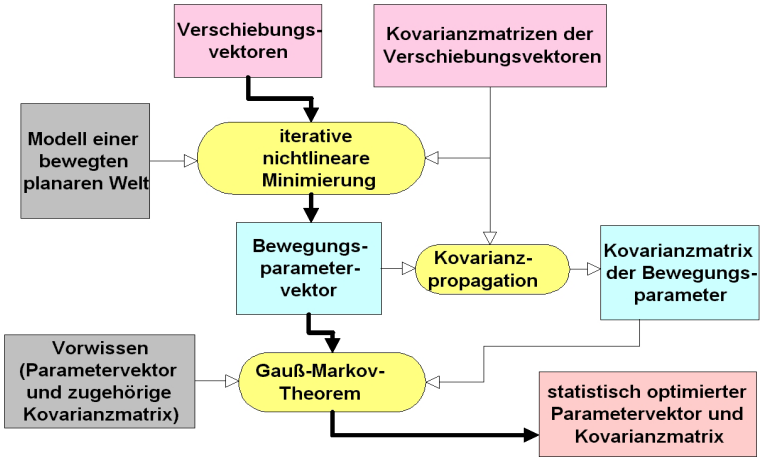


Bild 7.1: Das Verfahren der dreidimensionalen Bewegungsschätzung (Teil 3)

7.5 Ergebnisse

Im oberen Teil von Bild 7.2 sind zwei Bilder einer realen Bildsequenz dargestellt, bei der sich nach etwa 92 Bildern Geradeausfahrt eine Kurve anschließt. Die wahren Bewegungsparameter sind nicht bekannt; es kann jedoch aus Vergleichen mit synthetischen Bildsequenzen ein Nickwinkel von etwa zehn Grad und eine Rotation von etwa einem Grad bei der Kurvenfahrt abgeschätzt werden. Diese reale Verkehrssequenz stellt die größten Anforderungen an die Schätzverfahren, da in ihr sehr viele Ausreißer in den Verschiebungsvektoren aufgrund von dreidimensionalen Objekten und periodischen Strukturen auf der Straße auftreten.

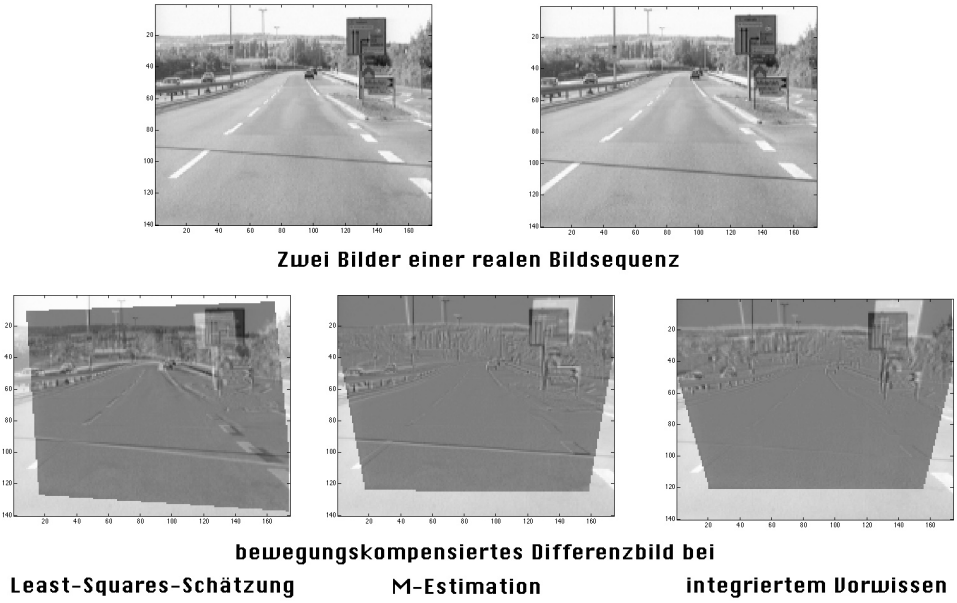


Bild 7.2: Zwei Bilder der Hildesheim-Sequenz mit ihren bewegungskompensierten Differenzbildern

In der unteren Zeile von Bild 7.2 sind die bewegungskompensierten Differenzbilder bei den verschiedenen Schätzmethoden dargestellt. Es ist deutlich erkennbar, daß durch die Integration von Vorwissen in die Schätzung, die Bewegungsparameter viel exakter bestimmt werden konnten, denn nur in dem rechten Bild erscheint die Straße in einem homogenen Grau (siehe hierzu auch Abschnitt 5.4).

In Bild 7.3 sind der geschätzte Nickwinkel der Kamera und der geschätzte Rotationswinkel[4] für den Fall der Least-Squares-Schätzung (strichpunktierte Linie), der M-Estimation (punktierte Linie) und der Integration des Vorwissens (durchgezogene Linie) für die Hildesheim-Sequenz dargestellt. Die punktierte, rote Linie zeigt deutlich, daß unter Verwendung von M-Estimatoren der Einfluß von Ausreißern auf die Bewegungsschätzung zwar reduziert, aber nicht eliminiert werden kann. In dieser Darstellung zeigt sich weiterhin, daß eine deutliche Stabilisierung der Nickwinkelschätzung bei Integration von Vorwissen erzielt werden konnte. Es kann an dieser Stelle nicht erwartet werden, daß die geschätzten Größen in Bild 7.3 glatt verlaufen, da einerseits Vibrationen vom Motor und andererseits Unebenheiten auf der Straßenoberfläche mehr oder weniger große Schwankungen der Bewegungsgrößen zur Folge haben. Außerdem ist an den Schätzungen

[4] um die geschätzte Rotationsachse

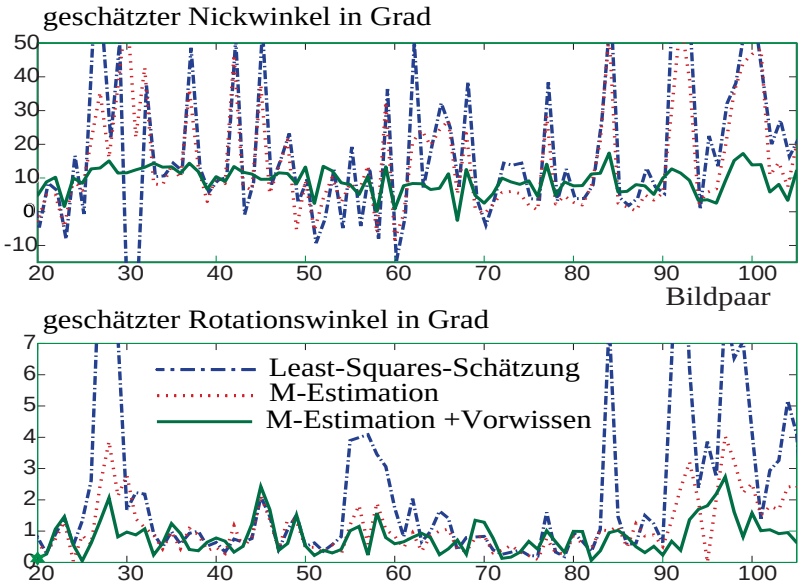


Bild 7.3: Geschätzter Rotations- und Nickwinkel für ca. 100 Bildpaare der Hildesheim-Sequenz bei Verwendung eines Least-Squares-Verfahrens, einer M-Esimation und einer M-Estimation mit zusätzlichem Vorwissen

des Rotationswinkels deutlich zu erkennen, daß die Verwendung von Vorwissen nicht notwendigerweise die Schätzwerte in Richtung der Mittelwerte „zieht“, denn das Vorwissen bezüglich der Rotation ist derart, daß keine Rotation vorliegt[5], aber trotzdem werden während der Rechtskurve Rotationen von bis zu 2.5° geschätzt. Wenn also die Zwei-Bild-Bewegungsschätzung einen bestimmten Parameter mit einer hohen Konfidenz ermitteln kann, so zeigen auch die Ergebnisse, daß das Vorwissen kaum einen Einfluß auf diesen Parameter hat.

Abschließend sind noch einige Bilder der Hildesheim-Sequenz mit den bewegungskompensierten Differenzbildern exemplarisch in Bild 7.4 dargestellt. Auch bei den anspruchvollsten Eingangsbildern der Sequenz wurde durch die Integration des Vorwissens eine zufriedenstellende Bewegungsschätzung erreicht.

[5]mit einer hohen Ungewißheit in den entsprechenden Komponenten der Kovarianzmatrix

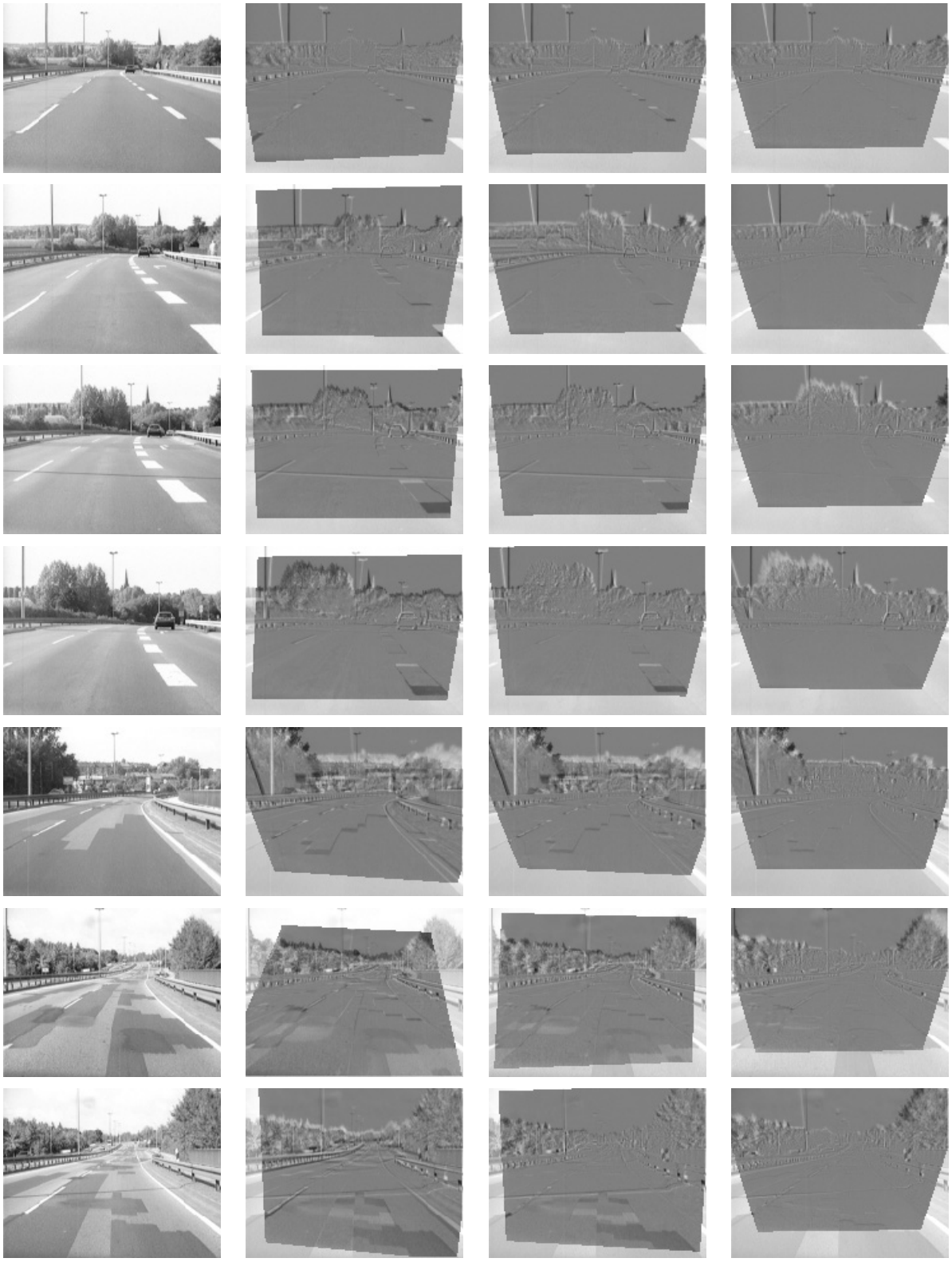


Bild 7.4: Einige Bilder der sog. Hildesheim-Sequenz (a) zusammen mit den bewegungskompensierten Differenzbildern bei der Least-Squares-Schätzung (b), bei der robusten Fair-Schätzung (c) und bei Integration von Vorwissen (d)

Teil II

Bewegungsschätzung mit neuronalen Methoden

Kapitel 8

Zellulare Neuronale Netzwerke

8.1 Neuronale Netzwerke

Eine Anordnung von miteinander verbundenen und wechselwirkenden Zellen wird üblicherweise als ein *Neuronales Netzwerk* bezeichnet. Ein natürliches neuronales Netz stellt beispielsweise das menschliche Gehirn dar und seine immense Leistungsfähigkeit in der Informationsverarbeitung hat Wissenschaftler dazu animiert, *Künstliche Neuronale Netzwerke (KNN)* zu definieren. In Anlehnung an das biologische Vorbild werden die zu verarbeitenden Informationen üblicherweise an die Eingangszellen eines Künstlichen Neuronalen Netzwerks angelegt, und nach einer gewissen Zeitspanne liegen die jeweiligen Ergebnisse an den sogenannten Ausgangszellen vor. Manche Künstlichen Neuronalen Netzwerke gehen noch einen Schritt weiter und lösen sich völlig von dem biologischen Vorbild. So spielt bei rekursiven Netzwerken, wie beispielsweise den Zellularen Neuronalen Netzen, die Rückkopplung der Zellausgänge auf die Zellzustände eine tragende Rolle. Es hat sich gezeigt, daß Künstliche Neuronale Netzwerke in vielen Fällen Systemen überlegen sind, die auf empirischem Wege[1] entwickelt wurden. So werden schon heute Künstliche Neuronale Netzwerke in den Bereichen Signalverarbeitung, Mustererkennung oder Robotik mit großem Erfolg eingesetzt.

8.2 Zellulare Neuronale Netze (CNN)

Eine spezielle, von Leon Chua [CY88] angegebene Form der Künstlichen Neuronalen Netzwerke bilden die *Zellularen Neuronalen Netze (CNN)*. Sie unterscheiden sich im wesentlichen von anderen KNN durch eine lokale Vernetzung der Zellen und eignen sich daher insbesondere für eine Implementierung als VLSI-Schaltung. CNN bilden ein neues Paradigma zur *nichtlinearen mehrdimensionalen Signalverarbeitung* und eröffnen dadurch völlig neue Möglichkeiten in der Bildverarbeitung. Die parallele Struktur der CNN führt außerdem zu einer extrem hohen Verarbeitungsgeschwindigkeit im Vergleich zu herkömmlichen Digitalrechnern, was den Einsatz von CNN in der *Bildverarbeitung* auch in dieser Hinsicht in hohem Maße interessant macht. Dabei wird jedem Neuron des CNN ein Bildpixel zugeordnet, so daß eingangsseitig direkt das zu verarbeitende Bild angelegt wird und ausgangsseitig das Ergebnisbild anliegt. Nicht nur viele der bereits etablierten Bildverarbeitungsalgorithmen, sondern auch völlig neuartige Anwendungen wurden bereits unter Verwendung von CNN realisiert. Die herausragenden Eigenschaften von

[1] beispielsweise mit statistischen Methoden

CNN sind ihre besondere Robustheit und die massiv parallele Informationsverarbeitung, die zu extrem hohen Verarbeitungsgeschwindigkeiten im Bereich von *Nano-Sekunden* führt.

8.2.1 Die mathematische Beschreibung von CNN

Ein CNN wird beschrieben durch ein System von gekoppelten gewöhnlichen nichtlinearen Differentialgleichungen. CHUA hat in [CR97] eine sehr allgemeine Definition von Zellularen Neuronalen Netzwerken angeführt:

> Ein CNN ist eine Anordnung von *lokal verknüpften* Zellen, von der jede ein dynamisches System darstellt, das einen *Eingang*, einen *Ausgang* und einen *Zustand*, der sich nach festgelegten dynamischen Gesetzen entwickelt, besitzt.

Durch die Beziehung

$$\begin{aligned} C\frac{\mathrm{d}x_{i,j}^{m}(t)}{\mathrm{d}t} &= -\frac{1}{R}x_{i,j}^{m}(t) + \sum_{m'}\Bigg(\sum_{k,l\in N_r(i,j)} A_{i,j,k,l}^{m'm}\left(y_{k,l}^{m'}(t)\right) \\ &\quad + \sum_{k,l\in N_r(i,j)} B_{i,j,k,l}^{m'm}\left(u_{k,l}^{m'}(t)\right) + \sum_{k,l\in N_r(i,j)} A'^{m'm}_{i,j,k,l}\left(y_{k,l}^{m'}(t-\tau_A)\right) \\ &\quad + \sum_{k,l\in N_r(i,j)} B'^{m'm}_{i,j,k,l}\left(u_{k,l}^{m'}(t-\tau_B)\right)\Bigg) + I_{i,j}^{m}(t)\,. \end{aligned} \tag{8.1}$$

ist eine spezielle Beschreibungsweise von CNN angegeben, in der die Zustandsänderungen der Zellen des Netzwerkes durch ein System von gekoppelten Differentialgleichungen beschrieben werden. Weiterhin gilt einheitlich für das gesamte Netzwerk die Ausgangsfunktion

$$y_{i,j}^{m}(t) = f\left(x_{i,j}^{m}(t)\right)\,. \tag{8.2}$$

Die verwendeten Größen haben dabei folgende Bedeutungen:

- $u_{i,j}^{m}(t)$ ist die Eingangsgröße der Zelle an der Position (i,j) in der m-ten Netzebene zum Zeitpunkt t,
- $x_{i,j}^{m}(t)$ ist der Zustand der Zelle an der Position (i,j) in der m-ten Netzebene zum Zeitpunkt t,
- $y_{i}^{m}(t)$ ist die Ausgangsaktivität der i-ten Zelle in der m-ten Netzebene zum Zeitpunkt t,
- $N_r(i,j)$ bezeichnet den Nachbarschaftsbereich um die Zelle an der Position (i,j) mit dem Radius r,
- C und R charakterisieren die für eine schaltungstechnische Realisierung wichtigen elektrischen Eigenschaften einer Zelle in Form von Kapazität und ohmschem Widerstand (siehe Abschnitt 8.2.2),
- $A_{i,j,k,l}^{m'm}(.)$ ist eine nichtlineare Funktion, welche die Rückkopplung des Ausgangs der Zelle an der Position (k,l) in der m'-ten Netzebene auf die Zelle an der Position (i,j) in der m-ten Netzebene charakterisiert,

- $B^{m'm}_{i,j,k,l}(.)$ ist eine nichtlineare Funktion, welche die nichtlinearen Eingangsgewichte der Zelle an der Position (k,l) in der m'-ten Netzebene auf die Zelle an der Position (i,j) in der m-ten Netzebene beschreibt,
- τ_A und τ_B werden häufig *delaytime* genannt und beschreiben die zeitliche Verzögerung mit der Rückkopplung und Eingang auf die Zelle i wirken,
- $I^m_{i,j}(t)$ ist eine zeitabhängige Schwellwertfunktion oder auch „Bias“ genannt, die für jede Zelle des Netzwerks unterschiedlich sein kann,
- $f(.)$ beschreibt schließlich, über welche Funktionsabhängigkeit der Ausgang einer Zelle mit ihrer Zustandsgröße zusammenhängt.

In praktischen Anwendungen kommen jedoch meist vereinfachte Formen von Zellularen Neuronalen Netzwerken zum Einsatz. So wird häufig die Kopplung der Zellen durch Polynomfunktionen beschrieben, da diese vergleichsweise einfach in schaltungstechnische Realisierungen umgesetzt werden können. Werden weiterhin translationsinvariante CNN betrachtet, bei denen die Kopplungsparameter und $I_i(t)$ nicht von der Lage der Zelle im Netz abhängig ist und nur noch relative Abstände von Bedeutung sind, so ergibt sich die Zustandsgleichung einer Zelle zu

$$\begin{aligned} C\frac{dx_{i,j}(t)}{dt} &= -\frac{1}{R}x_{i,j}(t) + \sum_{k,l\in\mathcal{N}_r(i,j)} \mathcal{A}_{i-k,j-l}(y_{k,l}(t)) + \sum_{k,l\in\mathcal{N}_r(i,j)} \mathcal{B}_{i-k,j-l}(u_{k,l}(t)) + I \quad (8.3)\\ \text{mit} \quad i &= 1,\ldots,N\ , \end{aligned}$$

wobei $\mathcal{A}_{i-k,j-l}(y_{k,l}(t))$ und $\mathcal{B}_{i-k,j-l}(u_{k,l}(t)))$ Polynome der Ausgangs- bzw. Eingangssignale darstellen, welche folgendermaßen spezifiziert werden können

$$\mathcal{A}_{i-k,j-l}(y_{k,l}(t)) = \sum_{d=1}^{D} k_{a,i-k,j-l,d}\cdot(y_{k,l}(t))^d \quad (8.4)$$

$$\mathcal{B}_{i-k,j-l}(u_{k,l}(t)) = \sum_{d=1}^{D} k_{b,i-k,j-l,d}\cdot(u_{k,l}(t))^d\ . \quad (8.5)$$

Der Parameter D stellt den Grad der Polynomfunktion dar, so daß beispielsweise $D=2$ eine quadratische Kopplung beschreibt. Die Polynomkoeffizienten $k_{a,i-k,j-l,d}$ und $k_{b,i-k,j-l,d}$ und der sogenannte Bias I bilden den Satz von Parametern, welche das Verhalten des CNN maßgeblich bestimmen. Zur eindeutigen Beschreibung des Verhaltens des CNN müssen außerdem die Anfangszustände der Zellen und die Randbedingungen des CNN gegeben sein.

Das sogenannte *Standard-CNN* stellt eine weitere Spezialisierung des CNN Konzeptes dar. Wie in (8.6) und (8.7) beschrieben, basiert es auf einem Netzwerk mit linear gekoppelten Zellen.

$$C\frac{dx_{i,j}(t)}{dt} = -\frac{1}{R}x_{i,j}(t) + \sum_{k,l\in\mathcal{N}_r(i,j)} a_{i-k,j-l}y_{k,l}(t) + \sum_{k,l\in\mathcal{N}_r(i,j)} b_{i-k,j-l}u_{k,l}(t) + I \quad (8.6)$$

$$y_{i,j} = f(x_{i,j}) \quad (8.7)$$

Werden die Koeffizienten $a_{i-k,j-l}$ und $b_{i-k,j-l}$ entsprechend Tabelle 8.1 für ein translationsinvariantes Netzwerk in Matrizenform angeordnet, so wird $\mathbf{A}$ üblicherweise als *Feedback-Template* und $\mathbf{B}$ als *Feedforward-Template* bezeichnet. Die Templategewichte zusammen mit dem Schwellwert ergeben also für das Standard-CNN 19 freie Gewichtsfaktoren. Werden diese 19 Gewichtsfaktoren des Standard-CNNs in einem Vektor der Form

$$\mathbf{G} = (a_{-1\,-1}, a_{-1\,0}, .., a_{1\,1}, b_{-1\,-1}, .., b_{1\,1}, I) \quad (8.8)$$

$$\mathbf{A} = \begin{array}{|c|c|c|}\hline a_{-1,-1} & a_{-1,0} & a_{-1,+1} \\ \hline a_{0,-1} & a_{0,0} & a_{0,+1} \\ \hline a_{+1,-1} & a_{+1,0} & a_{+1,+1} \\ \hline \end{array}, \quad \mathbf{B} = \begin{array}{|c|c|c|}\hline b_{-1,-1} & b_{-1,0} & b_{-1,+1} \\ \hline b_{0,-1} & b_{0,0} & b_{0,+1} \\ \hline b_{+1,-1} & b_{+1,0} & b_{+1,+1} \\ \hline \end{array}, \quad \mathrm{I}$$

Tabelle 8.1: Feedback-Template **A**, Feedforward-Template **B** und Schwellwert I für das translationsinvariante Standard-CNN

angeordnet, so wird **G** üblicherweise als *CNN-Gen* [Ch98] bezeichnet. Durch Angabe eines solchen CNN-Gens ist die dynamische Entwicklung eines CNN eindeutig bestimmt.

Als Ausgangsfunktion kommt in den meisten Fällen die *stückweise lineare Funktion*

$$f(x) = \frac{1}{2}(|x+1| - |x-1|) \tag{8.9}$$

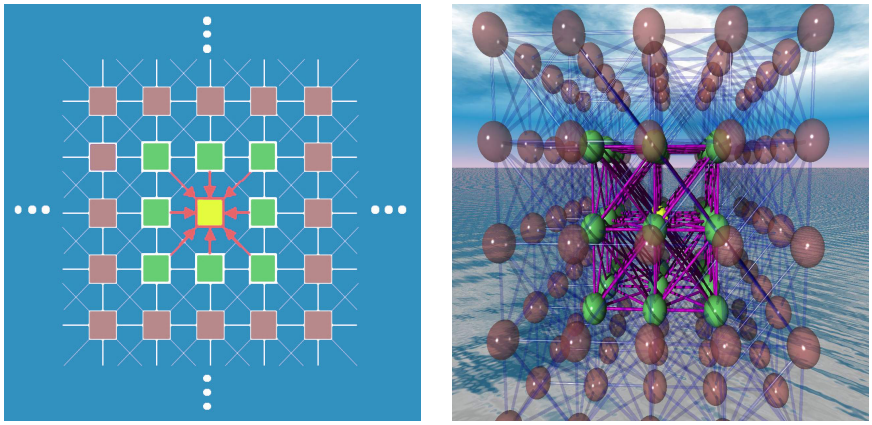
zum Einsatz. Aber auch die *Sigmoidfunktion*

$$f(x) = 1/(1+\exp(-\beta x)) \tag{8.10}$$

oder die *Sprungfunktion*

$$f(x) = \{-1 \; \forall \, x < 0;\; 1 \; \forall \, x \geq 0\} \tag{8.11}$$

finden als Ausgangsfunktion ihre Anwendung. Bild 8.1 visualisiert eine $3 \times 3-$ Nachbarschaft[2] für ein zwei- und ein dreidimensionales Zellulares Neuronales Netzwerk.



a) b)

Bild 8.1: Visualisierung eines a) zwei- und b) dreidimensionalen CNN

8.2.2 Schaltungstechnische Realisierung von CNN

Das Konzept der Zellularen Neuronalen Netzwerke erhielt in [Ro93] eine wesentliche Erweiterung zum *CNN Universal Machine*-Konzept (CNN-UM). Dieses Konzept erweitert CNN dahingehend, daß elektronische Schaltungen von CNN auch lokale Speicher und Logik-Operationen enthalten, so daß ein solcher CNN-UM-Chip eine parallele Anordnung von vollständigen Mikroprozessoren darstellt. Die Realisierung von CNN als elektronische Schaltkreise stellt eine wesentliche Voraussetzung für die Verwendung von CNN in industriellen Anwendungen dar, da erst dann

[2] oder auch „Einflußbereich“ genannt

die Geschwindigkeitsvorteile richtig zum Tragen kommen können. Die VLSI-Realisierung von Zellularen Neuronalen Netzwerken ist mittlerweile so weit fortgeschritten, daß nicht nur lineare CNN mit der Nachbarschaft $r = 1$ gebaut werden können (Abb. 8.2), sondern auch die Realisierung von nichtlinearen CNN mit einer Nachbarschaft von $r = 2$ heutzutage kein Problem mehr darstellt.

