

Local Features and Shape for Recognition

Roland Angst

2006

Eidgenössische Technische Hochschule Zürich
Departement Informatik
Pattern Analysis and Machine Learning Group
Prof. Joachim M. Buhmann

Betreuung:
Björn Ommer

Inhaltsangabe

Kapitel 1: Einleitung.....	5
1.1 Problembeschreibung.....	5
1.2 Kurzangabe der Semesterarbeit.....	5
1.3 Organisation der Semesterarbeit.....	7
Kapitel 2: Objekt Klassifizierung.....	8
2.1 Shape Matching.....	8
2.1.1 Mathematische Beschreibung.....	8
2.1.2 Unzulänglichkeiten der mathematischen Beschreibung.....	9
2.1.3 Korrespondenzen und das Huhn-Ei Problem.....	9
2.2 Compositional Graphical Models.....	10
Kapitel 3: Feature Deskriptoren.....	12
3.1 Motivation.....	12
3.2 Feature Deskriptor Beispiele.....	12
3.2.1 Geometric Blur.....	12
3.2.2 Localized Feature Histograms.....	13
3.2.3 Composed Localized Feature Histograms.....	14
3.2.4 Low Dimensional Localized Feature Histograms.....	15
3.2.5 Low Dimensional Composed Localized Feature Histograms.....	15
3.3 Vergleich von Feature Deskriptoren.....	15
3.3.1 Parameter des Geometric Blur.....	16
3.3.2 Parameter der Feature Histogramme.....	19
Kapitel 4: Korrespondenzen.....	22
4.1 Distanz zwischen Feature Deskriptoren.....	22
4.2 Verschiedene Arten von Korrespondenzen.....	22
4.2.1 Hardassignments.....	22
4.2.2 Softassignments.....	23
4.3 k nächste Nachbarn.....	23
Kapitel 5: Affine Transformation.....	25
5.1 Mathematische Einführung in affine Transformationen.....	25
5.2 Schätzung der affinen Transformation.....	27
Kapitel 6: Schätzung der Thin Plate Spline Transformation.....	30
6.1 Wahrscheinlichkeit und Regularisationstheorie.....	30
6.1.1 Modell.....	30
6.1.2 Reproducing Kernel Hilbert Spaces.....	30
6.1.3 Vom Unendlichen zum Endlichen: Die Kernel Property.....	31
6.1.4 Thin Plate Spline Basis Funktionen.....	32
6.2 Thin Plate Spline.....	34
6.2.1 Interpolation von eindimensionaler Daten.....	34
6.2.2 Interpolation von zweidimensionalen Daten.....	34
6.2.3 Regularisierte Thin Plate Spline.....	35
6.2.4 Gewichtete Thin Plate Spline.....	36
6.3 Thin Plate Splines im Vergleich mit kubischen B-Splines.....	38
6.4 Thin Plate Splines in Matlab.....	38
6.5 Neue Thin Plate Spline Implementierung.....	39

6.6 Angetroffene Probleme.....	40
6.7 Principal Warps.....	41
Kapitel 7: Objekt Klassifizierungsverfahren.....	44
7.1 Überblick.....	44
7.2 Interest Point Detection.....	46
7.2.1 Benutzer-definiert.....	46
7.2.2 Harris Interest Point Detector.....	46
7.2.3 Affine Interest Point Detector.....	46
7.2.4 Uniformes Gitter.....	46
7.3 Bestimmen der Feature Deskriptoren.....	46
7.4 Gewichtung von Korrespondenzpartnern.....	47
7.5 Schätzung der affinen Transformation.....	48
7.6 Iterationsphase.....	48
7.6.1 Dreiteilige Distanzfunktion zwischen zwei Bildpunkten.....	48
7.6.1 Varianten der Anzahl und Gewichtung berücksichtigter Korrespondenzen.....	50
7.7 Distanz zwischen zwei Bildern.....	50
7.8 Ähnlichkeit zu [CR00].....	51
7.8.2 Verfahren aus [CR00].....	51
7.8.2 Ähnlichkeiten und Unterschiede.....	52
Kapitel 8: Resultate.....	54
8.1 Stabilität des Verfahrens.....	54
8.2 Künstliche Beispiele.....	54
8.3 Bilder aus der Caltech101 Datenbank.....	55
Kapitel 9: Schlussfolgerung und weiterführende Ideen.....	56
Appendix A: Beispiele.....	58
Quellenverzeichnis.....	67

Kapitel 1: Einleitung

1.1 Problembeschreibung

Inhaltsbasierte Bildsuche wird immer wie wichtiger, sei es in der Medizin, der Wissenschaft oder für die Gewährleistung der öffentlichen Sicherheit. Das Erkennen von Objekten in Bildern stellt für einen Menschen in der Regel keine Schwierigkeit dar, weder im Falle von künstlich gerenderten Bildern, noch im Falle von realen Fotos. In der automatischen Bilderkennung ist jedoch das Gegenteil der Fall. Obwohl bereits seit längerer Zeit in diesem Bereich geforscht wird, sind noch keine rundum zufrieden stellende Methoden zur Objekterkennung entwickelt worden.

Um das eigentliche Problem der inhaltsbasierten Bildsuche besser angehen zu können, unterscheidet man mehrere, spezifischere Varianten, zum Beispiel das Erkennen des absolut selben Objekts unter geänderten Sichtpunkten oder das Zuordnen einer Objektkategorie aus einem Pool an Kategorien zu einem Bild. Die vorliegende Semesterarbeit beschäftigt sich hauptsächlich mit letzterer Variante. Die Schwierigkeit dieses Objektklassifizierungproblems röhrt von folgenden vier Faktoren her:

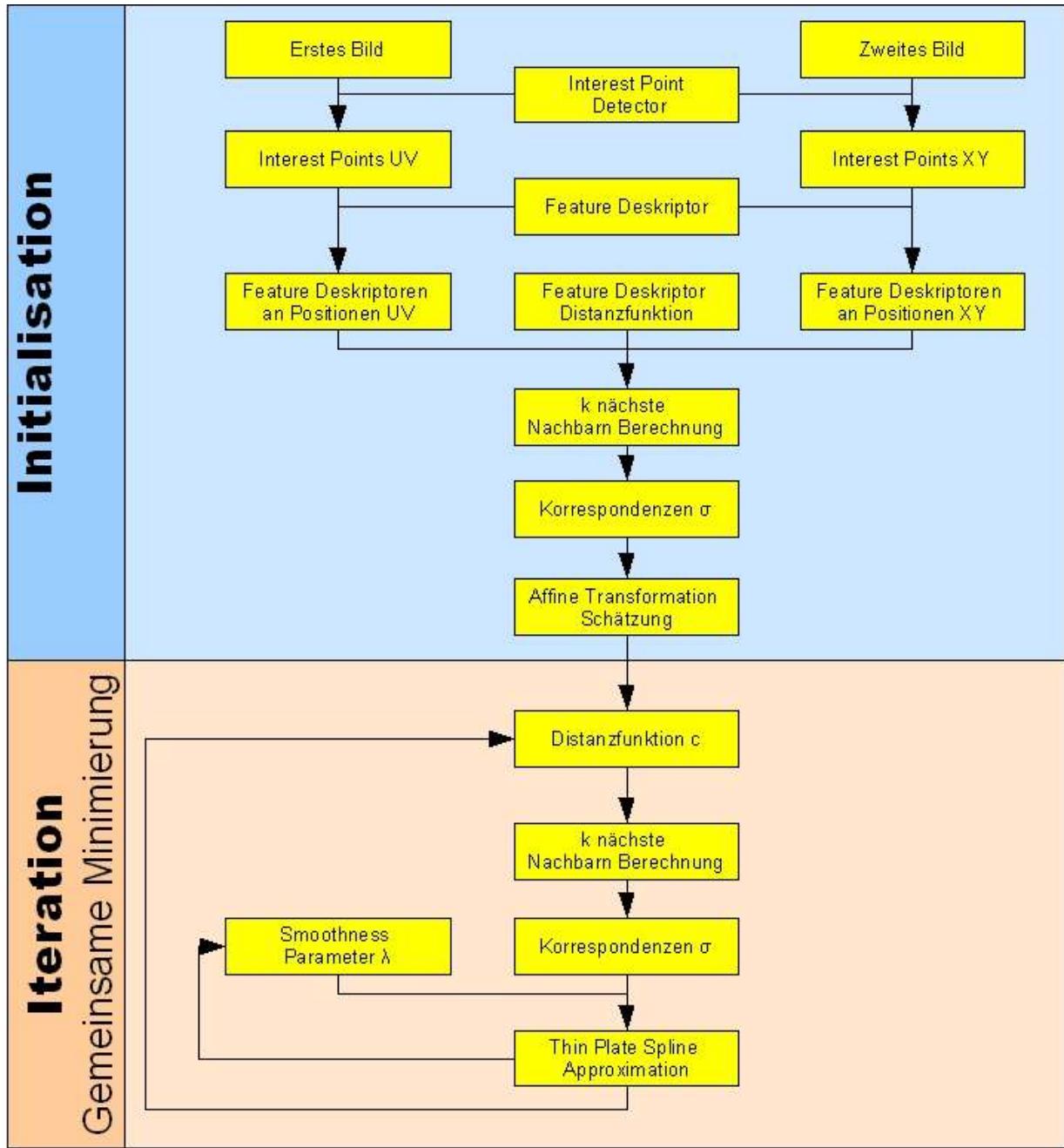
- Variationen des Gesichtspunkts führen zu einer geometrischen Verzerrung. Dabei kann dasselbe Objekt aus leicht geänderten Gesichtspunkten sehr unterschiedlich erscheinen.
- Beleuchtungsvariationen führen zu verschiedenen Beleuchtungsbedingungen, welche zum Beispiel zuvor nicht spiegelnde Bereiche nun spiegeln lassen.
- Objektvariationen, auch bekannt als Interclass Variationen, sind eine spezifische Eigenheit der Objektklassifizierung. Zum Beispiel würde ein Cabriolet von einem Menschen natürlich ebenso wie eine geschlossene Limousine zur Kategorie Auto gezählt. Diese Interclass Variationen werden einem automatischen Erkennungsprozess zum Beispiel mittels einem Machine Learning Verfahren durch Präsentieren von entsprechend aufbereiteten Beispielen beigebracht.
- Teilweise Objektverdeckungen und Stördaten erschweren den Prozess aus nahe liegenden Gründen.

Man kann hauptsächlich zwischen zwei Ansätzen zur Lösung des Objektklassifizierungsproblems unterscheiden. Der erste Ansatz vertraut auf die Verallgemeinerungsfähigkeit eines probabilistischen Lernverfahrens, das direkt auf den Bilddaten oder auf entsprechend aufbereiteten Information der Bilder trainiert wird. Dieser Ansatz benötigt natürlich ein grosses Trainingsset. In dieser Beziehung besser ist der zweite Ansatz, welcher auf der Beobachtung beruht, dass ein Mensch ein Objekt selbst in einer vereinfachten, eher abstrakten Version erkennen kann [B87]. Dies legt nahe, diese Tatsache algorithmisch auszunutzen, was zu den beiden im zweiten Kapitel näher beschriebenen Verfahren führt: Shape Matching und Compositional Graphical Models [OB05].

Shape Matching ist eine Sammelbezeichnung für Verfahren, welche auf einer bestimmten gemeinsam zu Grunde liegenden Idee basieren. Diese Idee wird später genauer erläutert. Demgegenüber ist Compositional Graphical Models ein spezifisches, nicht dem Shape Matching angehöriges Verfahren, welches beispielhaft als Alternative zum Shape Matching näher erklärt wird.

1.2 Kurzangabe der Semesterarbeit

Diese Semesterarbeit untersucht die Objekterkennung in Bildern beruhend auf dem Shape Matching Ansatz, unter Verwendung der Feature Deskriptoren aus dem Compositional Graphical Model Verfahren [OB05]. Im Verlaufe der Semesterarbeit wurde ein Verfahren entwickelt, um Distanzen zwischen zwei Bildern zu berechnen. Diese Distanz kann dann weiterverwendet werden, um das Objektklassifizierungsproblem zu lösen. Folgendes Diagramm gibt einen vereinfachten Überblick über den Algorithmus, welcher in zwei Phasen unterteilt werden kann: eine Initialisierungsphase und eine Iterationsphase.



Gegeben seien zwei Bilder mit dem Ziel, die Ähnlichkeit zwischen diesen beiden Bildern zu finden. Ausgehend von diskreten Punktmengen, jeweils eine für jedes Bild, wird versucht, eine Transformation vom einen Bild auf das andere Bild zu schätzen. Dazu werden Feature Deskriptoren, welche lokal die Geometrie und Farb- bzw. Grauwertverteilung des Bildes beschreiben, an den in den Punktmengen enthaltenen Punkten berechnet. Die Feature Deskriptoren des einen Bildes werden dann mit denjenigen des anderen Bildes verglichen, mit dem Ziel, Korrespondenzen σ zwischen den Punkten in der ersten und den Punkten in der zweiten Punktmenge zu finden. Eine Distanzfunktion bestimmt dazu die Ähnlichkeit zwischen zwei Feature Deskriptoren. Danach werden für jeden Punkt der ersten Punktmenge die k -nächsten Nachbarn (bezüglich eben dieser Distanzfunktion) in der zweiten Punktmenge bestimmt: dies entspricht 1-to- k Zuordnungen, welche im Folgenden auch Korrespondenzen genannt werden. Ausgehend von solchen gefundenen Korrespondenzen wird dann eine affine Transformation geschätzt. Diese affine Transformation wird zur Initialisierung einer zweiten Distanzfunktion c benötigt.

Die zweite Phase verläuft weitgehend analog zur ersten, jedoch werden nun komplexere Transformationen zugelassen.

Wie in vielen anderen Ansätzen auch werden Thin Plate Splines als Funktionsklasse der Transformationen verwendet. Iteriert wird über einen Smoothness Parameter λ , welcher die Komplexität der Thin Plate Spline Transformation steuert: in jedem folgenden Iterationsschritt werden etwas komplexere Transformationen zugelassen. Nun wird jedoch eine dreiteilige Distanzfunktion zur Korrespondenzbestimmung verwendet, bestehend aus einem Term, welcher die Ähnlichkeit der Feature Deskriptoren miteinbezieht, einem Term, welcher die zuvor berechnete affine Transformation zu erhalten versucht und einem letzten Term, welcher das Gedächtnis der vorherigen Iterationen darstellt. Dieses Gedächtnis, auch Momentumterm genannt, ist nötig, um ein Overfitting an die Daten durch die immer komplexeren Thin Plate Spline Approximationen zu verhindern. Am Ende eines Iterationsschritts wird dieses Gedächtnis jeweils aktualisiert, was mittels dem untersten Pfeil dargestellt wird.

Die Distanz zwischen zwei Bildern wird schliesslich unter Verwendung der gefunden Thin Plate Spline Transformation berechnet.

Da die Qualität der Feature Deskriptoren zentral für den Erfolg der Methode ist, wurden im Verlaufe der Semesterarbeit verschiedene Feature Deskriptoren näher untersucht: der Geometric Blur Feature Descriptor aus [BBM05], Localized Feature Histograms aus [OB05], sowie Variationen von letzterem.

1.3 Organisation der Semesterarbeit

Die vorliegende Semesterarbeit ist wie folgt aufgebaut. Im zweiten Kapitel wird näher auf zwei unterschiedliche Lösungsansätze zur Objekt Klassifizierung eingegangen. Beiden gemeinsam ist die Nutzung von Feature Deskriptoren, welche im dritten Kapitel beschrieben werden. Im vierten Kapitel werden verschiedene Arten von Korrespondenzen, welche sowohl in vorherigen Ansätzen verwendet, als auch in dieser Arbeit neu ausprobiert wurden, behandelt. In den Kapiteln fünf und sechs werden affine Transformationen respektive Thin Plate Splines näher beschrieben und analysiert. Im siebten Kapitel wird der entwickelte Objektklassifizierungsalgorithmus detailliert vorgestellt. Im achten Kapitel werden kurz die erzielten Resultate präsentiert, im neunten Kapitel werden weiterführende Ideen vorgeschlagen.

Kapitel 2: Objekt Klassifizierung

Dieses Kapitel beschreibt zwei verschiedene Ansätze zur Lösung des Objektklassifizierungsproblems. Einen Überblick über Methoden, die dem zuerst beschriebenen Ansatz folgen, gibt [VH99]. Der zweite Ansatz existierte zwar schon zuvor, wurde aber erst kürzlich, beschrieben in [OB05], zum ersten Mal im Bereich der bildbasierten Objektklassifizierung verwendet. Der zweite Ansatz wird zum Einen als exemplarische Alternative zum Shape Matching, zum Anderen, weil die dort verwendeten Feature Deskriptoren in dieser Semesterarbeit verwendet werden, kurz näher beschrieben.

2.1 Shape Matching

Shape Matching berücksichtigt, wie der Name schon sagt, lediglich die Form des Objekts, d.h. dessen Geometrie, welche durch Kanten, Konturen und Flächen beschrieben wird. Da diese Semesterarbeit auf Ideen und Verfahren des Shape Matchings aufbaut, wird in den folgenden Unterkapiteln näher darauf eingegangen.

2.1.1 Mathematische Beschreibung

Die hier verwendete mathematische Beschreibung des Shape Matchings folgt der Beschreibung und den Definitionen in [VH99]. Eine Shape wird wie folgt definiert:

Definition: Ein Pattern p ist eine Teilmenge von \mathbb{R}^2 .

Eine Kollektion von Patterns P und eine Transformationsgruppe

$$G = \{g | g : P \rightarrow P\}$$

bestimmen eine Familie von Shapes P/G . Die zu einem Pattern p zugehörige Shape entspricht dem durch p gehenden Orbit G

$$G(p) = \{g(p) | g \in G\}.$$

Der Shape Space entspricht der Kollektion aller dieser Orbits.

Um zwei Shapes miteinander vergleichen zu können, muss ein Ähnlichkeitsmaß auf dem Shape Space definiert werden. Dies direkt zu tun, stellt einem vor gewisse Schwierigkeiten. Darum definiert man ein Unähnlichkeitsmaß

$$\tilde{d} : P/G \times P/G \rightarrow \mathbb{R}$$

auf dem Shape Space ausgehend von einem bezüglich der Transformationsgruppe G invarianten Unähnlichkeitsmaß $d : P \times P \rightarrow \mathbb{R}$ auf dem Pattern Space. Ist d zudem eine Semi-Metrik, so besagt Theorem 3.1 in [VH99], dass

$$\tilde{d}(G(p_1), G(p_2)) = \inf \{d(g(p_1), p_2) | g \in G\}$$

eine Semi-Metrik auf dem Shape Space ist.

Mittels einem solchen Unähnlichkeitsmaß auf dem Shape Space ist es nun möglich, die Ähnlichkeit zwischen zwei Patterns unabhängig von Transformationen zu bestimmen. Betrachtet man als Beispiel affine Transformationen für die Transformationsgruppe G , so findet man leicht eine Metrik auf dem Pattern Space, welche die Voraussetzungen der Invarianz bezüglich affinen Abbildungen erfüllt, nämlich die Diskrete Metrik

$$d(p_1, p_2) = \begin{cases} 0 & \text{if } p_1 = p_2 \\ 1 & \text{otherwise} \end{cases}$$

Das Beispiel der affinen Transformationsgruppe ist insofern interessant, als dass affine Invarianz die Distanz zwischen zwei Patterns unabhängig vom gewählten Koordinatensystem macht.

Die Wahl der Gruppe von Transformationen G geht eng einher mit der Wahl der Distanzfunktion auf dem Pattern Space. Unterscheiden sich zwei Pattern p_1 und p_2 nur marginal voneinander, so sollte intuitiverweise erwartet werden können, dass auch $g(p_1)$ und $g(p_2)$ für ein $g \in G$ sich nur marginal voneinander unterscheiden, aber trotzdem nicht ganz identisch sind! Mathematisch ausgedrückt bedeutet das, dass jedes $g \in G$ ein Homeomorphismus zwischen den topologischen Räumen, dem Definitionsbereich und dem Bildbereich von g , sein muss. Zusammen mit der Forderung, dass die Metrik $d : P \times P \rightarrow \mathbb{R}$ invariant gegenüber der Transformationsgruppe G sein soll, folgt, dass jedes $g \in G$ eine Isometrie sein muss!

2.1.2 Unzulänglichkeiten der mathematischen Beschreibung

In der Praxis sind diese doch sehr theoretischen Formulierungen jedoch problematisch. Am Beispiel der Diskreten Metrik mit affinen Transformationen ist nämlich bereits ein Schwachpunkt dieser mathematischen Definitionen erkennbar. Lediglich die Forderung, dass d eine Semi-Metrik und jedes $g \in G$ eine Isometrie sein soll, reicht in der Praxis nicht aus: ist p_1 lediglich ein bisschen gegenüber p_2 gekrümmmt, so ist die durch die Diskrete Metrik induzierte Shape Distanz bereits maximal! Weitere Probleme kommen hinzu, zum Beispiel:

- Wie sollen die in einem Bild enthaltenen Informationen mittels einem entsprechenden Pattern repräsentiert werden? Eine diskrete Darstellung ist dafür unumgänglich. Die obige Definition eines Patterns wurde sehr generell gehalten. In dieser Semesterarbeit, wie auch in vielen anderen Publikationen, werden endliche Punktmengen als Patterns verwendet. Wenn im Folgenden darum von Patterns die Rede ist, sind immer endliche Punktmengen gemeint.
- Welche Gruppe von Transformationen G sollte berücksichtigt werden? Wünschenswert wäre, dass die Zuordnung zwischen Objektkategorien und Shapes injektiv ist, d.h. zwei unterschiedliche Objektkategorien sollten nicht zur selben Shape gehören. Um dies zu erreichen, muss die Mächtigkeit der Gruppe von Transformationen eingeschränkt werden. Nur genügend glatte Transformationen werden berücksichtigt. Eine Einschränkung auf lediglich genügend glatte Transformation lässt zudem einen Schluss von endlich vielen Punkten auf das ganze Bild zu.
- Wie sieht eine aussagekräftige Distanzfunktion d auf dem Pattern Space aus? Wie zuvor am Beispiel der Diskreten Metrik gezeigt, existieren Distanzfunktionen und Transformationsgruppen, welche zusammen die obigen theoretischen Bedingungen zwar erfüllen, praktisch aber nutzlos sind. Welche Forderungen müssen also an die Distanzfunktion auf dem Pattern Space gestellt werden, um eine aussagekräftige Distanzfunktion zu erhalten? Ist p_1 nur ein bisschen gegenüber p_2 „verändert“, dann muss von der Distanzfunktion gefordert werden, dass $d(p_1, p_2)$ klein ist, falls hingegen p_1 gegenüber p_2 stark „verändert“ ist, so muss $d(p_1, p_2)$ gross sein. Die Distanzfunktion auf dem Pattern Space muss also eine gewisse Stetigkeit bezüglich Veränderungen zwischen zwei Patterns erfüllen:

$$g \in G \text{ ist 'klein'} \Leftrightarrow d(g(p), p) \text{ ist klein} .$$

Durch das Addieren einer Art Norm der Transformation $|g|$ in der Distanzfunktion kann diese Stetigkeit erreicht werden:

$$\tilde{d}(G(p_1), G(p_2)) = \inf \{ d(g(p_1), p_2) + |g| \mid g \in G \}$$

Diese enge Kopplung der Distanzfunktion mit den Transformationen g führen zum im nächsten Unterkapitel beschriebenen Huhn-Ei Problem.