In Bild 8.2a ist eine schaltungstechnische Realisierung von CNN dargestellt. Es handelt sich dabei um den sogenannten *ACE4K*-Chip [Li99], der vom *Instituto de Microelectronica de Sevilla* (IMSE) in Spanien entwickelt wurde. Dieser auf CMOS ($0.8\mu m$) basierende Chip realisiert ein Netz von 64×64 Zellen und besitzt ein Auflösungsvermögen von 7 bis 8 Bit, so daß sich Bilder mit 256 Graustufen abbilden lassen. Auf diesem Chip sind weiterhin 32 lokale Templatespeicher und 4 lokale Speicher für Grauwertbilder integriert. In Bild 8.2b ist dargestellt, wie der Chip auf eine Plattform montiert ist, welche an einen „Personal Computer“ angeschlossen werden kann. Dies ermöglicht eine benutzerdefinierte Steuerung des Chips und eine darauf basierende Bildverarbeitung.

a) b)

Bild 8.2: CNN-Hardware: a) CNN-Chip ACE4K von IMSE-CNM, Sevilla (Spanien), b) CNN-Plattform von Analogic Computers Ltd., Budapest (Ungarn)

8.2.3 Die Bestimmung von CNN-Parametern

Die Bestimmung der CNN-Parameter für eine vorgegebene Anwendung aus dem Bereich der Bildverarbeitung kann in der Regel nur in sehr einfachen, meist binären Fällen analytisch erfolgen. In den allermeisten Fällen läuft die Parameterbestimmung für eine spezielle Anwendung über eine Optimierung. Dabei wird eine gewisse Anzahl von Trainingsbildern zusammen mit ihren Referenzbildern[3] verwendet. Die CNN-Ausgangsdaten und die Referenzbilder bilden ein Fehlermaß, welches mit einem Optimierungsverfahren minimiert wird. Stimmen schließlich die CNN-Ausgangsdaten und die Referenzbilder weitgehend überein, so wird ein Satz von CNN-Parametern ermittelt, bei dessen Verwendung die gestellte Aufgabe gelöst werden kann. Im Gegensatz zu anderen Neuronalen Netzwerken werden Zellulare Neuronale Netzwerke nur durch eine relativ geringe Anzahl von Parametern bestimmt, was die Optimierung wesentlich verein-

[3]Die Referenzbilder stellen die Ergebnisbilder für die vorgegebene Anwendung dar, welche auf anderem Wege ohne Verwendung von CNN generiert werden müssen. Hierzu kommen meist Standard-Bildverarbeitungsprogramme oder statistische Methoden zum Einsatz.

facht und beschleunigt. Trotzdem ist die Verlustfunktion[4] eines CNN durch eine hohe Anzahl von lokalen Minima charakterisiert, weshalb analytische Optimierungsverfahren meist keine zufriedenstellenden Ergebnisse liefern. In Kapitel 9 wird gezeigt, daß nur die Verwendung von statistischen Optimierungsverfahren garantiert, bei hinreichend großer Anzahl von Trainingsschritten einen passenden Parametersatz für das gestellte Problem zu ermitteln.

[4]Die Verlustfunktion sei definiert über die Quadratsumme der Grauwertdifferenzen des CNN-Ausgangsbildes zu dem Referenzbild.

Kapitel 9

Statistische Optimierungsverfahren

Die klassischen Gradientenverfahren der nichtlinearen Optimierung weisen allesamt einen großen Nachteil auf; sie finden aus lokalen Extrema nicht mehr heraus, da bei ihnen ausschließlich Informationen aus der näheren Umgebung des aktuellen Funktionswertes, wie beispielsweise über den Gradienten oder die Krümmung, für den nächsten Iterationsschritt verwendet werden. Abhilfe schaffen können die sogenannten *probabilistischen Methoden* der Optimierung, die mehr oder weniger zufällig den Parameterraum nach globalen Extrema absuchen. Es ist offensichtlich, daß solche Verfahren gerade bei Funktionen mehrerer Veränderlicher große Probleme im Hinblick auf Effizienz und Rechendauer aufweisen. Deshalb ist es von großer Bedeutung, daß statistische Optimierungsverfahren den Parameterraum *effizient* absuchen, wie dies beispielsweise bei den im folgenden vorgestellten Verfahren des *Simulated Annealings* (Abschnitt 9.2) und des *Iterative Annealings* (Abschnitt 9.5) der Fall ist. In diesem und den folgenden Kapiteln ist die Aufgabenstellung die Suche nach einem globalen Minimum[1].

9.1 Zufallssuche

Die einfachste aller probabilistischen Optimierungsmethoden stellt die Zufallssuche dar: die Parameter $\vec{x}_i$ werden völlig zufällig gewählt und die zu minimierende Funktion $f(\vec{x})$ ausgewertet. Bei der ständigen zufälligen Wahl der Parameter wird immer der Parametersatz $\vec{x}_{min}$ gespeichert, der den aktuell niedrigsten Funktionswert f_{min} liefert. Bei unendlich vielen Iterationsschritten kann damit das globale Minimum gefunden werden; in der Praxis jedoch führen begrenzte Rechen- und Zeitresourcen dazu, daß die Zufallssuche anderen Verfahren deutlich unterlegen ist.

9.2 Simulated Annealing

Eine Möglichkeit deutlich schneller zum Ziel zu gelangen, liefert das sog. *Simulated Annealing*. Der Begriff des *Simulated Annealings* wurzelt in einem physikalischen Phänomen, denn die Festkörperphysik bezeichnet mit *Annealing* einen langsamen Abkühlprozeß bei dem beispielsweise Kristalle ein globales Minimum der thermischen Energie erreichen, was sich in einer fehlerfreien symmetrischen Kristallstruktur äußert. Eine Beschreibung des Simulated Annealings

[1] Ist die Bestimmung des globalen Maximums das Ziel, so führt eine Negierung der Funktion dazu, daß wieder nach dem globalen Minimum gesucht werden kann.

findet sich in Tabelle (9.1). Die Funktion $f(\vec{x})$ stellt dabei die zu minimierende Funktion dar, wobei $\vec{x}$ den Parametervektor repräsentiert.

Algorithmus Simulated Annealing (Urform)

1. INITIALISIERUNG: SETZE STARTPARAMETERVEKTOR $\vec{x}_0 = \vec{0}, i = 0$
2. WÄHLE $\vec{y}$ IN EINER NACHBARSCHAFT VON $\vec{x}_i$
3. WENN $f(\vec{y}) < f(\vec{x}_i)$ SETZE $\vec{x}_{i+1} = \vec{y}$ UND GEHE ZU SCHRITT 5
4. WENN ZUFALLSZAHL $[0, 1[< \exp\left(\frac{f(\vec{y})-f(\vec{x}_i)}{T_i}\right)$, SETZE $\vec{x}_{i+1} = \vec{y}$
5. ERHÖHE i UM 1, VERRINGERE T_i UND GEHE ZU SCHRITT 2) FALLS ENDKRITERIUM NOCH NICHT ERREICHT

Tabelle 9.1: Die Urform des Simulated Annealings

Unter Anwendung dieses Algorithmus werden im allgemeinen Sprünge zu niedrigeren Funktionswerten vollzogen; allerdings sind mit einer gewissen Wahrscheinlichkeit auch Sprünge zu höheren Funktionswerten zulässig. Dadurch wird es möglich, eventuelle lokale Minima wieder zu verlassen. Der Faktor T_i wird in Analogie zum physikalischen Annealing als *Temperatur* bezeichnet und wird im Simulated Annealing ebenfalls schrittweise erniedrigt. Dies hat zur Folge, daß zu Beginn der Optimierung mit hoher Wahrscheinlichkeit *schlechtere* Funktionswerte akzeptiert werden, wohingegen es bei fortschreitender Zeit immer unwahrscheinlicher wird, daß Parameter mit größeren Funktionswerten akzeptiert werden. Es wurde mathematisch bewiesen [VA87], daß das Verfahren für alle Funktionen $f(\cdot)$ das globale Minimum findet, falls die Temperaturänderungen nur genügend klein sind. In der Praxis hat sich herausgestellt, das dieses Optimierungsverfahren den Parameterraum [In93] wesentlich effizienter absucht als die Zufallssuche aus Abschnitt 9.1.

9.3 Boltzmann Annealing

Eine direkte Anlehnung an die Thermodynamik zeigt das *Boltzmann Annealing*, bei dem genaue Angaben über die zu wählenden Wahrscheinlichkeitsverteilungen gemacht werden. Der Algorithmus zum Boltzmann Annealing stellt sich gemäß Tabelle 9.2 dar. Im Vergleich zur Urform des Simulated Annealings in Tabelle 9.1 ist beim Boltzmann Annealing in Tabelle 9.2 in Gleichung 9.1 die Wahrscheinlichkeitsverteilung der Zufallszahlen angegeben, mit dem der Parametervektor verändert wird. Da die Varianz dieser Normalverteilung direkt über den Temperaturparameter T bestimmt wird, und diese Temperatur in jedem Schritt entsprechend (9.4) erniedrigt wird, nimmt die Stärke der Veränderung des Parametervektors von Iteration zu Iteration ab. Werden also zu Beginn einer Optimierung große Teile des Parameterraums abgesucht, so konzentriert sich die Suche mit fortschreitenden Durchläufen immer mehr um den aktuell besten Parametervektor. Weiterhin erlaubt das Boltzmann Annealing entsprechend Gleichung 9.3 auch Sprünge zu schlechteren Funktionswerten, wodurch bei Verwendung dieses Verfahrens lokale Minima auf der Suche nach dem globalen Minimum verlassen werden können. In [Ge84] wurde gezeigt, daß die Vorschrift $T_i = \frac{T_0}{\ln i}$ zur Temperaturerniedrigung die schnellstmögliche Verringerung darstellt, bei der noch gewährleistet ist, daß das globale Minimum erreicht wird. Die Methode des

Algorithmus Boltzmann Annealing

1. INITIALISIERUNG: SETZE $x_0 = 0, i = 0$
2. VERÄNDERE $\vec{x}_i$ NACH DER WAHRSCHEINLICHKEITSVERTEILUNG

$$g(\Delta\vec{x}) = (2\pi T)^{-\frac{D}{2}} * \exp\left(\frac{-|\Delta\vec{x}|^2}{2T_i}\right) \tag{9.1}$$

$$\rightarrow \vec{y} = \vec{x}_i + \Delta\vec{x} \tag{9.2}$$

   (D: DIMENSION DES PARAMETERRAUMS)
3. WENN $f(\vec{y}) < f(\vec{x}_i)$ SETZE $\vec{x}_{i+1} = \vec{y}$ UND GEHE ZU SCHRITT 5
4. AKZEPTIERE $\vec{x}_i = \vec{y}$ ENTSPRECHEND DER WAHRSCHEINLICHKEITSVERTEILUNG

$$h(\Delta f) = \frac{1}{T_i C}\frac{\exp\left(\frac{-f(\vec{x}_{i+1})}{T_i}\right)}{\exp\left(\frac{-f(\vec{x}_{i+1})}{T_i}\right) + \exp\left(\frac{-f(\vec{x}_i)}{T_i}\right)} = \frac{1}{T_i C}\frac{1}{1 + \exp\left(\frac{\Delta f}{T_i}\right)} \tag{9.3}$$

   MIT $C = 0.693174$ UND $\Delta f = -f(x_i) + f(x_{i+1}) > 0$
5. ERHÖHE i UM 1, SETZE

$$T_i = \frac{T_0}{\ln i} \tag{9.4}$$

   (MIT ANFANGSTEMPERATUR T_0 "GROSS GENUG")
   UND GEHE ZU SCHRITT 2) FALLS ENDKRITERIUM NOCH NICHT ERREICHT

Tabelle 9.2: Boltzmann Annealing

Boltzmann-Annealings kann als eine Standard-Optimierungsmethode bezeichnet werden, da sie schon seit langem in vielen Anwendungsbereichen erfolgreich eingesetzt wird.

9.4 Fast Annealing

Obwohl das Boltzmann Annealing einen direkten physikalischen Bezug aufweist, existiert keine mathematische Begründung, die Veränderung des aktuellen Parametersatzes nicht auch durch Zufallszahlen mit anderen Verteilungsfunktionen als der Gaußverteilung durchzuführen. Bei einem solchen Vorgehen ist jedoch immer darauf zu achten, daß auch die Temperaturabkühlung in angemessener Weise durchgeführt wird, so daß ein sicheres Erreichen des globalen Minimums gewährleistet ist. Eine Möglichkeit ist, anstelle der Boltzmann-Verteilung, die *Cauchy-Verteilung* zu verwenden. Sie hat folgende Form:

$$g(\Delta\vec{x}) = \frac{T_i/\pi}{\left(|\Delta\vec{x}|^2 + {T_i}^2\right)^{\frac{D+1}{2}}} \tag{9.5}$$

Diese Verteilung besitzt den Vorteil, daß sie für große $|\Delta\vec{x}|$ wesentlich langsamer gegen Null konvergiert als die Boltzmann-Verteilung. Dadurch werden lokale Minima wesentlich effizienter

auf der Suche nach dem globalen Minimum übersprungen. Mit dieser Verteilung gelingt es also wesentlich schneller, lokale Minima zu verlassen. In [SH87] wurde gezeigt, daß dieses Verfahren selbst dann noch das globale Minimum erreicht, wenn als Abkühlungsvorschrift

$$T_i = \frac{T_0}{i} \tag{9.6}$$

verwendet wird. Ein Vergleich von (9.6) mit (9.4) zeigt, daß bei diesem Verfahren das globale Minimum exponentiell schneller erreicht wird als beim Boltzmann Annealing. Um das Fast Annealing programmtechnisch zu realisieren, benötigt man Cauchy-verteilte Zufallszahlen. In Anhang D wird eine analytische Methode der Erzeugung von beliebig verteilten Zufallszahlen allein unter Verwendung von gleichverteilten Zufallszahlen vorgestellt.

9.5 Iterative Annealing

Das nun beschriebene Optimierungsverfahren wurde im Rahmen dieser Dissertation speziell für den Einsatz in dem Frankfurter SCNN-System[2] entwickelt. Das Verfahren wird im folgenden mit *Iterative Annealing* bezeichnet. Es verlangt ein gewisses Vorwissen über den Parameterbereich und zeichnet sich insbesondere in den in Abschnitt 9.6 durchgeführten Vergleichstests durch zuverlässiges Erreichen des globalen Minimums aus. Die algorithmische Beschreibung ist in Tabelle 9.3 angegeben. Bild 9.1 enthält außerdem das Blockdiagramm des Iterative Annealings, anhand dem die Funktionsweise des Verfahrens leicht einzusehen ist.

Das Verfahren ist gekennzeichnet durch zwei Berechnungsvorgänge, von denen, wie in Bild 9.1 zu erkennen, die innere Schleife im wesentlichen einem Abkühlvorgang des *Simulated Annealings* entspricht und die äußere Schleife dem Verlassen von lokalen Minima dient. Im Unterschied zum Simulated Annealing werden allerdings anstelle von Gaußverteilten Zufallszahlen zur Veränderung des Parametervektors gleichverteilte Zufallszahlen verwendet. Dies hat einerseits den Nachteil, daß die Parameterveränderungen niemals über den Rand der Gleichverteilung hinaus erfolgen können und somit nur ein Teil des Parameterraumes abgesucht werden kann, aber andererseits führt dies zu einer wesentlichen Beschleunigung der Optimierung. Ein weiterer Unterschied zum Simulated Annealing liegt darin, daß nur bessere Parameter mit einer niedrigeren Verlustfunktion akzeptiert werden. Im Simulated Annealing werden ja mit abnehmender Wahrscheinlichkeit auch Parameter mit höheren Funktionswerten akzeptiert, was dazu führen soll, lokale Minima zu verlassen. Im Iterative Annealing übernimmt die äußere Schleife diese Aufgabe, so daß eine iterative[3] Wiederholung des Abkühlvorgangs, Sprünge aus lokalen Minima ermöglicht.

Die Funktion $f(\vec{x})$ repräsentiert das Fehlermaß, welches minimiert werden soll und $\vec{x}$ bezeichnet den D-dimensionalen Parametervektor. Die Größe s_{max} entspricht der maximalen Anzahl von Iterationsschritten und j_{max} bestimmt, wieviele Iterationen durchgeführt werden sollen. Außerdem wird das Verfahren maßgeblich durch den Startwert der Temperatur T_0 bestimmt, denn gemäß Schritt 4 des Algorithmus bestimmt dieser Parameter die maximal erlaubte Veränderung der Komponenten des Parametervektors pro Iterationsschritt. Die minimale Temperatur τ legt schließlich die Genauigkeit fest, mit der die einzelnen Komponenten des Parametervektors im

[2] SCNN ist ein Simulator für Zellulare Neuronale Netzwerke, der in der Lage ist, neben den zweidimensionalen linearen Standard-CNN auch dreidimensionale CNN, nichtlineare Kopplungen der Zellen über Polynomfunktionen oder Splines, Delay-Templates und mehrschichtige CNN zu verarbeiten [KLT00]. In SCNN sind weiterhin verschiedene Optimierungsverfahren integriert, um die CNN-Parameter für bestimmte Anwendungen zu ermitteln.

[3] daher auch der Name „Iterative Annealing“

Algorithmus Iterative Annealing

1. INITIALISIERUNG: WÄHLE STARTWERTE $x_0^k \; \forall \, k \in [1 \ldots D]$, SETZE $j = 0$, MAXIMALE SCHRITTANZAHL s_{max}, ANZAHL DER WIEDERHOLUNGEN j_{max}, STARTTEMPERATUR T_0 UND MINIMALE TEMPERATUR τ.
2. BERECHNE DIE SCHRITTWEITE ν MIT $\nu = (\tau/T_0)^{j_{max}/s_{max}}$
3. SETZE $T = T_0$ UND $i = 0$
4. VERÄNDERE $x_i^k \; \forall \, k \in [1 \ldots D]$ NACH

$$y_i^k = x_i^k + u^k \cdot T \tag{9.7}$$
$$u^k \in \text{GLEICHVERTEILUNG U}[-0.5, 0.5] \tag{9.8}$$

5. WENN $f(\vec{y}_i) < f(\vec{x}_i)$ SETZE $\vec{x}_{i+1} = \vec{y}_i$
6. ERNIEDRIGE TEMPERATUR ENTSPRECHEND $T = \nu \cdot T$
7. SETZE $i = i + 1$
8. FALLS $i < (s_{max}/j_{max})$, GEHE ZU SCHRITT 4
9. SETZE $j = j + 1$
10. FALLS $j < j_{max}$ GEHE ZU SCHRITT 3

Tabelle 9.3: Iterative Annealing

globalen Minimum bestimmt werden sollen. Für den Einsatz des Verfahrens in dem Frankfurter SCNN-Simulator wurden folgende Initialisierungswerte empirisch ermittelt,

$$T_0 = 20 \tag{9.9}$$
$$j_{max} = 100 \tag{9.10}$$
$$\tau = 0.0001 \; , \tag{9.11}$$

die sich für die in dieser Dissertationsschrift untersuchten Bildverarbeitungsaufgaben als eine gute Wahl herausgestellt haben.

In Bild 9.2 sind einige Iterationsschritte des Verfahrens für ein eindimensionales Beispiel graphisch dargestellt. Gestrichelte Linien bezeichnen erfolglose Schritte und durchgezogene Linien erfolgreiche Schritte, also Parameter mit einem kleineren Funktionswert als der des bis dahin besten Parameters. Bei jedem Schritt kühlt die Temperatur etwas ab, was in der Darstellung durch ein Intervall visualisiert ist. Dadurch reduziert sich der Suchbereich, bis schließlich T die minimale Temperatur τ erreicht (9.3a-9.3c). Dann beginnt der gesamte Prozeß von neuem mit $T = T_0$, was in Bild 9.3d dargestellt ist. Am Ende wird das globale Minimum erreicht.

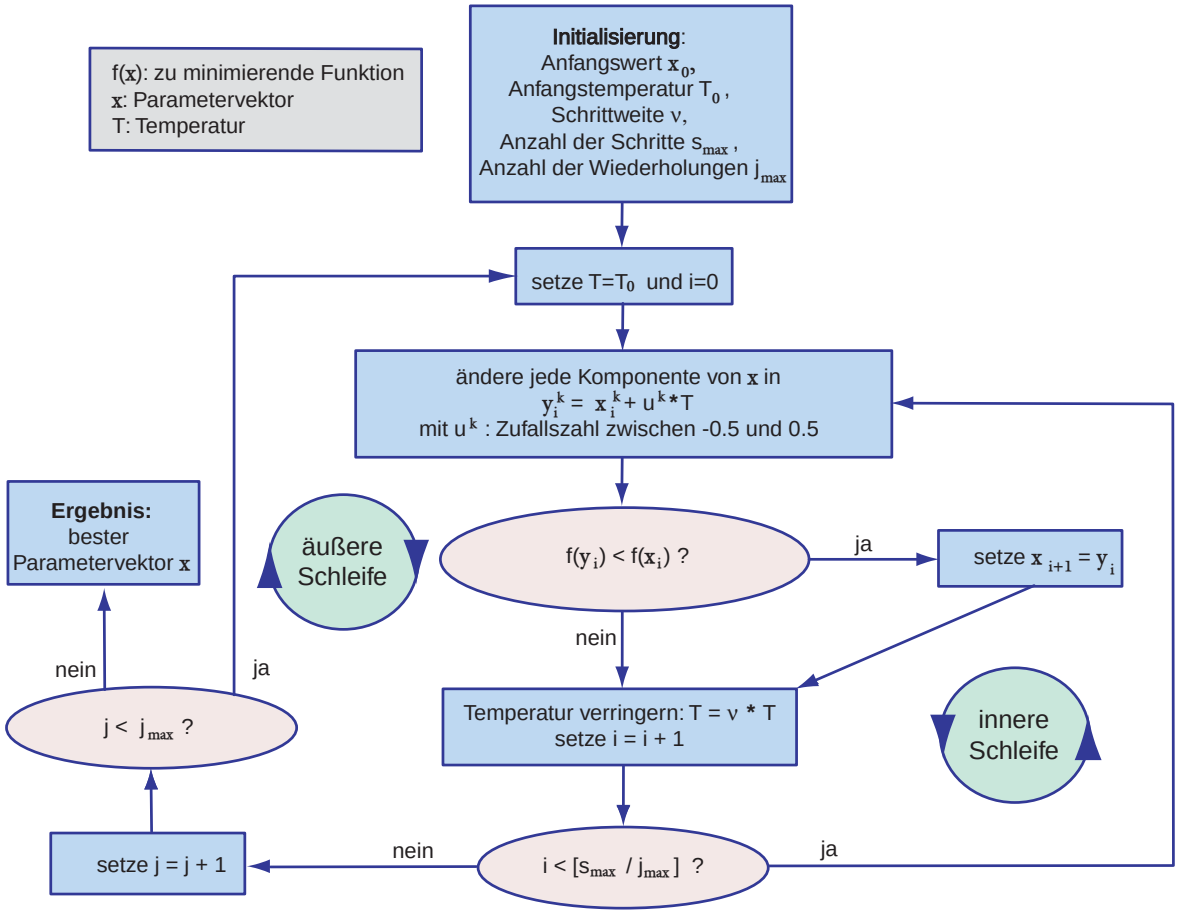


Bild 9.1: Blockdiagramm zum Iterative Annealing

9.6 Vergleichstests mit synthetischen Testfunktionen

9.6.1 Die zu untersuchenden Optimierungsverfahren

Neben den bereits in Abschnitt 9.4 und 9.5 vorgestellten statistischen Optimierungsverfahren „Fast Annealing“ und „Iterative Annealing“ werden ebenfalls die nicht-statistischen Optimierungsverfahren „Simplex“ und „Powell“ in die Untersuchungen mit einbezogen.

Das Simplex-Verfahren

Das Simplex-Verfahren wurde im Jahre 1965 von J. A. Nelder vorgestellt ([PTVF92], S. 408ff). Es basiert ausschließlich auf der Auswertung der Funktionswerte von $n+1$ Punkten im $\mathbb{R}^n$, den sogenannten *Vertizes*. Derjenige Vertex mit dem höchsten Funktionswert wird an dem Schwerpunkt der anderen n Vertizes gespiegelt. Anschließend wird der Funktionswert dieses neuen Vertex ermittelt und wenn er kleiner ist als die übrigen Vertizes, als neuer Punkt akzeptiert. Dieser Prozeß wiederholt sich so lange bis ein Minimum gefunden wurde. Um Oszillationen bei der Spiegelung der Vertizes zu verhindern, sind zusätzlich noch einige Regeln notwendig. Die Optimierungen mittels des Simplex-Verfahrens sind sehr zeitaufwändig, jedoch wird in sehr vielen Fällen das Minimum einer Funktion ermittelt.

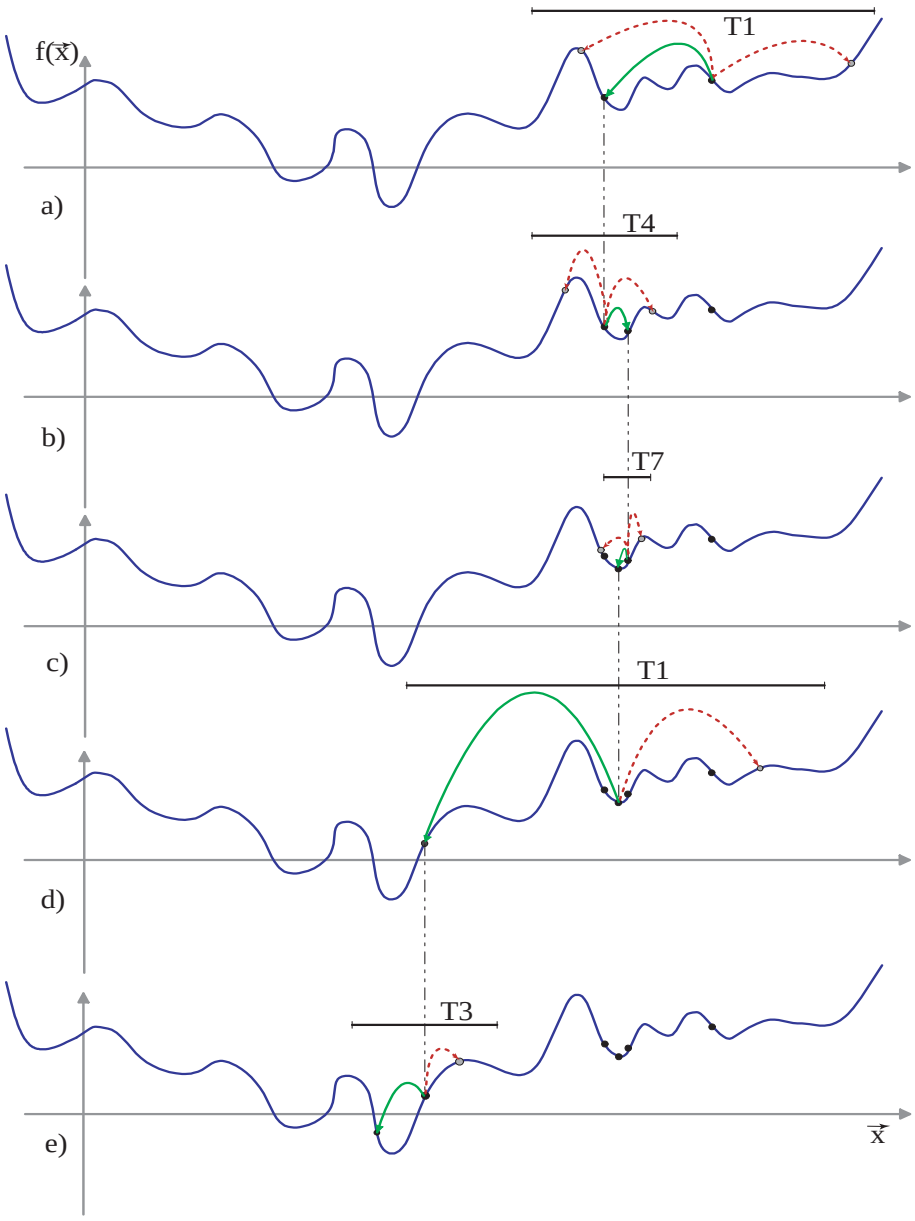


Bild 9.2: Einige Schritte des Iterative Annealing Verfahrens für die Minimierung einer eindimensionalen Funktion (Erläuterungen siehe Seite 71)

Das Powell-Verfahren

Das zweite Optimierungsverfahren, welches in die Untersuchungen miteinbezogen wurde, ist das Powell-Verfahren ([PTVF92], S. 412ff). Bei diesem iterativen Verfahren erfolgt jeweils eine eindimensionale Minimierung entlang von Q vorgegebenen Richtungen $\vec{u}_q$ mit $q = 1, \ldots, Q$, wobei zur Initialisierung die Einheitsvektoren gewählt werden. Aufgrund der Informationen über den Verlauf der zu minimierenden Funktion, die in allen bisherigen Iterationen gewonnen wurden, wird anschließend ein Satz neuer Richtungen $\vec{u}_q$ für die Minimierung erhalten. Das Verfahren ist sehr rechenintensiv, führt aber im allgemeinen zu einer sehr schnellen Ermittlung des Funktionsminimums.

9.6.2 Die Testfunktionen

Im folgenden werden die Testfunktionen, mit denen die Wirkungsweise der Optimierungsverfahren getestet wurde, analytisch und, wenn darstellbar, auch graphisch angegeben.

Funktion I

$$f_1(x) = 23.59127 + (x-2)^2 + \frac{\sin\left(x^3\right)}{\left(\cos(x) + 1.2\right)^2} \tag{9.12}$$

Diese Funktion besitzt um den Bereich $x \approx 9$ eine große Anzahl lokaler Minima. Das absolute Minimum der Funktion liegt bei $x \approx 3.1$ mit einem y-Wert von $y \approx 0$. Bild 9.3 zeigt den Verlauf dieser Funktion. Die Optimierung wurde an der Stelle $x = 12$ gestartet.

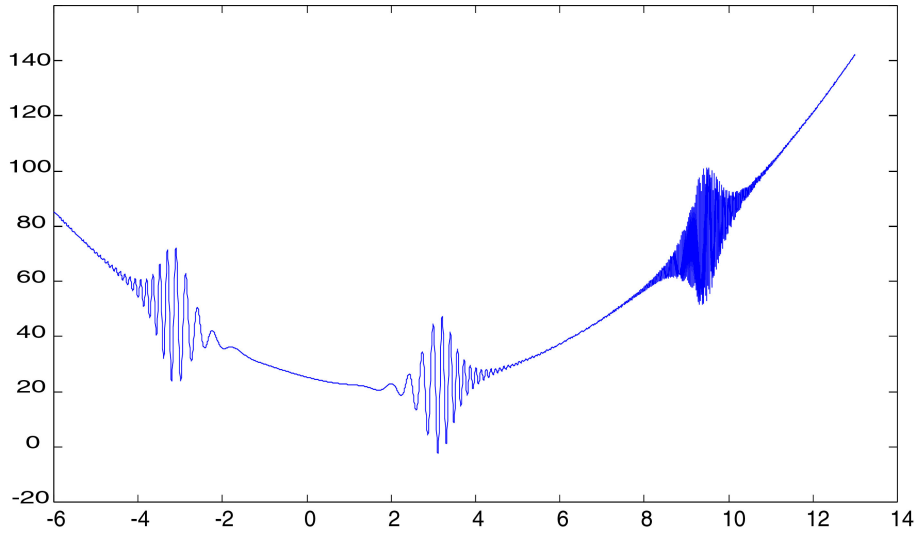


Bild 9.3: $f_1(x)$ in Abhängigkeit von x

Funktion II (Rosenbrocks Funktion)

$$f_2(x,y) = 100 * \left(y - x^2\right)^2 + (1-x)^2 \tag{9.13}$$

Diese Funktion wird häufig für Tests [PTVF92] hinsichtlich der Leistungsfähigkeit von Optimierungsverfahren verwendet. Bild 9.4 zeigt, daß die Funktion ein sehr flache und langgestreckte Region besitzt. Das absolute Minimum dieser Funktion liegt bei $x = y = 1$ mit einem Funktionswert von $f_2(1,1) = 0$. Der Startwert der Optimierung wurde auf $(x,y) = (50,50)$ gesetzt.

Funktion III

$$f_3(x,y) = (|x| + |y|)^2 + 20 * \sin(x * y) - 3 * \frac{\sin(R)}{R} + 15.9391 \tag{9.14}$$

$$\text{mit } R = \sqrt{(x^2 + y^2)} + 0.0000001; \tag{9.15}$$

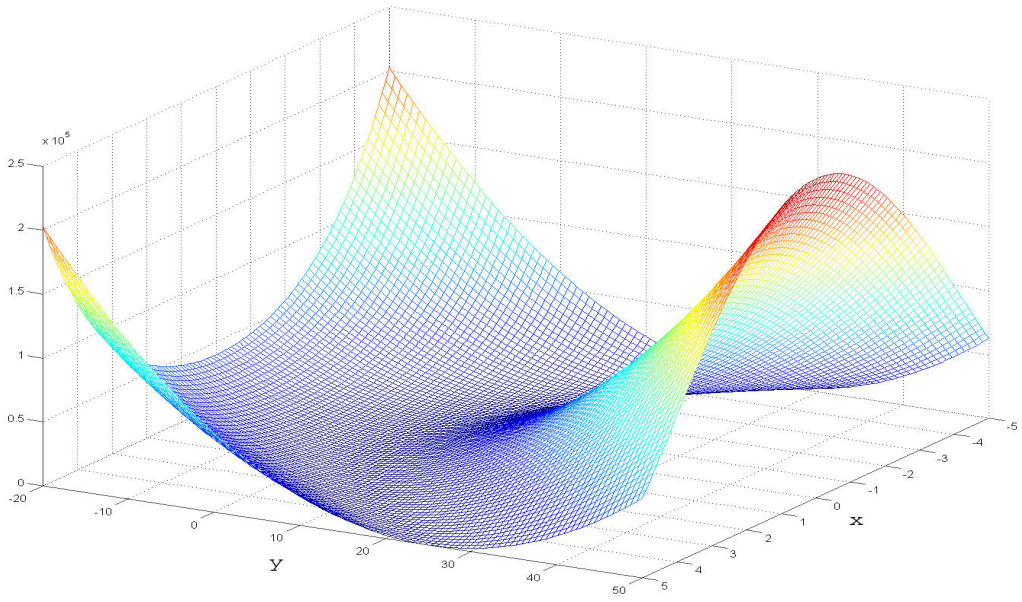


Bild 9.4: $f_2(x, y)$ in Abhängigkeit von x und y

Diese in Bild 9.5 dargestellte zweidimensionale Funktion wurde so konstruiert, daß sie viele lokale Minima besitzt. Weiterhin weist sie zwei absolute Minima an den Stellen $x = -1.1$, $y = 1.1$ und $x = 1.1$, $y = -1.1$ mit dem Funktionswert $f_3(x, y) \approx 0$ auf. Die Optimierungen wurden an der Stelle $(x, y) = (15, -23)$ gestartet.

Funktion IV

$$f_4(x, y) = (-13 + x + ((5 - y) * y - 2) * y)^2 + \tag{9.16}$$
$$(-29 + x + ((y + 1) * y - 14) * y)^2 \tag{9.17}$$

Diese zweidimensionale Funktion wurde so konstruiert, daß sie zwei Minima enthält. Das globale Minimum liegt bei $(x = 5 , y = 4)$ mit dem Funktionswert $f_4(x, y) = 0$. Die Optimierung wurde in der Nähe des lokalen Minimums $(x = 11.41 , y = -0.89)$ mit dem Funktionswert $f_4(x, y) \approx 49$ gestartet, um die Leistungsfähigkeit der Verfahren zu testen, aus einem lokalen Minimum herauszukommen. In der Literatur [BR95] wird diese Funktion als **Freudenstein-und-Roth-Funktion** (Bild 9.6) bezeichnet. Als Startwert der Optimierung wurde $(x, y) = (11, -1)$ gewählt, denn dieser Punkt liegt in der Nähe des lokalen Minimums.

Funktion V

Die Funktion $f_5(x, y, z)$ stellt eine dreidimensionale Funktion mit vielen lokalen Minima dar. Sie wird über den Algorithmus in Tabelle 9.4 definiert. Die Optimierung wird an der Stelle $(x, y) = (-1, 0, 1)$ gestartet. Das globale Minimum dieser Funktion liegt bei $x_1 = 0.398956$, $x_2 = 1$, $x_3 = 0$ mit einem Funktionswert von $f = 1.12 * 10^{-8}$.

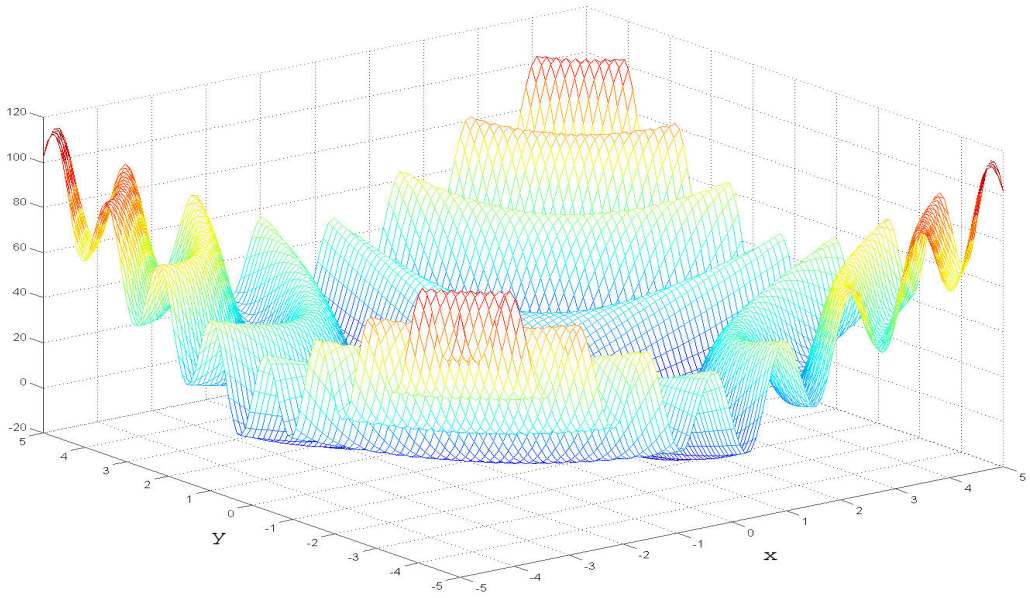


Bild 9.5: $f_3(x, y)$ in Abhängigkeit von x und y

Funktion VI

Mit Funktion $f_6(x_1, \ldots, x_6)$ wurde schließlich noch eine sechsdimensionale Funktion herangezogen, die ebenfalls eine große Anzahl lokaler Minima aufweist. Ihre Definition ist in Tabelle 9.5 dargestellt und sie wird als **Biggs-EXP6-Funktion** [MGH81] bezeichnet. Der Startwert der Optimierung wurde im Rahmen dieser Arbeit auf $\vec{x} = (0, 0, 0, 0, 0, 0)$ gesetzt. Das globale Minimum dieser Funktion liegt bei $\vec{x}_m = [1, 10, 1, 5, 4, 3]$ mit einem Funktionswert von $f(\vec{x}_m) = 0$.

9.6.3 Ergebnisse der Optimierungen

Im folgenden sind die Ergebnisse der Optimierungen bei Verwendung von vier verschiedenen Optimierungsmethoden dargestellt. Um die Optimierungsmethoden sinnvoll bewerten zu können, wurden jeweils *zehn* Einzelmessungen durchgeführt. Gerade bei den beiden statistischen Optimierungsverfahren wären einzelne Messungen wenig aussagekräftig. Dabei wurden die Startwerte jeweils leicht variiert, um sicher zu stellen, daß die analytischen Verfahren nicht ständig identische Ergebnisse liefern. Die Tabellen G.1 bis G.6 in Anhang G zeigen die Optimierungsergebnisse.

9.6.4 Auswertung der Vergleichsmessungen

Funktion I

Bei Anwendung des Simplex-Verfahrens verfängt sich jeder Optimierungsversuch der Testfunktion f_1 in einem lokalen Minimum. Bei Verwendung des Powell-Verfahrens hingegen wird in zwei von zehn Versuchen das globale Minimum gefunden, was im Hinblick auf die vielen lokalen Minima erstaunlich ist für ein analytisches Optimierungsverfahren. Das globale Minimum wird bei Benutzung des Fast-Annealing-Verfahrens immer gefunden und es werden dazu im Mittel 1277 Schritte benötigt. Die Anwendung des Iterative-Annealing-Verfahrens führt ebenfalls bei

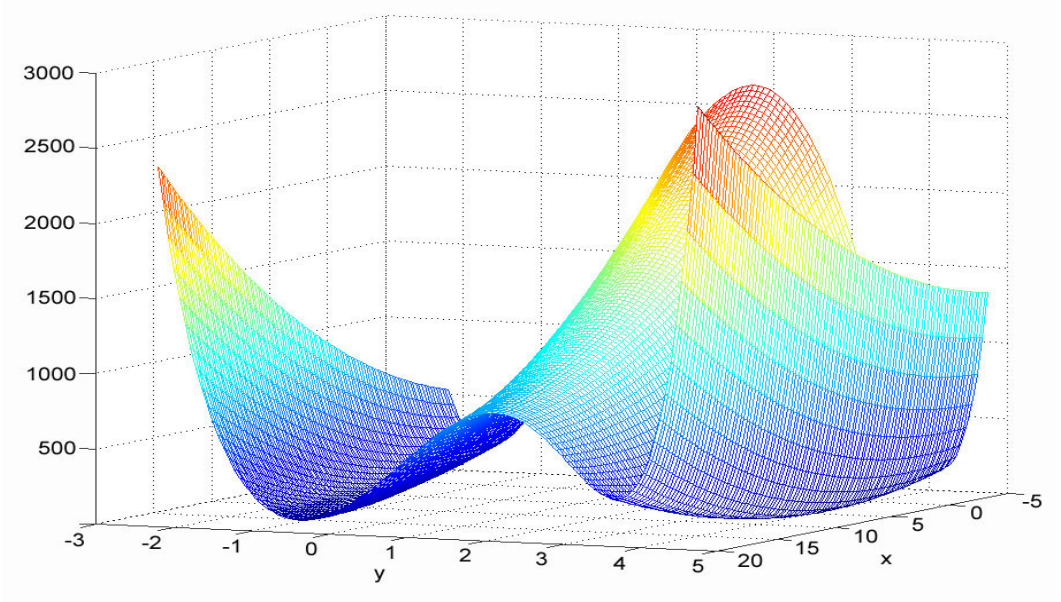


Bild 9.6: $f_4(x, y)$ in Abhängigkeit von x und y

allen zehn Versuchen zum Erreichen des globalen Extremums, hierzu werden allerdings mit einer durchschnittlichen Anzahl von 386 deutlich weniger Schritte benötigt als beim Fast Annealing.

Funktion II

Das globale Minimum der Rosenbrock-Funktion wird unter Anwendung aller getesteten Optimierungsverfahren sehr zuverlässig gefunden. Die Ergebnisse unterscheiden sich im wesentlichen nur darin, wie viele Schritte die Verfahren zum Erreichen des Minimums benötigen. Hier schneidet das Simplex-Verfahren mit 668 Schritten am besten ab, gefolgt von Powell (7893), Iterative Annealing (8200) und Fast-Annealing (13307).

Funktion III

Bei der Testfunktion $f_3(x, y)$ resultiert die Optimierung mittels des Simplex-Verfahrens ständig in lokalen Minima. Die Verwendung der anderen Optimierungsverfahren führt bei allen Versuchen zu eines der beiden globalen Minima, wobei das Iterative Annealing das schnellste Verfahren ist mit durchschnittlich 205 Schritten, gefolgt von Powell mit 390 Schritten und Fast Annealing mit 14290 Schritten im Mittel.

Funktion IV

Bei der Optimierung der Funktion $f_4(x, y)$ ist es bei Benutzung einer der beiden analytischen Verfahren (Simplex und Powell) nicht möglich, das lokale Minimum, in dessen Nähe die Optimierung gestartet wurde, wieder zu verlassen. Nur die Verwendung der beiden statistischen Verfahren führt dazu, daß das globale Minimum gefunden wird, wobei das Iterative-Annealing mit durchschnittlich 4022 Optimierungsschritten wesentlich schneller ist als das Fast-Annealing, welches im Mittel 16096 Schritte benötigt.

$$
\begin{aligned}
f_5(x,y,z) &= 0. \\
y &= [0.0009\,0.0044\,0.0175\,0.054\,0.1295\,0.2420\,0.3521\,0.3989 \\
&\quad 0.3521\,0.2420\,0.1295\,0.054\,0.0175\,0.0044\,0.0009\,0] \\
(*)\ \text{SETZE } i &= i+1 \\
&\quad t_i = \frac{8-i}{2} \\
&\quad df_i = x * \exp\left[\frac{-y * \left((t_i - z)^2\right)}{2}\right] - y_i \\
&\quad f_5(x,y,z) = f_5(x,y,z) + df_i^2 \\
\text{WENN } i &< 15 \text{ GEHE ZU } (*)
\end{aligned}
$$

Tabelle 9.4: Definition von $f_5(x, y, z)$

$$
\begin{aligned}
f_6(x_1,\ldots,x_6) &= 0. \\
(*)\ \text{SETZE } i &= i+1 \\
&\quad t_i = 0.1 * i \\
&\quad y_i = \exp(-t_i) - 5 * \exp(-10 * t_i) + 3 * \exp(-4 * t_i) \\
&\quad df_i = x3 * \exp(-t_i * x1) - x4 * \exp(-t_i * x2) \\
&\qquad + x6 * \exp(-t_i * x5) - y_i \\
&\quad f_6(x_1,\ldots,x_6) = f_6(x_1,\ldots,x_6) + df_i^2 \\
\text{WENN } i &< 13 \text{ GEHE ZU } (*)
\end{aligned}
$$

Tabelle 9.5: Definition von $f_6(x_1, \ldots, x_6)$

Funktion V

Bei der Funktion $f_5(x, y, z)$ führt der Einsatz des Powell- und des Fast-Annealing-Verfahrens bei keinem Versuch zur Bestimmung des globalen Minimums. Etwas besser ist hier das Simplex-Verfahren, bei dessen Anwendung in drei von zehn Fällen das globale Extremum erreicht wird. Das einzige Verfahren, welches gute Ergebnisse bei dieser Optimierungsaufgabe liefert, ist das Iterative-Annealing. Bei acht von zehn Fällen wird nach durchschnittlich 24997 Schritten das globale Minimum ermittelt.

Funktion VI

Die sechsdimensionale Funktion $f_6(x_1, \ldots, x_6)$ stellt von den sechs untersuchten Funktionen die höchsten Anforderungen an die Optimierungsverfahren. Einzig und allein die Anwendung des Iterative-Annealing-Verfahrens führt in den meisten Fällen, nämlich in sechs von zehn Versuchen,

zur Bestimmung des globalen Minimums. Allerdings werden dazu auch eine hohe Anzahl an Optimierungsschritten benötigt, im Mittel sind es 253737.

Resumée

Ganz allgemein läßt sich folgern, daß nicht-statistische Verfahren große Schwierigkeiten bereiten, ein globales Minimum einer Funktion zu finden, wenn viele lokale Minima auftreten. Hinzu kommt, daß die Minimierung mit nicht-statistischen Verfahren um so problematischer ist, je höher die Dimension des Parametervektors ist. Hier spielen die statistischen Verfahren, insbesondere das neue Iterative Annealing, ihre Stärken aus. Sie benötigen zwar meist eine hohe Anzahl von Optimierungsschritten, zeichnen sich aber durch eine besondere Robustheit bei der Ermittlung des globalen Minimums aus.

Kapitel 10

Generierung von markanten Bildbereichen mit CNN

In den folgenden Ausführungen soll nun gezeigt werden, daß das vorgestellte Verfahren der statistischen Bewegungsschätzung aus Kapitel 2 durch Verwendung von Zellularen Neuronalen Netzwerken realisiert werden kann. Zunächst wird demonstriert, wie der erste Schritt des Verfahrens, die Bestimmung eines Markantheitsbildes, mit einem CNN mit linearen Gewichtsfunktionen durchführbar ist. Das folgende Kapitel 11 zeigt anschließend, daß es unter Verwendung von CNN ebenfalls möglich ist, eine Verschiebungsvektorschätzung durchzuführen, und schließlich wird in Kapitel 12 eindrucksvoll demonstriert, daß mit CNN sogar das *gesamte* Verfahren der Hinderniserkennung *in einem Schritt* realisiert werden kann.

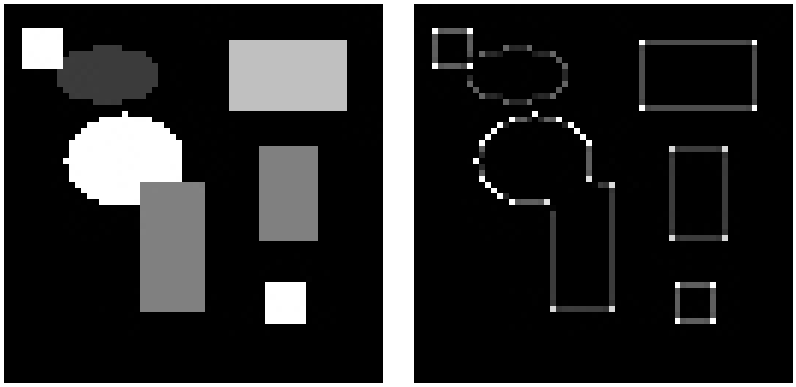
10.1 Generierung eines Markantheitsbildes mit CNN

Bild 10.1a zeigt ein synthetisch generiertes Bild. Die Aufgabe bestand nun darin, aus diesem Bild ein Markantheitsbild zu erzeugen, also ein Bild in dem die Ecken von Objekten deutlich hervortreten und gleichzeitig auch Objektkanten hervorgehoben sind. Der Grund für eine derartige Zielsetzung wurde bereits in Kapitel 3 erläutert. Objektecken sind für die Bewegungsschätzung wertvoller als Kanten, da sie mit einer hohen Konfidenz im zeitlich darauffolgenden Bild wiedergefunden werden können. Da aber meist nur wenige Objektecken im Bild zu finden sind, müssen auch Objektkanten detektiert werden. In Bild 10.1b ist das gewünschte Ergebnis zu sehen.

Mit diesem Trainingsbildpaar wurde versucht, die Parameter eines CNN mit linearen Gewichtsfunktionen für diese Anwendung zu ermitteln. Bild 10.1a diente dabei als Eingangsbild und Bild 10.1b als Referenzbild. Als Zustandsinitialisierung wurde ein komplett schwarzes Bild mit dem Zellzustandswert $x_{i,j} = -1$ verwendet. Entsprechend wird der Zellzustandswert $x_{i,j} = +1$ im folgenden als weiß definiert. Um eine isotrope[1] Verarbeitung des Eingangsbildes zu erhalten, wurden die Template-Elemente von $\mathbf{A}$ und $\mathbf{B}$ gemäß

$$\mathbf{A} = \begin{pmatrix} a_1 & a_2 & a_1 \\ a_2 & a_3 & a_2 \\ a_1 & a_2 & a_1 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} b_1 & b_2 & b_1 \\ b_2 & b_3 & b_2 \\ b_1 & b_2 & b_1 \end{pmatrix} \tag{10.1}$$

[1] richtungsunabhängige



a) b)

Bild 10.1: Trainingsbildpaar: a) Original, b) Markantheitsbild

symmetrisch angeordnet, so daß zusammen mit dem Schwellwert I nur 7 freie Parameter übrigbleiben.

Für die Bestimmung der freien CNN-Parameter wurde ein Parametertraining mit dem Frankfurter SCNN-Simulator[2] durchgeführt, wobei bei diesem Training alle in SCNN zur Verfügung stehenden Optimierungsverfahren eingesetzt wurden. In Tabelle H.1 im Anhang H sind die Ergebnisse der Optimierungsversuche unter Verwendung der unterschiedlichen Verfahren dargestellt. Das einzige Optimierungsverfahren, welches in den meisten Fällen ein zufriedenstellendes Ergebnis lieferte, war das *Iterative-Annealing-Verfahren*. Bei dessen Verwendung wurde in 7 von 10 Optimierungsversuchen ein Satz von CNN-Parametern ermittelt, so daß bei Anwendung des entsprechenden CNN die Generierung eines Markantheitsbildes realisiert werden kann. Der Einsatz der übrigen Optimierungsverfahren resultierte ständig in lokalen Minima der Verlustfunktion, so daß dabei keine geeigneten CNN-Parameter ermittelt werden konnten.

Der folgende Satz von CNN-Parameter

$$\begin{aligned} C &= (a_{-1\,-1}, a_{-1\,0}, .., a_{1\,1}, b_{-1\,-1}, .., b_{1\,1}, I) \\ &= (0, -0.03, 0, -0.03, 0.2, -0.03, 0, -0.03, 0, -0.2, -0.97, \\ &\quad -0.2, -0.97, 3.86, -0.97, -0.2, -0.97, -0.2, 1.67) \end{aligned} \tag{10.2}$$

wurde mit Hilfe des Optimierungsverfahrens „Iterative Annealing“ ermittelt. Unter Verwendung dieses CNN-Gens kann somit ein Markantheitsbild generiert werden, bei dem Objektecken heller sind als Objektkanten. Der resultierende Fehler zwischen Referenzbild und CNN-generiertem Bild, entsprechend der Formel (10.5), betrug dabei $5.1 * 10^{-5}$, was bedeutet, daß visuell keine Unterschiede mehr zwischen den Bildern festgestellt werden können. Zum Nachweis, daß die vorgestellte Markantheitsbildberechnung auch mit anderen Bildern als dem Trainingsbild funktioniert, wurden die in Bild 10.2 dargestellten Ausgangsbilder mittels eines CNN mit dem in Gleichung 10.2 angegebenen Gen verarbeitet. Das Ergebnis der CNN-Simulation ist in Bild 10.3 zu sehen. Wie gefordert, werden durch Anwendung des CNN Objektkanten und -ecken extrahiert, wobei Ecken stärker hervortreten als Kanten. Der Fehler zwischen dem statistischen Markantheitsbild und dem CNN-generierten Bild, entsprechend der Formel (10.5), betrug für

[2]SCNN [KLT00] [KTW96] ist ein am Institut für Angewandte Physik der Universität Frankfurt entwickeltes, universelles Simulationssystem für Zellulare Neuronale Netze, welches neben dreidimensionalen CNN, nichtlinearen Zellkopplungen und Delay-Funktionalität auch die Möglichkeit zur Verfügung stellt, mittels 9 verschiedener Optimierungsmethoden ein Parametertraining durchzuführen.

das linke Testbild $2.4 * 10^{-3}$ und für das rechte Bild $9.1 * 10^{-4}$, so daß auch bei den Testbildern visuell keine Unterschiede festgestellt werden können.