- Wie sollen diese Veränderungen $|g|$ quantitativ erfasst werden? Änderungen zwischen zwei Patterns werden durch die Transformationsgruppe modelliert, darum sollte auch das Mass der Veränderung zwischen zwei Patterns mittels der Transformationsgruppe definiert werden. Wie später beschrieben, besitzen Thin Plate Splines ein natürliches, nahe liegendes Änderungsmass.

Aus diesen Problemen folgt, dass praktische Verfahren sich nicht exakt an diese theoretischen Überlegungen halten können. Als fundierte Basis zum Entwerfen eines Verfahrens sind sie jedoch allemal nützlich.

2.1.3 Korrespondenzen und das Huhn-Ei Problem

Angenommen, man würde eine optimale Abbildung $\sigma: [1, \dots, n] \times [1, \dots, k] \rightarrow [1, \dots, m]$ zwischen den Indexen der n Punkte des ersten Patterns p_1 und den m Punkten des zweiten Patterns p_2 kennen. $\sigma(i, j)$ bezeichnet dabei zum Beispiel den Index des j -nächsten Nachbarn im zweiten Pattern bezüglich dem i -ten Punkt des ersten Patterns. Wird lediglich der nächste Nachbar berücksichtigt ($k = 1$) und ist diese Abbildung bijektiv, so entspricht sie einer Permutation der $n = m$ Indexe.

Folgende Distanzfunktion zwischen zwei Patterns wäre dann nahe liegend:

$$d(p_1, p_2) = \sum_{i=1, j=1}^{n, k} c(p_1[i], p_2[\sigma(i, j)]) ,$$

wobei c eine beliebig modellierte Distanzfunktion zwischen zwei Punkten sein kann. In c können zum Beispiel die Pixelkoordinaten und Feature Deskriptor Informationen der zwei Punkte eingehen.

Nun stellt sich folgendes dem Huhn-Ei Problem analoge Problem: würde man entweder die optimale Korrespondenzen $\sigma(i, j)$ oder die optimale Transformation g kennen, so wäre das Lösen des restlichen Problems viel einfacher. Sind die optimalen Korrespondenzen $\sigma(i, j)$ bekannt, so entsteht folgendes Optimierungsproblem:

$$\tilde{d}(G(p_1), G(p_2)) = \inf_g \left\{ |g| + \sum_{i=1, j=1}^{n, k} c(p_1[i], p_2[\sigma(i, j)]) \mid g \in G \right\} .$$

Wäre umgekehrt die optimale Transformation g zwischen zwei Patterns bekannt, so würde sich das Problem auf das Suchen der optimalen Korrespondenz σ beschränken:

$$\tilde{d}(G(p_1), G(p_2)) = \inf_{\sigma} \left\{ |g| + \sum_{i=1, j=1}^{n, k} c(p_1[i], p_2[\sigma(i, j)]) \mid g \in G \right\}$$

So einfach die einzeln betrachteten Optimierungen auch erscheinen mögen, eine gleichzeitige, gemeinsame Optimierung der Transformation und der Korrespondenz ist jedoch ein schwieriges, nicht-konvexes Optimierungsproblem. [BBM05], [BMP02], [BM01], [CR00] als auch diese Semesterarbeit versuchen dieses Optimierungsproblem, jeweils in einer etwas modifizierten Version, auf verschiedene Arten zu lösen. Diese Verfahren können anhand folgender Merkmale eingeteilt werden:

- Gemeinsames oder entkoppeltes Lösen des Minimierungsproblems
- Wahl der Distanzfunktion c
- Wahl der Informationen, welche in die Distanzfunktion c eingehen.
- Art der Korrespondenzen, über welche optimiert wird.

Zusammenfassend kann ein genereller Shape Matching Algorithmus wie folgt beschrieben werden:

Input: zwei Bilder
 Output: Ähnlichkeit zwischen den Objekten auf den zwei Bildern
 Algorithmus:
 1. Extrahiere Punktmengen: $p_1 = \text{extractPattern}(\text{image1})$;
 $p_2 = \text{extractPattern}(\text{image2})$;
 2. Minimiere folgendes Funktional über g und σ :
 $(g^*, \sigma^*) = \arg \min_{g, \sigma} E(g, \sigma) = \arg \min_{g, \sigma} \left\{ |g| + \sum_{i=1, j=1}^{n, k} c(p_1[i], p_2[\sigma(i, j)]) \right\} ;$
 3. Gebe $E(g^*, \sigma^*)$ zurück;

2.2 Compositional Graphical Models

Auf Shape Matching beruhende Verfahren basieren inhärent auf der Definition eines Ähnlichkeitsmasses für Shapes bzw. Patterns und Alignment Transformationen. Eine solche, möglichst allgemein gültige Definition zu finden ist schwierig, wenn nicht gar ganz unmöglich.

Compositional Graphical Models verhindern das Definieren eines solchen Ähnlichkeitsmass für Shapes, indem ein wahrscheinlichkeitsbasierter Ansatz benutzt wird. Diese Verfahren basieren auf der in [B87] präsentierten Idee, die visuelle Wahrnehmung ebenso wie die Sprache in ihren kleinsten Elementen zu beschreiben. In der Linguistik werden diese Elemente Phoneme genannt und jedes Wort jeglicher Sprache kann mittels einer sequentiellen Komposition von solchen Phonemen ausgedrückt werden. Für die visuelle Wahrnehmung erschwert sich dieser Prozess jedoch enorm:

- Das Finden der visuellen Basiselementen ist viel komplexer, da im Unterschied zur Sprache eine fast unbegrenzte Menge von Kompositionen existieren.
- Die möglichen Kompositionen sind nun zweidimensionale Projektionen von dreidimensionalen Basiselementen, die sich nun zudem gegenseitig verdecken können.

Anstatt nach solchen Basiselementen wie in [B87] explizit zu suchen, versucht in [OB05] ein graphisches Modell mittels Beispielen eben diese Basiselemente zu erlernen. Zudem versucht dieses Modell, die kategoriespezifischen Relationen zwischen diesen Basiselementen herauszufinden. Das graphische Modell ist ein Bayesian Network, wobei die Menge der Basiselemente durch ein Codebook an Feature Deskriptoren dargestellt wird und als Evidenz für den Input zur Inferenz der Objektklassifikation dient. Ein Vorteil dieser Methode liegt darin, dass sowohl Bottom-up Informationen durch die Feature Deskriptoren als auch Top-down Informationen durch die zuvor erlernten kategoriespezifischen Relationen im Bayesian Network propagiert werden können, um so eine verlässlichere Objektklassifikation zu erhalten. Da diese Methode wahrscheinlichkeitsbasiert ist, steht am Ende ein Konfidenzwert für die Objektklassifikation zur Verfügung. Dieser Konfidenzwert kann entfernt mit dem Ähnlichkeitsmass im Shape Matching verglichen werden, mit dem Unterschied, dass der Konfidenzwert eine klare Interpretation besitzt!

Die einzige Modellierungsschwierigkeit besteht nun darin, ein Verfahren zur Codebook Extraktion und eine Abbildung von den Feature Deskriptoren als Wahrscheinlichkeitsverteilung festzulegen.

Die Ähnlichkeit dieses Ansatzes mit Kenntnissen des menschlichen visuellen Systems sind erstaunlich. In [T86] wird vorgeschlagen, dass zwei Prozesse für das visuelle System verantwortlich sind. Ein paralleler Bottom-up Prozess (Preattentive Process) beschäftigt sich mit der Gruppierung von Informationen, die zur Unterscheidung von Vorder- und Hintergrund dienen. Daran schliesst ein sequenzieller Top-down Prozess (Attentive Process) an, der die noch immer abgetrennten Features eines Objekts selektiert und zu einem Ganzen verarbeitet.

Kapitel 3: Feature Deskriptoren

3.1 Motivation

Beide im vorhergehenden Kapitel erläuterten Vorgehensweisen werden von der Beobachtung motiviert, dass auch das visuelle System des Menschen die Bildverarbeitung auf mehrere, im Gehirn lokalisierbare Regionen aufteilt. So zeigen Experimente, dass Bereiche im Gehirn existieren, welche für einen ganz spezifischen Aspekt des Sehvermögens verantwortlich sind, zum Beispiel für Tiefe, Form, Bewegung oder Farbe. Unser Sehvermögen baut nun anhand paralleler Verarbeitungswege dieser Features eine Interpretation des Bildes auf. Diese Integration der parallelen Verarbeitungswege wird auch attentiver Prozess genannt. Es ist jedoch bis jetzt unklar, wie dieser attentiver Prozess genau funktionieren soll. Einen guten Einblick in die Neurologie des visuellen System gibt [KSJ00].

Zentral ist die Tatsache, dass das menschliche visuelle System einzelne Features aus einem „Bild“ der Retina extrahiert und weiterverarbeitet. Motiviert durch diese Erkenntnisse wurden zahlreiche Varianten entwickelt, dem Computer ebenfalls beizubringen, solche Features aus Bildern zu extrahieren. Diese so genannten Feature Deskriptoren versuchen nun möglichst viel Informationen aus der Umgebung eines Bildpunktes einzufangen, aber trotzdem möglichst invariant bezüglich Änderungen im gesamten Bild zu sein. Beim Design eines Feature Deskriptors muss also ein Kompromiss zwischen Informationsmenge und Grösse der berücksichtigten Umgebung getroffen werden.

Die Qualität der Feature Deskriptoren ist vor allem für die Korrespondenzsuche zentral. Lediglich aus aussagekräftige Feature Deskriptoren lassen sich zuverlässige und robuste Punkt-Korrespondenzen finden. Aber auch die Geschwindigkeit zur Berechnung der Feature Deskriptoren ist ein zu berücksichtigender Faktor. Eine schnellere Berechnung der Feature Deskriptoren eröffnet weitere Möglichkeiten, sei es die Extraktion von noch mehr Feature Deskriptoren oder die Realisierung einer Echtzeitanwendung.

3.2 Feature Deskriptor Beispiele

Im Verlaufe der Semesterarbeit wurden mehrere Feature Deskriptoren näher untersucht. Diese werden in den folgenden Unterkapiteln kurz vorgestellt.

3.2.1 Geometric Blur

Geometric Blur wurde zum ersten Mal in [BM01] beschrieben. Das Hauptproblem von Feature Deskriptoren ist, den richtigen Kompromiss zwischen der Grösse des berücksichtigten Bildfensters und daraus resultierenden Variationen, seien es Gesichtspunktvariationen, Objektvariationen oder Beleuchtungsvariationen, zu finden. Denn ein zu kleines Bildfenster verunmöglicht es dem Deskriptor, auf genügend Informationen für ein aussagekräftiges Feature zu zugreifen. Ein zu grosses Bildfenster führt hingegen zu zu grossen Variationen desselben Features.

Eine interessante Idee zur Lösung dieses Problems liefert der Geometric Blur. Zuerst wird versucht, das Erscheinungsbild eines Objekts unabhängig der Positionierung zu machen. Dies wird erreicht, indem das Bild in mehrere Kanäle unterteilt wird. Diese Kanäle sind stellen meist Filterantworten von auf dem Bild angewendeten Filtern dar. In der ursprünglichen Publikation [BM01] wurden zwölf Half-Wave Rectified Oriented Edge Responses verwendet. Es wird auch bemerkt, dass Geometric Blur am effektivsten auf spärlichen Signalen funktioniert. Es ist darum zum Beispiel von Vorteil, anstelle von binären Kantendetektoren probabilistische Kantendetektoren, welche Kanten eine Wahrscheinlichkeit zuordnen, zu verwenden. Dies zeigte sich auch in Versuchen, welche im Rahmen dieser Semesterarbeit durchgeführt wurden.

Um Unsicherheiten in der Positionierung zu berücksichtigen, ist folgende Beobachtung essentiell. Je weiter man sich von der zentralen Region des Feature Deskriptors entfernt, um so mehr Unsicherheit resultiert in der räumlichen Positionierung eines Signals auf Grund von zuvor genannten Variationen. In [BM01] wird eine theoretische Herleitung für diese intuitive Idee gegeben. Indem man das Bild mit einem Gausschen Filter verschmiert, kann der Betrag dieser Unsicherheit abhängig vom Einflussbereich des Filters in die Modellierung eingeflossen werden lassen. Da die Unsicherheit proportional zum Abstand zunimmt, wird im Geometric Blur ein im Abstand zum Feature Deskriptor Zentrum variabler Einflussbereich des Gauss Filter verwendet: je weiter entfernt, umso mehr wird das Signal verschmiert! Dieses räumliche Blurring wird auf jedem Bildkanal einzeln durchgeführt. Der Geometric Blur Descriptor an dem Punkt x_0 eines Kanals C kann als Konvolution mit einer Gaussfunktion also folgendermassen geschrieben werden:

$$GB_{x_0}(x) = C * G_{(\mu, \sigma)}(|x - x_0|) \text{ wobei:}$$

σ : Standardabweichung

μ : Mittelwert

Die Standardabweichung des Gauss Filters kann der Geschwindigkeit der Unsicherheitszunahme im Verhältnis zum Abstand angepasst werden. Der Mittelwert steuert die 'Basisunsicherheit' welche unabhängig vom Abstand in den Bildern vorhanden ist.

In einer leicht erweiterten Form fand der Geometric Blur dann auch in [BBM05] Verwendung. Der Geometric Blur Deskriptor ist für vom Feature Deskriptor Zentrum weit entfernte Punkte ziemlich gleichmässig. Der Geometric Blur Deskriptor kann darum anhand folgendem Samplingschema diskretisiert werden:

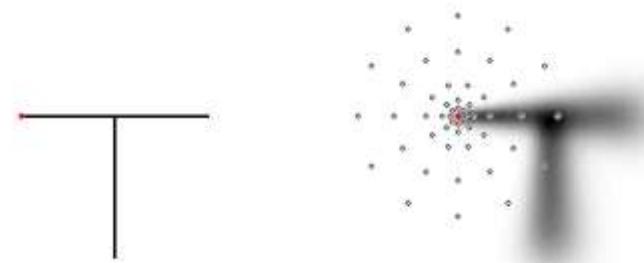


Bild aus [BBM05]

Originales Bild mit rot eingefärbter Feature Deskriptor Position

Diskreter Geometric Blur mit Sampling-Positionen als kleine weisse Kreise über stetigem Geometric Blur.

Dies hat einen weiteren Vorteil. Da Samples nur an fix vorgegebenen Abständen genommen werden, können für die einzelnen Kanäle jeweils die Filterantworten der durch die vorgegebenen Abstände festgelegten Gaussfilter vorausberechnet werden.

Die diskreten Geometric Blur Deskriptoren der einzelnen Kanäle, repräsentiert als Vektoren, werden anschliessend alle untereinander in einem grossen Vektor gesammelt, welcher dann auch der eigentlich Feature Deskriptor an diesem Bildpunkt darstellt. Als Abstandsmass für diese diskrete Version des Geometric Blur wird die negative Korrelation verwendet.

Ein klarer Nachteil der Geometric Blur Feature Deskriptoren sind der grosse Rechenaufwand für die Berechnung der einzelnen Kanäle und deren mehrfache Filterung mit verschiedenen Gaussfiltern. Zudem kommen noch sehr viele Parameter, die nur äusserst schwierig passend einzustellen sind. Parameter zu finden, die für alle Bilder passend sind, scheint fast unmöglich zu sein. In dieser Beziehung haben die nachher beschriebenen Localized Feature Histograms sicherlich die Nase vorn.

Die verwendete Implementierung des Geometric Blur basiert auf der Arbeit von Michael Wild und ist rein in Matlab geschrieben. Die ersten Arbeiten dieser Semesterarbeit involvierten diese im Entstehen begriffene Implementierung zu testen und zu bugfixen, um eine lauffähige Version des Geometric Blur später mit anderen Feature Deskriptoren vergleichen zu können.

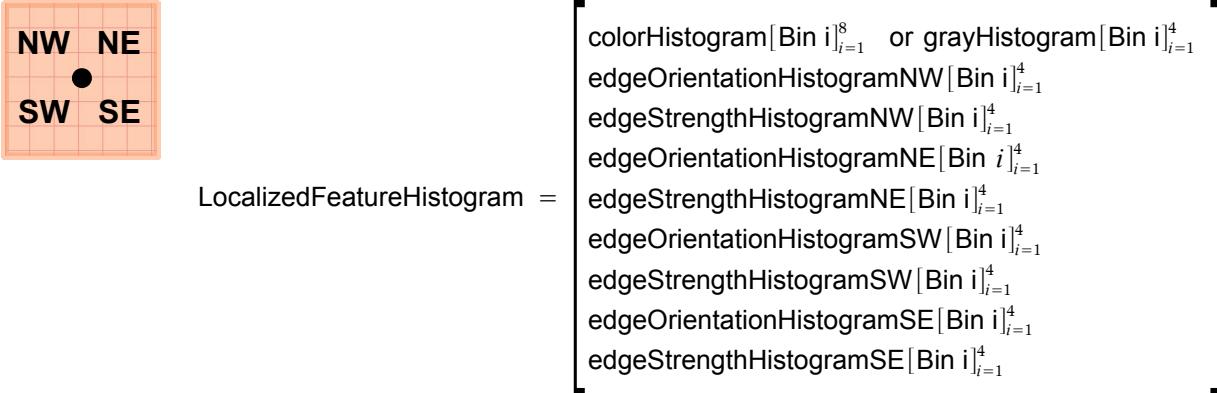
3.2.2 Localized Feature Histograms

Localized Feature Histograms wurden in [OB05] beschrieben. Localized Feature Histograms sind eine Kollektion von Histogrammen, welche von einem Patch in der Nähe eines Bildpunkts her stammen. Die Histogramme approximieren eine Wahrscheinlichkeitsverteilung im Bildpatch über:

- die Graustufen (4 Bins) oder Farben (8 Bins) im Falle von Farbbildern,
- die Kantenstärken (4 Bins)
- die Kantenorientierungen (4 Bins)

Die Kantenorientierungen werden im Histogramm mittels den Kantenstärken gewichtet. Ein Histogramm wird ebenfalls als Vektor dargestellt, der gesamte Feature Deskriptor besteht wiederum aus der Konkatenation aller Histogramme.

In der einfachsten Version wird ein Bildpatch um den Feature Deskriptor Punkt definiert, sodass der Punkt in der Mitte zu liegen kommt. Das Farb- oder Grauerthistogramm wird basierend auf diesem Patch berechnet. Für die Kantenorientierung- und Kantenstärkenhistogramme wird dieser Patch uniform in vier kleinere zerteilt, d.h. man erhält vier Kantenstärken- und vier Kantenorientierungshistogramme.

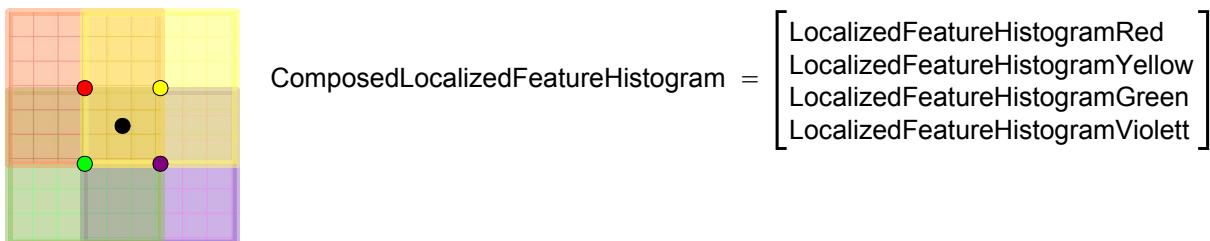


Da dieser Feature Deskriptor aus Histogrammen besteht, ist die naheliegendste Distanzfunktion die χ^2 Test Statistik.

Die verwendete Localized Feature Histogram Implementierung stammt von Michael Sauter und ist als mex Code in C++ geschrieben. Die Implementierung ist sehr flexibel gehalten, zum Beispiel können im CCConstants.h Headerfile fast alle Parameter geändert werden, so auch die Bingrösse der einzelnen Histogramme. Im Rahmen dieser Arbeit wurden aber immer Farbhistogramme der Grösse 8 und alle anderen Histogramme mit Grösse 4 verwendet. Zu Beginn liess diese Implementierung lediglich Feature Deskriptor Punkte an Knoten eines über das Bild gelegten regulären Gitters zu. Dies erklärt, weshalb viel des in dieser Semesterarbeit entstandenen Matlab Codes durch diese Gegebenheit verkompliziert wurde. Im Verlaufe der Arbeit wurde eine verbesserte Localized Feature Histogram Implementierung verfügbar, in welcher beliebige Patches spezifiziert werden können. So ist es nun möglich, einen beliebigen Feature Deskriptor Punkt zu wählen, indem die linke obere und rechte untere Koordinate des Basispatches dem Algorithmus als Input gegeben wird.

3.2.3 Composed Localized Feature Histograms

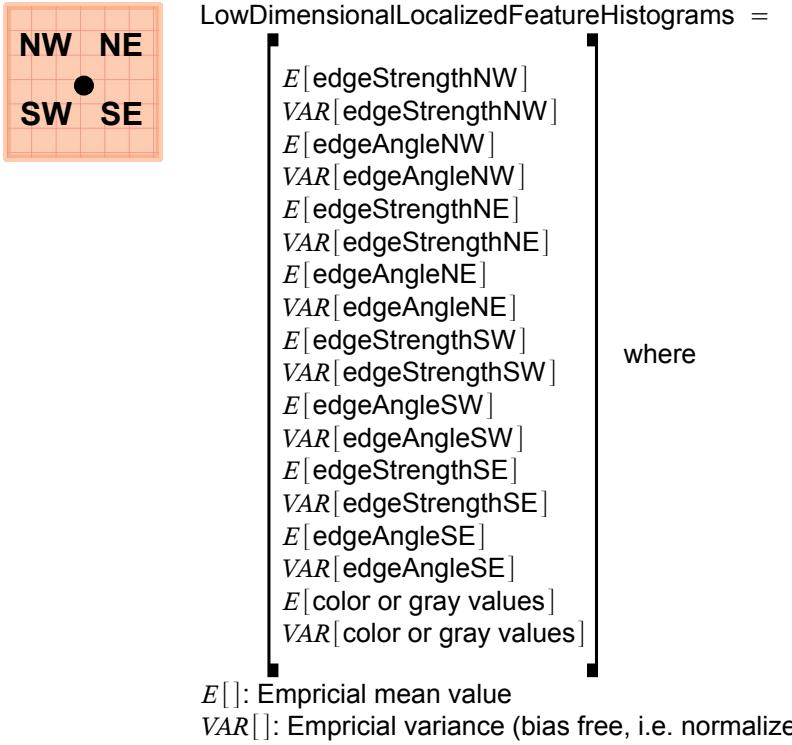
Einige Varianten des Localized Feature Histogram Feature Detektors wurden getestet. Die erste besteht aus einer Konkatenation vier überlappender Localized Feature Histograms. Dies entspricht einer einer einfachen, zweifachen oder sogar vierfachen Gewichtung der Kantenorientierung- und Kantenstärkehistogrammen. Die Farbhistogramme werden aber wie zuvor beschrieben über einen ganzen Patch eines Localized Feature Histograms berechnet und nicht wie die Kantenstärke- und Kantenorientierungshistogramme über Teilpatches. Darum fangen die vier Farbhistogramme im Composed Localized Feature Histogram die Wahrscheinlichkeitsverteilungen der Farben verstärkt abhängig von der relativen Position bezüglich dem Zentrum ein.