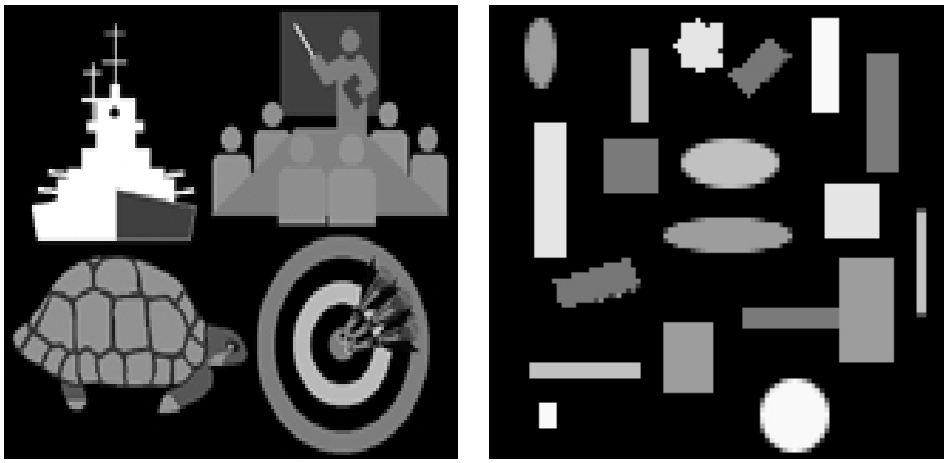


Bild 10.2: Die Originalbilder zum Test der CNN-basierten Markantheitsbildberechnung

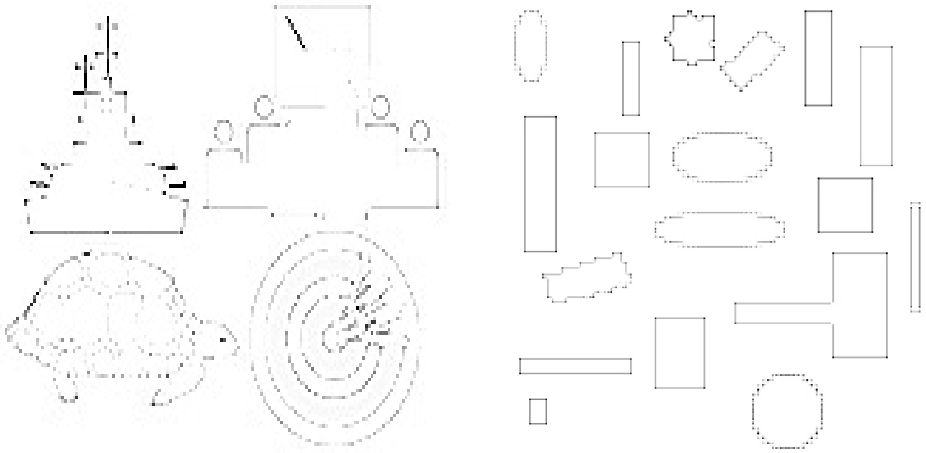


Bild 10.3: Invertierte Ergebnisse der Bestimmung der Markantheitsbilder bei Verwendung des ermittelten CNN-Gens (10.2)

10.2 Die Robustheit der ermittelten Parameter

Schaltungstechnische Realisierungen von Zellularen Neuronalen Netzwerken sind aufgrund dem in (8.1), (8.3) oder (8.6) geforderten dynamischen Verhalten einer Zelle durch analoge Bauteile aufgebaut. Dies führt im Vergleich zur Digitalelektronik zu einer Reihe von Schwierigkeiten. Die Digitaltechnik basiert auf einer zweiwertigen Logik, was bedeutet, daß Spannungen oberhalb eines bestimmten Wertes U_1 als „1“ oder „logisch Wahr“ interpretiert werden und Spannungen unterhalb eines Wertes U_2 als „0“ oder „logisch Falsch“. Sind die beiden Werte U_1 und U_2 nicht identisch, so liegt zwischen diesen Grenzen ein *verbotener Bereich.* Bei der Hardwarerealisierung einer Digitalschaltung muß sichergestellt sein, daß sich kein Spannungswert in diesem verbotenen

Bereich befinden kann. Im Fall von CNN-UM ist jedoch gerade der verbotene Bereich der Digitalelektronik von Interesse, da dieser den Arbeitsbereich des Netzwerks darstellt. Dabei spielen die einzelnen elektronischen Bauteile der Schaltung eine wesentliche Rolle; die physikalischen Eigenschaften dieser Bauteile weisen nämlich unvermeidbare bautechnische Abweichungen auf. Durch Vernetzung der CNN-Zellen untereinander und insbesondere durch Rückkopplungen zwischen diesen Zellen können diese Abweichungen eine große Auswirkung auf die Ausgangsaktivität der einzelnen Zellen haben, weshalb die Verwendung von *robusten Templates* besonders wichtig ist. „Robuste Templates“ bedeutet dabei, daß kleine Abweichungen in den Templateeinträgen keine großen Auswirkungen auf das Gesamtverhalten des Netzwerks zur Folge haben.

Im folgenden wird die Robustheit des ermittelten CNN-Gens zur Markantheitsbestimmung aus (10.2) untersucht. Dazu wurde jede Komponente des Gens durch normalverteilte Zufallswerte gemäß

$$C_i' = C_i + v_i \tag{10.3}$$

verändert, wobei v_i einer Gauß-verteilten Zufallszahl mit der Wahrscheinlichkeitsdichtefunktion

$$p(v_i) = \frac{1}{\sqrt{2\pi s^2}} \cdot e^{\frac{-v_i^2}{2s^2}} \tag{10.4}$$

entspricht. Anschließend wurde die Auswirkung auf die Berechnung des Markantheitsbildes festgestellt, wobei zwei unterschiedliche Fälle untersucht wurden:

1. der *invariante* Fall, bei dem das gleiche veränderte CNN-Gen auf alle Zellen des CNN angewandt wurde und
2. der *variante* Fall, bei dem alle Zellen des CNN unterschiedliche Gene erhalten. Diese unterschiedlichen Gene wurden jeweils durch eine Realisierung des gleichen stochastischen Prozesses erhalten.

In Tabelle 10.1 ist der genaue Ablauf der Untersuchungen zum invarianten Fall dargestellt. Eine

1. Veränderung jeder Komponente des CNN-Gens durch normalverteilte Zufallswerte mit einer Standardabweichung s gemäss (10.3) und (10.4)
2. Markantheitsbildberechnung mit dem Simulationssystem SCNN [KLT00]
3. Berechnung des mittleren Absolutfehlers (MAE) des Ausgangsbildes bei gestörtem Gen zu dem Referenzbild gemäss

$$MAE = \frac{1}{M * N} \sum_{i,j} |f_1(x_{i,j}) - f_2(x_{i,j})| , \tag{10.5}$$

   mit der ungestörten Ausgangsaktivität $f_1(x_{i,j})$, der Ausgangsaktivität bei verändertem CNN-Gen $f_2(x_{i,j})$ und der Grösse des Netzwerks $M * N$
4. fünfzigmalige Wiederholung der Schritte 1 bis 3
5. Berechnung des Mittelwertes der fünfzig MEA-Einzelmessungen

Tabelle 10.1: Der Ablauf der Untersuchung zum invarianten Fall

graphische Darstellung des Ergebnisses der Prozedur in Tabelle 10.1 bei Verwendung des Bildpaares 10.1 für zehn verschiedene Standardabweichungen s ist in Bild 10.4 zu finden. Weiterhin ist in Tabelle 10.2 der genaue Ablauf der Untersuchungen zum varianten Fall dargestellt und die graphische Darstellung der Ergebnisse bei Verwendung des Bildpaares 10.1 für vierzehn verschiedene Standardabweichungen s ist in doppelt logarithmischer Darstellung in Bild 10.5 zu finden.

1. zellabhängige Überlagerung der Gen-Elemente mit normalverteilten Zufallswerten mit einer Standardabweichung s gemäss (10.3) und (10.4)
2. Markantheitsbildberechnung mit dem Simulationssystem SCNN [KLT00]
3. Berechnung des MAE des Ausgangsbildes zu dem Referenzbild bei verändertem Gen gemäss (10.5)
4. hundertmalige Wiederholung der Schritte 1 bis 3
5. Berechnung des Mittelwertes der hundert MEA-Einzelmessungen

Tabelle 10.2: Der Ablauf der Untersuchung zum varianten Fall

Die beiden doppelt logarithmischen Darstellungen 10.4 und 10.5 zeigen, daß das ermittelte CNN-Gen zur Markantheitsberechnung robust ist gegenüber Parameterveränderungen, da der Einfluß der Störung auf das Ergebnisbild bis zu einer Standardabweichung von $s = 0.1$ vernachlässigbar ist. Besonders der an die Praxis angelehnte variante Fall zeigt, daß bis zu dem starken Funktionsanstieg bei $s \approx 0.1$ die Störungen keinen wesentlichen Effekt auf das Ausgangsbild haben. Somit kann das ermittelte CNN-Gen in Gleichung 10.2 auch bei praktischen Anwendungen zur Markantheitsbildberechnung eingesetzt werden.

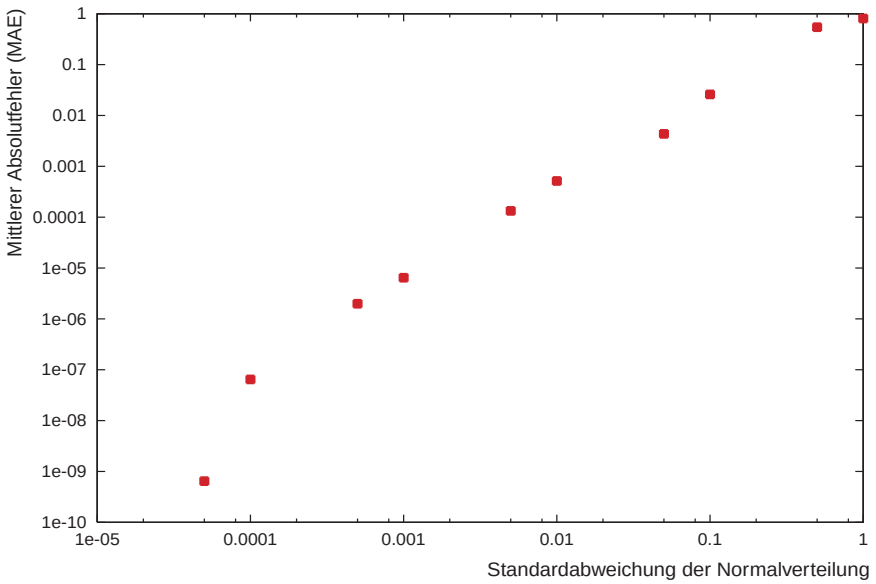


Bild 10.4: MAE in Abhängigkeit von s (invarianter Fall)

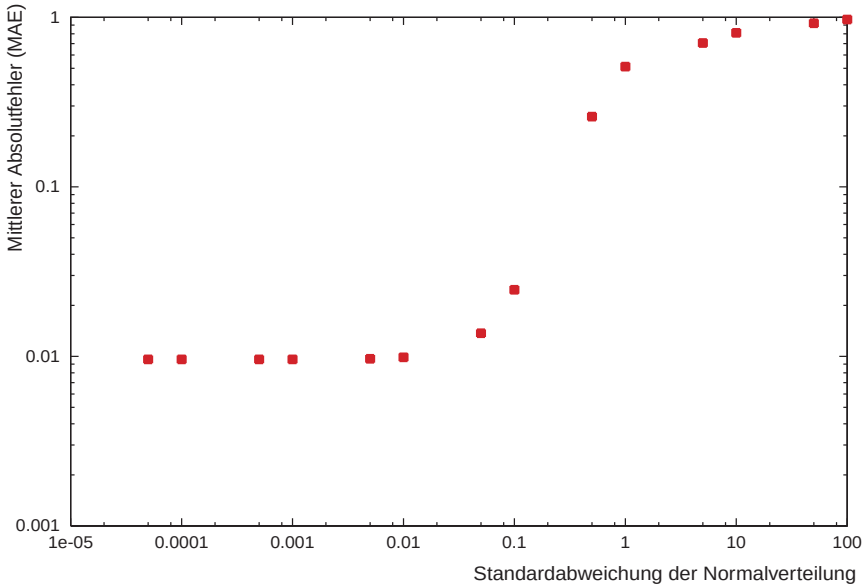


Bild 10.5: MAE in Abhängigkeit von s (varianter Fall)

Kapitel 11

Verschiebungsvektorschätzung mit CNN

In Kapitel 4 wurde die Bedeutung der Verschiebungsvektorschätzung für die Bereiche der 3D-Bewegungsschätzung und der Hinderniserkennung herausgestellt. Als weiteres sehr wichtiges Anwendungsgebiet der Verschiebungsvektorschätzung hat sich das Gebiet des sogenannten *Video Codings* herausgestellt, welches die Komprimierung von Videodaten zum Thema hat. Für all diese Anwendungen ist die Verarbeitungsgeschwindigkeit von großer Bedeutung, so daß nach schnellen Verschiebungsvektorschätzungen gefordert wird. Bedauerlicherweise ist die Verschiebungsvektorschätzung unter Verwendung statistischer Methoden sehr rechenintensiv, was für eine Echtzeitverarbeitung von großem Nachteil sein kann. Deshalb wird in diesem Kapitel eine neue Methode der Verschiebungsvektorschätzung vorgeschlagen, die auf der Verwendung von Zellularen Neuronalen Netzwerken basiert. Weiterhin wird gezeigt, daß das vorgestellte Verfahren robust gegenüber Störungen in der Umgebung der zu betrachtenden Bildobjekte ist.

11.1 Detektion verschobener Objekte mit CNN

Das Ziel der Verschiebungsvektorschätzung ist die Detektion verschobener Objekte in aufeinanderfolgenden Bildern einer Videosequenz, wobei die verschobenen Objekte auch von anderen Objekten umgeben sein können und ebenfalls Störungen auftreten. Im folgenden werden zwei unterschiedliche Bilder verwendet,

- das erste Bild enthält ausschließlich das Objekt, welches in einem folgenden Bild wiedergefunden werden soll und
- das zweite Bild enthält neben dem verschobenen Objekt zusätzlich weitere Objekte und additive Pixelstörungen.

Das gewünschte Ergebnis ist dann ein Bild, in dem ausschließlich das verschobene Objekt enthalten ist, da andere umgebende Objekte und die Pixelstörungen herausgefiltert wurden. Bevor der praxisrelevante zweidimensionale Fall behandelt wird, hat der folgende Abschnitt zunächst die Detektion von verschobenen, eindimensionalen Objekten zum Thema.

11.1.1 Der eindimensionale Fall

In Bild 11.1aa - 11.1ac sind verschiedene eindimensionale Trainingsmuster dargestellt. Als Bildobjekte werden zusammenhängende schwarze Pixel ($y_i = 1$) bezeichnet, also Liniensegmente mit einer Länge von mindestens zwei Bildpixeln. In Bild 11.1ab sind die verschobenen Liniensegmente mit umgebenden Pixelstörungen und anderen Objekten mit mindestens zwei Bildpixeln Länge gezeigt, wobei die Verschiebung bei diesen Beispielen immer 1 Pixel beträgt. In Bild 11.1ac sind schließlich die jeweils gewünschten CNN-Ausgangsbilder mit den extrahierten verschobenen Objekten präsentiert.

Die Optimierung der CNN-Parameter gestaltete sich derart, daß *alle* Trainingsmuster einem CNN mit bestimmten Parametern präsentiert wurde und der Gesamtfehler zwischen den Ausgangsakivitäten und den Trainingsreferenzmustern berechnet wurde. Dieser Gesamtfehler, der sich aus der Summe aller absoluten Pixeldifferenzen berechnet, wurde dann unter Variation der CNN-Parameter minimiert. Bei Verwendung der Lernmuster aus Bild 11.1a wurden durch Optimierung mit dem „Iterative Annealing-Verfahren“ geeignete Template-Elemente für ein CNN mit linear gekoppelten Zellen gemäß (8.6) ermittelt [Fe01]. Das ermittelte CNN-Gen lautet

$$(\alpha_{-1}, \alpha_0, \alpha_1, \beta_{-1}, \beta_0, \beta_1, I) = (2.35, 4.94, 2.35, 3.87, -0.76, 3.87, -2.18)\,. \tag{11.1}$$

Falls jedoch die Verschiebung des Objekts mehr als ein Pixel beträgt, führt die Optimierung eines CNN mit linearen Gewichtsfunktionen zu keinen verwendbaren Ergebnissen. Es hat sich allerdings herausgestellt, daß mittels eines CNN mit nichtlinearen Kopplungen, die Aufgabe der Detektion verschobener Objekte mit einer Verschiebung von mehr als einem Pixel gelöst werden kann. In Bild 11.1ba-bc sind dazu einige Trainingsbeispiele dargestellt. Die Optimierung mit diesen Trainingsmustern hat ergeben, daß unter Verwendung eines CNN mit polynomialer Kopplung[1] vom Grade 2 das beschriebene Problem gelöst werden kann, wobei

$$\begin{gathered}(\alpha^1_{-1}, \alpha^2_{-1}, \alpha^1_0, \alpha^2_0, \alpha^1_1, \alpha^2_1, \beta^1_{-1}, \beta^2_{-1}, \beta^1_0, \beta^2_0, \beta^1_1, \beta^2_1, I) = \\ (9.0, 24.9, 8.9, 0.3, 9.0, 24.9, 3.1, \text{-}28.5, \text{-}1.9, 20.5, 3.1, \text{-}28.5, \text{-}12.5)\end{gathered} \tag{11.2}$$

gilt.

11.1.2 Der zweidimensionale Fall

Im folgenden Teil wird der oben diskutierte eindimensionale Fall der Suche nach verschobenen Objekten auf den zweidimensionalen Fall erweitert, so daß von nun an Bilder die Datenbasis bilden. Bild 11.2a zeigt einen Satz von Trainingsmustern, in denen die Objekte jeweils um ein Pixel verschoben wurden und außerdem Störungen auftreten. Die Trainingsmuster wurden so ausgewählt, daß die Verschiebungsdetektion zum einen unabhängig von der Bewegungsrichtung ist, weiterhin unabhängig von der Form der verschobenen Objekte ist und außerdem für alle Arten von Störungen gleichermaßen funktioniert. Nur die Berücksichtigung dieser Vorgaben garantiert, daß die ermittelten CNN-Templates tatsächlich zu einer Verschiebungsvektorschätzung führen und nicht etwa eine Adaption an gewisse Formen oder spezielle Bewegungen darstellen.

Wie in Bild 11.1aa-ac enthält ein Bild in 11.2a(1) ein Objekt, welches im zweiten Bild in 11.2a(2) wiedergefunden werden soll. Das entsprechende Bild in 11.2a(2) zeigt das verschobene Objekt mit additiven Störungen und 11.2a(3) stellt das Referenzbild mit dem extrahierten verschobenen

[1] siehe Abschnitt 8.2.1

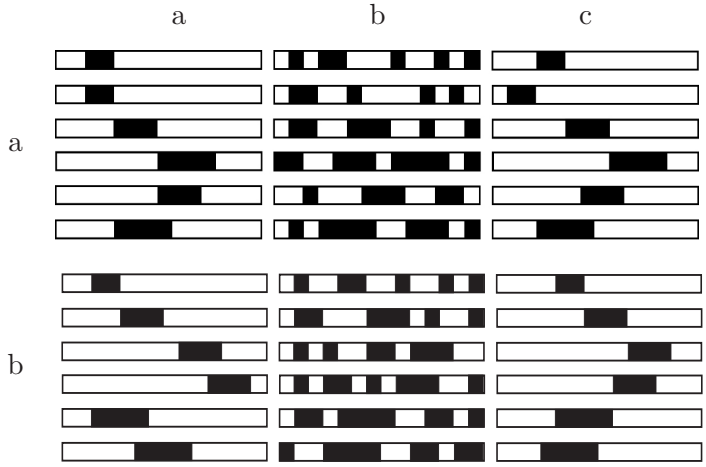


Bild 11.1: Lernmuster für die eindimensionale Detektion von Objekten, die um a) ein bzw. b) zwei Pixel verschoben wurden

Objekt dar. Im CNN-Kontext entspricht ein Bild aus 11.2a(1) der CNN-Eingangsaktivität, ein Bild aus 11.2a(2) dem Initialisierungszustand und ein Bild aus 11.2a(3) der erwünschten CNN-Ausgangsaktivität. Das Training wurde identisch zum eindimensionalen Fall durchgeführt; alle Trainingsmuster wurden einem bestimmten CNN präsentiert und der Gesamtfehler zwischen den CNN-Ausgängen im Gleichgewichtszustand und den Referenzbildern aus 11.2a(3) berechnet. Unter Verwendung der Iterative Annealing Optimierungsmethode wurde das CNN-Template

$$\begin{aligned}(&\alpha^1_{-1-1}, \alpha^2_{-1-1}, \alpha^1_{-10}, \alpha^2_{-10}, .., \alpha^2_{11}, \beta^1_{-1-1}, \beta^2_{-1-1}, .., \beta^2_{11}, I) = \\ &(5.5, \text{-}1.9, 3.1, 11.4, 5.5, \text{-}1.9, 3.1, 11.4, 47.7, \text{-}16.6, 3.1, 11.4, \\ &5.5, \text{-}1.9, 3.1, 11.4, 5.5, \text{-}1.9, 14.5, \text{-}18.9, 7.9, 23, 14.6, \text{-}18.9, \\ &7.9, 23, 1.3, \text{-}21.3, 7.9, 23, 14.6, \text{-}18.9, 7.9, 23, 14.6, \text{-}18.9, 14.1)\end{aligned} \tag{11.3}$$

mit polynomial gekoppelten Zellen zweiter Ordnung gemäß (8.3) mit der Nachbarschaft 1 und der Zellausgangsfunktion (8.9) ermittelt. Mittels des oben beschriebenen CNN ist es nun möglich, um ein Pixel verschobene Objekte zu detektieren. Um dies zu beweisen, wurde dem ermittelten CNN eine Anzahl von Testmustern vorgelegt, welche in dieser oder ähnlicher Form *nicht* für das Training verwendet wurden. Diese in Bild 11.2b dargestellten Testmuster wurden so ausgewählt, daß eine Überprüfung der Robustheit des ermittelten CNNs gegenüber Veränderungen in den Eingangsdaten ermöglicht wird. Die dargestellten Testmuster besitzen die folgenden Charakteristika:

1. zwei der Testobjekte (zweite und dritte Zeile) besitzen eine andere Form als die im Training berücksichtigten Objekte,

2. zwei der Testobjekte (erste und dritte Zeile) enthalten neben dem zu detektierenden Objekt weitere Objekte als Störung, die sogar Teil des Trainings waren und

3. ein Testobjekt in der vierten Zeile enthält zwei zu detektierende Objekte, welche sich in unterschiedliche Richtungen bewegen.

In der rechten Spalte von Bild 11.2b ist zu sehen, daß in allen Fällen die verschobenen Objekte fehlerfrei detektiert wurden.

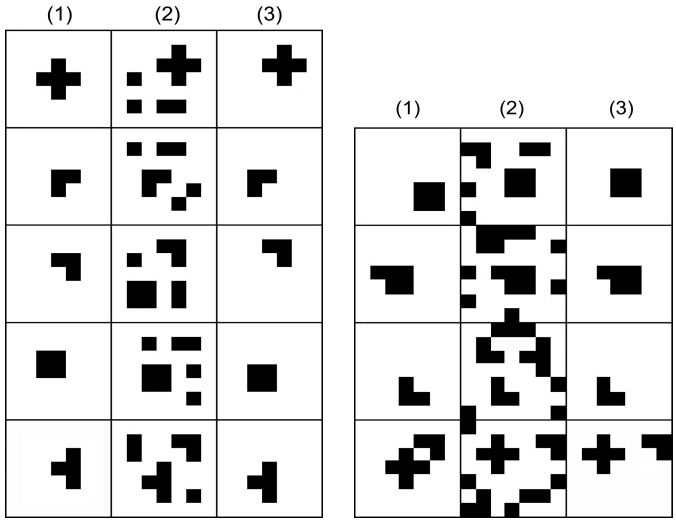


Bild 11.2: a) Lern- und b) Testmuster für die zweidimensionale Detektion von Objekten, die ein Pixel verschoben wurden; in (1) sind die Eingangsbilder mit den Originalobjekten dargestellt, in (2) die Bilder mit den verschobenen und von Störungen umgebenen Objekten und in (3) die Referenzbilder mit den extrahierten Objekten

Auch im zweidimensionalen Fall ist mit den ermittelten CNN nur die Objektdetektion für eine feste Verschiebung möglich. Bei den durchgeführten Untersuchungen hat es sich herausgestellt, daß es nicht möglich ist, mittels CNN mit polynomialer Kopplung eine variable Verschiebung von beispielsweise einem *und* zwei Pixeln Verschiebung zu detektieren. Deshalb wurden die CNN-Parameter ebenfalls für eine feste Verschiebung von zwei Pixeln ermittelt. Die Trainings- und Testmuster sind in Bild 11.3 dargestellt. Das Training verlief identisch zum zweidimensionalen Fall mit einem Pixel Verschiebung. Die ermittelten Parameter lauten

$$\begin{aligned}(\alpha^1_{-1-1}, \alpha^2_{-1-1}, \alpha^1_{-10}, \alpha^2_{-10}, .., \alpha^2_{11}, \beta^1_{-1-1}, \beta^2_{-1-1}, .., \beta^2_{11}, I) = \\ (6.2, 40.3, 8.4, \text{-}2.6, 6.2, 40.3, 8.4, \text{-}2.6, 22.5, 6.5, 8.4, \text{-}2.6, 6.2, \\ \text{-}40.3, 8.4, 2.6, 6.2, 40.3, 2.6, 17.6, 3.9, \text{-}53.2, 2.6, 17.6, 3.9, \\ \text{-}53.2, \text{-}11.8, \text{-}2.9, 3.9, \text{-}53.2, 2.6, 17.6, 3.9, \text{-}53.2, 2.6, 17.6, 13.6)\,.\end{aligned} \tag{11.4}$$

Prinzipiell sollte es ohne weiteres möglich sein, in entsprechender Weise CNN zu ermitteln, mit denen eine Objektverschiebung von drei oder mehr Pixeln detektiert werden kann. Es ist jedoch sehr wahrscheinlich, daß für diese Fälle CNN mit polynomialen Kopplungsfunktionen höherer Ordnung eingesetzt werden müssen, als dies bei der Verschiebung von maximal zwei Pixeln der Fall gewesen ist.

11.1.3 Die Detektion von bewegten Objekten in photographischem Bildmaterial

Um nachzuweisen, daß mit Hilfe der in Abschnitt 11.1.2 ermittelten CNN, ebenfalls photographisches Bildmaterial verarbeitet werden kann, wurden einige Bilder einer Videosequenz mittels eines CNN verarbeitet. In den Bildern 11.4a und 11.4b sind zwei aufeinanderfolgende Bilder einer

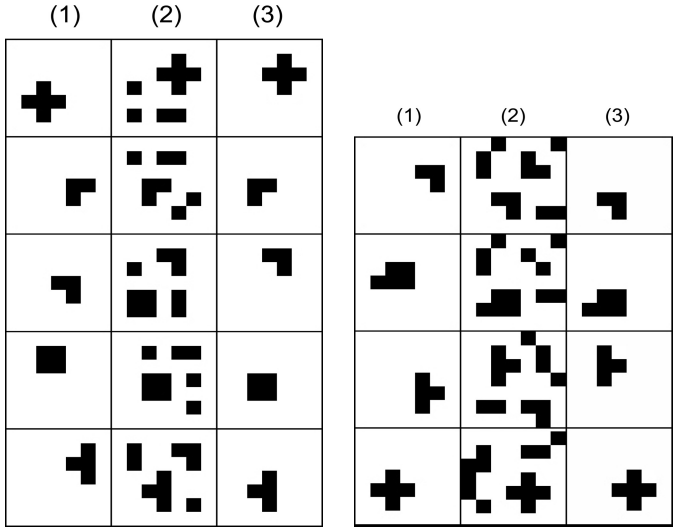


Bild 11.3: a) Lern- und b) Testmuster für die zweidimensionale Detektion von Objekten, die zwei Pixel verschoben wurden

Sequenz eines fliegenden Flugzeugs dargestellt. Der erste Verarbeitungsschritt stellt die Binarisierung ([cnnlib],S. 57) der Bilder dar, wie in 11.4c und 11.4e zu sehen ist. Die Gestalt des Flugzeugs nach der Binarisierung ist in beiden Bildern nicht identisch, was im Gegensatz zu künstlichem Bildmaterial bei realen Bildsequenzen nicht weiter verwundert, da die Konturen von realen Objekten fließende Übergänge aufweisen. Im folgenden wird gezeigt, daß trotz dieser Differenzen eine Detektion des verschobenen Flugzeugs mittels Zellularen Neuronalen Netzwerken möglich ist, was die *Robustheit* des ermittelten CNN gegenüber statistischen Ansätzen unterstreicht. Im Differenzbild 11.4g zwischen den Bildern 11.4c und 11.4e ist deutlich die Verschiebung zwischen den Flugzeugbildern zu erkennen. Bild 11.4d zeigt das extrahierte Flugzeug, welches aus Bild 11.4c über eine markierte Objektextraktion ([cnnlib], S. 26) erhalten wurde. Schließlich ist in Bild 11.4f das Resultat der Detektion des verschobenen Objektes unter Verwendung der Bilder 11.4d und 11.4e gegeben. Dabei wurden die in Abschnitt 11.1.2 ermittelten CNN-Parameter verwendet. Um zu zeigen, daß das Ergebnisbild 11.4d ausschließlich das verschobene Flugzeug enthält, wurde in Bild 11.4h ein Differenzbild zwischen den Bildern 11.4e und 11.4f gebildet. Bis auf einige Pixel am Rand des binarisierten Flugzeugs ist das Flugzeug verschwunden, was bedeutet, daß die Detektion des verschobenen Objektes erfolgreich durchgeführt wurde.

11.2 Das Verfahren der Verschiebungsvektorschätzung mit CNN

Viele Ansätze der Bewegungsschätzung aus Videosequenzen basieren auf einer Verschiebungsvektorschätzung (z.B. [Fe99]), bei der Bildbereiche aus einem Bild in einem anderen wiedergefunden werden sollen. Wie in Kapitel 4 beschrieben, ist das Verfahren des Block-Matchings eine statistische Methode zur Verschiebungsvektorschätzung. In diesem Kapitel wird nun ein CNN-Algorithmus beschrieben, mit dessen Hilfe eine Verschiebungsvektorschätzung durchgeführt werden kann. Das Verfahren weist neben seiner extrem hohen Verarbeitungsgeschwindigkeit eine besondere Robustheit gegenüber Störungen in der Umgebung des zu detektierenden Objektes auf.

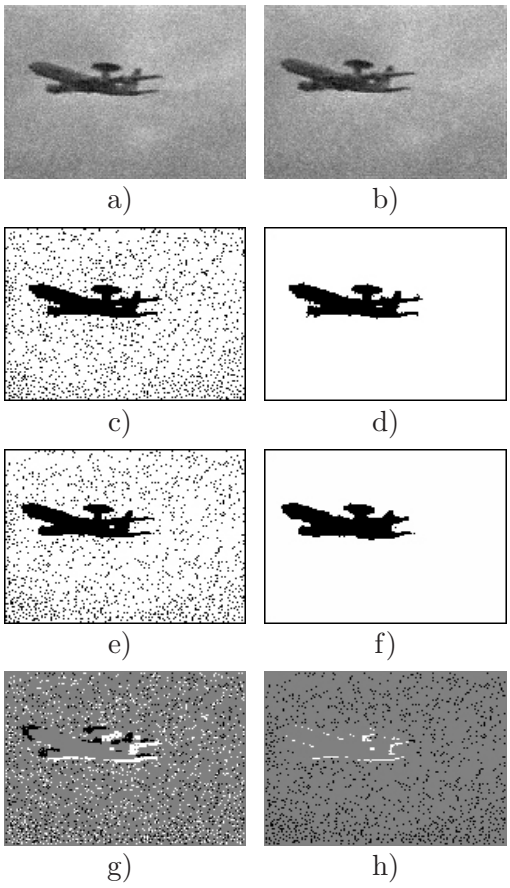


Bild 11.4: Detektion von bewegten Objekten in photographischem Bildmaterial

In Bild 11.5 ist das vorgeschlagene Verfahren dargestellt. Zuerst wird das zu betrachtende Objekt aus Eingangsbild 1 extrahiert. Anschließend wird das verschobene Objekt im Eingangsbild 2 detektiert. Der nächste Verarbeitungsschritt reduziert das Objekt in den beiden resultierenden Bildern auf einen Zentralpixel mittels eines CNN zur sogenannten „center point detection“, und schließlich werden diese Punkte horizontal und vertikal mittels einer sogenannten „CCD“ (connected component detection) auf die Randzeile bzw. -spalte projeziert. Um dann eine numerische Angabe des Verschiebungsvektors angeben zu können, müssen abschließend noch die Ränder der CCD-Bilder ausgewertet werden. Die Einzelschritte dieses Verfahrens werden in den folgenden Abschnitten genauer erläutert.

11.2.1 Extraktion eines maskierten Objektes

Bei diesem Verarbeitungsschritt enthält das Eingangsbild einen einzelnen schwarzen Pixel der ein Objekt innnerhalb des Initialzustandsbildes maskiert. Das Resultat der CNN-Operation ist ein Bild, welches nur das Gesamtobjekt an der entsprechenden Position enthält. Die Template-Elemente des linear gekoppelten CNN für die Extraktion eines maskierten Objektes [cnnlib] haben die Form

$$\mathbf{B}_1 \,..\, \mathbf{B}_4 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1.5 & 0 \\ 0 & 0 & 0 \end{pmatrix} ,$$
$$I_1 \,..\, I_4 = -1.5 \,,$$

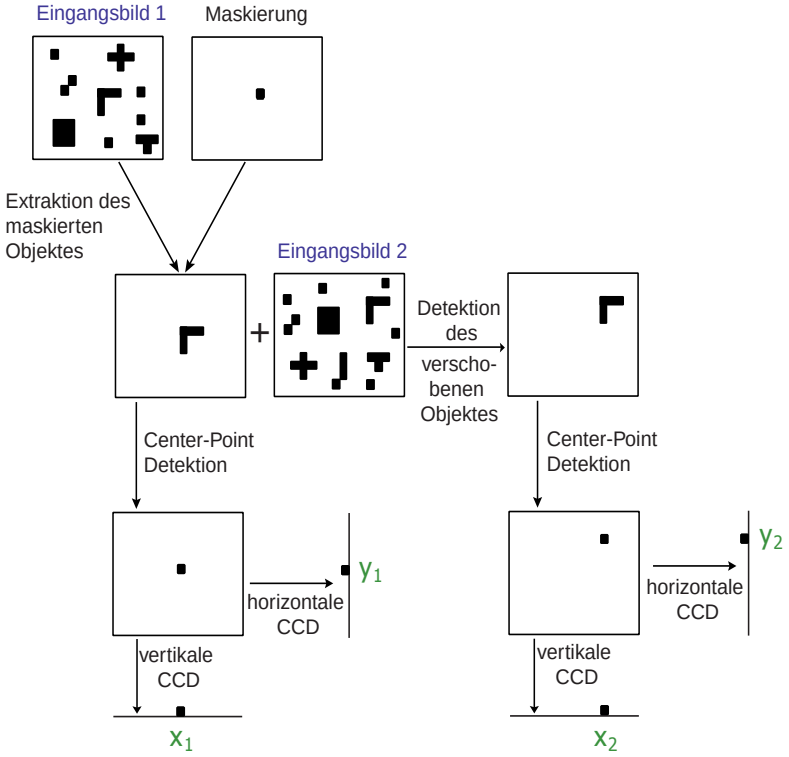


Bild 11.5: Das Verfahren der Verschiebungsvektorschätzung mit CNN

$$\mathbf{A}_1 = \begin{pmatrix} 0 & 0 & 0 \\ 1.5 & 3 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \mathbf{A}_2 = \begin{pmatrix} 0 & 1.5 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$
$$\mathbf{A}_3 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 3 & 1.5 \\ 0 & 0 & 0 \end{pmatrix} \text{ und } \mathbf{A}_4 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 1.5 & 0 \end{pmatrix}.$$

Mit jedem dieser Templatesätze kann ein Pixel des Objektes in der entsprechenden Richtung extrahiert werden. So entspricht z.B. $\mathbf{A}_1$, $\mathbf{B}_1$ und I_1 einer Extraktion von links nach rechts. Die entsprechenden vier CNNs müssen iterativ angewandt werden bis ein stationärer Zustand erreicht ist.

11.2.2 Detektion des verschobenen Objektes

Anschließend muß das extrahierte Objekt in dem zweiten Bild detektiert werden, wobei es sich um einige Pixel bewegt haben kann. Im zweiten Bild können ebenfalls Rauschen und andere Objekte auftreten. Bei diesem Schritt kommen die in Abschnitt 11.1.2 ermittelten CNN-Templates zum Einsatz und führen zu dem gewünschten Resultat.

11.2.3 Center-Point-Detektion

Bei der sogenannten Center-Point-Detektion [cnnlib] werden Objekte, die sich im Eingangsbild befinden, so lange verkleinert, bis sie im Ausgangsbild nur noch aus einem Pixel bestehen. Dazu

werden die folgenden acht unterschiedlichen CNN-Templates

$$\mathbf{B}_1 \,..\, \mathbf{B}_8 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} ,$$

$$I_1 \,..\, I_8 = -1 \,,$$

$$\mathbf{A}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 4 & -1 \\ 1 & 0 & 0 \end{pmatrix} , \mathbf{A}_2 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 6 & 0 \\ 1 & 0 & -1 \end{pmatrix} ,$$

$$\mathbf{A}_3 = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 4 & 0 \\ 0 & -1 & 0 \end{pmatrix} \,..\, \text{und } \mathbf{A}_8 = \begin{pmatrix} 1 & 0 & -1 \\ 1 & 6 & 0 \\ 1 & 1 & 1 \end{pmatrix}$$

iterativ angewandt. Jede Verwendung eines CNN mit einem dieser Templatesätze entfernt ein Pixel des Objekts in der entsprechenden Richtung, so daß am Ende im stabilen Zustand nur ein Zentralpixel übrig bleibt. Bild 11.6 zeigt ein Beispiel für eine Center-Point-Detektion. Es sind insgesamt drei Iterationen notwendig bis der stabile Zustand erreicht wird.

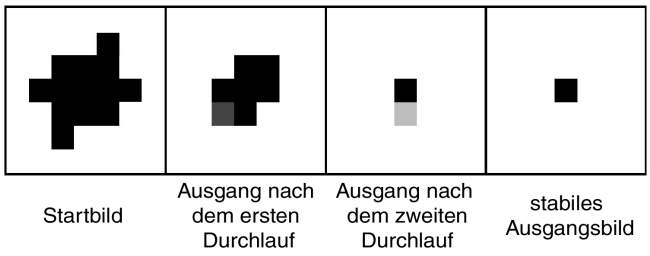


Bild 11.6: Beispiel einer Center-Point Detektion

11.2.4 Connected-Component-Detektion

Das Ziel der darauffolgenden sogenannten Connected-Component-Detektion ist es, die Zentralpixel des vorigen Verarbeitungsschrittes an den unteren bzw. rechten Rand des Bildes zu projezieren. Dies beschleunigt die Ermittlung der Objektposition erheblich, da abschließend nur die rechte Spalte und die untere Zeile des CCD-Bildes ausgewertet werden muß. Die horizontale CCD [cnnlib],

$$\mathbf{B} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} , \; I = 0, \; \mathbf{A} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 2 & -1 \\ 0 & 0 & 0 \end{pmatrix} , \tag{11.5}$$

und die vertikale CCD [cnnlib],

$$\mathbf{B} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} , \; I = 0, \; \mathbf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{pmatrix} , \tag{11.6}$$

können ebenfalls durch ein Zellulares Neuronales Netzwerk mit linear gekoppelten Zellen realisiert werden.

11.2.5 Ein Beispiel zur Verschiebungsvektorschätzung mit CNN

In Bild 11.7 ist ein Beispiel für die komplette Prozedur der Verschiebungsvektorschätzung mit CNN präsentiert. Die Ausgangsbilder 11.7a und 11.7d zeigen ein Kreuz als das zu detektierende Objekt und umgebende, additive Pixelstörungen. In Bild 11.7c ist das extrahierte Kreuz von Abb. 11.7a zu sehen, wobei der Markierungspunkt in Bild 11.7b verwendet wurde. Der wichtigste Schritt ist dann in Abb. 11.7e dargestellt, wo als Detektionsergebnis ein verschobenes Kreuz zu sehen ist. Danach werden die Zentralpixel der Bilder 11.7c und 11.7e bestimmt und es ergeben sich die Bilder 11.7f und 11.7i. Anschließend werden die Zentralpixel zum rechten (Fig. 11.7g, 11.7j) und zum unteren Bildrand (Fig. 11.7h, 11.7k) geschoben, was durch eine horizontale und eine vertikale CCD realisiert wird. Im abschließenden Verarbeitungsschritt müssen nur noch die

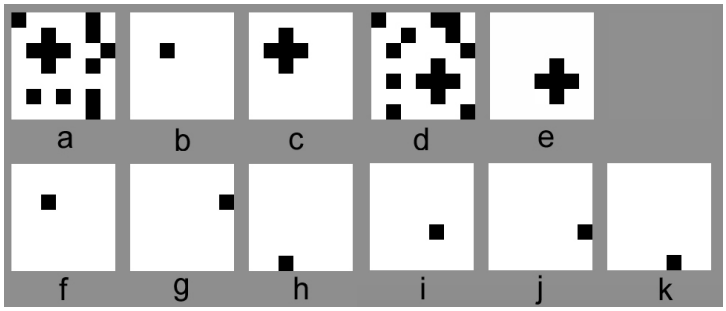


Bild 11.7: Ein Beispiel einer Verschiebungsvektorschätzung mit CNN (siehe Text)

Randzeilen der resultierenden CCD-Bilder ausgewertet werden und zu einem Verschiebungsvektor zusammengesetzt werden; als Ergebnis ergibt sich dabei

$$\vec{d} = \begin{pmatrix} y_2 - y_1 \\ x_2 - x_1 \end{pmatrix} = \begin{pmatrix} 3-5 \\ 5-3 \end{pmatrix} = \begin{pmatrix} -2 \\ 2 \end{pmatrix} . \tag{11.7}$$

In diesem Kapitel wurde gezeigt, daß eine Verschiebungsvektorschätzung nicht nur durch statistische Methoden realisiert werden kann, im Gegenteil, eine Verschiebungsvektorschätzung kann ebenso mit Hilfe von Zellularen Neuronalen Netzwerken erreicht werden. Bei der Verarbeitungsgeschwindigkeit und der Robustheit sind die CNN den statistischen Verfahren sogar deutlich überlegen.

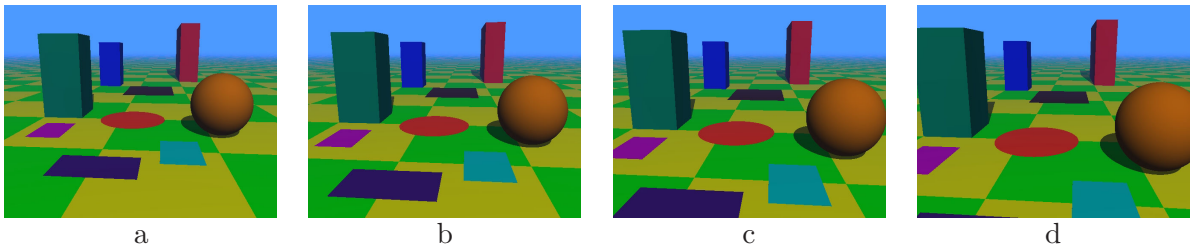
Kapitel 12

Direkte Hinderniserkennung mit CNN

Bisher wurde gezeigt, daß einzelne Schritte des statistischen Verfahrens der Bewegungsschätzung in planaren Welten aus monokularen Bildsequenzen mit einem CNN bearbeitet werden können. Wie bereits mehrfach erwähnt, sind die Vorteile eines solchen Vorgehens die gesteigerte Robustheit gegenüber Störungen und die immense Beschleunigung des Berechnungsvorgangs. Es verbleiben jedoch noch einige rechenaufwändige Schritte des statistischen Verfahrens der Hinderniserkennung, die analytisch gelöst werden müssen. So müßte beispielsweise weiterhin eine Schätzung der Bewegungsgrößen aus den ermittelten Verschiebungsvektoren und eine darauf aufbauende Hinderniserkennung erfolgen. Deshalb wurde im Rahmen dieser Arbeit ein Konzept entwickelt, durch Anwendung eines CNN eine *direkte* Hinderniserkennung zu realisieren. In diesem Kapitel wird in Abschnitt 12.1 und 12.2 beschrieben, wie sowohl aus synthetisch generierten Bildsequenzen als auch aus realen Verkehrssequenzen eine rein CNN-basierte Hinderniserkennung erzielt werden kann.

12.1 Hinderniserkennung in synthetisch erstellten Bildsequenzen

In Bild 12.1 sind einige Bilder einer synthetisch erstellten Bildsequenz dargestellt. Die Sequenz zeigt eine Fahrt über eine strukturierte Ebene, auf der dreidimensionale geometrische Figuren stehen. In Anlehnung an reale Verkehrsszenen ist die Bewegungsrichtung mit der Blickrichtung identisch.



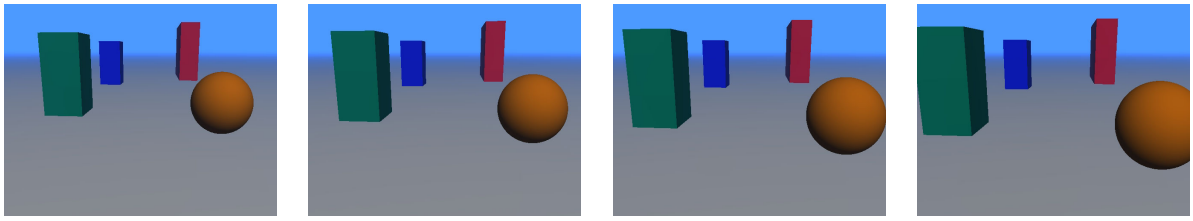
a b c d

Bild 12.1: Einige Bilder einer synthetisch generierten Fahrsequenz

Die Aufgabe bei der Hinderniserkennung mit CNN bestand nun darin, bei Vorgabe von zwei aufeinanderfolgenden Bildern dieser Sequenz, die dreidimensionalen Objekte zu extrahieren. Strukturen, die Teil der Ebene sind, sollen vom CNN herausgefiltert werden, so daß im Ergebnisbild ausschließlich die dreidimensionalen Objekte übrigbleiben, die dann als potentielle Hindernisse angesehen werden können. Bei diesem Ansatz zur Hinderniserkennung ist eine Modellierung der Ebene nicht länger notwendig, denn sie wird – für den Anwender unbemerkt – intern von dem CNN vorgenommen.

12.1.1 Das Verfahren

In den durchgeführten Untersuchungen[1] hat sich herausgestellt, daß es mit einem CNN nicht möglich ist, die dreidimensionalen Objekte aus der Bildsequenz komplett zu extrahieren, wie in Bild 12.2 gezeigt. Dies ist jedoch nicht weiter verwunderlich, denn es ist ohne eine genaue



a b c d

Bild 12.2: Aus synthetisch generierter Fahrsequenz extrahierte 3D-Objekte

Analyse der dargestellten Szene nicht feststellbar, ob ein Bildpixel, dessen Grauwert sich in den beiden Ausgangsbildern nicht verändert, einen Punkt repräsentiert, der Teil der Ebene ist oder auf der Oberfläche eines der 3D-Objekte liegt. Es liegt nahe, daß eine Klassifizierung von Punkten nur für Kantenpixel[2] realisiert werden kann, da sich ihre Grauwerte von einem Bild zum nächsten verändern. Aus diesem Grund wird das in Bild 12.3 dargestellte Verfahren zur Hinderniserkennung mittels CNN vorgeschlagen. Im ersten Schritt werden die beiden Ausgangsbilder einer Hochpaßfilterung unterzogen, so daß zwei Bilder generiert werden, in denen Objektkanten hervorgehoben sind. Dazu kommt ein CNN mit linearer Kopplung der Zellen mit dem CNN-Gen

$$\begin{aligned} C &= (a_{-1\,-1}, a_{-1\,0}, .., a_{1\,1}, b_{-1\,-1}, .., b_{1\,1}, I) \\ &= (0,0,0,0,2,0,0,0,0,-0.25,-0.25,-0.25, \\ &\quad -0.25,2,-0.25,-0.25,-0.25,-0.25,1.5) \end{aligned} \tag{12.1}$$

zum Einsatz. Im nächsten Schritt erfolgt eine Binarisierung der kantengefilterten Bilder, was ebenfalls durch Einsatz eines CNN realisiert werden kann. Das entsprechende CNN-Gen lautet wie folgt,

$$\begin{aligned} C &= (a_{-1\,-1}, a_{-1\,0}, .., a_{1\,1}, b_{-1\,-1}, .., b_{1\,1}, I) \\ &= (0,0,0,0,2,0,0,0,0, \\ &\quad 0,0,0,0,0,0,0,0,0,1.5). \end{aligned} \tag{12.2}$$

[1] Hierbei wurden CNN mit linearen Gewichtsfunktionen mit den Nachbarschaften 1 bis 3 untersucht, CNN mit polynomialen Gewichtsfunktionen vom Grade 2 bis 3 mit den Nachbarschaften 1 und 2 und CNN mit Delay-Funktionalität mit linearen Gewichtsfunktionen mit den Nachbarschaften 1 bis 2.

[2] sowohl Objekt- als auch Strukturkanten

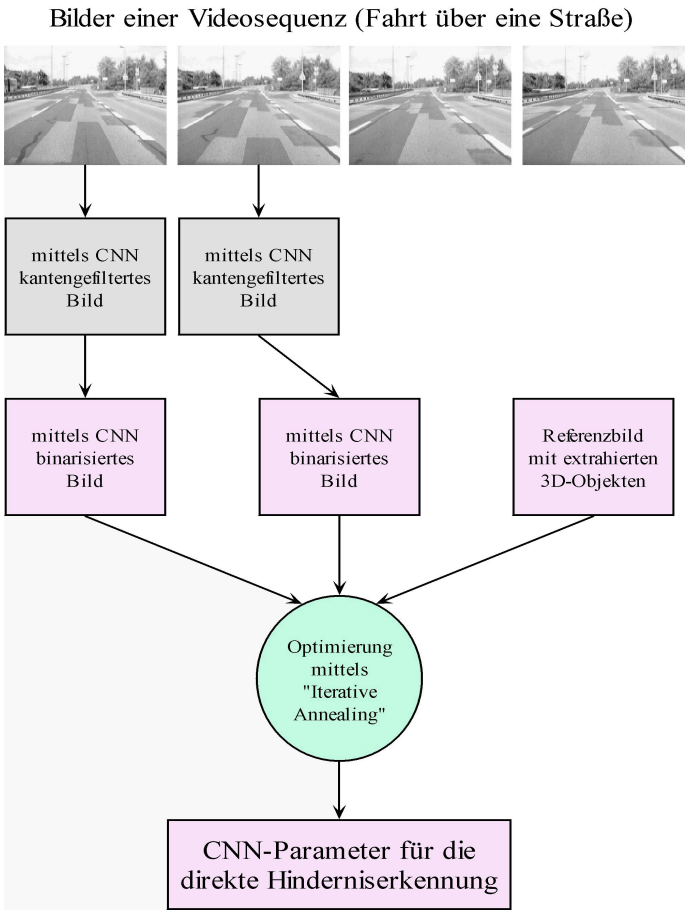
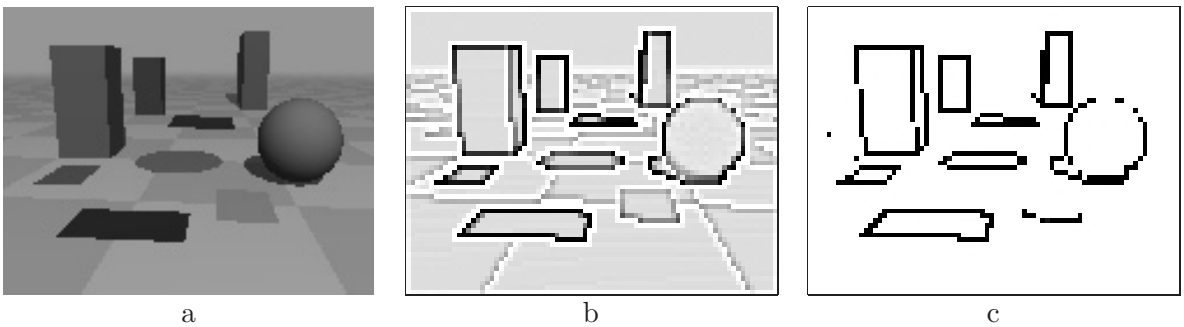


Bild 12.3: Das Verfahren der direkten Hinderniserkennung aus monokularen Bildsequenzen mittels CNN



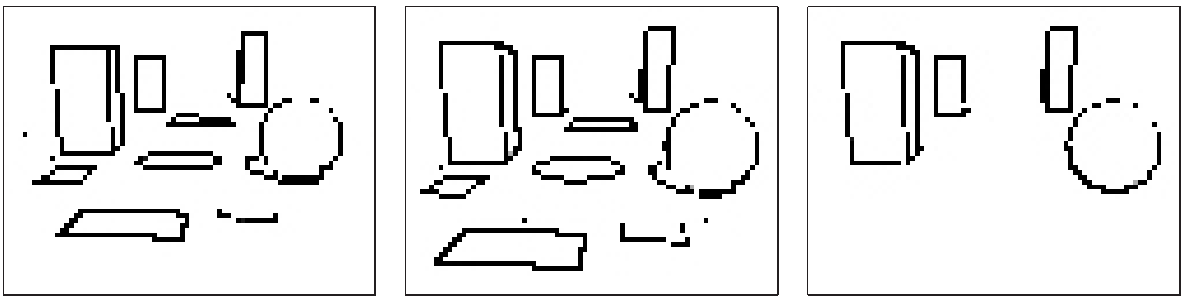
a b c

Bild 12.4: Originalbild (a), CNN-generiertes Kantenbild (b) und mittels CNN binarisiertes Kantenbild (c)

Anschließend werden die CNN-Parameter für die direkte Hinderniserkennung über eine Fehlerminimierung zwischen Referenzbild und CNN-Ausgangsbild mittels Iterative Annealing ermittelt. Die Grundlage dieser Optimierung bilden die beiden binarisierten Eingangsbilder zusammen mit dem Referenzbild, welches ausschließlich die binarisierten Kanten der 3D-Objekte enthält.

12.1.2 Ergebnisse

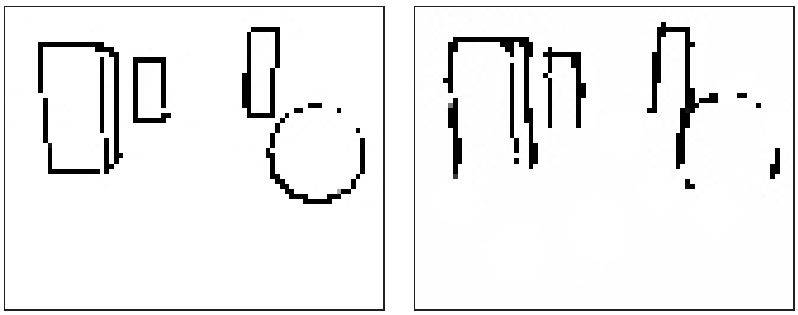
In diesem Abschnitt wird das in Bild 12.3 beschriebene Verfahren auf die synthetische Bildsequenz aus Bild 12.1 angewandt. Bild 12.4a zeigt ein Originalbild der Sequenz und Bild 12.4b das vom CNN generierte Kantenbild. In Bild 12.4c ist weiterhin das binarisierte Ergebnisbild dargestellt. Damit ergibt sich ein Satz von Trainingsbildern, wie er in Bild 12.5 zu sehen ist. Mit diesen Trainingsbildern wurde ein CNN ermittelt, welches in der Lage ist, die 3D-Objekte



a b

Bild 12.5: Die beiden binarisierten Trainingsbilder (a,b) und das Referenzbild (c)

aus der Szene zu extrahieren. Es hat sich herausgestellt, daß für diese Aufgabe ein CNN gemäß (8.3) mit der Nachbarschaft 2 und einer nichtlinearen Kopplung der Zellen vom Grade 3 notwendig ist. Der mittels Iterative Annealing ermittelte Satz von CNN-Parametern ist im Anhang I dargestellt.

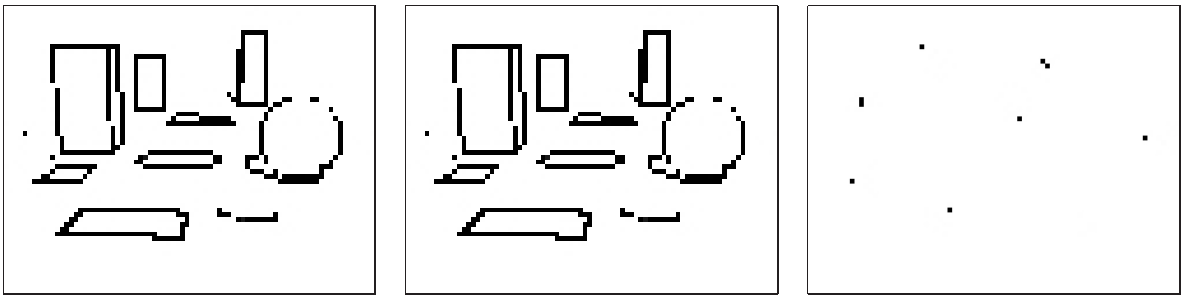


a b

Bild 12.6: Referenzbild (a) und das mit einem CNN berechnete Ergebnisbild (b)

Bild 12.6b zeigt das mit einem CNN berechnete Ausgangsbild. Es kann festgestellt werden, daß das CNN ausschließlich die dreidimensionalen Objekte extrahiert und Strukturen auf der Ebene entfernt hat. Es fällt jedoch auf, daß die Unterkanten der Objekte in dem Ausgangsbild fehlen, was aber keineswegs eine Unzulänglichkeit darstellt. Im Gegenteil, es bestätigt noch einmal, daß alle Objekte auf der Ebene entfernt worden sind, denn es ist prinzipiell nicht möglich, Bildpunkte von Strukturen auf der Ebene und Bildpunkte von Objektunterkanten zu unterscheiden, da sie sich in exakt derselben Weise bewegen.