Diese Variante stellte sich aber als wenig nützlich heraus, die Resultate waren ziemlich unbefriedigend. Dies röhrt wahrscheinlich daher, da die Patchgrösse einhalb mal grösser als bei den ursprünglichen Localized Feature Histograms ist. Das Verhältnis zwischen Patchgrösse und Stabilität bezüglich Bildvariationen verschiebt sich zu Gunsten der Patchgrösse, wodurch der Deskriptor nicht mehr gut funktioniert. Ein anderer Grund könnte die verstärkte Positionsabhängigkeit der Farbhistogramme sein.

3.2.4 Low Dimensional Localized Feature Histograms

Um die Anzahl an Dimensionen im Localized Feature Histogram Deskriptor (36 im Falle von Grauwertbildern, 40 im Falle von Farbbildern) zu verringern, wurde die Möglichkeit untersucht, Statistiken auf den Werten im Patch zu berechnen. Die Statistikberechnung erfolgt direkt auf den wahren Werten und nicht auf den diskretisierten, den Bins zugeordneten Werten. Für die Statistiken wurden der Mittelwert und die Varianzen der Werte der den Histogrammen zugrunde liegenden Patches berechnet. Dazu wurden die Standardformeln zur Mittelwert- bzw. Varianzschätzung verwendet. Die Mittelwertschätzung für die Kantenorientierung wurde ebenfalls mit den Kantenstärken gewichtet.



Die resultierenden Feature Deskriptoren besitzen also 18 Dimensionen. Die Implementierung wurde in das Bestehende mex-Framework der Localized Feature Histogramme integriert. Der zusätzlich Code wurde möglichst eigenständig gehalten. Die Details können CCStatistics.h und CCStatistics.cpp entnommen werden. Zudem kann in CCConstants.h eine zusätzliche Präprozessordirektive definiert werden, welche die Statistikberechnung an- und abstellt. Wie für die anderen Feature Deskriptoren auch, steht für diesen Deskriptor ein einfacher handhabbarer Wrapper für Matlab bereit.

Die Ergebnisse dieses Feature Deskriptor waren ebenfalls ziemlich ernüchternd. Eine direkte Verwendung dieser Feature Deskriptoren mit einer Distanzfunktion scheint nicht der richtige Weg zu sein. Eine interessante Möglichkeit, welche in einer weiterführenden Arbeit getestet werden könnte, wäre, die so aufbereiteten Informationen der ursprünglichen Localized Feature Deskriptoren in ein Machine Learning Lernverfahren zu speisen, welches dann seinerseits selbst anhand von Beispielen entscheiden soll, welche Statistiken für welche Objektkategorie aussagekräftig sind.

3.2.5 Low Dimensional Composed Localized Feature Histograms

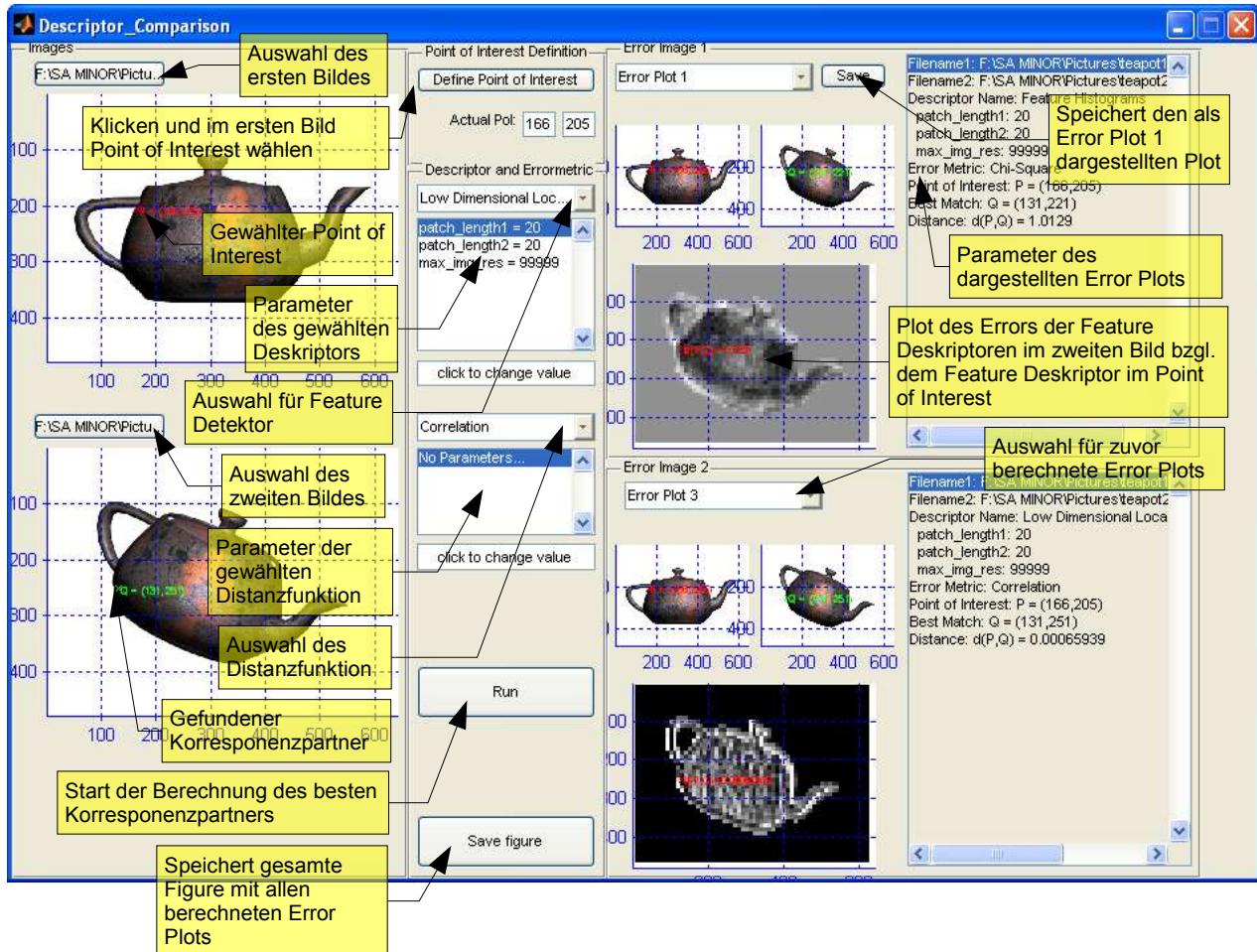
Als letzte Variante wurde die Komposition von Low Dimensional Localized Feature Histograms vierer überlappender Patches versucht. Das Vorgehen ist analog zu den Composed Localized Feature Histograms, mit dem Unterschied, dass nun einfach Low Dimensional Localized Feature Histograms anstelle von Localized Feature Histograms verwendet werden. Aber wie durch die schlechten Resultate der Low Dimensional Localized Feature Histograms zu erwarten war, war auch diese letzte Variante nicht sonderlich erfolgreich.

3.3 Vergleich von Feature Deskriptoren

Um die Feature Deskriptoren besser vergleichen zu können, wurde ein Matlab GUI entwickelt. Damit können die

Einflüsse verschiedener Parameterwerte auf die Feature Deskriptoren einfach untersucht werden. Dazu wird im ersten Bild ein Feature Deskriptor Punkt gewählt (Point of Interest, rot dargestellt). Im zweiten Bild wird dann der beste Korrespondenzpartner bezüglich einer gewählten Distanzfunktion zwischen den Feature Deskriptoren gesucht (grün dargestellt). Damit zwei Feature Deskriptoren gut miteinander verglichen werden können, stehen zwei Plotbereiche zur Verfügung. Mittels einem Drop Down Menu können dann zuvor berechnete Errorplots angewählt werden. Daneben werden die dafür verwendeten Parameter angezeigt. Die Parameter können in der Mitte durch Anklicken und anschliessendes Eintippen eines neuen Wertes in dem entsprechenden Bereich geändert werden. Es können die Parameter der Feature Deskriptoren als auch die Parameter der Distanzfunktion, sofern vorhanden, geändert werden. Feature Deskriptoren können beliebig mit Distanzfunktionen kombiniert werden. Es kann zudem die ganze Matlab Figure oder aber ein einzelner Error Plot gespeichert werden.

Die Standardmässig eingestellten Parameter wurden so bestimmt, dass sie meist die zuverlässigsten Resultate lieferten.



3.3.1 Parameter des Geometric Blur

Der Geometric Blur Feature Deskriptor besitzt folgende Parameter:

Name	Zweck	Mögliche Werte	
filter	<p>Definiert die zu verwendenden Kanäle als Liste von Komma-separierten Filternamen (keine Abstände zwischen den Filternamen). Beispiel: pbCanny,sobel,prewitt</p> <p>Die folgenden Filter können unter http://www.cs.berkeley.edu/projects/vision/grouping/seg_bench/ bezogen werden.</p>	sobel	Matlab sobel filter
		prewitt	Matlab prewitt filter
		roberts	Matlab roberts filter
		log	Matlab log filter
		zerocross	Matlab zerocross filter
		canny	Matlab canny filter
		pbCanny	Probabilistic canny filter
		pbGM	Gradient magnitude filter
		pbGM2	Gradient magnitude on multiple scales
		pbBG	Brightness gradient
		pbCG	Color gradient only for RGB images
		pbTG	Texture gradient
		pbCGTG	Color and texture gradient only for RGB images
		pbBGTG	Brightness and texture gradient
nr_radii	Definiert die Anzahl an Samplinggradien für den diskreten Geometric Blur Deskriptor.	Integer grösser als Null	
		Integer grösser als Null	
nr_theta	Definiert die Anzahl an Samplingpositionen auf einem Radius.	Integer grösser als Null	
alpha	Definiert die Standardabweichung des Gaussfilters	Nicht negative Zahl	
beta	Definiert den Mittelwert des Gaussfilters (Grösse der Basisunsicherheit)	Nicht negative Zahl	
max_radius	Fließt in die Berechnung der einzelnen Radien ein.	Positive, reelle Zahl	
radii	Legt die Radien der Samplingpositionen fest. Durch Ändern dieser Werte können diese Radien direkt geändert werden. Durch Ändern von max_radius oder linear_weight werden diese Werte anhand der unten gegebenen Formel geändert.	Positive reelle Zahlen als Vektor in Matlabnotation. Die Länge der Vektoren muss für beide Deskriptoren gleich sein, d.h. radii1 und radii2 müssen die selbe Anzahl Elemente besitzen.	

Name	Zweck	Mögliche Werte
theta	Legt die Samplingpositionen auf einem Radius fest. Durch Ändern dieser Winkel können die Winkel der Samplingpositionen direkt geändert werden. Durch Ändern von nr_theta werden die Winkel zwischen zwei benachbarten Samplingpositionen uniform gewählt.	Reelle Zahlen als Vektor in Matlabnotation. Die Länge des Vektoren muss für beide Deskriptoren gleich sein, d.h. theta1 und theta2 müssen die selbe Anzahl Elemente besitzen.
linear_weight	Beeinflusst die Berechnung der Samplinggradien. Siehe untere Formel.	Reelle Zahl in [0,1]

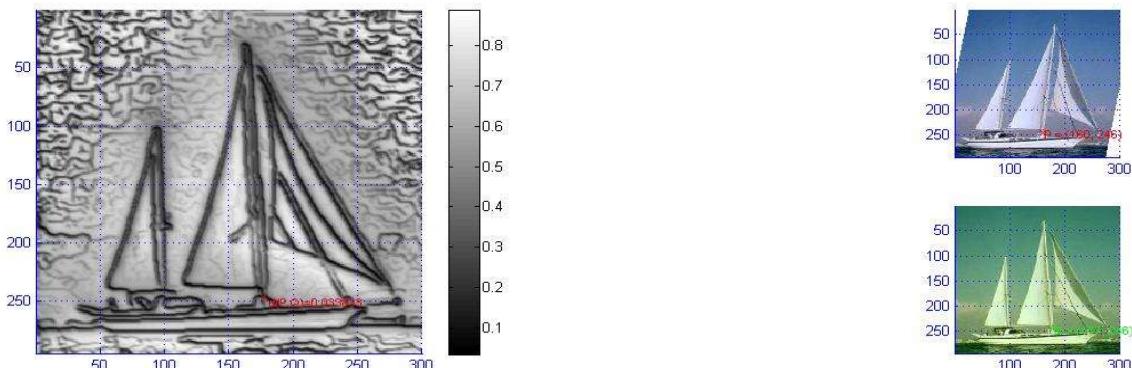
In seiner ursprünglichen Beschreibung in [BM01] nimmt die Stärke des Blurrings proportional zum Logarithmus des Abstands zum Feature Punkt zu. Bei der diskreten Version dieses Deskriptors stellte sich allerdings heraus, dass so zu viel Gewicht auf die unmittelbare Umgebung des Feature Punkts fällt, da die ersten paar Samplinggradien zu nahe zueinander zu liegen kommen. Anstelle einer rein logarithmisch basierten Blurringstärke wurde eine Konvexkombination einer linearen und einer logarithmisch basierten Blurringstärke getestet:

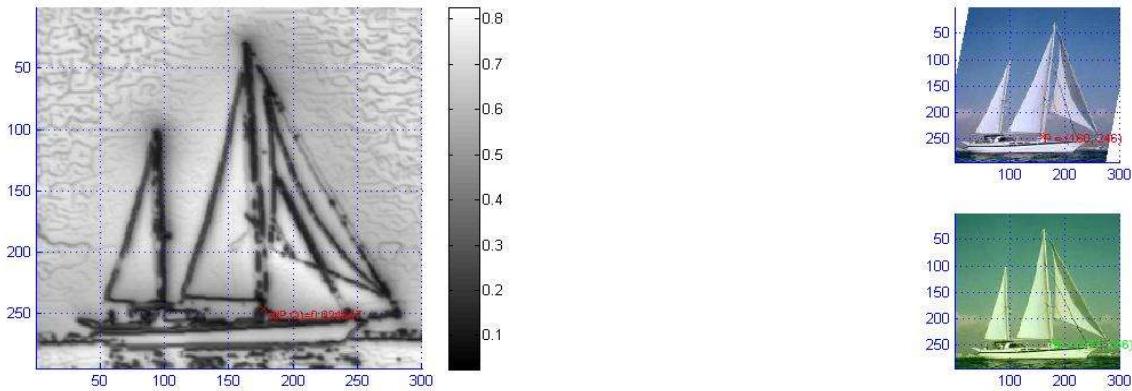
$$\text{radii} = [0, \text{linear_weight} \cdot \text{linspace}(1, \text{max_radius}, \text{nr_radii}) + (1 - \text{linear_weight}) \cdot \text{logspace}(\log_{10}(1), \log_{10}(\text{max_radius}), \text{nr_radii})];$$

Dieser lineare Term hat relativ zum logarithmischen Term nahe dem Zentrum viel Gewicht, hingegen weiter entfernt wird er vom logarithmischen Term dominiert.

An obiger Tabelle erkennt man wiederum einen grossen Nachteil des Geometric Blur: er benötigt sehr viele, korrekt eingestellte Parameter. Abgesehen von den zwei redundanten Parametern, sind alle für eine gute Funktionsweise enorm wichtig. Ein weiterer Nachteil ist die benötigte Rechenzeit: werden mehrere Kanäle gewählt, so steigt die Rechenzeit enorm an.

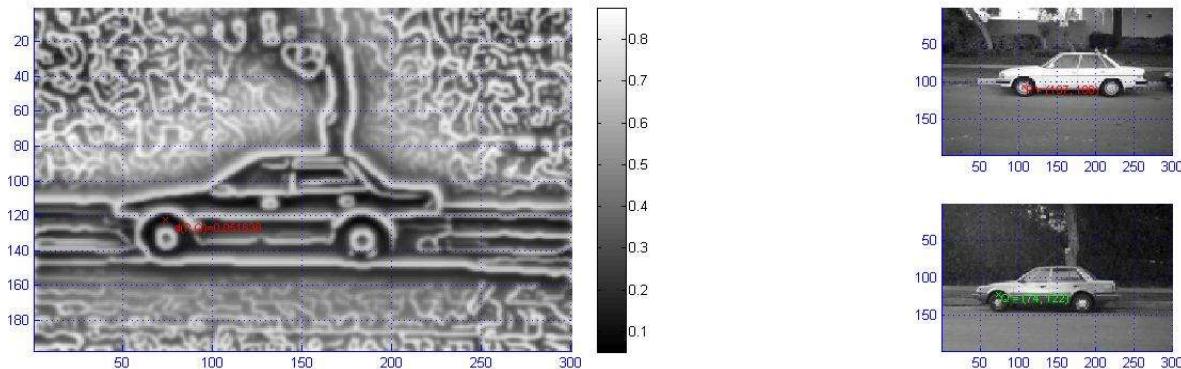
Zwei Beispiele sollen repräsentativ die Funktionsweise des Geometric Blur zeigen. Das erste Bild ist jeweils eine leicht gescherte Version des ursprünglichen Bildes, das zweite Bild besitzt einen verstärkten Grünanteil und soll die Stabilität bezüglich Farbunterschiede zeigen. Beide Berechnungen wurden mit Standardwerten für die Parameter gemacht, als Distanzfunktion wurde 'Correlation' verwendet. Das erste Beispiel verwendet lediglich einen Kanal (pbCanny), das zweite Beispiel verwendet deren drei (pbCanny, sobel, pbGM).





Man bemerkt, dass die Distanz zwischen den Geometric Blur Deskriptoren überall dort klein ist, wo sich auch Kanten in der Nähe befinden, und dies invariant gegenüber der Kantenorientierungen! Dies ist eigentlich nicht wünschenswert, da der Geometric Blur die lokale Geometrie auffassen soll!

Nun noch ein Beispiel, um Variationen innerhalb derselben Klasse zu berücksichtigen. Das Vorderrad wurde fast korrekt erkannt. Aber man bemerke auch hier, dass sich der eigentliche Punkt exakt im Mittelpunkt der Radkappe befand, der Geometric Blur aber wiederum sehr stark auf die Kanteninformation des rundum liegenden Reifens ansprach.

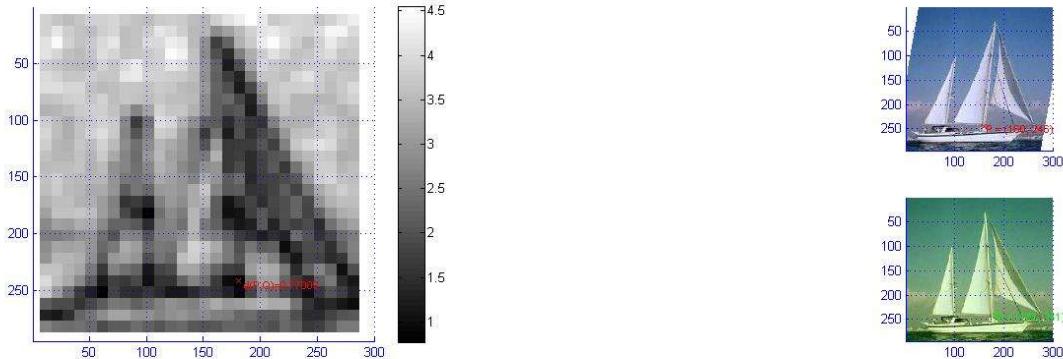


3.3.2 Parameter der Feature Histogramme

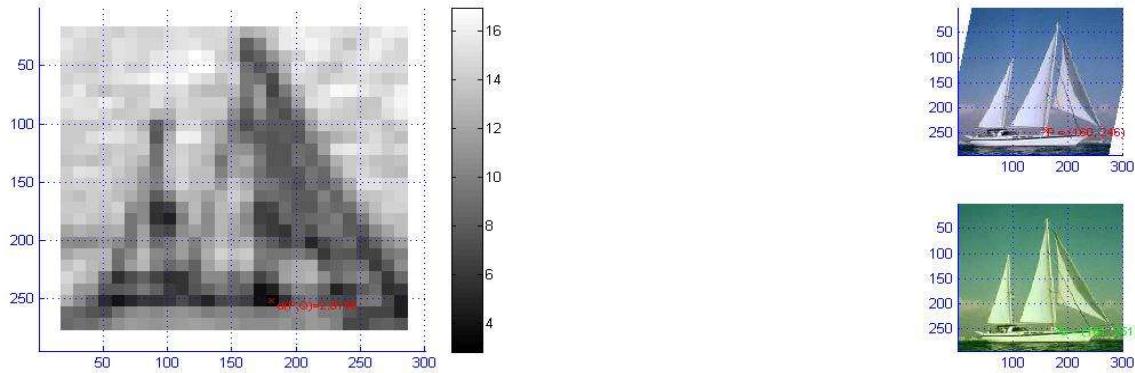
Alle auf den Localized Feature Histogrammen basierenden Feature Deskriptoren besitzen nur zwei Parameter, nämlich die berücksichtigte Patchgrösse und die Bingrössen. Der Parameter 'max_img_res' sollte auf dem Standardwert belassen werden. Dessen Einfluss kann in der Beschreibung zur Localized Feature Histogram Implementierung nachgeschaut werden, für unsere Zwecke ist er eigentlich überflüssig.

Bei den Composed Feature Histograms traten teilweise 'Memory out of Bound' Fehlermeldungen in Matlab auf. Um dies zu verhindern, werden nicht an allen Pixelkoordinaten im zweiten Bild sondern lediglich auf einem regulären Gitter Feature Deskriptoren bestimmt. Knoten in diesem Gitter sind in x- und y-Richtung jeweils um eine halbe Patchlänge voneinander entfernt. Um trotzdem ein durchgehendes Errorbild zu erhalten, werden die erhaltenen Fehler an den Gitterknoten auf einen ganzen Block extrapoliert. Aus diesem Grund können die Ergebnisse zwischen dem Geometric Blur und auf Localized Feature Histogrammen basierten Deskriptoren nur mit Vorsicht verglichen werden. Denn bei Feature Histogrammen kann es passieren, dass ein charakteristischer Kantenzug durch das verwendete reguläre Gitter nicht im Zentrum des Deskriptors zu liegen kommt, was beim Geometric Blur nicht passieren kann, da dort für jeden Bildpixel ein Deskriptor berechnet wird.

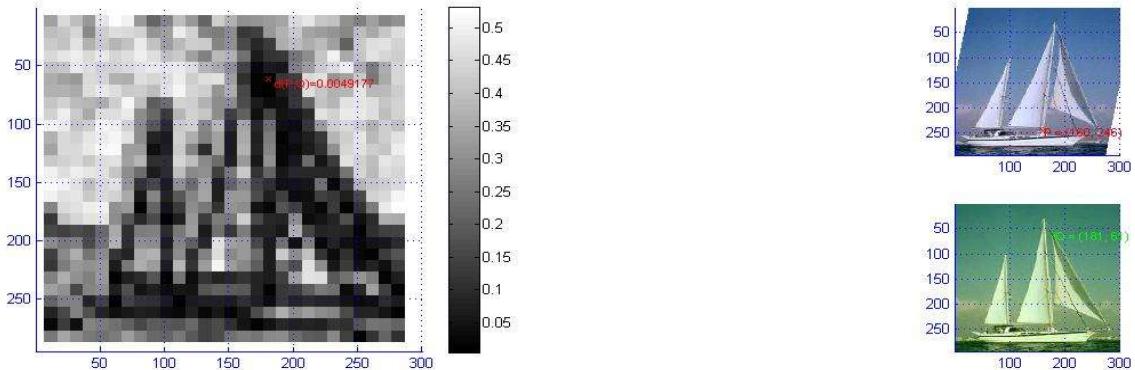
Das folgende Beispiel zeigt einen Error Plot mit Localized Feature Histograms. Es wurden Standardwerte für die Parameter verwendet, die Distanzfunktion ist χ^2 .



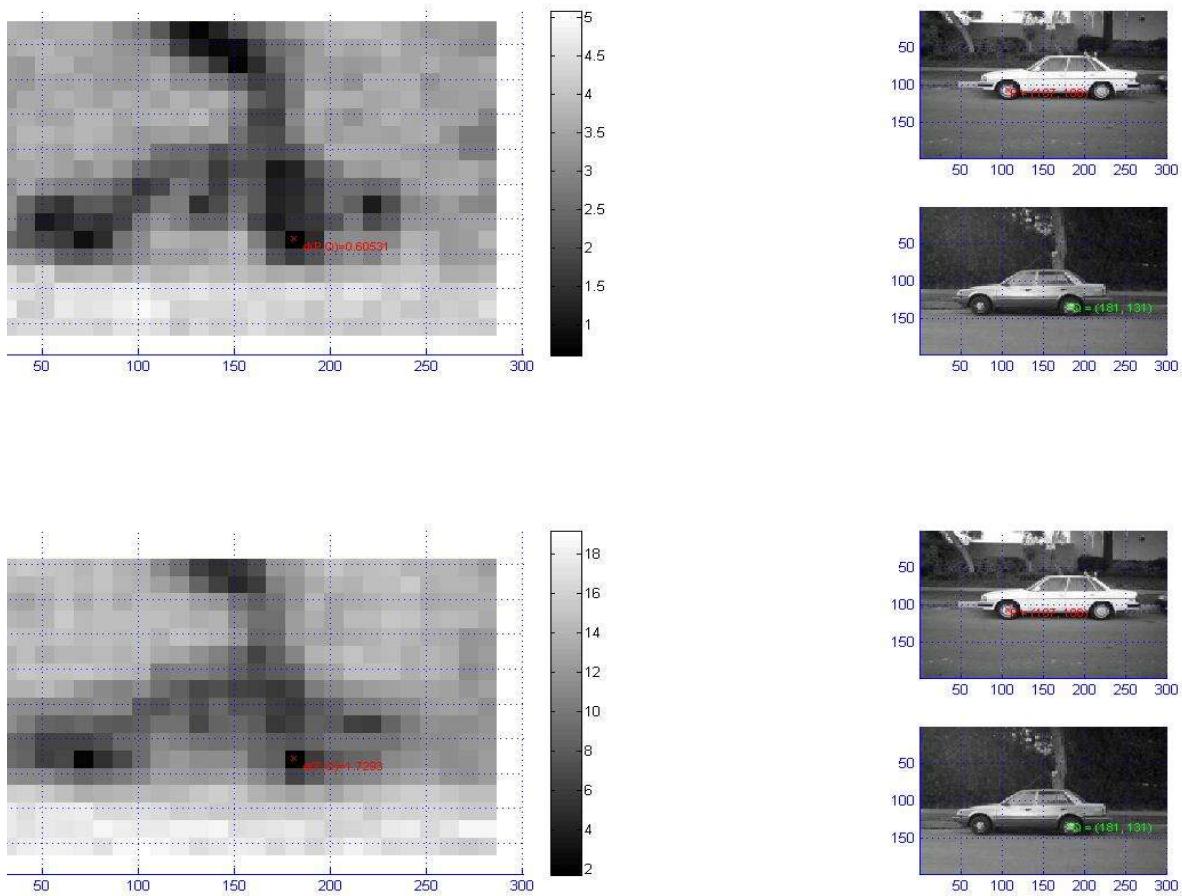
Dasselbe Beispiel mit Composed Localized Feature Histograms:



Selbst folgendes relative einfache Beispiel funktioniert mit dem Low Dimensional Feature Histogram Detektor nicht.
Als Distanzfunktion wurde die Korrelation verwendet.



Auch hier wieder ein Beispiel für die interklass Variation. Für das erste Beispiel kamen Localized Feature Histogramme zum Zug, für das zweite Composed Localized Feature Histogramme. Alle Parameter wurden jeweils auf dem Standardwert belassen, die Distanzfunktion war χ^2 .



Lediglich das Hinterrad wurde mit dem Vorderrad verwechselt, was aber trotzdem ein sehr gutes Resultat ist, da sich das Vorder- vom Hinterrad lokal betrachtet eigentlich fast nicht unterscheiden.

Kapitel 4: Korrespondenzen

Ein Teil des Optimierungsproblems im Shape Matching ist das Bestimmen von optimalen Punkt-Korrespondenzen zwischen zwei Patterns. In diesem Kapitel wird das Finden einer optimalen Korrespondenz unabhängig von der Suche nach einer optimalen Transformation betrachtet. Die Erkenntnisse dieses Kapitels lassen sich aber auch auf die referenzierten Verfahren übertragen, die eine gemeinsame Optimierung dieser zwei Elemente durchführen. Denn wie später seien beschrieben wird, wird die gemeinsame Optimierung durch zwei einzelne, jeweils sequentiell repeteierte Schritte erreicht.

4.1 Distanz zwischen Feature Deskriptoren

Die Verwendung der im zweiten Kapitel herausgearbeitete Distanz- oder Energiefunktion verlangt die Definition eines Distanzmasses c zwischen den beiden Punkten eines Korrespondenzenpaars. Dazu werden meist Feature Deskriptoren und Pixelkoordinaten dieser Punkte verglichen. Dies wiederum macht die Wahl einer Distanzfunktion zwischen Feature Deskriptoren nötig. Die meisten Feature Deskriptoren implizieren jedoch eine bestimmte Distanzfunktion, sodass eigentlich keine Wahlmöglichkeit bestehen bleibt.

Sollen die Feature Deskriptoren eine Wahrscheinlichkeitsverteilung mittels Histogrammen approximieren, wie zum Beispiel im Falle der Localized Feature Histograms in [OB05] oder der Shape Contexts in [BMP02], so ist die natürlichste Wahl der Distanzfunktion zwischen zwei Histogrammen h_i und h_j die χ^2 Test-Statistik:

$$\chi^2(h_i, h_j) = \frac{1}{2} \sum_{k=1}^{\text{#Bins}} \frac{(h_i[k] - h_j[k])^2}{h_i[k] + h_j[k]}$$

In [BBM05] wird als Distanzfunktion die negative Korrelation verwendet, welche auch als Auswahlmöglichkeit im Feature Deskriptor Vergleich zu finden ist:

$$\text{negCorr}(fd_i, fd_j) = 1 - \frac{\text{corr}_{\text{Pearson}}(fd_i, fd_j) + 1}{2} \quad \text{wobei:}$$

fd_i : Feature Deskriptor im i-ten Punkt
 fd_j : Feature Deskriptor im j-ten Punkt
 $\text{corr}_{\text{Pearson}}$: Pearson Korrelation

Die Korrelation misst bekanntlich den Grad der linearen Abhängigkeit zwischen zwei Zufallsvariablen, oder im empirischen Fall zwischen Realisierungen dieser Zufallsvariablen. Die oben definierte Distanzfunktion ist somit klein, wenn eine hohe positive Korrelation besteht. Es ist aber zu beachten, dass eine Korrelation von Null nicht impliziert, dass die zwei Zufallsvariablen unabhängig voneinander sind! Dieses Distanzmass ist somit mit Vorsicht zu geniessen. Zudem entsprechen negative Korrelationen einer grossen Distanz, was auch fraglich sein kann. Denn eine hohe negative Korrelation lässt auf eine hohe Abhängigkeit schliessen.

Passt keine andere Distanzfunktion, so ist die Summe der Quadrate (SSD: Sum of squared distances) meist die Standardwahl. Ausser im euklidischen Raum ist diese Distanzfunktion meist eine ziemlich schlechte Wahl. Trotzdem wird sie oft verwendet, meist weil keine besseren Alternativen zur Verfügung stehen. Im RGB Farbraum wird sie zum Beispiel oft verwendet, obwohl keine theoretische Fundierung dafür existiert.

4.2 Verschiedene Arten von Korrespondenzen

Ein wichtiger Aspekt ist, über welche Klasse von Korrespondenzen optimiert wird. Die beiden Patterns müssen nicht aus exakt gleich viele Punkten aufgebaut sein, da zur Steigerung der Robustheit sowieso eine bestimmte Anzahl an „Outlier-Partner“ in die Patterns eingefügt werden. Punkte des anderen Patterns können nun mit einem dieser Outlier-Partner in Beziehung gesetzt werden, ev. unter Bestrafung durch einen konstanten Penaltybetrag. Mittels dieser Outlier-Partner können nun beide Patterns auf dieselbe Punkteanzahl gebracht werden.

Zwei Haupttypen an Korrespondenzen können unterschieden werden: Hard- und Softassignments.

4.2.1 Hardassignments

Hardassignments können durch eine binäre Matrix $M \in \{0, 1\}^{n \times m}$ dargestellt werden, wobei $m_{ij} = 1$ falls der i-te Punkt des ersten dem j-ten Punkt des zweiten Pattern zugeordnet wird und $m_{ij} = 0$ sonst.

Bei Hardassignments ist eine grundlegende Entscheidung, ob 1-to-1, 1-to-many, many-to-1 oder many-to-many Korrespondenzen gesucht werden. Diese Bezeichnungen sagen aus, wie viele Korrespondenzpartner ein Punkt des ersten (zweiten) Patterns im zweiten (ersten) Pattern haben kann. Eine 1-to-many Korrespondenz verlangt also zum Beispiel, dass jedem Punkt des ersten Patterns exakt einem Punkt des zweiten Patterns zugeordnet wird, die Anzahl an Punktpartner eines Punktes im zweiten Pattern aus dem ersten Pattern ist aber gleichgültig. Im Falle von 1-to-1 Korrespondenzen, entspricht eine Korrespondenz also einer Permutation.

Die Korrespondenzsuche wird meist als Optimierungsproblem mit Nebenbedingungen formuliert. Ein oft gewählter Ansatz versucht die Summe der Distanzen zwischen den Partnern zu minimieren, unter der Nebenbedingung der korrekten Klassenzugehörigkeit der Korrespondenzrelation. Diese Nebenbedingungen können einfach als Bedingungen an die Zeilen- und Kolonnensummen der Matrix M formuliert werden. Im Falle von 1-to-1 Korrespondenzen resultiert daraus ein weighted bipartite matching Problem, was in $O(N^3)$ gelöst werden kann.

Eine Optimierung über einen Integerbereich $[0, 1]^{n \times m}$, auch bekannt als 0-1 Integer Programming, ist viel komplexer als eine Optimierung über den reellen Bereich $[0, 1]^{n \times m}$. Ein Vorgehen besteht nun darin, die integerwertige Nebenbedingung $M \in [0, 1]^{n \times m}$ in eine reellwertige Nebenbedingung $M \in [0, 1]^{n \times m}$ zu relaxieren. Unter bestimmten Voraussetzungen (nämlich falls die Matrix der Nebenbedingungen des Integer Programming Problems total unimodular und die rechte Seite der Nebenbedingungen ebenfalls ganzzahlig sind) kann die so gefundene reellwertige optimale Lösung M_{real} in eine optimale ganzzahlige Lösung M_{integer} umgewandelt werden, ohne dass sich der Wert der Zielfunktion ändert. Dies ist für 1-to-1 Korrespondenzen der Fall.

4.2.2 Softassignments

All diese durch Integer Programming verursachten Schwierigkeiten können umgangen werden, wenn reellwertige Assignments $M \in [0, 1]^{n \times m}$ von Beginn an zugelassen werden. Diesen so genannten Softassignments kann zudem eine natürliche Interpretation zugeordnet werden: m_{ij} entspricht der Wahrscheinlichkeit, dass, gegeben der i-te Punkt des ersten Pattern, dieser i-te Punkt dem j-ten Punkt des zweiten Patterns zugeordnet ist. Diese bedingten Wahrscheinlichkeiten müssen sich natürlich auf Eins summieren: wird eine soft 1-to-1 Korrespondenz gesucht, so muss verlangt werden, dass

$$\sum_{i=1}^n m_{ij} = 1 \quad \text{und} \quad \sum_{j=1}^m m_{ij} = 1 ,$$

wird jedoch eine soft 1-to-many Korrespondenz gesucht, so reicht die Forderung

$$\sum_{j=1}^m m_{ij} = 1$$

aus.

Softassignments haben zudem den Vorteil, dass sie einem nicht zu einer „Alles-oder-Nichts“ Entscheidung bei der Wahl der Korrespondenzpartners zwingen. Es ist gut möglich, dass mehrere Kandidaten als Partner für einen Punkt etwa gleich gut in Frage kommen. Eine binäre Entscheidung zu fällen ist fehleranfällig und vernachlässigt unter Umständen auch Informationen.

Die Idee der soft 1-to-1 Korrespondenzen wird in [CR00] unter Verwendung von Deterministic Annealing verfolgt. Auf diese Methode wird später noch näher eingegangen, da sie, obwohl erst nach dem Entwickeln des in dieser Semesterarbeit verfolgten Ansatzes entdeckt, jener erstaunlich ähnlich ist.

4.3 k nächste Nachbarn

In dieser Semesterarbeit wird eine neue Variante der Korrespondenzsuche untersucht. Anstatt irgendwelche Optimierungen mit Nebenbedingungen zu lösen, werden jedem Punkt des ersten Patterns die k nächsten Nachbarn der Punkte des zweiten Pattern bezüglich der Distanzfunktion c zugeordnet. Die berücksichtigten Korrespondenzen können also als hard k-to-many bezeichnet werden. Im Folgenden bezeichnet $\sigma(i, j)$ wiederum den Index des j-nächsten Nachbarn im zweiten Pattern des i-ten Punktes des ersten Pattern.

Nachdem zu jedem Punkt des ersten Patterns p_1 die k nächsten Nachbarn des zweiten Patterns p_2 gefunden sind, können diese Korrespondenzen $\sigma(i, j)$ basierend auf ihrer Distanz gewichtet werden. Dazu wird ihre Distanz als Energie interpretiert und mittels einer Gibbs-Verteilung in eine Wahrscheinlichkeit umgewandelt. Dabei spielt die verwendete

Temperatur T eine wichtige Rolle:

$$\tilde{w}_{i,\sigma(i,j)} = \exp\left(-\frac{1}{T} c(p_1[i], p_2[\sigma(i,j)])\right)$$

Je grösser T ist, umso uniformer wird die daraus resultierende Wahrscheinlichkeitsverteilung. Die gewichteten Korrespondenzen werden so normalisiert, dass sie zu Eins summieren und somit einer Wahrscheinlichkeitsverteilung entsprechen:

$$\tilde{w}_{i,\sigma(i,j)} = \frac{\tilde{w}_{i,\sigma(i,j)}}{\sum_{i=1, j=1}^{n,k} \tilde{w}_{i,\sigma(i,j)}}.$$

Diese gewichteten k nächsten Nachbarn entsprechen somit wiederum eher einer soft Korrespondenz.

Eine andere Möglichkeit wäre, die Gewichte

$$\tilde{w}_{i,\sigma(i,j)}$$

jeweils für jeden Punkt i des ersten Patterns nur bezüglich der Punkte j des zweiten Patterns zu normalisieren:

$$\tilde{w}_{i,\sigma(i,j)} = \frac{\tilde{w}_{i,\sigma(i,j)}}{\sum_{j=1}^k \tilde{w}_{i,\sigma(i,j)}}.$$

Dadurch könnte eine soft 1-to-many Korrespondenz erreicht werden. Das Problem mit einer solchen Normalisierung besteht darin, dass Punkten i', zu welchen nur schlecht passende Nachbarn gefunden werden können (d.h. $\sum_{j=1}^k \tilde{w}_{i,\sigma(i,j)}$ ist klein) zu viel Bedeutung zugemessen wird. Es müsste eine Vorverarbeitung durchgeführt werden, welche diese Outlier aussortiert. Doch dies erschwert den Prozess erheblich, weshalb darauf verzichtet wurde.

Kapitel 5: Affine Transformation

Ist eine optimale oder näherungsweise optimale Korrespondenz zwischen den Punkten des ersten und des zweiten Patterns bekannt, so kann ausgehend von dieser eine Transformation geschätzt werden, welche die Punkte des ersten Patterns auf die korrespondierenden Punkte des zweiten Patterns abbildet. Dieses Problem ist so natürlich schlecht gestellt, denn es existieren unendlich viele Lösungstransformationen f . Um dies zu verhindern wird a priori Wissen über die zu schätzende Transformation f verwendet. Eine natürliche Annahme ist, dass f „glatt“ ist. Diese Glattheit kann auf verschiedene Arten quantitativ definiert werden, wodurch jeweils eine andere optimale Transformationen gefunden wird. Diese Glattheitsannahme wurde bereits zuvor angesprochen, denn sie lässt die Extrapolation der Transformation zwischen den zwei Patterns vom ganzen Definitionsbereich in den ganzen Bildbereich zu: wären nicht glatte Transformationen oder sogar unstetige Transformationen möglich, so wäre der Schluss von den diskreten Punkten der Patterns auf den gesamten Definitionsbereich und Bildbereich der Bilder sicherlich nicht möglich.

Nebst den im nächsten Kapitel beschriebenen Thin Plate Splines stellen affine Transformationen einen zentralen Bestandteil des Algorithmus dar. Affine Transformationen sind natürlich glatt, da sie überhaupt keine Krümmung enthalten. Zudem sind affine Transformationen als Spezialfall in der Thin Plate Spline Funktionsklasse enthalten.

Die Schätzung einer guten affinen Transformation ist essentiell für die weiteren Schritte im Algorithmus. Denn sie wird benutzt, um zwei der drei Terme in der Distanzfunktion der Iterationsphase des Algorithmus zu initialisieren. Als Input für die Schätzung dienen die Koordinaten der zuvor berechneten Korrespondenzpaare und deren Gewichte.

In diesem Kapitel wird zuerst eine kurze mathematische Einführung in affine Transformationen gegeben und zudem die verwendete Notation erklärt. Danach wird die entwickelte Methode zur Schätzung einer affinen Transformation vorgestellt.

5.1 Mathematische Einführung in affine Transformationen

Die hier gegebene Einführung fasst die wichtigsten, für diese Arbeit relevanten Aspekte von affinen Transformationen zusammen. Einen etwas tieferen Einblick gibt zum Beispiel [G00], woran sich auch diese Einführung stark orientiert. Zuerst muss definiert werden, was unter einem affinen Raum verstanden wird:

Definition: Ein affiner Raum ist ein Tripel

$$\begin{aligned} & \langle E, \vec{E}, + \rangle \text{ wobei:} \\ & E: \text{nichtleere Menge} \\ & \vec{E}: \text{Vektorraum} \\ & +: E \times \vec{E} \rightarrow E \end{aligned}$$

Dabei muss die Funktion $+$ folgende Axiome erfüllen:

$$\begin{aligned} & \forall a \in E: a + \vec{0} = a \\ & \forall a \in E: \forall \vec{u}, \vec{v} \in \vec{E}: (a + \vec{u}) + \vec{v} = a + (\vec{u} + \vec{v}) \\ & \forall a, b \in E: \exists \vec{u} \in \vec{E}: a + \vec{u} = b \wedge \vec{u} \text{ ist eindeutig} \end{aligned}$$

Der eindeutige Vektor \vec{u} sodass $a + \vec{u} = b$ wird auch als \vec{ab} geschrieben.

Das zweite wichtige Konzept für diese Arbeit ist dasjenige eines affinen Frames.

Definition: Gegeben sei ein affiner Raum $\langle E, \vec{E}, + \rangle$. Ein affiner Frame mit Ursprung a_0 ist ein Paar $(a_0, (\overrightarrow{a_0a_1}, \dots, \overrightarrow{a_0a_m})) \in E \times \vec{E}^m$, sodass $(\overrightarrow{a_0a_1}, \dots, \overrightarrow{a_0a_m})$ eine Basis von \vec{E} ist. Ein affiner Frame $(a_0, (\overrightarrow{a_0a_1}, \dots, \overrightarrow{a_0a_m})) \in E \times \vec{E}^m$ kann auch mittels Matrixnotation geschrieben werden:
 $(a_0, (\overrightarrow{a_0a_1}, \dots, \overrightarrow{a_0a_m})) \equiv (a_0, A)$ wobei: $A = [\overrightarrow{a_0a_1} \dots \overrightarrow{a_0a_m}] \wedge A$ hat vollen Kolonnenrang

Aus der linearen Algebra folgt nun, dass, gegeben ein affiner Frame $(a_0, (\overrightarrow{a_0a_1}, \dots, \overrightarrow{a_0a_m})) = (a_0, A)$, jedes $x \in E$ eine eindeutige Repräsentation als Koordinatenvektor $(x_1 \dots x_m)^T$ bezüglich diesem affinen Frame besitzt:

$$x = a_0 + \underbrace{\overrightarrow{a_0 x}}_{\in \vec{E} \equiv \{z | z = A y \wedge y \in R^m\}} = a_0 + x_1 \overrightarrow{a_0 a_1} + \dots + x_m \overrightarrow{a_0 a_m} = a_0 + A \begin{pmatrix} x_1 \\ \dots \\ x_m \end{pmatrix}$$

Aus dieser Koordinatenrepräsentation können nun einfach die eindeutigen baryzentrischen Koordinaten

$$\vec{\lambda}: \sum_{i=0}^m \lambda_i = 1$$

von x bezüglich dem affinen Frame $(a_0, (\overrightarrow{a_0 a_1}, \dots, \overrightarrow{a_0 a_m}))$ hergeleitet werden:

$$x = a_0 + \sum_{i=1}^m x_i \overrightarrow{a_0 a_i} = \left(1 - \sum_{i=1}^m x_i\right) a_0 + \sum_{i=1}^m x_i a_i = \sum_{i=0}^m \lambda_i a_i \quad \text{wobei:}$$

$$\lambda_0 = 1 - \sum_{i=1}^m x_i \quad \text{und} \quad \forall i \in \{1, \dots, m\}: \lambda_i = x_i$$

Eine Linearkombination in einem Vektorraum entspricht also einer baryzentrischen Kombination in einem affinen Raum!