Um auszuschließen, daß das CNN nur die im Bild enthaltenden Formen gelernt hat, wurden dem CNN als Test zwei identische Bilder vorgesetzt. In Bild 12.7c ist das Ergebnis dargestellt, und wie man sieht, wurden keinerlei Objekte extrahiert. Dies verdeutlicht, daß aus nur *einer* Ansicht einer Szene keine Rückschlüsse auf deren dreidimensionale Struktur möglich sind.



a b

Bild 12.7: Die beiden (identischen) Eingangsbilder (a,b) und das Resultat nach einer Berechnung mit CNN (c)

Als ein weiterer Test für die Verwendbarkeit der ermittelten CNN-Parameter wurde das dazugehörige CNN zur Verarbeitung von Daten eingesetzt, die im Parametertraining nicht berücksichtigt wurden. Als Eingangsdaten wurden die zeitlich folgenden Bilder der bereits vorgestellten synthetischen Szene verwendet. In Bild 12.8b und 12.8d sind die Ausgangsbilder des CNN dargestellt, wobei die oben vorgestellten CNN-Parameter verwendet wurden. Zum Vergleich sind ebenfalls die „optimalen“ Ergebnisbilder in Abb. 12.8a und 12.8c dargestellt. Man erkennt auch in den Ausgangsbildern 12.8b und 12.8d das Verhalten, daß ausschließlich die 3D-Strukturen extrahiert und alle Strukturen auf der Ebene[3] eliminiert wurden.

12.2 Hinderniserkennung in realen Verkehrssequenzen

Nachdem in Abschnitt 12.1 gezeigt wurde, wie eine direkte Hinderniserkennung in planaren Welten bei künstlich generierten Bildsequenzen realisiert werden kann, wird das entwickelte Verfahren in diesem Abschnitt ebenfalls an realen Verkehrssequenzen getestet. Bild 12.9 zeigt einige Bilder einer solchen Fahrsequenz.

Die einzelnen Schritte der Vorverarbeitung sind in Bild 12.10 dargestellt. Die erste Zeile zeigt die Kantenbilder, die mit dem gleichen CNN-Template wie in Abschnitt 12.1.1 ermittelt wurden und in der zweiten Zeile sind die mittels CNN binarisierten Kantenbilder dargestellt.

Damit ergibt sich ein Satz von Trainingsbildern, wie sie in Bild 12.11 zu sehen sind. Das CNN-Eingangsbild stellt Abb.12.11a dar, als Initialisierungszustand des CNN dient Abb.12.11b, und das Referenzbild mit dem erwünschten Ergebnis ist in Abb.12.11c zu sehen.

12.2.1 Ergebnisse

Mittels verschiedener Optimierungsverfahren[4] wurde versucht, einen Satz von CNN-Parametern für ein polynomial gekoppeltes CNN zu erhalten. Der Grad der polynomialen Zellkopplungen

[3] auch die Unterkanten der 3D-Objekte

[4] Iterative Annealing, Fast Annealing, Powell und Simplex

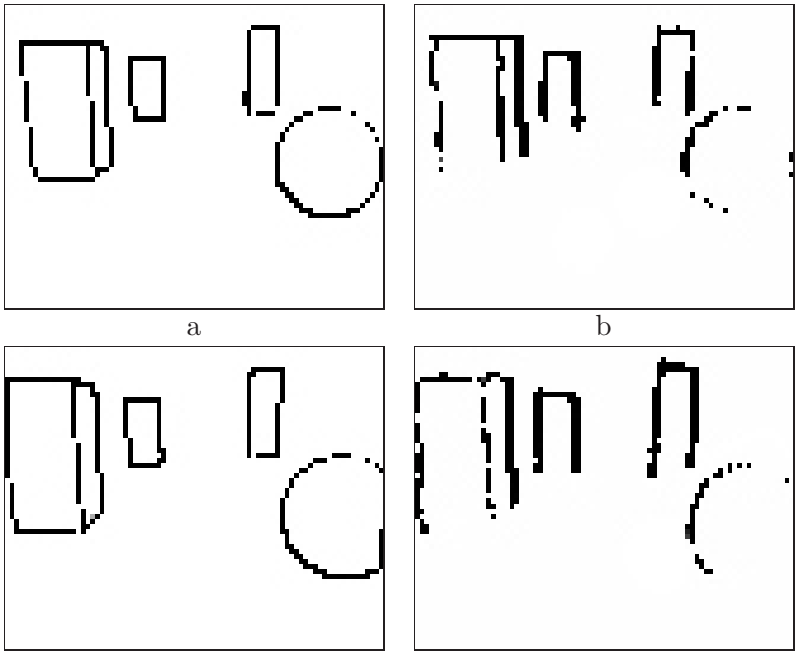


Bild 12.8: Die vom CNN generierten Ergebnisbilder (b und d) für die folgenden Bilder der künstlichen Sequenz und die „optimalen“ Ergebnisbilder (a und c)

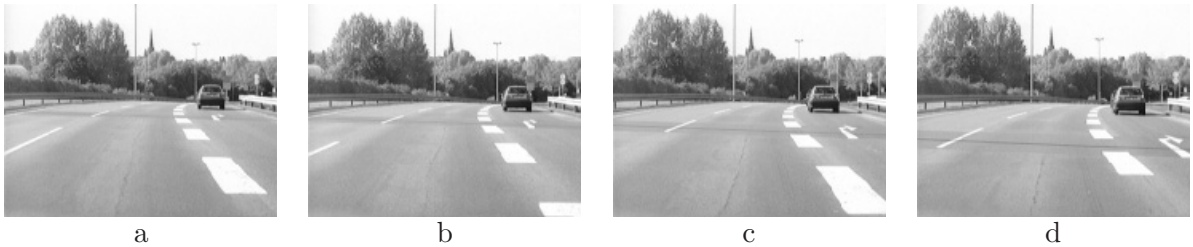


Bild 12.9: Einige Bilder einer realen Fahrsequenz

wurde von 1 bis 5 variiert. Es wurden zahlreiche Optimierungsversuche durchgeführt, wovon einige zu geeigneten CNN-Parametern für die CNN-basierte Hinderniserkennung in realen Verkehrsszenen führten. Ebenso wie in Abschnitt 12.1 wurden die besten Ergebnisse mit einem CNN der Nachbarschaft 2 bei einer nichtlinearen Kopplung der Zellen von Grade 3 erhalten. Die ermittelten CNN-Parameter sind im Anhang I angeführt.

In Bild 12.12 sind die Ergebnisse der besten Optimierung dargestellt. Es kann festgestellt werden, daß die Aufgabe der Beseitigung von Strukturen innerhalb der Straßenebene für das ausgehende Bildpaar sehr gut erlernt wurde. Dies zeigt sich daran, daß das Bild 12.12a nahezu identisch zum Referenzbild 12.11c ist. Auch für das darauffolgende Bildpaar funktioniert die Strukturbeseitigung auf der Straße noch recht gut, wie in Abb. 12.12b zu erkennen ist. Erst bei dem dritten Bildpaar ist das ermittelte CNN nicht mehr in der Lage, eine Objektextraktion durchzuführen, was in Bild 12.12c dargestellt ist. An dieser Stelle müßte eine weitere Optimierung stattfinden. Der Grund für ein solches Verhalten liegt möglicherweise darin, daß ein CNN mit einer polynomialen Zellkopplung vom dritten Grade noch nicht die optimale Konfiguration zur Lösung des gestellten Problems darstellt. Möglicherweise sind CNN mit noch höheren Polynomgraden in

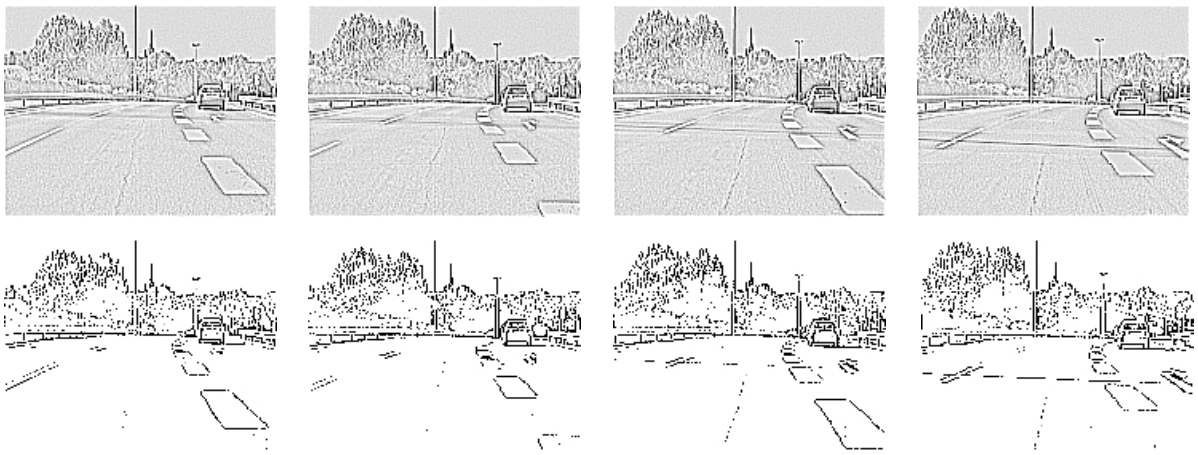
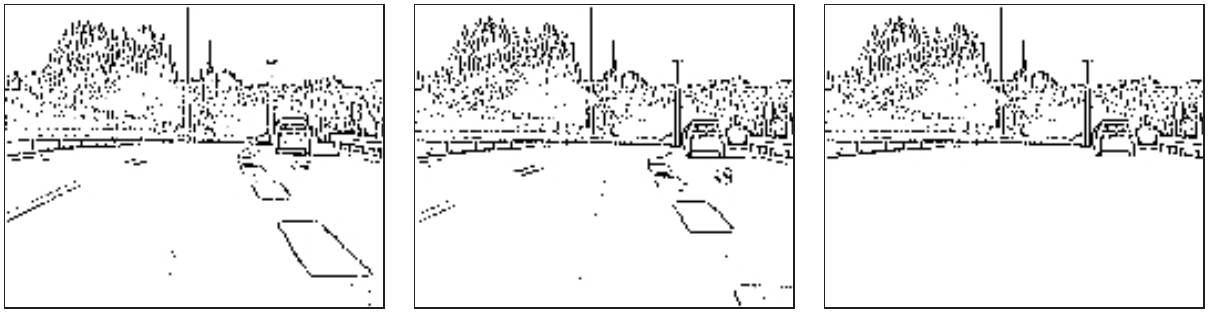
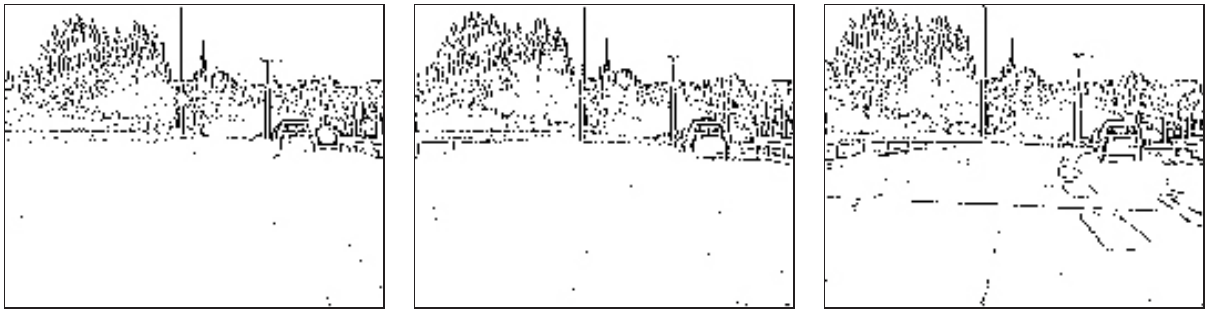


Bild 12.10: Die CNN-generierten Kantenbilder der realen Fahrsequenz und dessen Binarisierungsergebnis



a b c

Bild 12.11: Zwei binarisierte Trainingsbilder (a,b) und das Referenzbild (c)



a b c

Bild 12.12: Die mittels CNN generierten Ergebnisbilder

der Kopplung der Zellen oder mit noch größeren Nachbarschaften oder mit Delay-Funktionalität besser geeignet, das Problem der Hinderniserkennung in planaren Welten zu lösen. Hierzu sind allerdings noch weitergehende Untersuchungen notwendig. Jedoch zeigen diese vielversprechenden ersten Resultate sehr eindrucksvoll, welche enorme Leistungsfähigkeit Zellulare Neuronale Netzwerke für die Bewegtbildverarbeitung aufweisen.

Teil III

Vergleich

Kapitel 13

Vergleich der Hinderniserkennung unter Verwendung statistischer Methoden und CNN

13.1 Einleitung

In diesem Kapitel erfolgt ein Vergleich der statistischen Hinderniserkennung mit der CNN-basierten Hinderniserkennung. Aus den folgenden beiden Gründen erweist sich ein solcher Vergleich als schwierig. Einerseits unterliegen statistische und neuronale Methoden grundlegend unterschiedlichen Vorgehensweisen. Beim statistischen Ansatz erfolgt eine genaue Analyse des Problems durch Aufstellung der beschreibenden Formeln, die anschließend entweder analytisch oder numerisch gelöst werden. Bei den neuronalen Netzen hingegen wird häufig ausgehend von der Aufstellung eines möglichst vollständigen Trainingsdatensatzes, ein Training mittels eines geeigneten Optimierungsverfahrens durchgeführt. Wenn die gewählte Netztopologie zur Lösung des Problems geeignet ist und ein Satz von geeigneten Parametern gefunden wurde, dann erfolgt ein Test mit einem Datensatz, der keinesfalls Bestandteil des Trainingsdatensatzes sein sollte. Die erste Schwierigkeit liegt also in der unterschiedlichen Herangehensweise an ein Problem. Der zweite Grund, warum ein Vergleich zwischen statistischen und CNN-basierten Methoden schwierig ist, liegt darin, daß die Ausgangsgrößen eines Zellularen Neuronalen Netzwerkes Bilder sind. Man muß also eine geeignete Kodierung der CNN-Ausgangsbilder finden, um beispielsweise vektorielle Größen[1] zu bestimmen.

13.2 Die Suche nach markanten Bildbereichen

Die Suche nach markanten Bildbereichen besteht aus zwei Teilen: der Bestimmung eines Markantheitsbildes, in dem Objektkanten und -ecken hervorgehoben sind und der Generierung eines Satzes von markanten Bildbereichen aus einem solchen Markantheitsbild. Wie in Abschnitt 3 gezeigt wurde, ist insbesondere die Ermittlung des Markantheitsbildes mit statistischen Methoden sehr rechenintensiv. Allerdings kann dieser Verarbeitungsschritt ebenfalls von Zellularen Neuronalen Netzwerken mit der gleichen Qualität erledigt werden, wie dies in Kapitel 10 verdeutlicht wurde. Der Vorteil von CNN für diesen Anwendungsfall liegt im wesentlichen in der

[1] wie beispielsweise ein Translationsvektor oder ein Normalenvektor einer Ebene

hohen Verarbeitungsgeschwindigkeit. Während die analytische Berechnung des Markantheitsbildes auf den momentan schnellsten Personalcomputern einige Millisekunden beansprucht, kann dies mittels einer CNN-basierten schaltungstechnischen Realisierung in der Größenordnung von 100 Nanosekunden erledigt werden. Der Vergleich der Bestimmung des Markantheitsbildes fällt somit eindeutig zugunsten des CNN aus.

Für die Bestimmung der Liste von markanten Bildbereichen aus dem Markantheitsbild, ist anschließend der Einsatz von analytischen Methoden sinnvoll.

13.3 Verschiebungsvektorschätzung

Die Verschiebungsvektorschätzung nach der Methode des Blockmatchings ist, wie in Kapitel 4 dargestellt, sehr rechenintensiv. Hier wird insbesondere dann ein großer Anteil an Rechenleistung aufgewandt, wenn der Suchbereich nicht durch Vorwissen über die Art der Bewegung eingeschränkt werden kann. Deshalb ist der Einsatz von CNN an dieser Stelle besonders sinnvoll. In Kapitel 11 wurde gezeigt, daß unter Verwendung eines CNN Verschiebungsvektoren zwischen korrespondierenden Bildbereichen geschätzt werden können. Wie in Abschnitt 11.1 beschrieben, sind hierzu CNN mit polynomial gekoppelten Zellen notwendig, zu denen allerdings schon erste Hardwarerealisierungen [LPKH01] vorliegen. Weiterhin wurde in [SFT02] gezeigt, daß die quadratische Kopplung bei der Verschiebungsvektorschätzung mit CNN durch eine Hintereinanderschaltung von zwei linear gekoppelten CNN realisiert werden kann, so daß auch Hardwarerealisierungen von CNN mit linear gekoppelten Zellen eingesetzt werden können.

In dieser Arbeit wurde gezeigt, daß eine Verschiebungsvektorschätzung sowohl für binäres Eingangsmaterial als auch für Graustufenbilder mittels CNN realisiert werden kann. Neben der Robustheit von CNN gegenüber Störungen ist auch hier die extrem hohe Verarbeitungsgeschwindigkeit von besonderer Bedeutung für eine anwendungsorientierte Echtzeitverarbeitung. Deshalb spricht auch bei diesem Verarbeitungsschritt alles für den Einsatz eines CNN für die Verschiebungsvektorschätzung.

13.4 Hinderniserkennung

Das Verfahren der statistischen Hinderniserkennung in planaren Welten aus monokularen Bildsequenzen besteht aus vielen Teilschritten, wie in den Kapiteln 2 bis 7 ausführlich beschrieben. Das gesamte Verfahren ist sehr komplex und rechenintensiv.

In Kapitel 12 wurde gezeigt, daß es prinzipiell möglich ist, eine Hinderniserkennung in planaren Welten mit Hilfe von CNN in einem einzigen Verarbeitungsschritt zu erledigen. Wie ausführlich beschrieben, sind dazu, wie bei der Verschiebungsvektorschätzung, CNN mit polynomialen Zellkopplungen notwendig. Da die bisherigen Ergebnisse allerdings nur auf der Optimierung von zwei Sequenzen beruhen — einer synthetisch erzeugten und einer realen Verkehrssequenz — ist es jedoch notwendig, eine Verifizierung der erhaltenen Ergebnisse durchzuführen. Bei diesen Untersuchungen müssen beispielsweise noch die folgenden Fragen geklärt werden:

- Für wieviele zeitlich aufeinander folgende Bildpaare einer Videosequenz ist ein optimiertes CNN-Template anwendbar?

- Sind die optimierten Templates von dem Bildinhalt abhängig oder nur von Betrag und Richtung der Bewegung?
- Wie wirken sich Erschütterungen aufgrund von Fahrbahnunebenheiten auf die CNN-Hinderniserkennung aus?
- Wie können in einem weiteren Verarbeitungsschritt potentielle Hindernisse von anderen dreidimensionalen Objekten unterschieden und extrahiert werden?

Die vielversprechenden ersten Ergebnisse zur Hinderniserkennung mit CNN zeigen bereits zweifelsfrei, daß Zellulare Neuronale Netzwerke ein ausgezeichnetes Werkzeug für die Bewegungsschätzung und die Hinderniserkennung aus Videosequenzen darstellen.

Kapitel 14

Zusammenfassung

Im Rahmen dieser Arbeit wurde untersucht, inwieweit eine Bewegungsschätzung aus monokularen Bildsequenzen von Staßenverkehrsszenen und eine darauf aufbauende Hinderniserkennung mit Hilfe von statistischen oder neuronalen Methoden realisiert werden kann. Bei dem zugrunde liegenden mathematischen Modell wird angenommen, daß die Umgebung, in der sich ein Fahrzeug bewegt, im wesentlichen eben ist, was für Verkehrsequenzen in guter Näherung erfüllt ist.

Im ersten Teil dieser Arbeit wurde ein statistisches Verfahren zur Bewegungsschätzung vorgestellt und diskutiert. Der erste Schritt dieses Verfahrens stellt die Generierung eines sogenannten *Markantheitsbildes* dar, in welchem Objektkanten und Objektecken visuell hervorgehoben werden. Für die daraus resultierende Liste von markanten Bildbereichen werden anschließend unter Verwendung einer sogenannten *Verschiebungsvektorschätzung*, Korrespondenzen im zeitlich folgenden Bild ermittelt. Ausgehend von dem resultierenden Verschiebungsvektorfeld, werden in dem nächsten Schritt des Verfahrens die *Bewegungsgrößen* ermittelt, also die Rotationsmatrix und der Translationsvektor des Fahrzeugs, beziehungsweise der Kamera. Um abschließend eine Hinderniserkennung realisieren zu können, erfolgt unter Verwendung der Bewegungsgrößen eine *Bewegungskompensation* der Bilddaten. Bei einer solchen Bewegungskompensation wird unter Verwendung der ermittelten Bewegungsgrößen und dem Modell der bewegten Ebene eine Rücktransformation jedes Bildpixels durchgeführt, so daß bei der Bildung eines Differenzbildes zwischen dem bewegungskompensierten Bild und dem tatsächlich aufgenommenen Bild, dreidimensionale Strukturen, die ja das Ebenenmodell verletzen, deutlich hervortreten und somit auf potentielle Hindernisse hinweisen.

Es hat sich gezeigt, daß Fehlmessungen in den Verschiebungsvektoren, welche beispielsweise durch periodische Strukturen auf der Ebene auftreten können, die Bewegungsschätzung und die Hinderniserkennung empfindlich stören. Diese statistischen Ausreißer bewirken, daß trotz der Verwendung von *robusten Schätzmethoden*, eine stabile Hinderniserkennung nur durch die Einbeziehung von *Vorwissen über die Art der Bewegung des Fahrzeugs* realisiert werden kann. Weiterhin führen die Komplexität des Verfahrens und die damit verbundenen hohen Anforderungen an die Rechenleistung der eingesetzten Hardware dazu, daß die für die praktische Anwendbarkeit so wichtige *Echtzeitfähigkeit des Verfahrens* bisher nur für Eingangsbilder mit geringer Auflösung ermöglicht werden konnte.

Speziell für die Bildverarbeitung hat sich das neue Paradigma der *Zellularen Neuronalen Netzwerke* als außerordentlich leistungsfähig erwiesen. Neben der extrem hohen Verarbeitungsgeschwindigkeit von CNN-basierten schaltungstechnischen Realisierungen zeichnen sie sich durch eine hohe Robustheit bei verrauschten oder fehlerhaften Eingangsdaten aus. Für nahezu jedes aktuelle Problem der Bildverarbeitung wurde bisher ein geeignetes CNN bestimmt. Auch für komplexe Aufgabenstellungen aus der Bildverarbeitung, wie beispielsweise die Texturklassifikation, die Spurverfolgung oder die Gewinnung von Tiefeninformation konnten bereits CNN-Programme implementiert und schaltungstechnisch verwirklicht werden.

So konnte auch im zweiten Teil dieser Arbeit gezeigt werden, daß die einzelnen Schritte der Hinderniserkennung aus monokularen Bildsequenzen ebenfalls unter Verwendung eines CNN realisierbar sind. Es wurde demonstriert, daß für die Generierung eines Markantheitsbildes bereits ein Standard-CNN mit linearer Kopplungsfunktion und der Nachbarschaft $r = 1$ verwendet werden kann. Das rechenaufwändige statistische Verfahren der Markantheitsbildberechnung kann somit durch einen einzigen CNN-Verarbeitungsschritt durchgeführt werden. Weiterhin wurde im Rahmen dieser Arbeit gezeigt, daß auch der folgende, rechenintensive Schritt des statistischen Verfahrens der Hinderniserkennung, nämlich die Verschiebungsvektorschätzung, mittels CNN verwirklicht werden kann. Hierzu sind CNN mit polynomialen Kopplungsfunktionen und der Nachbarschaft $r = 1$ notwendig. Bei den durchgeführten Untersuchungen hat sich herausgestellt, daß die CNN-basierten Verarbeitungsschritte den statistischen Methoden in den Punkten *Robustheit* und *Verarbeitungsgeschwindigkeit* deutlich überlegen sind.

Abschließend wurde in dieser Arbeit gezeigt, daß mit Hilfe von CNN sogar eine direkte Hinderniserkennung aus monokularen Bildsequenzen — ohne den Umweg über die Bestimmung der Verschiebungsvektoren und der Bewegungsgrößen — realisiert werden kann. In dem vorgestellten Verfahren wird nach zwei Vorverarbeitungsschritten, die Hinderniserkennung in einem einzigen Schritt unter Verwendung eines CNN mit polynomialen Zellkopplungsgewichten vom Grade $D = 3$ und der Nachbarschaft $r = 2$ durchgeführt. Das vorgeschlagene Verfahren führt zu einer wesentlichen Vereinfachung der Hinderniserkennung in monokularen Bildsequenzen, da die Bewegegungsschätzung aus dem statistischen Verfahren nicht länger notwendig ist. Die Umgehung der expliziten Bewegungsschätzung hat weiterhin den Vorteil, daß der Rechenaufwand stark reduziert wurde und durch den Wegfall der Verschiebungsvektorschätzung und dem damit verketteten Problem der Ausreißer, ist das vorgestellte CNN-basierte Verfahren außerdem sehr robust. Die ersten Resultate, die unter Verwendung von synthetischen und natürlichen Bildsequenzen erhalten wurden, sind überaus vielversprechend und zeigen, daß CNN ausgezeichnet zur Verarbeitung von Videosequenzen geeignet sind.

Anhang A

Lösung von Least Squares Problemen mit allgemeiner Kovarianzmatrix

In diesem Abschnitt wird gezeigt, wie ein Least-Squares-Schätzproblem gelöst werden kann, wenn die Kovarianzmatrix der Meßfehler *kein* Vielfaches der Einheitsmatrix ist, sondern in Form einer beliebigen positiv definiten Kovarianzmatrix gegeben ist. Den Ausgangspunkt bildet die allgemeine Beobachtungsgleichung

$$\vec{z} = \mathbf{H}\vec{x} + \vec{v} \tag{A.1}$$

mit dem Meßwertvektor $\vec{z}$, der Beobachtungsmatrix $\mathbf{H}$, dem Beobachtungsvektor $\vec{x}$ und dem Fehlervektor $\vec{v}$. Es gelte außerdem

$$\mathsf{E}\left[\vec{v}\right] = \vec{0},\ \mathbf{C} = \mathsf{Cov}\left[\vec{v}\right] = \mathsf{E}\left[\vec{v}\vec{v}^T\right] \ .$$

Da die Kovarianzmatrix $\mathbf{C}$ positiv definit ist, ist auch deren Inverse $\mathbf{C}^{-1}$ positiv definit. Entsprechend dem Satz, daß sich jede reelle symmetrische positiv definite Matrix $\mathbf{M}$ in der Form $\mathbf{N}\mathbf{N}^T$ faktorisieren läßt, kann $\mathbf{C}^{-1}$ in der Form $\mathbf{C}^{-1} = \mathbf{R}\mathbf{R}^T$ geschrieben werden. Die Matrix $\mathbf{R}$ wird dabei oft als „Wurzel“ von $\mathbf{C}^{-1}$ bezeichnet, und sie wird benötigt, um eine Gewichtung der Komponenten von $\vec{z}$ entsprechend ihrer Fehlerstruktur vorzunehmen.

Es wird nun gezeigt, daß sich durch eine Transformation mit der Matrix $\mathbf{R}$ ein vorliegendes allgemeines Least-Squares-Problem auf das spezielle Problem mit $\mathsf{Cov}\left[\vec{v}\right] = \sigma \cdot \mathbf{I}$ zurückführen läßt. Dazu wird eine Transformation des Meßwertvektors $\vec{z}$ gemäß

$$\vec{z}' = \mathbf{R}^T\vec{z} \tag{A.2}$$

durchgeführt. Daraus ergibt sich dann

$$\vec{z}' = \mathbf{R}^T\vec{z} = \mathbf{R}^T\mathbf{H}\vec{x} + \mathbf{R}^T\vec{v} := \mathbf{H}'\vec{x} + \vec{v}' \ .$$

Für die Kovarianzmatrix des transformierten Fehlervektors $\vec{v}'$ erhält man schließlich

$$\mathsf{Cov}\left[\vec{v}'\right] = \mathsf{E}\left[\vec{v}' \cdot (\vec{v}')^T\right] = \mathsf{E}\left[\mathbf{R}^T\vec{v} \cdot \vec{v}^T\mathbf{R}\right] = \mathbf{R}^T\mathsf{E}\left[\vec{v} \cdot (\vec{v})^T\right]\mathbf{R} = \mathbf{R}^T\mathbf{C}\mathbf{R}$$

$$= \mathbf{R}^T\left(\mathbf{C}^{-1}\right)^{-1}\mathbf{R} = \mathbf{R}^T\left(\mathbf{R}\mathbf{R}^T\right)^{-1}\mathbf{R} = \mathbf{R}^T\left(\mathbf{R}^T\right)^{-1}\mathbf{R}^{-1}\mathbf{R} = \mathbf{I} \ .$$

Dies bedeutet, daß wenn man den Meßwertfehlervektor mit der Wurzel seiner Kovarianzmatrix multipliziert, dann läßt sich das Problem der Least-Squares-Schätzung mit allgemeiner Kovarianzmatrix des Fehlervektors auf den Fall des Least-Squares-Problems mit einer einfach strukturierten Kovarianzmatrix [SW94] zurückführen.

Anhang B

Quaternionen

B.1 Definition von Quaternionen

Quaternionen ([Te95],[Po95]) sind eine Erweiterung des Körpers $\mathbb{C}$ der komplexen Zahlen. Man erhält sie dadurch, daß zu den reellen Zahlen nicht nur eine[1], sondern drei imaginäre Einheiten hinzugefügt werden, die man mit i, j, k bezeichnet.

Ein Quaternion wird in der Form $\underline{q} = a + bi + cj + dk$ dargestellt, wobei a, b, c, d reelle Zahlen sind. Die Menge K aller Quaternionen stellt einen vierdimensionalen Vektorraum mit der Basis $(1, i, j, k)$ dar. Man nennt a den *Realteil* und $bi + cj + dk$ den *Vektorteil* von $\underline{q}$.

B.2 Das Einheitsquaternion

Als *Einheitsquaternion* $\underline{q}_e$ bezeichnet man ein Quaternion $\underline{q} = (a, b, c, d)^T$ vom Betrag 1, also

$$a^2 + b^2 + c^2 + d^2 = 1 \; . \tag{B.1}$$

Einheitsquaternionen liegen auf einer vierdimensionalen Hyperkugel mit dem Radius 1 und dem Nullpunkt als Mittelpunkt. Für Einheitsquaternionen gilt

$$\underline{q}_e{}^{-1} = \underline{q}_e{}^{*} \; , \tag{B.2}$$

wobei der Stern für „konjugiert komplex“ steht. Einheitsquaternionen eignen sich sehr gut zur Beschreibung von Rotationen im dreidimensionalen Raum.

B.3 Beschreibung von Rotationen durch Quaternionen

Jedes Quaternion mit dem Betrag 1 kann in der Form

$$\underline{q}_e = \begin{pmatrix} \cos\phi \\ \sin\phi \cdot \vec{n} \end{pmatrix} \tag{B.3}$$

[1] wie bei den komplexen Zahlen

dargestellt werden, wobei der Vektor $\vec{n} = (n_1, n_2, n_3)^T$ den Betrag 1 haben soll. Dabei ist ϕ der Drehwinkel um die Achse $\vec{n}$ im dreidimensionalen Raum.

Die Rotationsmatrix für eine Rotation im dreidimensionalen Raum ergibt sich entsprechend der Gleichung

$$\mathbf{R} = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_2 q_1 - q_0 q_3) & 2(q_3 q_1 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{pmatrix} \tag{B.4}$$

aus einem Einheitsquaternion, wobei also die Nebenbedingung $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$ gelten muß.

Es sei darauf hingewiesen, daß ein Quaternion eine Rotation nicht eindeutig beschreibt. Ein Einheitsquaternion, dessen Komponenten alle negiert sind ($\underline{q}'_e = -q_0 - iq_1 - jq_2 - kq_3$) beschreibt dieselbe Rotation wie ($\underline{q}_e = q_0 + iq_1 + jq_2 + kq_3$).

Abschließend wird gezeigt, wie der Richtungsvektor der Drehachse $\vec{n}$ und der Drehwinkel ϕ um diese Achse aus einem Einheitsquaternion $\underline{q}_e = q_0 + iq_1 + jq_2 + kq_3$ direkt bestimmt werden kann,

$$\phi = 2 \arccos q_0, \quad \vec{n} = \frac{(q_1, q_2, q_3)^T}{|(q_1, q_2, q_3)^T|} \qquad \text{wenn} \quad q_0 \geq 0 \tag{B.5}$$

bzw.

$$\phi = 2(\pi - \arccos q_0), \quad \vec{n} = -\frac{(q_1, q_2, q_3)^T}{|(q_1, q_2, q_3)^T|} \qquad \text{wenn} \quad q_0 \leq 0\ . \tag{B.6}$$

In diesem Abschnitt wurde gezeigt, daß Einheitsquaternionen Rotationen im dreidimensionalen Raum repräsentieren können. Aufgrund ihrer Vorteile gegenüber Eulerwinkeln ist es sehr sinnvoll bei Schätzproblemen, in denen Rotationen auftreten, Quaternionen als Parametrisierung der Rotationsmatrix zu verwenden. Der Nachteil des Einsatzes von Quaternionen zur Parametrisierung von Rotationen ist allerdings, daß bei der Lösung von Schätzproblemen eine Nebenbedingung eingehalten werden muß.

Anhang C

Lagrange-Multiplikatoren

Gesucht wird das Extremum der skalaren Funktion $f(\vec{x}),\ \vec{x} \in \mathbb{R}^n$ unter den m Nebenbedingungen

$$g_1(\vec{x}) = 0,\ \ldots,\ g_m(\vec{x}) = 0. \tag{C.1}$$

Ein möglicher Lösungsweg läuft über die Bildung der *Lagrange-Funktion*

$$L(\vec{x}) = f(\vec{x}) + \mu_1 g_1(\vec{x}) + \ldots + \mu_m g_m(\vec{x}). \tag{C.2}$$

Dabei bezeichnet man die Größen $\mu_1 \ldots \mu_m$ als *Lagrange-Multiplikatoren*. Die n Gleichungen

$$\frac{\partial L}{\partial x_1} = 0,\ \ldots,\ \frac{\partial L}{\partial x_n} = 0 \tag{C.3}$$

stellen zusammen mit den m Randbedingungen ein System von $m+n$ Gleichungen für die $m+n$ Variablen $x_1, \ldots, x_n, \mu_1, \ldots, \mu_m$ dar. Die Lösung dieses Gleichungssystems liefert schließlich [Br95] diejenigen Parameter, die unter Berücksichtigung der Nebenbedingungen, die Funktion $f(\vec{x})$ minimiert bzw. maximiert.

Anhang D

Erzeugung von Zufallszahlen

Der Algorithmus in Tabelle D.1 kann zur Erzeugung beliebig verteilter, kontinuierlicher Zufallsvariablen x aus gleichverteilten Zufallszahlen r genutzt werden [Pa91]. Ein Beweis folgt im Anschluß.

1. Sei $p(x)$ die vorgegebene Wahrscheinlichkeitsdichte der Zufallszahlen x
2. Berechne die Verteilungsfunktion $\Phi(x)$ aus der Wahrscheinlichkeitsdichte $p(x)$ gemäss:
$$\Phi(x) = \int_{-\infty}^{x} p(x')dx' \tag{D.1}$$
3. Bilde die Umkehrung von $\Phi(x) =: r$
$$\Rightarrow x = \Phi^{-1}(r) \tag{D.2}$$
4. Erzeuge Zufallzahlen r, die in $0 \leq r < 1$ gleichverteilt sind, und berechne daraus $x = \Phi^{-1}(r)$; diese sind dann mit der Wahrscheinlichkeitsdichte p(x) verteilt

Tabelle D.1: Analytisches Verfahren zur Generierung von Zufallszahlen mit beliebiger Wahrscheinlichkeitsverteilung

Beweis 1 *Sei $F(x) = P\{\xi < x\}$ die Verteilungsfunktion der Zufallsgröße ξ. Es gilt $0 \leq F(x) \leq 1$. Da $F(x)$ außerdem stetig und streng monoton ist, existiert auch die Umkehrfunktion G zu F: $x = G(y)$, so daß für alle y mit $0 < y \leq 1$ gilt:*

$$G\left(F(x)\right) = x \quad \textit{und} \quad F\left(G(y)\right) = y. \tag{D.3}$$

Sei weiterhin eine im Intervall $[0, 1]$ gleichverteilte Zufallsgröße γ gegeben. Es gilt:

$$P\left\{G(\gamma) < x\right\} = P\left\{F\left(G(\gamma)\right) < F(x)\right\} = P\{\gamma < F(x)\} \tag{D.4}$$

Da γ im Intervall $[0,1]$ gleichverteilt ist, ist die Wahrscheinlichkeit

$$P\{\gamma < F(x)\} = P\{0 < \gamma < F(x)\} \tag{D.5}$$

gleich der Länge des Intervalls $[0, F(x)]$, d.h. gleich $F(x)$. Also gilt

$$P\{G(\gamma) < x\} = F(x) = P\{\xi < x\}. \tag{D.6}$$

Somit haben $G(\gamma)$ und ξ die gleiche Wahrscheinlichkeitsverteilung.

Anhang E

Kombination zweier Parameterschätzungen

E.1 Das Gauß-Markov-Theorem

Unter Verwendung des linearen Beobachtungsmodells

$$\vec{z} = \mathbf{H}\vec{x} + \vec{v} \tag{E.1}$$

mit

$$\mathsf{E}\left[\vec{v}\right] = \vec{0} \quad \text{und} \quad \mathbf{C} = \mathsf{Cov}\left[\vec{v}\right] = \mathsf{E}\left[\vec{v}\vec{v}^T\right] \text{ beliebig, aber positiv definit,}$$

ergibt sich die erwartungstreue Schätzung des Parametervektors $\vec{x}$ mit minimaler Varianz zu [SW94]

$$\hat{\vec{x}} = \left(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{C}^{-1}\vec{z}. \tag{E.2}$$

Der Schätzfehler $\vec{r} = \hat{\vec{x}} - \vec{x}$ weist dann eine Kovarianzmatrix von

$$\hat{\mathbf{C}} = \left(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H}\right)^{-1} \tag{E.3}$$

auf.

E.2 Parameterschätzung aus mehreren Beobachtungen

Sollen mehrere Beobachtungen zu einer Schätzung kombiniert werden, so kann der folgende Weg eingeschlagen werden. Ausgehend von den beiden Beobachtungsgleichungen,

$$\vec{z}_1 = \mathbf{H_1}\vec{x} + \vec{v}_1 \quad \text{und} \quad \vec{z}_2 = \mathbf{H_2}\vec{x} + \vec{v}_2\,, \tag{E.4}$$

mit

$$\mathsf{E}\left[\vec{v}_1\right] = \vec{0}, \quad \mathsf{E}\left[\vec{v}_2\right] = \vec{0}, \quad \mathsf{E}\left[\vec{v}_1\vec{v}_2^T\right] = \vec{0} \quad \text{und}$$
$$\mathbf{C_1} = \mathsf{E}\left[\vec{v}_1\vec{v}_1^T\right], \quad \mathbf{C_2} = \mathsf{E}\left[\vec{v}_2\vec{v}_2^T\right]$$

kann die erweiterte Beobachtungsgleichung

$$\vec{z}_e = \mathbf{H_e}\vec{x} + \vec{v}_e \tag{E.5}$$

mit

$$\vec{z}_e := \begin{pmatrix} \vec{z}_1 \\ \vec{z}_2 \end{pmatrix}, \quad \mathbf{H}_e := \begin{pmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \end{pmatrix} \quad \text{und} \quad \vec{v}_e := \begin{pmatrix} \vec{v}_1 \\ \vec{v}_2 \end{pmatrix}$$

aufgestellt werden. Da laut Vorraussetzung die Fehlervektoren $\vec{v}_1$ und $\vec{v}_2$ unkorreliert sind, gilt

$$\mathbf{C}_e = \begin{pmatrix} \mathbf{C}_1 & \vec{0} \\ \vec{0} & \mathbf{C}_2 \end{pmatrix}, \tag{E.6}$$

und somit kann auf Gleichung E.5 das Gauß-Markov-Theorem aus Abschnitt E.1 angewandt werden. Es ergibt sich als optimaler Schätzwert für $\vec{x}$

$$\hat{\vec{x}} = \left(\mathbf{H}_e^T \mathbf{C}_e^{-1} \mathbf{H}_e\right)^{-1} \mathbf{H}_e^T \mathbf{C}_e^{-1} \vec{z}_e\,, \tag{E.7}$$

mit der Kovarianzmatrix des Schätzfehlers von

$$\hat{\mathbf{C}} = \left(\mathbf{H}_e^T \mathbf{C}_e^{-1} \mathbf{H}_e\right)^{-1}. \tag{E.8}$$

Da

$$\mathbf{C}_e^{-1} = \begin{pmatrix} \mathbf{C}_1^{-1} & \vec{0} \\ \vec{0} & \mathbf{C}_2^{-1} \end{pmatrix} \tag{E.9}$$

gilt, kann $\mathbf{H}_e^T \mathbf{C}_e^{-1}$ berechnet werden zu

$$\mathbf{H}_e^T \mathbf{C}_e^{-1} = \begin{pmatrix} \mathbf{H}_1^T & \mathbf{H}_2^T \end{pmatrix} \begin{pmatrix} \mathbf{C}_1^{-1} & \vec{0} \\ \vec{0} & \mathbf{C}_2^{-1} \end{pmatrix} = \begin{pmatrix} \mathbf{H}_1^T \mathbf{C}_1^{-1} & \mathbf{H}_2^T \mathbf{C}_2^{-1} \end{pmatrix}. \tag{E.10}$$

Daraus ergibt sich dann für $\mathbf{H}_e^T \mathbf{C}_e^{-1} \mathbf{H}_e$,

$$\mathbf{H}_e^T \mathbf{C}_e^{-1} \mathbf{H}_e = \begin{pmatrix} \mathbf{H}_1^T \mathbf{C}_1^{-1} & \mathbf{H}_2^T \mathbf{C}_2^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \end{pmatrix} = \mathbf{H}_1^T \mathbf{C}_1^{-1} \mathbf{H}_1 + \mathbf{H}_2^T \mathbf{C}_2^{-1} \mathbf{H}_2\,. \tag{E.11}$$

Werden die Gleichungen E.10 und E.11 in Gleichung E.7 eingesetzt, dann resultiert der optimale Schätzwert für die Kombination zweier Beobachtungen zu

$$\hat{\vec{x}} = \left(\mathbf{H}_1^T \mathbf{C}_1^{-1} \mathbf{H}_1 + \mathbf{H}_2^T \mathbf{C}_2^{-1} \mathbf{H}_2\right)^{-1} \cdot \left(\mathbf{H}_1^T \mathbf{C}_1^{-1} \vec{z}_1 + \mathbf{H}_2^T \mathbf{C}_2^{-1} \vec{z}_2\right) \tag{E.12}$$

und dessen Kovarianzmatrix zu

$$\hat{\mathbf{C}} = \left(\mathbf{H}_1^T \mathbf{C}_1^{-1} \mathbf{H}_1 + \mathbf{H}_2^T \mathbf{C}_2^{-1} \mathbf{H}_2\right)^{-1}. \tag{E.13}$$

E.2.1 Sonderfall: die Beobachtungsmatrizen sind Einheitsmatrizen

Für den Sonderfall, daß die Beobachtungsmatrizen $\mathbf{H}_1$ und $\mathbf{H}_2$ Einheitsmatrizen sind, ergibt sich aus den Gleichungen E.12 und E.13

$$\hat{\vec{x}} = \left(\mathbf{C}_1^{-1} + \mathbf{C}_2^{-1}\right)^{-1} \cdot \left(\mathbf{C}_1^{-1} \vec{z}_1 + \mathbf{C}_2^{-1} \vec{z}_2\right) \tag{E.14}$$

und

$$\hat{\mathbf{C}} = \left(\mathbf{C}_1^{-1} + \mathbf{C}_2^{-1}\right)^{-1}. \tag{E.15}$$

Dieses Ergebnis gibt an, wie zwei unabhängige Schätzergebnisse zusammen mit ihren Kovarianzmatrizen in statistisch optimaler Weise[1] miteinander zu einer einzigen Schätzung kombiniert werden können.

[1] im Sinne von Least-Squares

Anhang F

Ermittlung von Vorwissen über die Bewegung eines Fahrzeugs

In diesem Abschnitt wird erläutert, wie ein Erwartungswert der Bewegungsparameter und dessen Kovarianzmatrix für die Bewegung eines Fahrzeugs auf einer Straße ermittelt werden kann. Diese Momente können aus einer Betrachtung des statischen und dynamischen Verhaltens eines Fahrzeuges abgeschätzt werden. Die Beschreibung der Bewegung erfolgt über den Parametervektor $\vec{p} = (p_1 \dots p_{10}) := (\epsilon_0, \epsilon_1, \epsilon_2, \epsilon_3, t_1, t_2, t_3, b_1, b_3, \lambda)$. Die ersten vier Komponenten von p beschreiben die Rotation der Kamera – und somit auch des Fahrzeugs – mit Hilfe eines Einheitsquaternions. Der Parameter λ resultiert aus der Nebenbedingung des Einheitsquaternions $\sum_{i=0}^{3} \epsilon_i^2 = 1$ und dessen Berücksichtigung in der Minimierung der Verlustfunktion 6.6 unter Verwendung von Lagrange-Multiplikatoren (siehe auch Anhang C). Die Komponenten (t_1, t_2, t_3) repräsentieren den Translationsvektor der Kamera und (b_1, b_3) die unabhängigen Komponenten des Normalenvektors der Ebene. Alle Parameter werden im folgenden im Fahrzeugkoordinatensystem angegeben. Im folgenden wird weiterhin angenommen, daß die Bildwiederholfrequenz 25 Hz beträgt, also die Zeitdauer, die zwischen der Aufnahme zweier Bilder verstreicht, 40 ms beträgt.

F.1 Erwartungswert der Bewegungsparameter

Über die Erwartungswerte der einzelnen Komponenten des Parametervektors können folgende Überlegungen angestellt werden:

1. Da die Wahrscheinlichkeiten für die Durchfahrt einer Rechts- oder Linkskurve gleich groß sind, kann angenommen werden, daß im Mittel keine Rotation des Fahrzeugs auftritt. Das bedeutet für das Einheitsquaternion $\mathbf{E}(\vec{\epsilon}) = (1, 0, 0, 0)$.

2. Aus dem gleichen Grund kann davon ausgegangen werden, daß im Mittel der Translationsvektor nur eine Komponente ungleich Null besitzt, nämlich die z-Komponente, welche in Fahrtrichtung zeigt. Wenn davon ausgegangen wird, daß ein PKW im Schnitt 50 km/h oder 13.89 m/s schnell fährt, so legt es in 40 ms eine Strecke von 0.56 m zurück. Daraus ergibt sich ein mittlerer Translationsvektor von $\mathbf{E}(\vec{t}) = (0, 0, -0.56\,\mathrm{m})$.

3. Da ein Fahrzeug im Schnitt parallel zur Fahrbahn steht, hat der Normalenvektor der Straße nur eine Komponente in y-Richtung. Per Konvention zeigt der Normalenvektor der

Straße nach oben, in Fahrzeugkoordinaten also in die negative y-Richtung. Es gilt somit $\mathbf{E}(\vec{n}) = (0, -1, 0)$. In Abschnitt 5.2 wurde beschrieben, daß nur fünf der sechs Komponenten des Translations- und Normalenvektors bestimmt werden können, so daß hier nur die Komponenten b_1 und b_3 als variabel angesehen werden.

4. Über den Lagrange-Parameter kann keine Angabe gemacht werden, so daß er als eine mittelwertfreie Zufallsgröße mit großer Unsicherheit angesehen werden kann, also $\mathbf{E}(\lambda) = 0$.

Zusammengefaßt ergibt sich also für den Erwartungswert des Parametervektors

$$\mathbf{E}(\vec{p}) = (1, 0, 0, 0, 0, 0, -0.56, 0, 0, 0)\,. \tag{F.1}$$

F.2 Kovarianzmatrix der Bewegungsparameter

Durch eine physikalische Analyse der folgenden Punkte kann eine Abschätzung der Kovarianzmatrix der Bewegungsparameter erfolgen:

1. Ein Fahrzeug neigt sich beim Beschleunigen und beim Bremsen. Die Extremsituation zur Abschätzung des maximalen *Nickwinkels* und des Normalenvektors der Ebene in Kamerakoordinaten ist die *Vollbremsung* auf trockenem Umtergrund. Eine Betrachtung von Gleitreibung und Massenträgheit führt hier zu den gewünschten Varianzen.

2. Wenn ein Fahrzeug eine Kurve durchfährt, dann giert es, d.h. es dreht sich um seine Hochachse. Unter Verwendung des minimalen Wendekreises eines Fahrzeugs und seiner Haftreibung auf trockener Straße, kann die maximale *Gierrate* zwischen zwei Bildern einer Videosequenz abgeschätzt werden. Außerdem kann bei dieser Betrachtung die maximale Änderung des Translationsvektors ermittelt werden.

3. Die Unebenheiten von Fahrbahnen führen ebenfalls zu starken Änderungen des Translationsvektors und des Normalenvektors der Ebene. Zur Abschätzung der maximalen Änderungen muß hier ausgehend von einem ca. 10cm-hohen Hindernis auf der Straße die Rückstellkraft der Stoßdämpfer und die daraus resultierende Nickbewegung des Fahrzeugs betrachtet werden.

4. Eine Fahrt durch eine enge Kurve hat ein Kippen der Hochachse des Fahrzeugs aufgrund seiner Trägheitskräfte zur Folge. Der entsprechende Winkel, dessen Varianz durch die physikalische Analyse dieser Situation, unter Verwendung der Rückstellkraft der Stoßdämpfer, der Spurbreite und der Massenträgheit, ermittelt werden kann, ist der sog. *Rollwinkel*.

An dieser Stelle sei nur das Endresultat der dynamischen Analyse angegeben. Für die Kovarianzmatrix des Einheitsquaternions in Kamerakoordinaten ergibt sich

$$\mathsf{Cov}\left[\vec{\epsilon}\right] = \begin{pmatrix} 4.865 \cdot 10^{-7} & 0 & 0 & 0 \\ 0 & 5.199 \cdot 10^{-4} & 0 & 0 \\ 0 & 0 & 5.199 \cdot 10^{-4} & 0 \\ 0 & 0 & 0 & 5.199 \cdot 10^{-4} \end{pmatrix}, \tag{F.2}$$

die Kovarianzmatrix des Translationsvektors lautet

$$\mathsf{Cov}\left[\vec{t}\right] = \begin{pmatrix} 0.000001 & 0.000019 & 0.000187 \\ 0.000019 & 0.001125 & 0.010695 \\ 0.000187 & 0.010695 & 0.101756 \end{pmatrix} \tag{F.3}$$

und die Kovarianzmatrix des Normalenvektors der Ebene ergibt sich zu

$$\mathsf{Cov}\left[\vec{t}\right] = \begin{pmatrix} 0.004067 & 0 & 0 \\ 0 & 0.000336 & 0.002002 \\ 0 & 0.002002 & 0.019172 \end{pmatrix} . \tag{F.4}$$

Zu dem Lagrangeparameter λ kann keine Abschätzung erfolgen, da er völlig unbestimmt ist. Somit gilt für seine Kovarianz, bzw. Varianz

$$\mathsf{Cov}\left[\lambda\right] = \infty . \tag{F.5}$$

Anhang G

Optimierungsergebnisse der Testfunktionen f_1 bis f_6

Die Tabellen G.1 bis G.6 zeigen die Optimierungsergebnisse der Funktionen I bis VI aus Abschnitt 9.6.2, wobei die Optimierungsmethoden *Simplex, Powell, Fast Annealing* und *Iterative Annealing* untersucht wurden. Die Auswertung der Ergebnisse befindet sich in Abschnitt 9.6.3.