Das letzte benötigte Konzept ist nun die affine Transformation selbst.

Definition: Gegeben seien zwei affine Räume $\langle E, \vec{E}, + \rangle$ und $\langle E', \vec{E}', +' \rangle$. Eine Funktion $f: E \rightarrow E'$ ist eine affine Abbildung (oder affine Transformation), falls

$$\forall \lambda_i \in \mathbb{R}, a_i \in E: \sum_i \lambda_i = 1 \rightarrow f\left(\sum_i \lambda_i a_i\right) = \sum_i \lambda_i f(a_i)$$

Eine affine Abbildung bewahrt also Barycenter. Aus diesen Kenntnissen folgt nun die entscheidende Folgerung. Sie wird hier nicht bewiesen, interessierte Leser werden auf [G00] verwiesen.

Satz: Für jede affine Abbildung $f: E \rightarrow E'$ existiert eine eindeutige lineare Abbildung $\vec{f}: \vec{E} \rightarrow \vec{E}'$ sodass:

$$\forall a \in E: \forall \vec{v} \in \vec{E}: f(a + \vec{v}) = f(a) +' \vec{f}(\vec{v})$$

Umgekehrt gilt für jede lineare Abbildung $\vec{h}: \vec{E} \rightarrow \vec{E}'$ und jedes beliebige $a \in E, b \in E'$, dass

$$f: E \rightarrow E': f(a + \vec{v}) = b +' \vec{h}(\vec{v})$$

eine affine Abbildung ist.

Aus diesem Satz ist einfach ersichtlich, dass eine affine Abbildung nichts anderes als eine lineare Abbildung mit anschliessender Addition +' ist. Diese Addition wird normalerweise natürlich Translation genannt.

Eine lineare Abbildung kann bekanntlich mittels einer Matrix-Vektor Multiplikation ausgedrückt werden. Verwendet man homogene Koordinaten, so kann man eine affine Abbildung ebenfalls als Matrix-Vektor Multiplikation formulieren. Die zugrundeliegenden Räume E und E' werden um eine Dimension erweitert und es wird verlangt, dass die zusätzlichen Koordinaten der Koordinatenrepräsentation gleich Eins sind. Es wird also in einer $n+1$ dimensionale Hyperebene gearbeitet.

$$f: E \rightarrow E': f\left(\begin{array}{c} x \\ \vdots \\ a \\ + \\ \vec{v} \end{array}\right) = b +' \vec{h}(\vec{v}) \equiv t +' A \vec{u} = \begin{bmatrix} A_{2 \times 2} & t \\ O & 1 \end{bmatrix} \begin{pmatrix} \vec{u} \\ 1 \end{pmatrix} \quad \text{wobei:}$$

t : Koordinatenrepräsentation von b in einem affinen Frame $(a_0, (\overrightarrow{a_0 a_1}, \dots, \overrightarrow{a_0 a_m})) \in E' \times \vec{E}'^m$.

u : Koordinatenrepräsentation von \vec{v} in \vec{E}

$A_{2 \times 2}$: Lineare Transformation der Koordinaten von \vec{E} nach \vec{E}'

Die letzte Koordinate der Koordinatendarstellung im Raum E' bleibt im Rahmen dieser Arbeit immer gleich Eins, sie kann darum weggelassen werden (für projektive Abbildungen würde sie jedoch eine wichtige Rolle spielen).

In dieser Arbeit wird lediglich eine zweidimensionale affine Abbildung

$$f: UV \subset \mathbb{R}^2 \rightarrow XY \subset \mathbb{R}^2: f(a + \vec{v}) \equiv t + A\vec{u} = \begin{bmatrix} x_u & x_v & t_x \\ y_u & y_v & t_y \end{bmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} A_{2 \times 2} & , & \begin{pmatrix} t_x \\ t_u \end{pmatrix} \end{bmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}$$

benötigt, zudem wird implizit das affine Frame

$$\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \left(\begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) \right)$$

im affinen Raum $\langle E, \vec{E}, + \rangle$ gewählt. Eine solche affine Abbildung besitzt somit sechs Freiheitsgrade $(x_u, x_v, t_x, y_u, y_v, t_y)$. Im nächsten Unterkapitel wird beschrieben, wie über exakt diese sechs Freiheitsgrade optimiert wird.

Um zu verstehen, wie eine bestimmte affine Abbildung den einen Raum abbildet, kann die Untermatrix $A_{2 \times 2}$ analysiert werden. Sie bildet die lineare Abbildung der affinen Abbildung und somit umfasst sie Skalierungen, Scherungen, Rotationen, Spiegelungen, etc. Der Vektor t in der letzten Kolonne der affinen Abbildungsmatrix entspricht einer Translation um eben diesen Vektor.

5.2 Schätzung der affinen Transformation

Das Schätzen der affinen Transformationen wird als Optimierungsproblem formuliert. Mittels den zuvor berechneten Gewichten der Korrespondenzen kann ein Weighted Linear Least Squares Problem formuliert werden:

$$A^* = \arg \min_A \sum_{i=1}^n \sum_{j=1}^k w_{i,\sigma(i,j)} \left\| \begin{pmatrix} x_{\sigma(i,j)} \\ y_{\sigma(i,j)} \end{pmatrix} - A \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} \right\|^2$$

Dabei werden homogene Koordinaten verwendet und die sigma-Notation soll die gefundenen Korrespondenz darstellen. Durch Lösen obigen Problems zeigte sich aber, dass oft eine optimale Transformation gefunden wurde, welche eine sehr starke Kontraktion des ersten Bildes in die Mitte des zweiten Bildes beinhaltete. Um dies zu vermeiden wurde in einem ersten Versuch versucht, einen Penaltyterm zur Flächenerhaltung in die Zielfunktion einzubauen:

$$A^* = \arg \min_A \left\{ \sum_{i=1}^n \sum_{j=1}^k w_{i,\sigma(i,j)} \left\| \begin{pmatrix} x_{\sigma(i,j)} \\ y_{\sigma(i,j)} \end{pmatrix} - A \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} \right\|^2 + \lambda \left((\det(A_{2 \times 2}))^2 - \text{ratio} \right)^2 \right\}$$

wobei: $\det(A_{2 \times 2})$ = Determinante der Untermatrix von A ohne Translationsteil

Bekanntlich beschreibt der Betrag der Determinante der Jacobimatrix J einer Funktion f die relative Flächenänderung zwischen dem Definitions- und Bildbereich von f:

$$f: U \rightarrow X: dV_X = \sqrt{\det(J(f)^T J(f))} dV_U \underset{\text{für } J \in \mathbb{R}^{n \times n} \text{ und J regulär}}{=} |\det(J(f))| dV_U$$

Die Jacobimatrix einer affinen Abbildung entspricht der Untermatrix $A_{2 \times 2}$ da die letzte Koordinate konstant gleich 1 ist. Die 2×2 Unterdeterminante $A_{2 \times 2}$ des linearen Anteils der affinen Transformation ist quadratisch und, abgesehen von Spezialfällen, auch regulär. Die vereinfachte zweite Formel kann also zur Flächenberechnung benutzt werden. Nähme man direkt den Betrag der Determinante in der Zielfunktion, so würde die Zielfunktion nicht stetig ableitbar sein. Darum wurde das Quadrat der Determinanten verwendet. Der Parameter 'ratio' steuert die gewünschte Vergrößerung oder Verkleinerung der Abbildung. Für möglichst flächenerhaltende Abbildungen ist er demnach gleich 1. Um sowohl Vergrößerungen, als auch Verkleinerungen zuzulassen wird diese Differenz ins Quadrat gesetzt. Dadurch bleibt die Zielfunktion stetig ableitbar. Es ist also leicht ersichtlich, dass der Penaltyterm gleich Null ist, falls die Abbildung Flächen im Verhältnis 'ratio' verzerrt, ansonsten jedoch um so grösser ist, je mehr die Flächenänderung von 'ratio' variiert.

Die Ableitung der Zielfunktion ist nun durch den zusätzlichen Term nicht mehr linear und muss iterativ gelöst werden. Es zeigte sich aber auch hier, dass dieser Ansatz unzureichend war. Der Parameter λ war nicht allgemein gültig einzustellen: um die Kontraktion zu verhindern musste er so gross gewählt werden, dass der eigentliche Datenterm zu viel an Einfluss verlor.

Eine andere Möglichkeit, aber immer noch im Framework der Optimierungstheorie, besteht darin, anstelle eines Penaltyterms zur Flächenerhaltung die Flächenerhaltung als Nebenbedingung zu formulieren. Natürlich darf die Flächenerhaltung nicht strikt gefordert werden, sondern eher mit einer oberen und unteren Schranke an die Unterdeterminante. Es ergibt sich also folgendes Optimierungsproblem mit Nebenbedingungen, wobei α ein positiver, zuvor festgelegter Parameter ist (meist verwendet wurde $\alpha = 0.2$):

$$A^* = \min_A \sum_{i=1}^n \sum_{j=1}^k w_{i,\sigma(i,j)} \| \begin{pmatrix} x_{\sigma(i,j)} \\ y_{\sigma(i,j)} \end{pmatrix} - A \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} \|^2$$

sodass:

$$\begin{aligned} |\det(A_{2 \times 2})| &< \text{ratio} + \alpha \\ |\det(A_{2 \times 2})| &> \text{ratio} - \alpha \end{aligned}$$

Da in der verwendeten Bilddatenbank alle Bilder etwa dieselbe Grösse aufweisen, wurde ratio auf 1 gesetzt. Da Bilder in derselben Kategorie eventuell gespiegelt oder rotiert sind, wird die Optimierungsroutine mit verschiedenen Startpunkten aufgerufen: der Identität, Spiegelung in x-Richtung, Spiegelung in y-Richtung, Spiegelung in x- und y-Richtung (= Rotation um 180°) und Rotationen um +90° und -90°. Um gute Startpunkte für die Lösungsmethode zu finden, wird die mittlere Verschiebung der Korrespondenzpartner zuvor geschätzt und in die Startlösungen integriert. Zudem wird diese geschätzte Verschiebung für jede der verschiedenen Startlösungen mehrmals zufällig leicht verschoben.

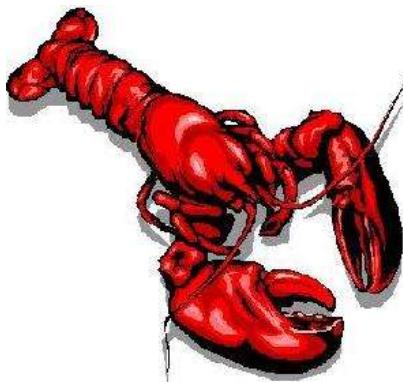
$$A_0 = \begin{bmatrix} +1 & 0 & t_x + N(0, 10) \\ 0 & +1 & t_y + N(0, 10) \\ -1 & 0 & t_x + N(0, 10) \\ 0 & -1 & t_y + N(0, 10) \end{bmatrix} \vee \begin{bmatrix} -1 & 0 & t_x + N(0, 10) \\ 0 & +1 & t_y + N(0, 10) \\ 0 & -1 & t_x + N(0, 10) \\ +1 & 0 & t_y + N(0, 10) \end{bmatrix} \vee \begin{bmatrix} +1 & 0 & t_x + N(0, 10) \\ 0 & -1 & t_y + N(0, 10) \\ 0 & +1 & t_x + N(0, 10) \\ -1 & 0 & t_y + N(0, 10) \end{bmatrix} \vee \\ t = \begin{pmatrix} t_x \\ t_y \end{pmatrix} = \sum_{i=1}^n \sum_{j=1}^k w_{i,\sigma(i,j)} \left(\begin{pmatrix} x_{\sigma(i,j)} \\ y_{\sigma(i,j)} \end{pmatrix} - \begin{pmatrix} u_i \\ v_i \end{pmatrix} \right)$$

$N(\mu, \sigma)$: Normalverteilung mit Mittelwert μ und Standardabweichung σ

Von all diesen Versuchen wird dann die Lösung mit dem kleinsten resultierenden Zielfunktionswert gewählt.

Es ist interessant zu bemerken, dass in fast allen Versuchen die zweite Nebenbedingung aktiv war, d.h. die ganze erlaubte Verkleinerung wurde ausgenutzt, was mit der zuvor gemachten Beobachtung übereinstimmt.

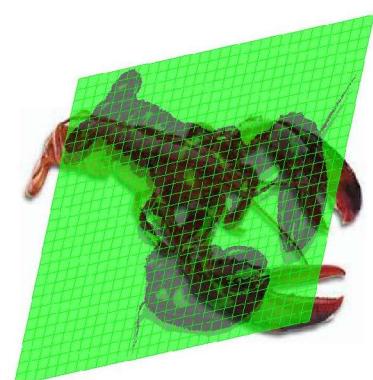
Erstes Bild im UV Bereich



Zweites Bild im XY Bereich



Vom UV in den XY Bereich affin transformiertes erstes Bild.



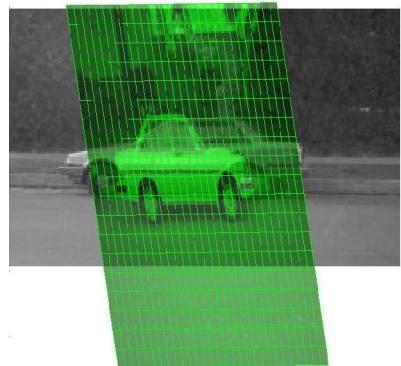
Erstes Bild im UV Bereich



Zweites Bild im XY Bereich



Vom UV in den XY Bereich affin transformiertes erstes Bild.



Kapitel 6: Schätzung der Thin Plate Spline Transformation

In diesem Kapitel wird zunächst eine kurze Einführung in die Regularisationstheorie gegeben. Danach wird auf eine Subklasse der Probleme in der Regularisationstheorie eingegangen, auf Funktionen in durch Kernels generierten Funktionsräumen. Eine spezifische Wahl eines solchen Kernels führt schliesslich zu den Thin Plate Splines, welche einen wichtigen Teil dieser Semesterarbeit ausmachen. Thin Plate Splines erweisen sich schon seit einiger Zeit als nützliches Hilfsmittel im Shape Matching. Das Konzept der Thin Plate Splines ist einfach, es gibt jedoch sehr interessante, weniger triviale Erkenntnisse, welche hier aber lediglich angeschnitten werden. Es wäre sicherlich interessant, diese Erkenntnisse in einer weiterführenden Arbeit näher zu untersuchen.

6.1 Wahrscheinlichkeit und Regularisationstheorie

6.1.1 Modell

Gegeben sei eine Menge an unabhängigen Datenpunkten

$$s = \{(u_i, z_i) \in \mathbb{R}^n \times \mathbb{R}\}_{i=1}^N ,$$

die von einer unbekannten Funktion $f \in H : \mathbb{R}^n \rightarrow \mathbb{R}$ in Gegenwart von Rauschen erhalten wurden. Diese Funktion soll nun anhand dieser Daten gefunden werden. Wählt man einen wahrscheinlichkeitstheoretischen Ansatz, so können die Datenpunkte als auch die Funktion als Realisierungen zweier Zufallsvariablen S und F betrachtet werden. Verwendet man einen Bayes Schätzer, so wird die bedingte Wahrscheinlichkeit $P\{F=f|S=s\} \propto P\{S=s|F=f\} P\{F=f\}$ benötigt. Der einfachste Bayes Schätzer resultiert aus Verwendung der 0-1 Verlustfunktion, was nämlich zum Maximum a Posteriori Schätzer führt:

$$f^* = \arg \max_{f \in H} P\{F=f|S=s\} = \arg \max_{f \in H} (P\{S=s|F=f\} P\{F=f\})$$

Unter der zusätzlichen Annahme von unabhängigen, Gausschem Rauschen lässt sich letztere bedingte Wahrscheinlichkeit folgendermassen schreiben:

$$P\{S=s|F=f\} \propto \exp\left(-\sum_{i=1}^N \frac{1}{2\sigma_i^2} (z_i - f(u_i))^2\right) = \exp\left(-\sum_{i=1}^N w_i (z_i - f(u_i))^2\right)$$

wobei: σ_i : Varianz des Rauschen bzgl. i-tem Datenpunkt

Anstelle von Varianzen werden im letzten Term Gewichte w_i eingeführt, da den Korrespondenz Gewichte zugeordnet werden. Die Interpretationen von Gewichten und Varianzen sind jedoch invers zueinander: ist die Varianz einer Korrespondenz zum Beispiel klein, so ist diese Korrespondenz verlässlich und darum ihr Gewicht entsprechend hoch.

Die Wahl für die Prior-Wahrscheinlichkeit von f wird nun unter Verwendung eines Operators I und einer passenden Norm auf dem Funktionsraum $I[H]$ zu:

$$P\{F=f\} \propto e^{-\lambda \|I[f]\|_H^2}$$

wobei: $\lambda \in \mathbb{R}_+$

Damit erhält man:

$$\begin{aligned} f^* &= \arg \max_{f \in H} P\{F=f|S=s\} = \arg \max_{f \in H} (P\{S=s|F=f\} P\{F=f\}) \\ &= \arg \max_{f \in H} \left(\exp\left(-\sum_{i=1}^N w_i (z_i - f(u_i))^2 - \lambda \|I[f]\|_H^2\right) \right) = \arg \min_{f \in H} \left(\sum_{i=1}^N w_i (z_i - f(u_i))^2 + \lambda \|I[f]\|_H^2 \right) \end{aligned}$$

Diese letzte Funktion entspricht exakt der zu minimierenden Funktion in der Regularisationstheorie!

6.1.2 Reproducing Kernel Hilbert Spaces

Ein Reproducing Kernel Hilbert Space wird wie folgt definiert:

Definition: Sei X ein beliebige begrenzte Menge und H ein Hilbertraum von komplexwertigen Funktionen auf X. H ist genau dann ein Reproducing Kernel Hilbert Space, wenn die Evaluationsfunktionale $F_x[f] = f(x) \quad \forall f \in H$ linear

und begrenzt sind.

Reproducing Kernel Hilbert Spaces sind insofern interessant, als dass zu jedem Reproducing Kernel Hilbert Space eine eindeutige positiv semidefinite Funktion $K(x,y)$, genannt Reproducing Kernel von H , mit folgender Reproducing Property genannten Eigenschaft existiert:

$$\forall f \in H : \langle f(y), K(y, x) \rangle_H = f(x)$$

Dieser Reproducing Kernel entspricht also quasi der Dirac-Funktion des L_2 (aber L_2 ist kein Reproducing Kernel Hilbert Space!).

Wählt man den Hilbertraum als

$$H = \left\{ f \mid f(x) = \sum_{n=0}^{\infty} c_n \phi_n(x) \right\} \text{ wobei: } \phi_n(x) \text{ linear unabhängige Funktionen}$$

und definiert das Skalarprodukt als

$$\langle \sum_{n=0}^{\infty} a_n \phi_n(x), \sum_{n=0}^{\infty} d_n \phi_n(x) \rangle_H \equiv \sum_{n=0}^{\infty} \frac{a_n d_n}{\lambda_n},$$

dann kann auf folgende Weise ein Reproducing Kernel auf H konstruiert werden:

$$K(x, y) \equiv \sum_{n=0}^{\infty} \lambda_n \phi_n(x) \phi_n^*(y) \text{ wobei: } \lambda_n \in \mathbb{R}_+ \text{ und } \sum_{n=0}^{\infty} \lambda_n^2 < \infty.$$

Durch die Wahl des Skalarprodukts wird natürlich auch eine Norm auf dem Hilbertraum impliziert. Dass dieser Kernel die Reproducing Property besitzt, kann durch einfaches Einsetzen überprüft werden:

$$\langle f(y), K(y, x) \rangle_H = \sum_{n=0}^{\infty} \frac{a_n \lambda_n \phi_n(x)}{\lambda_n} = \sum_{n=0}^{\infty} a_n \phi_n(x) = f(x)$$

Dieser Zusammenhang zwischen Reproducing Kernel Hilbert Space und Kernel Funktion gilt auch umgekehrt: während die Wahl eines Reproducing Kernel Hilbert Spaces eine eindeutig bestimmte und wie oben definierte Form besitzende Kernelfunktion impliziert, lässt sich zu jeder Funktion K der obigen Form einen Reproducing Kernel Hilbert Space konstruieren.

Die Dimension des Reproducing Kernel Hilbert Space wird als die Anzahl der Basisfunktionen $\phi_n(x)$ definiert. Es existieren unendlich dimensionale Reproducing Kernel Hilbert Spaces (z.B. durch die Wahl $\phi_n(x) = e^{inx}$), als auch endlich dimensionale. In letzterem Fall werden alle unendlichen Summen durch endliche Summen ersetzt.