Funktion I

Methode der Optimierung	glob. Minimum gefunden?	Funktionswert im Minimum	Anzahl der Schritte bis zum glob. Minimum	mittl. Anz. der Rechenschritte
Simplex		15.7		
”		87.2		
”		26.1		
”		87.2		
”		87.2		
”		87.2		
”		87.2		
”		87.2		
”		87.2		
”		87.2		
Powell		26.2		
”		26.2		
”		3.33		
”		3.32		
”	√	$5.3 * 10^{-6}$	160	
”	√	$5.2 * 10^{-7}$	162	
”		26.1		
”		26.1		
”		26.1		
”		26.2		
Fast	√	$7.1 * 10^{-5}$	1276	
Annealing	√	$5.2 * 10^{-6}$	1291	
”	√	$5.2 * 10^{-6}$	1719	
”	√	$4.2 * 10^{-5}$	1387	
”	√	$4.0 * 10^{-5}$	1316	} 1277
”	√	$4.0 * 10^{-5}$	1317	
”	√	$7.0 * 10^{-5}$	789	
”	√	$6.4 * 10^{-5}$	1140	
”	√	$6.3 * 10^{-5}$	1297	
”	√	$5.2 * 10^{-5}$	1240	
Iterative	√	$6.0 * 10^{-5}$	251	
Annealing	√	$7.5 * 10^{-5}$	252	
”	√	$7.0 * 10^{-5}$	252	
”	√	$9.3 * 10^{-5}$	710	
”	√	$9.4 * 10^{-5}$	90	} 386
”	√	$1.1 * 10^{-5}$	196	
”	√	$1.1 * 10^{-5}$	190	
”	√	$8.2 * 10^{-6}$	1494	
”	√	$5.0 * 10^{-5}$	288	
”	√	$4.9 * 10^{-5}$	143	

Tabelle G.1: Ergebnisse zur Bestimmung des globalen Minimums von Funktion f_1

Funktion II

Methode der Optimierung	glob. Minimum gefunden?	Funktionswert im Minimum	Anzahl der Schritte bis zum glob. Minimum	mittl. Anz. der Rechenschritte
Simplex	√	$< 10^{-5}$	773	
”	√	$< 10^{-5}$	676	
”	√	$< 10^{-5}$	675	
”	√	$< 10^{-5}$	676	
”	√	$< 10^{-5}$	734	} 668
”	√	$< 10^{-5}$	734	
”	√	$< 10^{-5}$	734	
”	√	$< 10^{-5}$	487	
”	√	$< 10^{-5}$	478	
”	√	$< 10^{-5}$	720	
Powell	√	$< 10^{-5}$	7163	
”		8.07		
”	√	$< 10^{-5}$	7975	
”	√	$< 10^{-5}$	9320	
”	√	$< 10^{-5}$	6920	} 7893
”	√	$< 10^{-5}$	7802	
”	√	$< 10^{-5}$	7199	
”	√	$< 10^{-5}$	7567	
”	√	$< 10^{-5}$	9199	
”		0.12		
Fast	√	$< 10^{-5}$	13387	
Annealing	√	$< 10^{-5}$	12798	
”	√	$< 10^{-5}$	12341	
”	√	$< 10^{-5}$	13469	
”	√	$< 10^{-5}$	9941	} 13307
”	√	$< 10^{-5}$	13106	
”	√	$< 10^{-5}$	13074	
”	√	$< 10^{-5}$	14703	
”	√	$< 10^{-5}$	17216	
”	√	$< 10^{-5}$	13047	
Iterative	√	$< 10^{-5}$	2921	
Annealing	√	$< 10^{-5}$	2316	
”	√	$< 10^{-5}$	26188	
”	√	$< 10^{-5}$	1735	
”	√	$< 10^{-5}$	8712	} 8200
”	√	$< 10^{-5}$	11233	
”	√	$< 10^{-5}$	19015	
”	√	$< 10^{-5}$	3260	
”	√	$< 10^{-5}$	3472	
”	√	$< 10^{-5}$	3167	

Tabelle G.2: Ergebnisse zur Bestimmung des globalen Minimums von Funktion f_2

Funktion III

Methode der Optimierung	glob. Minimum gefunden?	Funktionswert im Minimum	Anzahl der Schritte bis zum glob. Minimum	mittl. Anz. der Rechenschritte
Simplex		1384		
”		1384		
”		1384		
”		1384		
”	√	$< 10^{-5}$	399	
”		1384		
”		1384		
”		1384		
”		1384		
”		1384		
Powell	√	$< 10^{-5}$	281	
”	√	$< 10^{-5}$	475	
”	√	$< 10^{-5}$	395	
”	√	$< 10^{-5}$	476	
”	√	$< 10^{-5}$	384	} 390
”	√	$< 10^{-5}$	475	
”	√	$< 10^{-5}$	375	
”	√	$< 10^{-5}$	279	
”	√	$< 10^{-5}$	374	
”	√	$< 10^{-5}$	390	
Fast	√	$< 10^{-5}$	13704	
Annealing	√	$< 10^{-5}$	11815	
”	√	$< 10^{-5}$	13385	
”	√	$< 10^{-5}$	13416	
”	√	$< 10^{-5}$	16143	} 14209
”	√	$< 10^{-5}$	14986	
”	√	$< 10^{-5}$	14086	
”	√	$< 10^{-5}$	15598	
”	√	$< 10^{-5}$	14877	
”	√	$< 10^{-5}$	14085	
Iterative	√	$< 10^{-5}$	189	
Annealing	√	$< 10^{-5}$	179	
”	√	$< 10^{-5}$	223	
”	√	$< 10^{-5}$	215	
”	√	$< 10^{-5}$	225	} 205
”	√	$< 10^{-5}$	222	
”	√	$< 10^{-5}$	202	
”	√	$< 10^{-5}$	195	
”	√	$< 10^{-5}$	218	
”	√	$< 10^{-5}$	189	

Tabelle G.3: Ergebnisse zur Bestimmung des globalen Minimums von Funktion f_3

Funktion IV

Methode der Optimierung	glob. Minimum gefunden?	Funktionswert im Minimum	Anzahl der Schritte bis zum glob. Minimum	mittl. Anz. der Rechenschritte
Simplex		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
Powell		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
”		48.98		
Fast	$\surd$	$< 10^{-5}$	17520	
Annealing	$\surd$	$< 10^{-5}$	15963	
”	$\surd$	$< 10^{-5}$	14142	
”	$\surd$	$< 10^{-5}$	14757	
”	$\surd$	$< 10^{-5}$	16964	} 16096
”	$\surd$	$< 10^{-5}$	15856	
”	$\surd$	$< 10^{-5}$	16436	
”	$\surd$	$< 10^{-5}$	14593	
”	$\surd$	$< 10^{-5}$	17926	
”	$\surd$	$< 10^{-5}$	16812	
Iterative	$\surd$	$< 10^{-5}$	3216	
Annealing	$\surd$	$< 10^{-5}$	4733	
”	$\surd$	$< 10^{-5}$	5871	
”	$\surd$	$< 10^{-5}$	4365	
”	$\surd$	$< 10^{-5}$	2602	} 4022
”	$\surd$	$< 10^{-5}$	4455	
”	$\surd$	$< 10^{-5}$	2870	
”	$\surd$	$< 10^{-5}$	5291	
”	$\surd$	$< 10^{-5}$	4745	
”	$\surd$	$< 10^{-5}$	2073	

Tabelle G.4: Ergebnisse zur Bestimmung des globalen Minimums von Funktion f_4

Funktion V

Methode der Optimierung	glob. Minimum gefunden?	Funktionswert im Minimum	Anzahl der Schritte bis zum glob. Minimum	mittl. Anz. der Rechenschritte
Simplex		0.56		
”		0.56		
”		0.56		
”	√	$1.5 * 10^{-3}$	352	
”		0.56		
”	√	$1.5 * 10^{-3}$	773	
”		0.56		
”		0.56		
”		0.56		
”	√	$1.5 * 10^{-3}$	310	
Powell		0.56		
”		0.29		
”		0.26		
”		0.26		
”		0.31		
”		0.56		
”		0.26		
”		0.40		
”		0.26		
”		0.26		
Fast		0.56		
Annealing		0.56		
”		0.56		
”		0.56		
”		0.40		
”		0.56		
”		0.40		
”		0.40		
”		0.56		
”		0.40		
Iterative	√	$1.5 * 10^{-3}$	37593	
Annealing	√	$1.5 * 10^{-3}$	31599	
”	√	$1.5 * 10^{-3}$	40736	
”		0.269		} 24997
”	√	$1.5 * 10^{-3}$	67459	
”	√	$1.5 * 10^{-3}$	1641	
”	√	$1.5 * 10^{-3}$	12578	
”	√	$1.5 * 10^{-3}$	7718	
”	√	$1.5 * 10^{-3}$	659	
”		0.28		

Tabelle G.5: Ergebnisse zur Bestimmung des globalen Minimums von Funktion f_5

Funktion VI

Methode der Optimierung	glob. Minimum gefunden?	Funktionswert im Minimum	Anzahl der Schritte bis zum glob. Minimum	mittl. Anz. der Rechenschritte
Simplex		0.24		
”		0.0056		
”		0.0056		
”		0.18		
”		0.24		
”		0.24		
”	√	$< 10^{-4}$	2095	
”		0.0056		
”		0.024		
”		0.0056		
Powell		9.86		
”		0.005		
”		9.86		
”		0.24		
”		0.83		
”		5.45		
”		0.30		
”		0.24		
”		0.24		
”		0.005		
Fast		0.30		
Annealing		0.014		
”		0.014		
”		0.013		
”		0.014		
”		0.0056		
”		0.0056		
”		0.013		
”		0.0056		
”		0.0056		
Iterative		$1.3 * 10^{-3}$		
Annealing		0.0056		
”	√	$< 10^{-4}$	178000	
”	√	$< 10^{-4}$	17423	
”	√	$< 10^{-4}$	557000	} 253737
”		$2.3 * 10^{-3}$		
”	√	$< 10^{-4}$	251000	
”	√	$< 10^{-4}$	113000	
”	√	$< 10^{-4}$	406000	
”		$3.8 * 10^{-4}$		

Tabelle G.6: Ergebnisse zur Bestimmung des globalen Minimums von Funktion f_6

Anhang H

Optimierungsergebnisse der Markantheitsberechnung mittels CNN

In Tabelle H.1 sind die Optimierungsergebnisse der Markantheitsberechnung aus Kapitel 10 unter Verwendung eines Zellularen Neuronalen Netzwerks dargestellt. Ausschließlich das Iterative Annealing Verfahren findet zuverlässig einen guten CNN-Parametersatz für das gestellte Problem.

Methode der Optimierung	glob. Minimum gefunden?	Funktionswert im Minimum	Anzahl der Schritte bis zum glob. Minimum
Simplex		0.013	
”		0.013	
”		0.013	
”		0.013	
”		0.013	
”		0.013	
”		0.013	
”		0.013	
”		0.013	
”		0.013	
Powell		0.013	
”		0.013	
”		0.013	
”		0.013	
”		0.013	
”	√	$1.1 * 10^{-5}$	7650
”		0.013	
”		0.013	
”		0.013	
”		0.013	
Fast Annealing		0.0048	
		0.0048	
”		0.011	
”		0.0048	
”		0.012	
”		0.013	
”		0.0048	
”		0.012	
”		0.0048	
”		0.0048	
Iterative Annealing	√	$1.6 * 10^{-4}$	49796
		0.0048	
”	√	$7.1 * 10^{-5}$	17416
”	√	$2.5 * 10^{-4}$	49799
”		0.0047	
”	√	$8.3 * 10^{-5}$	49792
”		0.0029	
”	√	$1.6 * 10^{-4}$	49777
”	√	$2.3 * 10^{-4}$	14799
”	√	$5.1 * 10^{-5}$	49912

Tabelle H.1: Optimierungsergebnisse der Markantheitsberechnung mittels CNN

Anhang I

CNN-Parameter zur direkten Hinderniserkennung

Für die Hinderniserkennung in synthetisch generierten Bildsequenzen, wie sie beispielsweise in Bild 12.1 dargestellt ist, wurde das CNN-Gen in Tabelle I.1 ermittelt, welches die Parameter eines CNN gemäß (8.3) mit der Nachbarschaft 2 und eine polynomiale Zellkopplung vom Grade 3 darstellt.

Um eine Hinderniserkennung in realen Verkehrsszenen, wie sie beispielsweise in Bild 12.9 dargestellt ist, zu realisieren, wurde das CNN-Gen in Tabelle I.2 ermittelt, welches – wie im synthetischen Fall – die Parameter eines CNN gemäß (8.3) mit der Nachbarschaft 2 und eine polynomiale Zellkopplung vom Grade 3 darstellt.

Feedback-	1.16832332375	-5.56019524633	-11.1319370834	-4.07095669872
Template:	13.3020772112	18.8209874767	-4.0060047348	-1.76288677012
-8.16775566914	-1.03145297196	-2.23022646953	5.36234401405	4.99736085602
4.9125061886	3.08071884651	2.4237054196	-8.60627367629	-0.738833529227
-10.3171823969	4.94326498831	10.0624127299	4.98229509659	0.978300557096
2.31615769945	-2.81831837523	-6.22720744157	-5.95071587638	3.10504333522
1.17700735566	17.7610157503	0.365959591026	0.852652299339	0.830591486775
-2.97612988003	-7.02546387614	-0.229319338222	1.85278363784	3.34992600866
-4.15095914971	-1.96797113679	-1.86896068959	-6.07060950735	2.55664953361
-2.76011696072	-1.68939686366	-6.21190688967	1.46962684042	-3.77151138368
9.05113731582	-5.21279352014	-1.82096902666	-3.65628018064	-7.86167043667
9.67247358372	0.916777241186	-6.90876077902	-10.7470829304	1.28953735282
-9.24421884748	-3.19331975776	6.63940677261	-3.6649409708	-7.47525014693
-4.39190426109	-0.153339208654		4.31718530597	-14.4525749546
-10.2885859885	2.04087911523	Feedforward-	0.380292240725	-6.9811798987
-9.31014965709	-1.60640923172	Template:	-2.10235235304	1.48242223012
-2.80442931504	7.73347354266	8.61250875799	-3.0964255297	-1.00247604563
-2.35948645283	-6.57109642369	-0.298504347829	4.84533867579	3.61549446412
1.98609432518	8.93370793935	-3.52163360381	6.35977525942	-3.05800547888
-0.764854994011	10.7947228834	-4.02953230199	-1.17842284162	1.62196643089
8.96342485173	-4.60171936688	-0.686643526094	2.70472389133	-0.57055750703
1.60855179261	6.30555866642	-1.66338442011	6.56262686815	2.18072081769
-1.57111038044	8.21673238892	5.78680535614	-0.363497756746	0.427033881648
4.1666017435	-8.37150144776	-6.66844272552	-3.59867571855	0.0138795154254
-7.4073850849	16.0249749628	-10.1715970863	-1.72569823643	6.47034743433
4.91400198212	1.23744038184	4.49335460678	3.40568701997	7.29495709398
0.034058153054	-0.967476335693	-5.49487779921	4.07955938902	4.33638216983
-5.32540110038	-12.1864335341	-1.08213072334	0.146460739685	
5.33492639784	-2.31248999502	-1.1906383795	9.33438959595	Bias:
0.833156207849	-1.02209426324	-10.4514978317	-3.03920412039	-4.021
-4.81385106256	-7.54269621465	-6.72080130275	10.5679229853	
-3.31804121507	-5.97268688038	3.17336953141	10.5652771304	

Tabelle I.1: CNN-Parameter für die Hinderniserkennung in synthetisch generierten Bildsequenzen

Feedback-	-0.0508725036718	-0.0413559002235	-0.0622679189352	-0.0381951392829
Template:	0.0103516061208	0.0145637503262	-0.0413348160901	0.058599273352
-0.0328567569418	-0.125335040283	0.0716494477832	-0.0300708042195	0.0283834428174
0.013232955599	-0.066125294714	-0.0458596843964	0.0360286846382	-0.0593109821612
-0.0435094934456	-0.0754407738674	-0.0422690103742	0.0262430018784	0.0284342800969
-0.0407652747966	0.0280077949304	0.0131931689444	-0.0853009763112	0.0480873736698
0.00689391510538	-0.253307286084	0.0567430570958	-0.0188138351154	0.0449467356777
-0.0899772985451	-0.0873346758049	0.0296501622079	0.0205006007226	0.0664305113835
0.0545608454193	-0.155333779446	0.00257277784887	0.0198925802412	-0.0546944393933
-0.0138203366697	-0.0922862821089	0.0331622722413	0.00981368716677	-0.0213033095885
0.0219361783872	0.0422364046126	0.0484263813652	-0.0603653762166	-0.0421264883539
0.0330001870324	-0.0162044635087	-0.0422974358171	-0.00888137474716	0.0248121349536
-0.0933919921883	-0.0554674506517	0.0555734710226	-0.0464090165216	0.0348567164989
0.0532408215163	-0.140921664876		0.00771667923734	-0.0274637373389
0.0408126425334	-0.0116426574913	Feedforward-	-0.0157470292526	-0.0110779024738
-0.0465531086561	-0.0245025792383	Template:	-0.0158146360688	-0.0132713523942
-0.0291548702506	-0.0684443854416	-0.0427997654385	0.0846736354147	0.104251580031
-0.0325937658971	-0.0434626780077	-0.0525735077043	-0.0482477573206	-0.0139073579438
0.0539566994037	-0.100709820255	0.0329093121117	-0.106047661316	0.0660948233749
-0.0110653555589	-0.0199535040019	-0.0423526870894	0.0279763434881	0.00324064537925
-0.105181943545	0.0292649920194	0.0165892747507	-0.0277757368693	-0.0956404603637
0.0091160823287	-0.079162229687	0.0460776825066	0.0105905637554	-0.0348837514869
0.038546375252	-0.0284783077832	0.0351504115925	0.023043119823	0.0467763793781
-0.0658374920755	-0.00377649919809	-0.107164593308	0.0646442741899	0.021172744547
0.0458492461168	0.0325167313048	-0.0292333761512	0.00477101355861	0.0534876907368
-0.061942219777	0.00391521257779	-0.0166379663142	-0.0609075116051	-0.0409123355191
0.0649869531869	0.145970801497	0.0736592996568	0.031765552505	-0.0509647740242
-0.0884350820813	0.0324281698761	0.0117129937176	0.0850787069434	
-0.0242065799122	0.0743871765762	0.0178865384315	-0.0527599886956	Bias:
0.0193599692934	-0.0364344069463	-0.00517475741859	0.0264408435265	0.0892567
0.0235123438124	-0.0522412500766	-0.040106098395	0.0236991320562	
-0.0965744571443	-0.0796877457481	-0.0107376013213	-0.0700376501419	

Tabelle I.2: CNN-Parameter für die Hinderniserkennung in realen Verkehrssequenzen

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[AAD94] Adorni, G., D'Andrea, V., Destri, G.: *A Massively Approach to Cellular Neural Networks Image Processing*, Proceedings of the CNNA '94, pp. 423-428, Rome, Italy, 1994.

[AADM96] Adorni, G., D'Andrea, V., Destri, G., Mordinini, M.: *Shape Searching in Real World Images: A CNN-based Approach*, Proceedings of the CNNA '96, pp. 213-218, Seville, Spain, 1996.

[BA98] Benning, W., Aussems, Th.: *Mobile mapping by a car driven survey system (CDSS)*, in KAHMEN (ed.), Symposium on Geodesy for Geotechnical and Structural Engineering, Eisenstadt (A), 1998.

[BR95] Bigildeyeva, T.B., Rolshikov, V.Y.: *Numerical methods for optimizing discontinuous functions.* J. Comput. Syst. Sci. Int., 33, Nr.5, S. 49-56, 1995.

[Bi77] Biermann, G. J.: *Factorization Methods for Discrete Sequential Estimation*, Academic Press, New York, 1977.

[Br95] Bronstein, I. N., Semendjajew, K. A.: *Taschenbuch der Mathematik*, Verlag Harri Deutsch, Thun, Frankfurt a. M., 1995.

[CC95] Crounse, K., Chua, L. O.: *Methods for Image Processing and Pattern Formation in Cellular Neural Networks: A Tutorial*, IEEE Transactions on Circuits and Systems, Vol. 42, No. 10, pp. 583-601, 1995.

[CC96] Crounse, K., Chua, L. O.: *The CNN Universal Machine is as Universal as a Turing Machine*, IEEE Transactions on Circuits and Systems, Vol. 43, No. 4, pp. 352-355, 1996.

[Ch98] Chua, L. O.: *Cellular Neural Networks: A Paradigm for Complexity,* World Scientific, Series A, Vol. 31, Singapore, New Jersey, London, Hong Kong, 1997.

[CR97] Chua, L. O., Roska, T.: *Cellular Neural Networks - Foundations and Primer,* Lecture Notes/EE129, University of California, Berkley, 1997.

[CY88] Chua, L. O., Yang, L.: *Cellular Neural Networks: Theory and Applications*, IEEE Trans. on Neural Networks, Vol. 35, No. 10, p. 1257-1272, 1988.

[CE90] Carlsson, S., Eklundh, J.-O.: *Object detection using model based prediction and motion parallax*, in: Lecture Notes in Computer Science 427, ECCV'90, Springer-Verlag, pp. 297-306, 1990.

[cnnlib] Roska, T., Kék, L.: *CSL: CNN Software Library*, Version 7.3, Budapest, 1999.

[DV96] Dellaert, F., Vanderwalle, J.: *Automatic Design of Cellular Neural Networks by means of Genetic Algorithms: Finding a Feature Detector*, 6th Int. Workshop on CNNs and their Applications Proceedings, Seville, Spain, 1996.

[Fe01] Feiden, D., Tetzlaff, R.: *Iterative Annealing: A New Efficient Optimization Method for Cellular Neural Networks*, in: Konferenzband zur International Conference on Image Processing (ICIP) 2001, Thessaloniki, 2001.

[Fe01b] Feiden, D., Tetzlaff, R.: *Feature Extraction in Motion Estimation with Cellular Neural Networks using Iterative Annealing*, Konferenzband zur ECCTD 2001 (European Conference on Circuit Theory and Design), Helsinki, 2001

[Fe99] Feiden, D., Mühlich, M., Mester, R.: *Robuste Bewegungsschätzung aus monokularen Bildsequenzen von planaren Welten*, in: Förstner W.: Mustererkennung 99, 21. DAGM-Symposium, Bonn, 1999.

[Foe91] Förstner, W.: *Statistische Verfahren für die automatische Bildanalyse und ihre Bewertung bei der Objekterkennung und -vermessung*, Habilitationsschrift, Verlag der Bayerischen Akademie der Wissenschaften, München, 1991.

[Ge98] Geis, G.: *Untersuchungen der Auswirkungen von Parameterabweichungen auf das Verhalten von Zellularen Neuronalen Netzwerken,* Diplomarbeit, Institut für Angewandte Physik, Frankfurt am Main, 1998

[Ge84] Geman, S., und Geman, D.: *Stochastic relaxation, Gibbs distribution and the Bayesian restoration in images*, IEEE Trans. Patt. Anal. Mac. Int. 6(6), 721-741, 1984.

[Go94] Gonzalez, R.: *Digital Image Processing*, Addison-Wesley, 1994.

[Ha94] Haralick, H.: *Propagating Covariances in Computer Vision*, Seattle, 1994.

[HS93] R. M. Haralick, L. G. Shapiro: *Computer and Robot Vision, Volume II*, Addison-Wesley, 2. Aufl., 1993.

[Hu81] Huber, P.: *Robust Statistics*, Wiley, New York, 1981.

[In92] Ingber, L.: *Genetic Algorithms and Very Fast Simulated Reannealing: A Comparison*, Math. and Comp. Mod. 16(11), 87-100 (1992).

[In93] Ingber, L.: *Simulated Annealing: Practice Versus Theory*, Math. and Comp. Mod. 18(11), 29-57, 1993.

[K87] Koch, K. R. *Parameterschätzung und Hypothesentests*, Dümmler Verlag, Bonn, 1987.

[KLT00] Kunz, R., Loncar, A., Tetzlaff, R.: *SCNN 2000, Part I and II*, 6th Int. Workshop on CNNs and their Applications Proceedings, p. 123ff, 2000.

[KTW96] Kunz, R., Tetzlaff, R., Wolf, D.: *SCNN: A Universal Simulator for Cellular Neural Networks*, 6th Int. Workshop on CNNs and their Applications Proceedings, Seville, Spain, pp. 255-259, 1996.

[Ku96] Kunz, R.: *Simulation Zellularer Neuronaler Netzwerke,* Diplomarbeit, Institut für Angewandte Physik, Frankfurt am Main, 1996

[Li99] LIÑÁN, G. ET AL.: *A 0.5 0.5μm CMOS 10^6 Transistors Analog Programmable Array Processor for Real-Time Image Processing*: Proc. of the 25th European Solid-State Circuits Conference, pp. 358-36, Duisburg, Germany, Sept., 1999.

[Lo99] LONCAR, A.: *Kennlinienverfahren zur Modellierung nichtlinearer Systeme mit Zellularen Neuronalen Netzwerken,* Diplomarbeit, Institut für Angewandte Physik, Frankfurt am Main, 1999.

[LP98] LANG, C. B., PUCKER, N.: *Mathematische Methoden in der Physik*, Spektrum Akademischer Verlag, 1998.

[LPKH01] LAIHO, M., PAASIO, A., KANANEN, A., HALONEN, K.: *Discrete Time Analog Polynomial Type CNN with Digital State*, Proceedings of the IEEE International Symposium on Circuits and Systems, S. III-497-500, Sydney, 2001 .

[Me95] MENDEL, J.: *Lessons in Estimation Theory for Signal Processing, Communications and Control*, Prentice Hall, Englewood Cliffs, 1995.

[MH95] MESTER, R.; HÖTTER, M.: *Zuverlässigkeit und Effizienz von Verfahren zur Verschiebungsvektorschätzung*, In: Sagerer, G. et al. (eds.): *Mustererkennung 1995*, S. 285 – 294, Springer, 1995.

[MH95b] MESTER, R.; HÖTTER, M.: *Robust displacement vector estimation including a statistical error analysis*, Proc. 5th International Conference on Image Processing and its Applications, Edinburgh, UK, 1995.

[MGH81] MORE, J. J. , GARBOW, B. S., HILLSTROM, K. E.: *Algorithm 566 - Testing unconstrained optimization software*, ACM Transsactions on Mathematical Software, Volume 7, S. 17-41, 1981.

[Pa91] PAPOULIS, A.: *Probability, Random Variables and Stochastic Processes*, 3rd Ed., McGraw-Hill, 1991.

[Po95] PONTRJAGIN, L. S.: *Verallgemeinerungen der Zahlen*, Verlag Harri Deutsch, Frankfurt am Main, 1995.

[PTVF92] PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T. AND FLANNERY, B. P.: *Numerical Recipes in C*, Cambridge University Press, Cambridge, UK, 1992.

[Ra94] RADVANYI, A. G.: *Solution of Stereo Correspondence in Real Scene: An Analogic CNN Algorithm*, IEEE Proceedings of the CNNA '94, Italy, Spain, pp. 231 - 236, 1994.

[Ro93] ROSKA, T., CHUA, L. O.: *The CNN Universal Machine*, IEEE Trans. on Circ. and Syst., Vol. 40, pp. 163-173, 1993.

[Ro94] ROSKA, T.: *Analogic Algorithms Running on the CNN Universal Machine*, IEEE Proceedings of the CNNA '94, Rome, Italy, 1994.

[Ro95] ROSKA, T., CHUA, L. O., WOLF, D., KOZEK, T., TETZLAFF, R., PUFFER, F. *Simulating Nonlinear Waves and Partial Differential Equations via CNN - Part I: Basic Techniques*, IEEE Trans. on Circuits and Systems, Vol. 42, No. 10, pp. 807-815, 1995.

[RC95] Roska, T., Chua, L. O. *On a Framework of Complexity of Computations on Flows -Implemented on the CNN Universal Machine*, DNS-15-1995, Technical Report, Analogical and Neural Computing Systems Laboratory of the Comp. and Aut. Inst., Hung. Acad. of Sci., Budapest, 1995.

[Ro01] Rottensteiner, F. *Semi-automatic extraction of buildings based on hybrid adjustment using 3D surface models and management of building data in a TIS*, Dissertationsschrift an der TU Wien, 2001.

[RUS94] Rekeczky, C., Ushida, A., Roska, T.: *Analogic CNN Algorithm Exploring the Shortest Path: Possible Applications in Routing Problems*, Technical Report, Institute of Electronics, Information and Communication Engineers NLP '94, Vol. 94, No. 259, pp. 61-68, Tokyo, 1994.

[RRO96] Rekeczky, C., Roska, T., Ushida, A.: *CNN Based Self-Adjusting Nonlinear Filters*, Proceedings of the CNNA '96, pp. 369-373, Seville, Spain, 1996.

[SFT02] Schönmeyer, R., Feiden, D., Tetzlaff, R.: *Multi-Template Training for Image Processing with Cellular Neural Networks*, Konferenzband zur CNNA 2002 (7th IEEE International Workshop on Cellular Neural Networks and their Applications), Frankfurt, 2002.

[Sch02] Schönmeyer, R.: *Analyse und Optimierung von schaltungstechnischen Realisierungen Zellularer Neuronaler Netze zur nichtlinearen Informationsverarbeitung*, Diplomarbeit, Institut für Angewandte Physik, Frankfurt am Main, 2002.

[Si98] Sindhuber, A.: *Ergänzung und Fortführung eines digitalen Landschaftsmodelles mit multispektralen und hochauflösenden Fernerkundungsaufnahmen*, Ricks Verlag, Wien, 1998.

[SIV96] Sivia, D. S.: *Data Analysis*, Clarendon Press, Oxford, 1996.

[Sl90] Slot, K.: *Determination of Cellular Neural Networks Parameters for Feature Detection of Two-Dimensional Images*, Proceedings of the CNNA '90, Budapest, Hungary, 1996.

[SH87] Szu, H. und Hartley, R.: *Fast Simulated Annealing*, Phys. Lett. A, 122(3-4), 157-162, 1987.

[SW94] Stark, H; Woods, J.W.: *Probability, Random Processes and Estimation Theory for Engineers*, 2nd edition, Prentice Hall, 1994.

[SC98] Szirányi, T., Csapodi, M.: *Texture Classification and Segmentation by Cellular Neural Network using Genetic Learning*, Computer Vision and Image Understanding, Vol. 71, No. 3, pp. 255-270, 1998

[Te95] Tekalp, A. M.: *Digital Video Processing*, Prentice Hall, 1995.

[Tr99] Trautwein, S.: *Gemeinsame Erfassung von Bewegungsvorgängen aus drei Ansichten einer Szene*, Diplomarbeit, Institut für Angewandte Physik, Universität Frankfurt, 1999.

[THZ82] Tsai, R., Huang, T. und Zhu, W.: *Estimating three-dimensional motion parameters of a rigid planar patch, II: Singular value decomposition*, IEEE Transactions on Acoustics, Speech and Signal Processing, ASSP-30(4): 525 – 534, 1982.

[TZ96] Taraglio, S., Zanela, A.: *Cellular Neural Networks for the Stereo Matching Problem*, IEEE Proceedings of the CNNA '96, pp. 93-98, Seville, Spain, 1997.

[VA87] Van Laarhoven, P. J. M., Aarts, E. H. L.: *Simulated annealing: theory and applications*, Reidel, 1997.

[WHA93] J. Weng, T. S. Huang und N. Ahuja: *Motion and Structure from Image Sequences*, Bd. 29 von *Series in Information Science*, Springer Verlag, Berlin, Heidelberg, New York, 1993.

[YC90] G.S.J. Young, R. Chellappa: *3-D Motion Estimation Using a Sequence of Noisy Stereo Images: Models, Estimation and Uniqueness Results* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No. 8, p. 735-759, 1990.

[Zh96] Zengyou Zhang: *Parameter Estimation Techniques: A Tutorial with Application to Conic fitting*, Inria-Research-Report Nr. 2676, 1996.

Stichwortverzeichnis