6.1.3 Vom Unendlichen zum Endlichen: Die Kernel Property

Beschränkt man sich bei der Priorwahl für $P\{F=f\} \propto e^{-\lambda \|I[f]\|_H^2}$ auf eine semi-Norm definiert auf einem Reproducing Kernel Hilbert Space, so vereinfacht sich die zu minimierende Funktion

$$f^* = \arg \min_{f \in H} \left(\sum_{i=1}^N w_i (z_i - f(u_i))^2 + \lambda \|I[f]\|_H^2 \right) = \arg \min_{\{c_n\}_{0}^{\infty}} \left(\sum_{i=1}^N w_i \left(z_i - \sum_{n=0}^{\infty} c_n \phi_n(u_i) \right)^2 + \lambda \sum_{n=0}^{\infty} \frac{c_n^2}{\lambda_n} \right)$$

enorm. Wie zum Beispiel im Appendix A von [GJP95] in einer einfachen Beweisskizze gezeigt wird, reduziert sich die Lösung des obigen Optimierungsproblems nämlich von einer unendlichen Anzahl an Koeffizienten c_n auf eine endliche Anzahl! Diese Eigenschaft ist unter dem Namen Kernel Property bekannt. Es wird weiter gezeigt, dass die Form der Lösungsfunktion folgende Form hat:

$$f(u)^* = \sum_{i=1}^N c_i K(u, u_i) + \sum_{i=1}^k a_i \psi_i(u) \text{ wobei: } \{\psi_i(u)\}_{i=1}^k \text{ eine Basis des Nullraums } H_0 = \{f \mid \|f\|_H^2 = 0\}$$

Der Nullraum H_0 besteht also aus allen Funktionen, welche gemessen in der semi-Norm Null ergeben. Die semi-Norm lässt sich nun mit Hilfe des Kernels sehr einfach ausdrücken:

$$\begin{aligned}
 \|f\|_H^2 &= \left\| \sum_{i=1}^N c_i K(\cdot, u_i) + \sum_{i=1}^k a_i \psi_i(\cdot) \right\|_H^2 = \left\langle \sum_{i=1}^N c_i K(\cdot, u_i) + \sum_{i=1}^k a_i \psi_i(\cdot), \sum_{j=1}^N c_j K(\cdot, u_j) + \sum_{j=1}^k a_j \psi_j(\cdot) \right\rangle_H \\
 &= \sum_{i=1}^N \sum_{j=1}^N \langle c_i K(\cdot, u_i), c_j K(\cdot, u_j) \rangle_H + \sum_{i=1}^N \sum_{j=1}^k \langle c_i K(\cdot, u_i), a_j \psi_j(\cdot) \rangle_H \\
 &\quad + \sum_{i=1}^k \sum_{j=1}^N \langle a_i \psi_i(\cdot), c_j K(\cdot, u_j) \rangle_H + \sum_{i=1}^k \sum_{j=1}^k \langle a_i \psi_i(\cdot), a_j \psi_j(\cdot) \rangle_H \\
 &= \sum_{i=1}^N \sum_{j=1}^N c_i c_j K(u_i, u_j) = \mathbf{c}^T \mathbf{K} \mathbf{c}
 \end{aligned}$$

Die zweite Zeile folgt aus der Linearität des Skalarprodukts, die letzte Zeile folgt durch die Reproducing Property der Kernelfunktion und durch die Orthogonalität von H und H_0 . Der Prior $P\{F=f\} \propto e^{-\lambda \|I[f]\|_H^2}$ für Funktionen aus einem Reproducing Kernel Hilbert Space lässt sich somit als einfache Quadratische Form des positiv-semidefiniten Kernels \mathbf{K} schreiben! Der Eintrag K_{ij} der Kernelmatrix entspricht $K(u_i, u_j)$. Zusammenfassend erhält man also folgendes Linear Least Squares Problem für die unbekannten Parameter:

$$\begin{aligned}
 f^* &= \arg \min_{f \in H} \left(\sum_{i=1}^N w_i (z_i - f(u_i))^2 + \lambda \|I[f]\|_H^2 \right) \\
 &= \arg \min_{\{c_n\}_0^\infty} \left(\sum_{i=1}^N w_i \left(z_i - \left(\sum_{j=1}^N c_j K(u_i, u_j) + \sum_{j=1}^k a_j \psi_j(u_i) \right) \right)^2 + \lambda \mathbf{c}^T \mathbf{K} \mathbf{c} \right)
 \end{aligned}$$

6.1.4 Thin Plate Spline Basis Funktionen

Es ist intuitiver, eine (semi-) Norm mit gewünschten Eigenschaften auf dem Reproducing Kernel Hilbert Space zu definieren, als direkt eine Kernelmatrix \mathbf{K} zu definieren. Das Problem beim Definieren einer Norm besteht jedoch darin, die der Norm zugrunde liegende Kernelfunktion zu finden.

Geht man lediglich von radialen Basisfunktionen aus, so kann die Kernelfunktion folgendermassen geschrieben werden ([EPP00]):

$$K(x, y) \equiv \sum_{n=0}^{\infty} \lambda_n \phi_n(x) \phi_n(y) \equiv \sum_{n=0}^{\infty} \lambda_n e^{i 2 \pi n x} e^{-i 2 \pi n y}$$

Jeder positiv definite radiale Kernel definiert somit einen Reproducing Kernel Hilbert Space über $[0,1]$. Dabei impliziert der Kernel folgendes Skalarprodukt:

$$\begin{aligned}
 \langle f, g \rangle_H &\equiv \left\langle \sum_{n=0}^{\infty} a_n e^{i 2 \pi n \cdot}, \sum_{n=0}^{\infty} d_n e^{i 2 \pi n \cdot} \right\rangle \equiv \sum_{n=0}^{\infty} \frac{\tilde{f}(n) \tilde{g}^*(n)}{\lambda_n} \quad \text{wobei} \\
 \tilde{f} &\text{ die Fouriertransformierte von } f \text{ ist}
 \end{aligned}$$

Der zugehörige Reproducing Kernel Hilbert Space ist

$$H_K = \left\{ f \in L_2([0, 1]^d) \mid \|f\|_K^2 = \sum_{n=1}^{\infty} \frac{|\tilde{f}(n)|^2}{\lambda_n} < \infty \right\} .$$

Um diese Ergebnisse auf Reproducing Kernel Hilbert Spaces von Funktionen, welche auf dem ganzen d dimensionalen Euklidischen Raum definiert sind, zu übertragen, gehen die Summen in Integrale über:

$$K(x, y) \equiv \int_s \tilde{K}(s) e^{isx} e^{-isy} ds \quad \text{und} \quad \langle f, g \rangle_H \equiv \int_s \frac{\tilde{f}(s) \tilde{g}^*(s)}{\tilde{K}(s)} ds \quad \text{wobei } \lambda_n \text{ in } \tilde{K}(s) \text{ überging}$$

Die Reproducing Property ist durch diesen Kernel erfüllt:

$$\langle f(\cdot), K(\cdot, y) \rangle = \int_s \frac{\tilde{f}(s) \tilde{K}^*(s, y)}{K(s)} ds = \int_s \frac{\tilde{f}(s) \tilde{K}(s) e^{-iys}}{K(s)} ds = f(y)$$

Der zugehörige Reproducing Kernel Hilbert Space ist

$$H_K = \left\{ f \in L_2(\mathbb{R}^d) \mid \|f\|_K^2 = \int_s \frac{|\tilde{f}(s)|^2}{K(s)} ds < \infty \right\}$$

Dieser Vorkenntnisse erlauben einem nun, sowohl durch Wahl von $\tilde{K}(s)$ die Norm der Funktionen zur Glattheitssteuerung intuitiv zu definieren, als auch die der Norm zugrunde liegenden Kernelfunktionen zu finden.

Thin Plate Spline Basisfunktionen resultieren aus der Wahl

$$\tilde{K}(s) = \frac{1}{\|s\|^{2m}}$$

für $m = 2$. Die zugehörigen Kernelfunktionen sind:

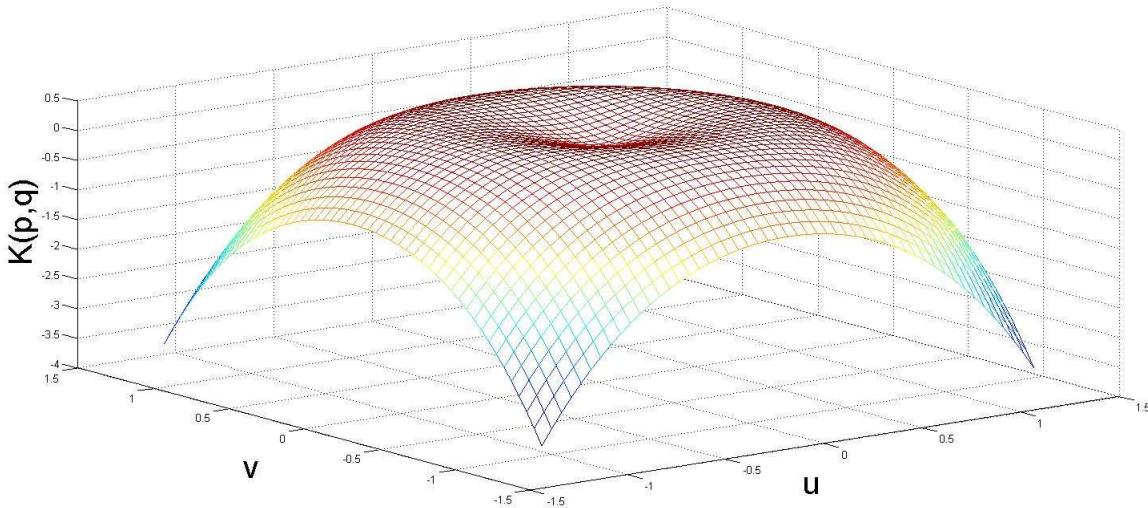
$$K(x, y) = \begin{cases} \|x - y\|^{2m-d} \ln \|x - y\| & \text{falls } 2m > d \wedge d \text{ is even} \\ \|x - y\|^{2m-d} & \text{sonst} \end{cases}$$

Im zweidimensionalen Fall ($d=2$) besteht der Nullraum H_0 drei dimensional und er besteht aus allen Polynomen vom maximalen Grad 1. Der Thin Plate Spline Filter Kernel $\tilde{K}(s)$ entspricht einem Hochpassfilter, d.h. er verstärkt hohe Frequenzen s mehr als niedrige. Das ist wünschenswert, denn hohe Frequenzen entsprechen weniger glatten Funktionen und diese werden mittels diesem Hochpassfilter in der zu minimierenden Norm (für $m = 2$)

$$I[f] = \|f\|_H^2 = \langle f, f \rangle_H = \int_s \frac{|\tilde{f}(s)|^2}{\tilde{K}(s)} ds = \int_s \|s\|^4 |\tilde{f}(s)|^2 ds = c^T K c$$

bestraft.

Folgendes Bild dient zur Illustration dieser Basisfunktionen für den in dieser Semesterarbeit wichtigen zweidimensionalen Fall. Dabei bezeichnen p und q Punkte aus der zweidimensionalen Ebene, welche mittels u und v Koordinaten parametrisiert ist.



6.2 Thin Plate Spline

6.2.1 Interpolation von eindimensionaler Daten

Physikalisch werden Thin Plate Splines dadurch motiviert, dass ein dünnes Metallblech in der uv-Ebene unter Einfluss von kleinen Kräften in z-Richtung versucht, in einen Zustand geringster Krümmungsenergie zu gelangen. Sei die z Koordinate des Metallblechs durch eine Funktion $z: \mathbb{R}^2 \rightarrow \mathbb{R}$ beschrieben. Dann ist die Krümmungsenergie von z gegeben durch:

$$\begin{aligned} I[z] &= \iint_{\mathbb{R}^2} \left(\left(\frac{\partial^2 z}{\partial u^2} \right)^2 + 2 \left(\frac{\partial^2 z}{\partial u \partial v} \right)^2 + \left(\frac{\partial^2 z}{\partial v^2} \right)^2 \right) du dv \\ &\propto \iint \left((s^2 z(s,t))^2 + 2(s t z(s,t))^2 + (t^2 z(s,t))^2 \right) ds dt = \iint ((s^2 + t^2) z(s,t))^2 ds dt \\ &= \int \|r\|^4 |\tilde{z}(r)|^2 dr \quad \text{wobei } r^2 = \|(s,t)\|^2 \end{aligned}$$

Man erkennt, dass das physikalisch motivierte Funktional der erste Zeile äquivalent zum zuvor hergeleiteten Funktional für die zweidimensionalen Thin Plate Spline Basisfunktionen ist.

Es ist zu bemerken, dass $I[z]$ lediglich eine linearisierte Version des physikalisch korrekten Funktionalen ist. Beide Versionen stimmen jedoch überein, falls die Fläche, deren Krümmungsenergie berechnet werden soll, isometrisch parametrisiert ist. Dies ist im Allgemeinen nicht der Fall, trotzdem liefert das linearisierte Funktional zuverlässige Ergebnisse, solange die Abweichung zur isometrischen Parametrisierung klein ist. Im hier betrachteten Fall für eine Fläche $F: \mathbb{R}^2 \rightarrow \mathbb{R}^3: (u,v) \rightarrow (u,v,z(u,v))$ ist diese Bedingung erfüllt, falls die partiellen Ableitungen von z klein sind.

Der Thin Plate Spline Interpolant $z(u,v)$ minimiert die obige Krümmungsenergie unter der Bedingung der exakten Interpolation aller Datenpunkte (u_i, v_i) . Er gehört zur Klasse der Radialen Basis Funktionen Interpolanten mit Polynomgrad 1, als Basis Funktion wird eben die Thin Plate Spline verwendet:

$$\begin{aligned} f: \mathbb{R}^2 \rightarrow \mathbb{R}: f(u,v) &= a_1 + a_u u + a_v v + \sum_{i=1}^n c_i U(\left\| \begin{pmatrix} u_i \\ v_i \end{pmatrix} - \begin{pmatrix} u \\ v \end{pmatrix} \right\|) \\ U: \mathbb{R} \rightarrow \mathbb{R}: U(r) &= r^2 \log r \end{aligned}$$

Um zweifach integrierbare zweite Ableitungen zu erhalten, was wir bei der Minimierung der Krümmungsenergie voraussetzen, muss zudem gefordert werden:

$$\sum_{i=1}^n c_i = 0 \quad \text{und} \quad \sum_{i=1}^n c_i u_i = 0 \quad \text{und} \quad \sum_{i=1}^n c_i v_i = 0 .$$

Eine andere Erklärung für diese Nebenbedingungen ist die Forderung, dass die Polynome vom maximalen Grad 1 dem Nullraum H_0 des Reproducing Kernel Hilbert Spaces entsprechen müssen. Dies wiederum heisst, dass diese Polynome orthogonal zu Funktionen des Hilbertraums H sein müssen.

Die unbekannten Parameter des Thin Plate Spline Interpolanten gehen lediglich linear in den Interpolanten ein, darum ist eine Formulierung als Linear Least Squares Problem möglich, was ja bereits im Kapitel 6.1.3 gezeigt wurde.

Dieser Interpolant hat eine interessante Eigenschaft. Er kann in einen affinen Teil und eine Summe über nicht lineare Basis Funktionen geteilt werden. Der affine Teil bestimmt das Verhalten der Interpolation im Unendlichen, die Summe der Basisfunktionen ist beschränkt und asymptotisch flach.

6.2.2 Interpolation von zweidimensionalen Daten

Um ein Warping, d.h. eine Transformation $f: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ welche (u,v) in (x,y) überführt, unter Verwendung der Thin Plate Spline Interpolation zu erhalten, werden einfach zwei separate Interpolationen durchgeführt: eine für die x-Dimension und eine für die y-Dimension. Wie in [B89] festgestellt worden ist, sind diese beiden Thin Plate Spline Interpolationen aber nicht, wie vermutet werden könnte, völlig unabhängig voneinander, sondern sie sind zwei unterschiedliche Linearkombinationen derselben so genannten Principal Warps! Denn die zugrunde liegende Kernelmatrix K ist dieselbe.

Eine einfach Formulierung als Linear Least Squares Problem lässt auf die unbekannten Gewichte a und c schliessen:

$$\begin{aligned}
 f^* &= \arg \min_{[c_n]_1^n, a} \left(\sum_{i=1}^N \left\| \begin{pmatrix} x_i \\ y_i \end{pmatrix} - \underbrace{\begin{bmatrix} a_{ux} & a_{vx} & a_{tx} \\ a_{uy} & a_{vy} & a_{ty} \\ 1 & 1 & 1 \end{bmatrix} \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix}}_{\text{affiner Teil}} + \sum_{j=1}^n \begin{pmatrix} c_{ix} \\ c_{iy} \end{pmatrix} U(\left\| \begin{pmatrix} u_i \\ v_i \end{pmatrix} - \begin{pmatrix} u_j \\ v_j \end{pmatrix} \right\|) \right\|^2 \right) \\
 \nabla_{(c, a)} f^* &= 0 \rightarrow \begin{bmatrix} K & P \\ P^T & O_{3 \times 3} \end{bmatrix} \begin{bmatrix} c \\ a \end{bmatrix} = L \begin{pmatrix} c \\ a \end{pmatrix} = \begin{bmatrix} z \\ 0 \end{bmatrix} \quad \text{wobei:} \\
 K_{ij} &= U\left(\left\| \begin{pmatrix} u_i \\ v_i \end{pmatrix} - \begin{pmatrix} u_j \\ v_j \end{pmatrix} \right\|\right) \\
 P &= \begin{bmatrix} \vdots & \vdots & \vdots \\ u_i & v_i & 1 \\ \vdots & \vdots & \vdots \end{bmatrix} \in \mathbb{R}^{n \times 3} \\
 c &= \begin{bmatrix} \vdots & \vdots \\ c_{ix} & c_{iy} \\ \vdots & \vdots \end{bmatrix} \in \mathbb{R}^{n \times 2} \\
 a &= \begin{bmatrix} a_{ux} & a_{uy} \\ a_{vx} & a_{vy} \\ a_{tx} & a_{ty} \end{bmatrix} \\
 z &= \begin{bmatrix} \vdots & \vdots \\ x_i & y_i \\ \vdots & \vdots \end{bmatrix} \in \mathbb{R}^{n \times 2} \\
 L &= \begin{bmatrix} K & P \\ P^T & O \end{bmatrix}
 \end{aligned}$$

L ist regulär und kann darum invertiert werden wodurch die Unbekannten zu bestimmen sind.

Der zweidimensionale Thin Plate Spline Interpolant sieht also folgendermassen aus:

$$f : \mathbb{R}^2 \rightarrow \mathbb{R}^2: f(u, v) = \begin{bmatrix} a_{ux} & a_{vx} & a_{tx} \\ a_{uy} & a_{vy} & a_{ty} \\ 1 & 1 & 1 \end{bmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} + \sum_{i=1}^n \begin{pmatrix} c_{ix} \\ c_{iy} \end{pmatrix} U(\left\| \begin{pmatrix} u \\ v \end{pmatrix} - \begin{pmatrix} u_i \\ v_i \end{pmatrix} \right\|)$$

Bezeichne L_n^{-1} den linken, oberen n -bei- n Block von L^{-1} . Dann ist die Krümmungsenergie $I[f]$ des Thin Plate Spline Interpolanten eine quadratische Form von L_n^{-1} in x und y , aber auch eine quadratische Form von K in c_{*x} und c_{*y} , wie zuvor hergeleitet wurde:

$$I[f] = \frac{1}{8\pi} (x^T c_{*x} + y^T c_{*y}) \propto x^T (L_n^{-1} x) + y^T (L_n^{-1} y) = x^T L_n^{-1} x + y^T L_n^{-1} y = c_{*x}^T K c_{*x} + c_{*y}^T K c_{*y}$$

Interessant ist, dass eine solche Thin Plate Spline Interpolation, solange sie sich nicht faltet, ein Diffeomorphismus ist, d.h. eine differenzierbare Abbildung zwischen Mannigfaltigkeiten, welche zudem eine differenzierbare Inverse besitzt.

6.2.3 Regularisierte Thin Plate Spline

Oft ist es wünschenswert, vor allem wenn Unsicherheiten und Rauschen in den Daten vorhanden sind, dass die Datenpunkte nicht exakt interpoliert, sondern lediglich approximiert werden. Um eine glätttere Thin Plate Spline Approximation zu erhalten, wird nun Datentreue gegen Glattheit eingetauscht, indem Prior Informationen wie zuvor beschrieben in die zu minimierende Funktion einfließen lassen werden:

$$\begin{aligned}
 H[f] &= \sum_{i=1}^n (z_i - f(u_i, v_i))^2 + \lambda I[f] \\
 &= \left(z - [K \ P] \begin{pmatrix} c \\ a \end{pmatrix} \right)^T \left(z - [K \ P] \begin{pmatrix} c \\ a \end{pmatrix} \right) + \lambda c^T K c
 \end{aligned}$$

Wenn in obiges Funktional der Thin Plate Spline Approximant eingesetzt wird, die immer noch zwingenden Nebenbedingungen für zweifach integrierbare zweite Ableitungen berücksichtigt und der Gradient nach den unbekannten Gewichten gleich Null gesetzt wird, ist eine kleine Änderung erkennbar, die an der ursprünglichen Gleichung für die Unbekannten vorgenommen werden muss:

$$\begin{aligned}
 \nabla_{c,a} \left(\left(z - [K \ P] \begin{pmatrix} c \\ a \end{pmatrix} \right)^T \left(z - [K \ P] \begin{pmatrix} c \\ a \end{pmatrix} \right) + \lambda c^T K c \right) &= 0 \\
 = -2[K \ P]^T z + 2[K \ P]^T [K \ P] \begin{pmatrix} c \\ a \end{pmatrix} + \lambda \begin{bmatrix} K+K^T \\ O \end{bmatrix} c & \\
 \equiv -2 \begin{bmatrix} K^T & P \\ P^T & O \end{bmatrix} \begin{pmatrix} z \\ 0 \end{pmatrix} + 2 \begin{bmatrix} K^T & P \\ P^T & O \end{bmatrix} \begin{bmatrix} K & P \\ P^T & O \end{bmatrix} \begin{pmatrix} c \\ a \end{pmatrix} + \lambda \left(\underbrace{\begin{bmatrix} K & P \\ P^T & O \end{bmatrix} + \begin{bmatrix} K^T & P \\ P^T & O \end{bmatrix}}_{=2 \begin{bmatrix} K & P \\ P^T & O \end{bmatrix}} \right) \begin{pmatrix} c \\ 0 \end{pmatrix} &= 0 \\
 \equiv - \begin{pmatrix} z \\ 0 \end{pmatrix} + \underbrace{\begin{bmatrix} K & P \\ P^T & O \end{bmatrix} \begin{pmatrix} c \\ a \end{pmatrix}}_{=\begin{bmatrix} I & O \\ O & I \end{bmatrix}} + \lambda \begin{bmatrix} K & P \\ P^T & O \end{bmatrix}^{-1} \begin{bmatrix} K & P \\ P^T & O \end{bmatrix} \begin{pmatrix} c \\ 0 \end{pmatrix} &= 0 \\
 \equiv \begin{pmatrix} c \\ a \end{pmatrix} = \begin{bmatrix} K+\lambda I & P \\ P^T & O \end{bmatrix}^{-1} \begin{pmatrix} z \\ 0 \end{pmatrix} &
 \end{aligned}$$

In der dritten Zeile wurden drei Zeilen eingefügt, die gemäss Nebenbedingung Null ergeben müssen. Die vierte Zeile folgt durch Multiplikation von

$$\left(\begin{bmatrix} K^T & P \\ P^T & O \end{bmatrix} \right)^{-1} \stackrel{\text{L is symmetric}}{=} \begin{bmatrix} K & P \\ P^T & O \end{bmatrix}^{-1}$$

von links. K muss also lediglich durch $K+\lambda I$ ersetzt werden. Diese Herleitung kann analog auf zweidimensionale Approximationen erweitert werden.

Ein skalenunabhängiger Regulierungsparameter λ_0 kann, wie in [BMP02] ebenfalls gemacht, eingeführt werden, indem der Regulierungsparameter λ durch $\lambda = \alpha^2 \lambda_0$ ersetzt wird, dabei entspricht α der mittleren Distanz zwischen zwei Punkten (u_i, v_i) und (u_j, v_j) des ersten Patterns.

Für $\lambda \rightarrow \infty$ geht die Thin Plate Spline Approximation in die normale Least Squares Approximation einer Ebene über.

6.2.4 Gewichtete Thin Plate Spline

Wie im 4. Kapitel beschrieben, wurde im Rahmen dieser Arbeit eine neue Art der Korrespondenzen untersucht, nämlich die k nächsten Nachbarn jeden Punktes des ersten Patterns. Zudem können diese k-to-many Korrespondenzen basierend auf ihren Distanzen gewichtet werden. Die bisher gegebenen Herleitungen der Thin Plate Spline Approximation wurden aber unter Annahme von 1-to-1 Korrespondenzen gemacht: Punkt $P_i=(x_i, y_i, 1)$ entspricht Punkt $z_j=(x_j, y_j)$ und somit ist $k = 1$ und alle Gewichte $w_{i,\sigma(i,j)}$ sind gleich gross. Gewichtete Korrespondenzen können nun aber als weighted Least Squares Ansatz eingebracht werden. Dabei ändert sich die Formel geringfügig, wie im folgenden Abschnitt hergeleitet wird.

Um die Notation einfach zu halten und Doppelindizes zu verhindern, werden die Gewichte $w_{i,\sigma(i,j)}$ der Korrespondenzpaare $(p_1[i], p_2[\sigma(i,j)])$ linear untereinander in einen Vektor w geschrieben. Die Koordinaten der zu einem solchen Korrespondenzpaar zugehörigen Punkte werden ebenfalls an entsprechender Stelle in der drei-Spalten Matrix P im Falle der Punkte des ersten Patterns respektive in einem Vektor x und y im Falle der Punkte des zweiten

Patterns vermerkt.

Es wird erneut lediglich die eindimensionale Datenapproximation hergeleitet. Die Erweiterung auf zweidimensionale Approximationen ist analog, da im mehrdimensionalen Fall lediglich mehrere separate Thin Plate Splines verwendet werden.

Es ist also folgendes Funktional zu minimieren:

$$\begin{aligned}
 H[f] &= \sum_{i=1}^n w_i (z_i - f(u_i, v_i))^2 + \lambda \iint_{\mathbb{R}^2} \left(\left(\frac{\partial^2 f}{\partial u^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial u \partial v} \right)^2 + \left(\frac{\partial^2 f}{\partial v^2} \right)^2 \right) du dv \\
 &\equiv \sum_{i=1}^n w_i \left(z_i - \left(a_1 + a_u u_i + a_v v_i + \sum_{j=1}^n c_j U(\| \begin{pmatrix} u_i \\ v_i \end{pmatrix} - \begin{pmatrix} u_j \\ v_j \end{pmatrix} \|) \right) \right)^2 + \lambda c^T K c \\
 &= \left(z - [K \ P] \begin{bmatrix} c \\ a \end{bmatrix} \right)^T W \left(z - [K \ P] \begin{bmatrix} c \\ a \end{bmatrix} \right) + \lambda c^T K c \\
 \text{s.t. } & \sum_{i=1}^n c_i = 0 \quad \text{und} \quad \sum_{i=1}^n c_i u_i = 0 \quad \text{und} \quad \sum_{i=1}^n c_i v_i = 0
 \end{aligned}$$

wobei W eine Diagonalmatrix mit den Gewichten w_i als Diagonalelemente. Dies ist eine Optimierungsaufgabe mit Nebenbedingungen, darum definiert man die Lagrangefunktion und setzt deren Gradienten gleich Null:

$$\begin{aligned}
 L(c, a, \omega) &= \left(z - [K \ P] \begin{bmatrix} c \\ a \end{bmatrix} \right)^T W \left(z - [K \ P] \begin{bmatrix} c \\ a \end{bmatrix} \right) + \lambda c^T K c + \omega^T P^T c \\
 \nabla_{(c, a, \omega)} L(c, a, \omega) &= -2 \begin{bmatrix} K^T \\ P^T \\ O_{3 \times n} \end{bmatrix} W z + 2 \begin{bmatrix} K^T \\ P^T \\ O_{3 \times n} \end{bmatrix} W [K \ P] \begin{pmatrix} c \\ a \end{pmatrix} + \lambda \begin{bmatrix} K + K^T \\ O_{6 \times n} \\ P^T \end{bmatrix} c + \begin{bmatrix} O_{n \times n} & P \\ O_{3 \times n} & O_{3 \times 3} \\ P^T & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ \omega \end{pmatrix} = 0 \\
 &= -2 \begin{bmatrix} K^T & P \\ P^T & O_{3 \times 3} \\ O_{3 \times n} & O_{3 \times 3} \end{bmatrix} \begin{bmatrix} W & O_{n \times 3} \\ O_{3 \times 3} & I_{3 \times 3} \end{bmatrix} \begin{pmatrix} z \\ 0 \end{pmatrix} + 2 \begin{bmatrix} K^T & P \\ P^T & O_{3 \times 3} \\ O_{3 \times n} & O_{3 \times 3} \end{bmatrix} \begin{bmatrix} W & O_{n \times 3} \\ O_{3 \times n} & I_{3 \times 3} \end{bmatrix} \begin{bmatrix} K & P \\ P^T & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ a \end{pmatrix} \\
 &+ \lambda \left(\begin{bmatrix} K^T & P \\ P^T & O_{3 \times 3} \\ O_{3 \times n} & O_{3 \times 3} \end{bmatrix} + \begin{bmatrix} K & P \\ P^T & O_{3 \times 3} \\ O_{3 \times n} & O_{3 \times 3} \end{bmatrix} \right) \begin{pmatrix} c \\ 0 \\ \omega \end{pmatrix} + \begin{bmatrix} O_{n \times n} & P \\ O_{3 \times n} & O_{3 \times 3} \\ P^T & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ \omega \end{pmatrix}
 \end{aligned}$$

Multipliziert man nun

$$\begin{bmatrix} L^{-1} & O_{n+3 \times 3} \\ O_{3 \times n+3} & I_{3 \times 3} \end{bmatrix} \text{ wobei } L = \begin{bmatrix} K & P \\ P^T & O_{3 \times 3} \end{bmatrix}$$

von links so erhält man:

$$\begin{aligned}
 \nabla_{(c, a, \omega)} L(c, a, \omega) &= -2 \begin{bmatrix} W & O_{n \times 3} \\ O_{3 \times n} & I_{3 \times 3} \\ O_{3 \times n} & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} z \\ 0 \\ 0 \end{pmatrix} + 2 \begin{bmatrix} W & O_{n \times 3} \\ O_{3 \times n} & I_{3 \times 3} \\ O_{3 \times n} & O_{3 \times 3} \end{bmatrix} \begin{bmatrix} K & P \\ P^T & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ a \\ 0 \end{pmatrix} + 2\lambda \begin{bmatrix} I_{n+3 \times n+3} \\ O_{3 \times n+3} \end{bmatrix} \begin{pmatrix} c \\ 0 \\ \omega \end{pmatrix} \\
 &+ \begin{bmatrix} L^{-1} & O_{n+3 \times 3} \\ O_{3 \times n+3} & I_{3 \times 3} \end{bmatrix} \begin{bmatrix} O_{n \times n} & P \\ O_{3 \times n} & O_{3 \times 3} \\ P^T & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ \omega \\ 0 \end{pmatrix} \\
 &= -2 \begin{bmatrix} W & O_{n \times 3} \\ O_{3 \times n} & I_{3 \times 3} \\ O_{3 \times n} & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} z \\ 0 \\ 0 \end{pmatrix} + 2 \begin{bmatrix} WK + \lambda I_{n \times n} & WP \\ P^T & O_{3 \times 3} \\ O_{3 \times n} & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ a \\ 0 \end{pmatrix} + \begin{bmatrix} O_{n \times n} & O_{n \times 3} \\ O_{3 \times n} & I_{3 \times 3} \\ P^T & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ \omega \\ 0 \end{pmatrix} = 0
 \end{aligned}$$

Die letzte Zeile folgt aus einer 2-bei-2 Blockzerlegung und anschliessender Inversion der Matrix L:

$$\text{mittels } 2 \times 2 \text{ Blockzerlegung: } L^{-1} \begin{bmatrix} P \\ O_{3 \times 3} \end{bmatrix} = \begin{bmatrix} O_{n \times 3} \\ I_{3 \times 3} \end{bmatrix}$$

Da $P^T c = 0$ sein muss, ist leicht ersichtlich, dass:

$$-2 \begin{bmatrix} O_{3 \times n} & I_{3 \times 3} \end{bmatrix} \begin{pmatrix} z \\ 0 \end{pmatrix} + 2 \begin{bmatrix} P^T & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ a \end{pmatrix} + \begin{bmatrix} O_{3 \times n} & I_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ \omega \end{pmatrix} = I_{3 \times 3} \omega = 0 \rightarrow \omega = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Man erhält also folgendes einfache lineare Gleichungssystem:

$$\begin{bmatrix} W & O_{n \times 3} \\ O_{3 \times n} & I_{3 \times 3} \end{bmatrix} \begin{pmatrix} z \\ 0 \end{pmatrix} = \begin{bmatrix} W K + \lambda I_{n \times n} & W P \\ P^T & O_{3 \times 3} \end{bmatrix} \begin{pmatrix} c \\ a \end{pmatrix}$$

Eine wichtige Tatsache, die während der Semesterarbeit bemerkt wurde, sei noch angemerkt. Der Gradienten der Lagrangefunktion bildet schon vor den gemachten Vereinfachungen ein lineares Gleichungssystem. Das Problem ist aber, dass sowohl $L^T W L$ als auch $L^T L$ oft nahezu singulär sind. Dadurch entstanden durch deren Inversion enorme numerische Ungenauigkeiten. Es war darum unumgänglich, eine vereinfachte Form zu suchen. Dies wurde auch mit Erfolg erreicht, wie oben ersichtlich. Diese vereinfachte Form gab in keinem der gemachten Beispiele numerische Probleme!

6.3 Thin Plate Splines im Vergleich mit kubischen B-Splines

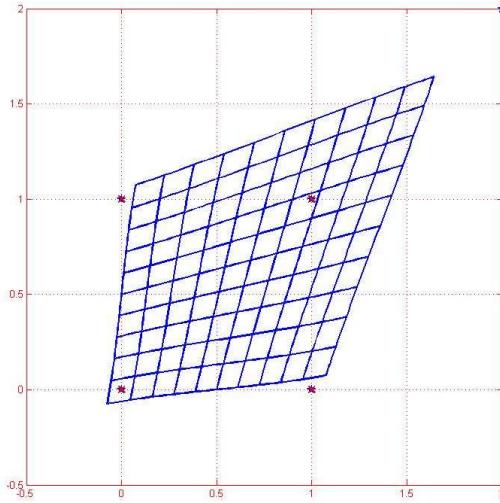
Thin Plate Splines sind das mehrdimensionale Analogon zu Cubic B-Splines. Cubic B-Splines finden zwar auch mittels Tensorprodukten in mehrdimensionalen Problemen Anwendung. Die Datenpunkte müssen dann jedoch in einem regulären Gitter vorliegen. Zudem ist es nicht möglich einen einzelnen Datenpunkt p_{ij} mit einem Gewicht w_{ij} zu gewichten: es ist lediglich möglich, eine ganze Zeile i bzw. eine ganze Kolonne j des regulären Gitters zu gewichten. Das heisst jedoch, dass alle Datenpunkte, die in dieser Zeile respektive in dieser Kolonne liegen, von diesem Gewicht mit beeinflusst werden.

Im Rahmen dieser Semesterarbeit wurden auch Tests mit der Cubic B-Spline Approximationsfunktion 'csaps' von Matlab gemacht. Es stellte sich jedoch heraus, dass diese Methode lediglich Nachteile gegenüber der Thin Plate Spline Approximation besitzt.

6.4 Thin Plate Splines in Matlab

Einige Bemerkungen bezüglich der Implementierung der Thin Plate Spline Approximation in Matlab sind angebracht. Die 'tpaps' Matlab Funktion wird mit drei Argumenten aufgerufen: die Datenpunkte (u_i, v_i) im Definitionsbereich, die entsprechenden Funktionswerte im Bildbereich z_i und einem Regularisierungsparameter p . Dieser Parameter muss in $[0,1]$ liegen und er steuert die Glattheit der Approximation. Für $p=0$ erhält man eine Ebene, welche mittels Least Squares an die Daten angepasst wird, für $p=1$ erhält man den Thin Plate Spline Interpolanten. Dieses Verhalten ist umgekehrt zur eigentlichen Regularisationstheorie, wo ein grosser Wert für λ einer glatten Funktion entspricht.

Die Beschreibung in der Matlab Hilfe besagt, dass die Datenpunkte (u_i, v_i) alle unterschiedlich sein müssen. Dies würde zu Problemen mit den k nächsten Nachbarn Korrespondenzen führen. Denn dabei werden ja denselben Datenpunkt (u_i, v_i) k Funktionswerte zugeordnet. Merkwürdigerweise funktioniert die Matlab 'tpaps' Funktion aber auch in diesem Fall problemlos, wie folgendes Beispiel zeigt:



Die roten Datenpunkte im Definitionsbereich sind

$$UV = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} .$$

Die blauen Datenpunkte im Bildbereich sind

$$XY = \begin{bmatrix} 0 & 0 & 1 & 1 & 2 & 2 \\ 0 & 1 & 0 & 1 & 2 & 2 \end{bmatrix} .$$

Das blaue Gitter repräsentiert die Thin Plate Spline Approximation unter Verwendung dieser Daten mittels Matlab Funktion tpaps ($p = 0.5$). Man erkennt gut, wie die doppelte Gewichtung des Datenpunktes (2,2) gegenüber (1,1) die Thin Plate Spline Approximation in die rechte obere Ecke 'zieht'.

Die Thin Plate Splines in Matlab lassen aber keine Gewichtung der einzelnen Datenpunkte zu. Das Verhalten der Thin Plate Splines im vorigen Beispiel motiviert jedoch, eine stärkere Gewichtung eines Datenpunkts dadurch zu erreichen, indem man den Datenpunkt mehrmals in die Approximation einfließen lässt. Eine Punktcorrespondenz wird also umso häufiger in die Datenpunktvektoren der Thin Plate Spline Approximation eingefügt, je grösser sein Gewicht relativ zu den anderen Korrespondenzen ist. Diese Idee wurde im Verlaufe dieser Semesterarbeit implementiert.

Es gibt aber auch Nachteile der 'tpaps' Funktion in Matlab. Das mehrmalige Einfügen desselben Korrespondenzpaars führt zu einem stark erhöhten Rechenaufwand. Darum wurde die maximale Anzahl an in die Thin Plate Spline Approximation eingehenden Korrespondenzpaaren nach oben hin begrenzt, in den meisten Beispielen auf etwa 600. Dies führt dazu, dass Korrespondenzpaare mit zu kleinem Gewicht eventuell gar nicht in die Approximation eingehen, da ihr Gewicht relativ zu den restlichen Korrespondenzpaaren zu klein ist. Um dies zu verhindern, wurde auch eine Variante untersucht, in der die 'diskretisierte Gewichtung' immer aufgerundet wird, d.h. dass jedes Korrespondenzpaar mindestens einmal in die Approximation einfließt. Diese Methode stellte sich allerdings als ungeeignet heraus.

Der grösste Nachteil der Thin Plate Spline Approximation 'tpaps' besteht jedoch darin, dass es nicht möglich ist, ein Bild mit der 'imtransform' Matlab Funktion unter Verwendung der von 'tpaps' gefundenen Thin Plate Spline Approximation zu warpen. Diese Unzulänglichkeit verunmöglicht interessante, iterative Shape Matching Ideen. Darum wurde beschlossen, eine eigene Thin Plate Spline Approximationsmethode basierend auf 'TFORM' Objekten zu implementieren. Diese 'TFORM' Objekte werden ebenfalls von der 'imtransform' Funktion verwendet, womit nichts mehr einem Warping von Bildern im Wege steht.

6.5 Neue Thin Plate Spline Implementierung

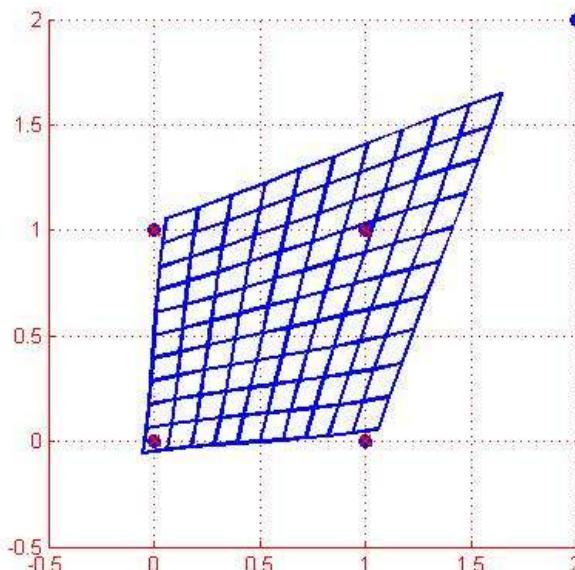
Die eigene Thin Plate Spline Implementierung verwendet exakt die unter 6.2.4 beschriebenen Gleichungen um die unbekannten Parameter zu berechnen. Das Interface dieser Methode ist sehr einfach: die benötigten Korrespondenzen

werden als zwei zwei-Kolonnen Vektoren UV und XY als Argumente übergeben, zudem muss der Wert für den Regularisationsparameter und die Gewichte der Korrespondenzpaare als Vektor mitgegeben werden.

Als Rückgabewert liefert diese Implementierung ein TFORM Objekt, mit Vorwärts- und Rückwärtstransformation. Diese Objekte werden von der Matlab Image Processing Toolbox verwendet und somit sind alle auf TFORM Objekten basierenden Funktionen, im speziellen 'imtransform', anwendbar. Es ist zu beachten, dass die zurückgegebenen Vor- und Rückwärtstransformationen jeweils nicht die Umkehrfunktion der anderen ist! Sie stellen lediglich eine Näherung an diese Umkehrfunktionen dar. Die Vorwärtstransformation wird wie zuvor beschrieben berechnet: zu jedem Punkt (u_i, v_i) im Definitionsbereich werden die k nächsten Nachbarn der Punkte im Bildbereich zugeordnet, ev. gewichtet mit einer Distanzfunktion definiert auf den Punkten. Für die Berechnung der Rücktransformation werden lediglich die beiden Punktvektoren ausgetauscht. Die k nächsten Nachbarn im ursprünglichen Bildbereich werden nun als die Datenpunkte im Definitionsbereich angesehen, während die Punkte im ursprünglichen Definitionsbereich als die korrespondierenden Datenpunkte im Bildbereich interpretiert werden. Es werden also dieselben Korrespondenzen mit demselben Gewicht verwendet, lediglich der Definitionsbereich und Bildbereich werden miteinander vertauscht.

Eine technische Schwierigkeit erschwerte die Implementierung. Für eine Evaluation des Thin Plate Spline Approximanden muss die Kernelfunktion K n mal ausgewertet werden, für jeden Datenpunkt einmal. Aus Effizienzgründen werden die Berechnungen hauptsächlich unter Verwendung von Matrix-Vektor Operationen durchgeführt. Sind viele Datenpunkte in die Approximation eingeflossen und sollen zudem viele Punkte auf einmal evaluiert werden, so stößt Matlab schnell an seine Speichergrenze. Um die Summen nicht mittels langsamen Schleifen zu berechnen, wird die Evaluation in mehreren Blöcken durchgeführt. Einzelheiten können dem Source Code in der Datei 'tps.m' entnommen werden.

Die Vorteile dieser Implementierung gegenüber der 'tpaps' Funktion sind offensichtlich: durch das Vermeiden des mehrfachen Einfügens von Korrespondenzpaaren können viel mehr Korrespondenzpaare in die Approximation eingehen. Zudem können diese Paare exakt, d.h. reellwertig, gewichtet werden.



Folgendes Beispiel dient dazu, die Funktionalität der Implementierung zu zeigen. Es werden dieselben Datenpunkte wie im vorherigen Beispiel verwendet, mit der Ausnahme, dass anstelle des doppelten Punktes $(2,2)$ dieser Punkt nur noch einmal in die Approximation einfließt, jedoch mit doppeltem Gewicht! Die vollständigen Parameter sehen folgendermassen aus:

$$UV = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix} \text{ und } XY = \begin{bmatrix} 0 & 0 & 1 & 1 & 2 \\ 0 & 1 & 0 & 1 & 2 \end{bmatrix} \text{ und } w = [1 \ 1 \ 1 \ 1 \ 2] \text{ und } \lambda = 1;$$

Die Ähnlichkeit dieser zwei Approximationen ist ziemlich überzeugend.

6.6 Angetroffene Probleme

Ein Problem ist, dass die Thin Plate Spline Approximationen oft nicht die gewünschte Glätte besässen. Vor allem die Matlab 'tpaps' Methode ist sehr anfällig auf die Wahl des Regularisationsparameters p . Wird p gleich Null gesetzt, so erhält man eine affine Least Squares Lösung. Wird p aber lediglich ein bisschen grösser als Null gesetzt, so treten

bereits sehr starke Krümmungen auf. Eine mögliche Erklärung ist die folgende: wie zuvor gesagt wird eine linearisierte Version des eigentlichen physikalischen Funktionals für die Krümmungsenergie verwendet. Ist die Fläche isometrisch parametrisiert, so stimmt das linearisierte und das tatsächliche, physikalisch motivierte Funktional überein. Weicht die Parametrisierung aber zu stark von einer Isometrie ab, so bricht das lineare Modell vielleicht zusammen und das linearisierte Funktional liefert unzuverlässige Werte.

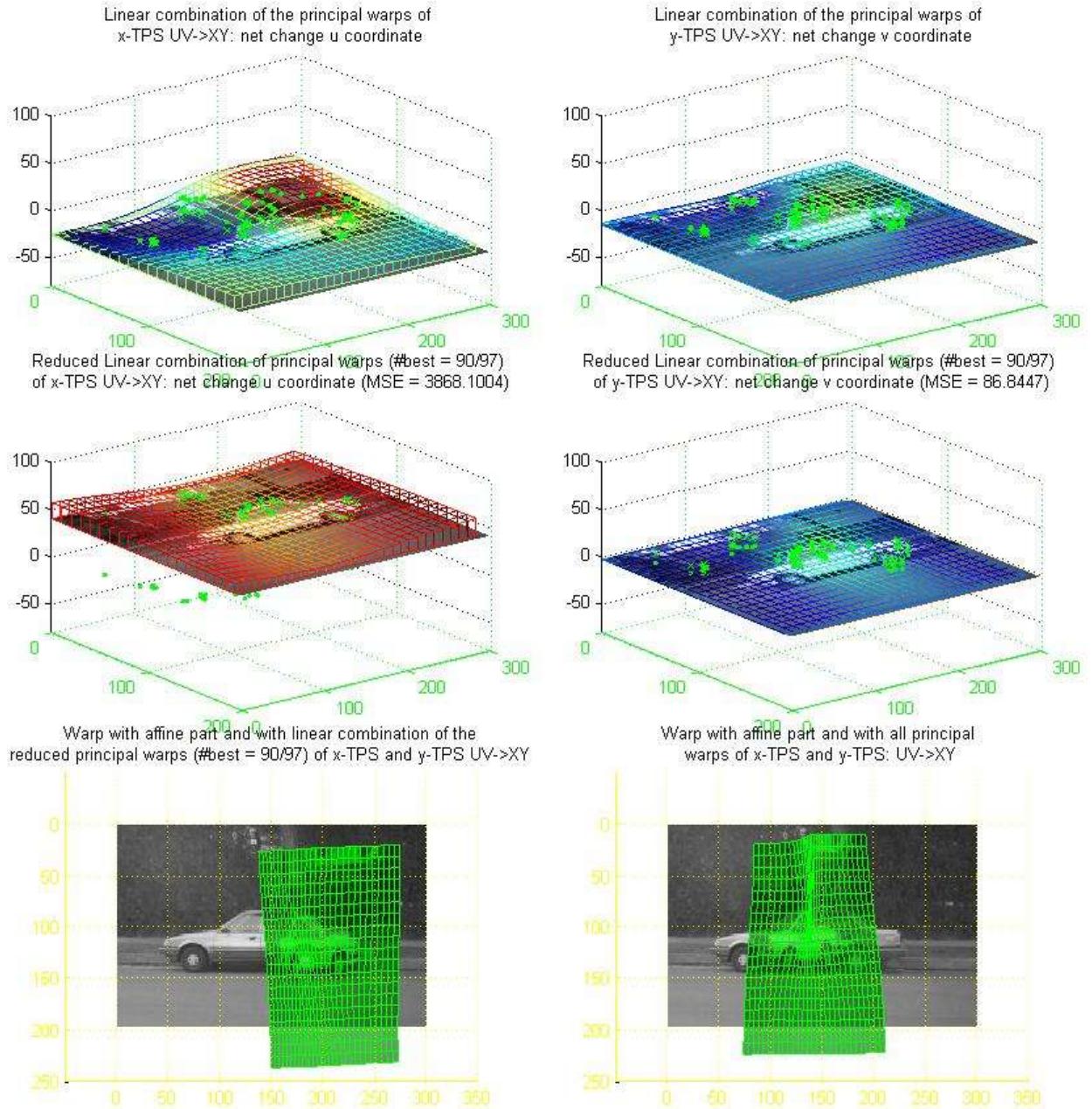
Ein weiteres Problem wird in [CBC01] angesprochen. Die Konditionierung von L wird um so schlechter, je mehr Korrespondenzpaare in die Approximation einfließen. Es muss also der richtige Kompromiss zwischen Verschlechterung der Konditionierung und der Informationsmenge der berücksichtigten Korrespondenpaare gefunden werden.

6.7 Principal Warps

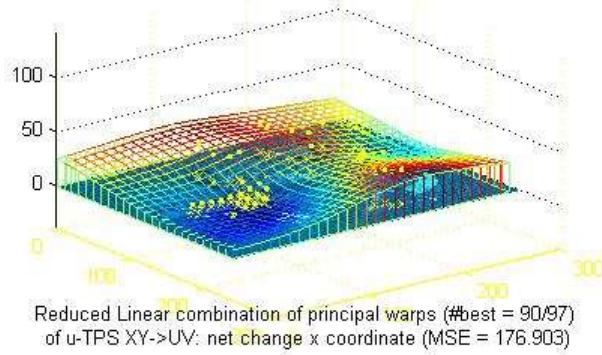
Thin Plate Splines haben einige interessante Eigenschaften. Für eine zweidimensionale Dateninterpolation werden zwei Thin Plate Splines verwendet, diese sind aber nicht, wie zuerst vermutet werden mag, völlig unabhängig voneinander. Sie besitzen nämlich dieselbe Kernelmatrix K , da die Datenpunkte sich im Wertebereich zwar unterscheiden, aber im Definitionsbereich dieselben sind. Es existiert darum eine starke verborgene Struktur in mehrdimensionalen Thin Plate Spline Approximationen.

Eine Möglichkeit diese Strukturen zu analysieren sind Principal Warps, wie sie zum Beispiel in [B89] vorgestellt werden. Principal Warps sind Eigenfunktionen der Krümmungsenergiematrix. Principal Warps ermöglichen es, die x- und y-Thin Plate Spline Approximation als Linearkombination derselben Eigenfunktionen darzustellen. Die Eigenfunktionen können zudem anhand der Krümmung, welche sie in der Approximation verursachen, geordnet werden. Principal Warps sind daher für Thin Plate Splines das Analogon zur Principal Component Decomposition. Sie ermöglichen, dass hohe Krümmung verursachende „Dimensionen“ vernachlässigt werden können, was zwar einen kleinen Fehler impliziert, dafür aber eine glattere Approximation ermöglicht. Für n Datenpunkte existieren lediglich $n-3$ Principal Warps. Eine nähere Erläuterung würde den Rahmen dieser Arbeit sprengen, Details können in [B89] gefunden werden. Principal Warps wurden zudem auch lediglich zu Versuchszwecken und aus eigenem Interesse implementiert, sie werden in der aktuellen Version des Algorithmus nicht benötigt.

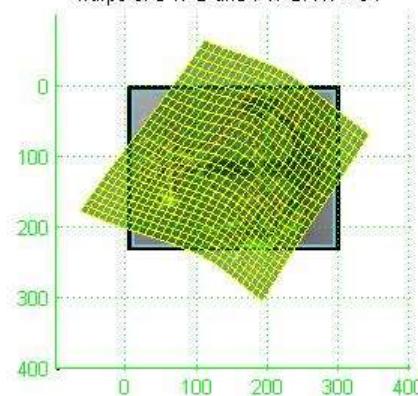
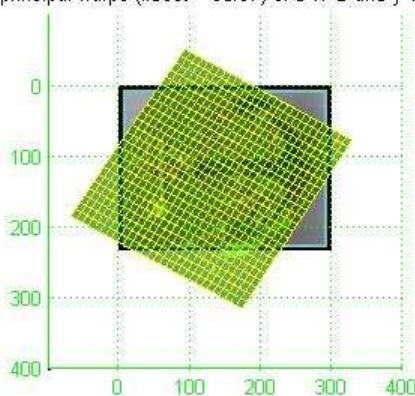
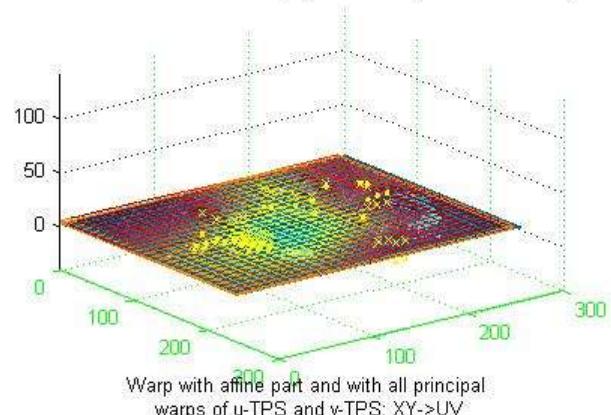
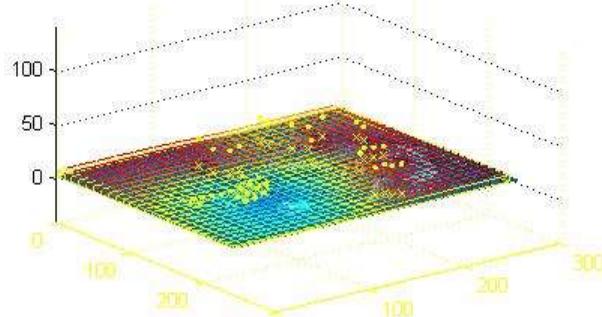
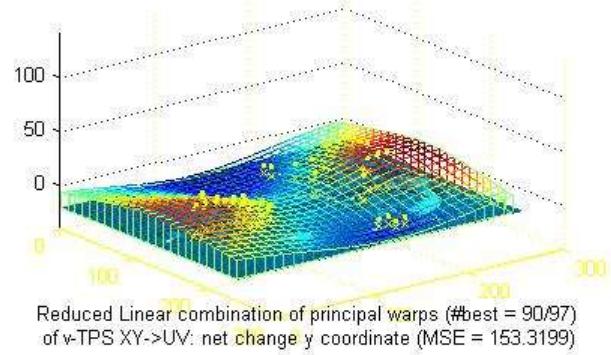
Unten sind zwei Bilder, welche die Funktionsweise der Principal Warps verdeutlichen sollen. Das erste Bild ist eine Thin Plate Spline vom UV Raum in den XY Raum, während im zweiten Bild die Thin Plate Spline den Raum XY in den Raum UV abbildet. Vor allem im zweiten Bild sieht man den Effekt des Weglassens von sieben Principal Warps sehr gut. Es ist aber zu bemerken, dass diese Beispiele mit Bedacht ausgewählt worden sind. Das Weglassen von Principal Warps führt nämlich häufig zu fehlender Verschiebung des ganzen Bildes. Dieser Effekt kann auch leicht im ersten Bild beobachtet werden. Weiter anzumerken ist, dass die Principal Warps Implementierung auf der Beschreibung von [B89] beruht. Es könnte aber durchaus sein, dass sich die Formeln ändern, falls gewichtete Thin Plate Splines benutzt werden. Es wäre sicher interessant zu untersuchen, inwiefern die Formeln angepasst werden müssten, falls überhaupt. Bei beiden Beispielen wurden gewichtete Thin Plate Splines verwendet, in Klammern wird der resultierende mittlere quadratische Fehler angegeben.



Linear combination of the principal warps of u-TPS XY->UV: net change x coordinate



Linear combination of the principal warps of v-TPS XY->UV: net change y coordinate



Kapitel 7: Objekt Klassifizierungsverfahren

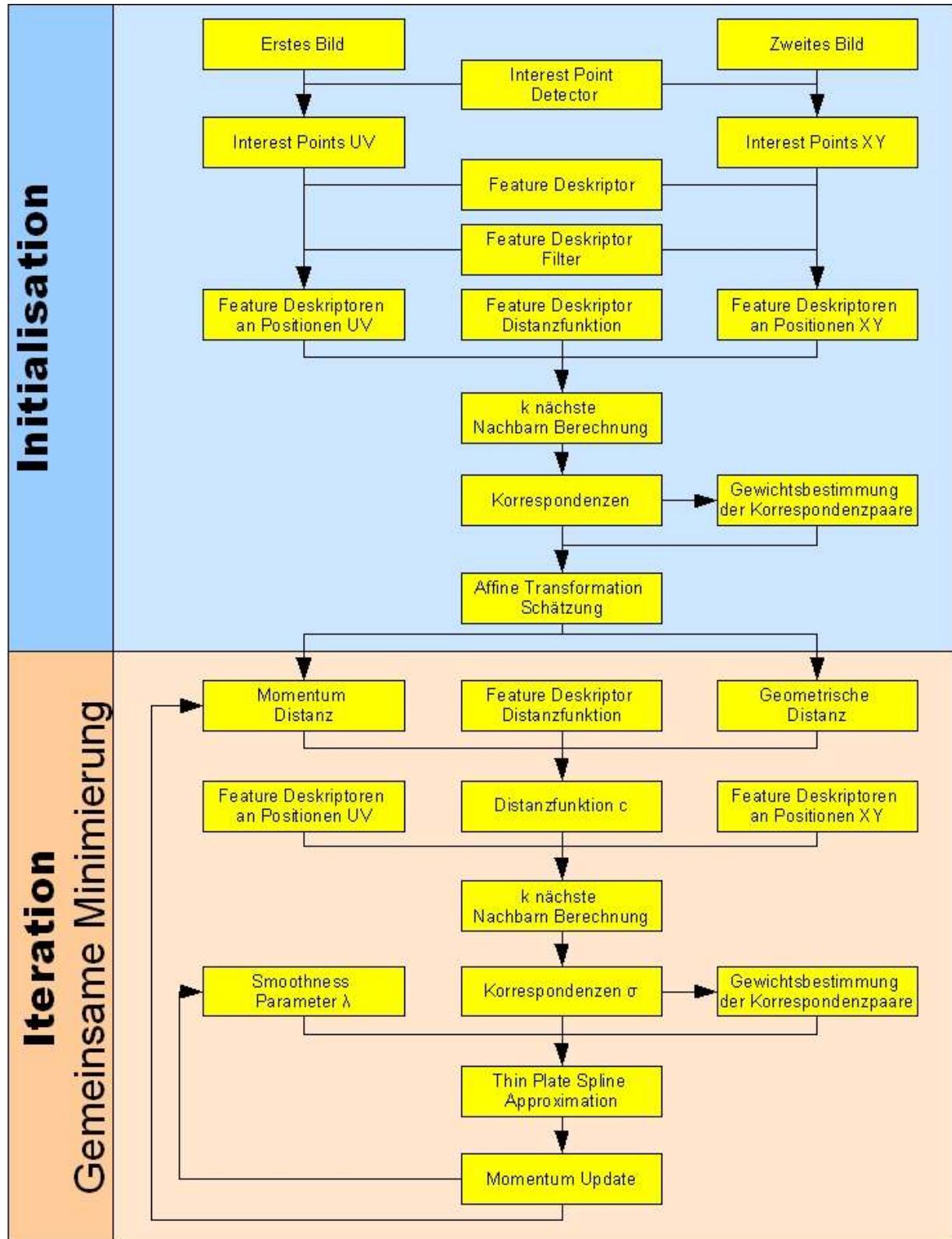
Im Verlaufe dieser Semesterarbeit wurde ein Verfahren entwickelt, um die zuvor präsentierten Ideen zu testen. Dieses wird im Folgenden ausführlich präsentiert. Das Hauptziel des Verfahrens ist die Vermeidung des Lösen eines Integer Programming Problem, wie es zum Beispiel beim Finden von 1-to-1 Korrespondenzen auftritt.

Zuerst wird ein Überblick über den Algorithmus gegeben. Danach werden die einzelnen Schritte des Algorithmus ausführlich behandelt.

7.1 Überblick

Es wird versucht, iterativ sowohl gute Korrespondenzen, als auch eine gute Thin Plate Spline Transformation zu schätzen. Dazu werden die k-nächsten Nachbarn der Punkte des ersten Patterns verwendet. Mit jeder folgenden Iteration wird der Krümmung der Thin Plane Spline weniger Gewichtung zugemessen, sodass komplexere Transformationen möglich werden. Dies wird durch Anpassen des Smoothing Parameters erreicht. Dieser Parameter steuert den Tradeoff zwischen Datentreue einerseits und Prior Information andererseits. Die Prior Information stellt Bedingungen an die Glattheit der Thin Plate Spline Approximation.

Folgende Graphik gibt einen Überblick über den Programm- und Informationsfluss. Dabei wird der Definitionsbereich des ersten Bildes als UV-Raum bezeichnet, derjenige des zweiten Bildes als XY-Raum.



Die Input-Argumente des Algorithmus bestehen aus den zwei Bildern, der Rückgabewert besteht aus der Distanz zwischen diesen beiden Bildern. Der Algorithmus wurde bereits im ersten Kapitel grob beschrieben, darum wird hier lediglich auf die noch nicht genau beschriebenen Details eingegangen. Ganz zentral für den Algorithmus ist die

Distanzfunktion c zwischen zwei Punkten zweier Patterns. Diese Distanzfunktion wird benutzt, um die k nächsten Nachbarn zu berechnen. Sie besteht aus drei Teilen:

- Distanz zwischen den zwei Feature Deskriptoren der zwei Punkte: dieser Term fasst die lokale Ähnlichkeit in den Bildern an den zwei Punkten auf.
- Gedächtnis-Term (Momentum) vorheriger Iterationen: dadurch soll verhindert werden, dass durch die verminderte Gewichtung der Krümmung nicht plötzlich sehr stark gekrümmte und somit sehr spezifische Transformationen entstehen. Denn die verwendeten Thin Plate Spline Transformationen sind sehr mächtig und können dadurch auch eher unrealistische Transformationen hervorbringen.
- Geometrischer Term basierend auf der ursprünglich geschätzten affinen Transformation: Dieser Term versucht, Korrespondenzen in den zwei Bildern räumlich lokal an derselben Stelle zu finden. Er soll also verhindern, dass Korrespondenzen räumlich kreuz und quer verteilt sind und die Thin Plate Spline Transformation des Bildes 'gefaltet' wird.

Im Folgenden wird nun detailliert auf die einzelnen Stufen des Algorithmus eingegangen.

7.2 Interest Point Detection

Der Algorithmus arbeitet mit Patterns, welche einer endlichen Punktmenge entsprechen. Dazu müssen also interessante Punkte in den Bildern gefunden werden. Dazu stehen vier Möglichkeiten zur Verfügung.

7.2.1 Benutzer-definiert

Wie der Name schon sagt, ist es mit dieser Methode dem Benutzer überlassen, wichtige Punkte manuell auszuwählen. Dies stellt für eine automatische Bilderkennung natürlich eine unbefriedigende Lösung dar, für Testzwecke und zum Testen der Stabilität des Verfahrens ist diese Methode aber sehr nützlich.

7.2.2 Harris Interest Point Detector

Dieser Interest Point Detector basiert auf der Beschreibung und der Implementierung von [MS05]. Die Implementierung ist lediglich Linux-kompatibel, da ein Linux System-Call aufgerufen wird. Unter Windows kann dieser Interest Point Detector darum nicht verwendet werden. Zudem muss das Bild im ppm-Format vorliegen. Falls dies nicht der Fall ist, so wird das Bild jedoch vom zusätzlich implementierten Skript automatisch in eine temporäre ppm-Bilddatei konvertiert.

7.2.3 Affine Interest Point Detector

Die Implementierung dieses Interest Point Detectors ist rein Matlab basierend und wurde entwickelt und implementiert in [L03]. Dieser Interest Point Detector besitzt einige Input-Argumente:

- sigma: Dieses Argument ist ein Vektor, dessen Elemente die Größen der berücksichtigten Interest Points bestimmen. Gute Standardwerte für die Elemente von Sigma sind 4, 16, 32, 64.
- nr_of_PoI: dieses Argument bestimmt die maximale Anzahl der zurückgegebenen (stärksten) Interest Points.

Zusätzlich können die gefunden Interest Points mit diesem Algorithmus räumlich optimiert werden. Dies ist jedoch ziemlich zeitaufwändig und brachte keine bemerkenswerte Vorteile. Darum wurden die meisten Beispiele ohne diese Optimierung durchgeführt.

7.2.4 Uniformes Gitter

Interessante Punkte in einem Bild zu finden ist für sich selbst bereits ein sehr schwieriges Problem. Die Resultate lassen oft zu wünschen übrig. Will man möglichst viele potentielle Korrespondenzpartner haben, so besteht die naheliegendste Lösung darin, ein uniformes Gitter über das Bild zu legen und jeden Knotenpunkt als Interest Point zu betrachten. Auch dieser Ansatz birgt jedoch Probleme, da zum Beispiel keinerlei Kanteninformationen zur Interest Point Bestimmung benutzt werden und die Knotenpunkte möglicherweise immer etwas neben Kanten zu liegen kommen.

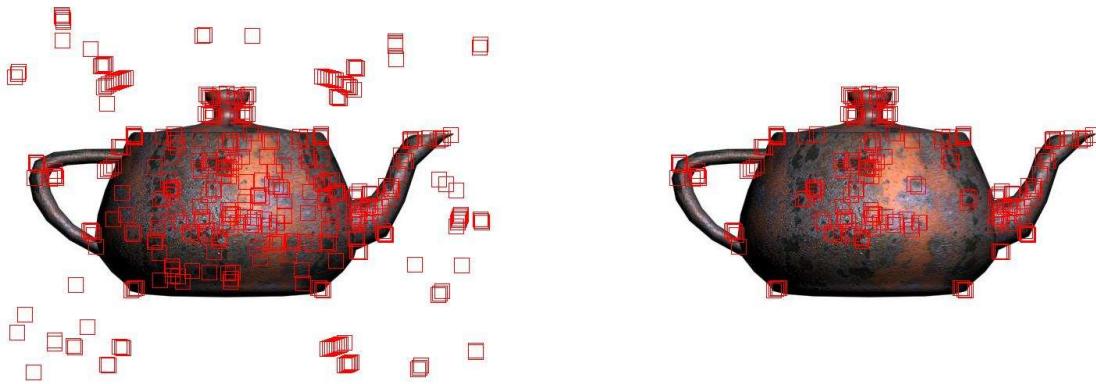
7.3 Bestimmen der Feature Deskriptoren

Um semantische Informationen aus den Bildern zu gewinnen, werden Feature Deskriptoren an den zuvor bestimmten Interest Points bestimmt. Es kann zwischen allen zuvor beschriebenen Feature Deskriptoren ausgewählt werden, mit

Ausnahme des Geometric Blur. Die meisten Beispiele und Tests wurden jedoch mit dem Localized Feature Histogram Detektor durchgeführt, da dieser die vielversprechendsten Resultate lieferte.

Es stellte sich heraus, dass eine Nachbearbeitung der berechneten Deskriptoren nötig ist. Es ist möglich, dass Interest Points in Regionen mit fast uniformer Farbverteilung oder sehr schwacher Kantenstärken zu liegen kommen (zum Beispiel bei der Wahl eines uniformen Gitters). Solche Deskriptoren sind nicht sehr aussagekräftig, trotzdem würden sie auf ähnliche Deskriptoren, zum Beispiel ebenfalls mit fast uniformer Farbverteilung, sehr stark ansprechen. Darum wird ein Filter-Schritt nötig, in welchem nicht aussagekräftige Feature Deskriptoren mit ihrem zugehörigen Interest Point verworfen werden und somit im weiteren Verlauf nicht mehr verwendet werden.

Ein Feature Deskriptor wird nicht aussortiert, falls die mittlere Kantenstärke und die Standardabweichung der Grauwerte im Grauwertbild des zugrunde liegenden Patches über einem Schwellenwert liegen. Diese Schwellenwerte betragen 20.0 und 0.001 für die Grauwertstandardabweichung respektive für die mittlere Kantenstärke. Die mittlere Kantenstärke wird mittels dem in Matlab implementierten Sobel Kantendetektor bestimmt.



Ungefilterte Interest Points mit zugehörigen Feature Deskriptor Patches

Gefilterte Interest Points mit zugehörigen Feature Deskriptor Patches

7.4 Gewichtung von Korrespondenzpartnern

Damit gute Korrespondenzpaare, d.h. solche mit kleiner Distanz, stärkeren Einfluss bei der Transformationsschätzung ausüben können, werden den gefundenen Korrespondenzpaaren Gewichte zugeordnet. Diese Gewichte lassen sich aus der Feature Deskriptor Distanz zwischen den zwei Korrespondenzpartnern berechnen. Dieselbe Gewichtungsmethode wird später dann auch für die dreiteilige Distanzfunktion zwischen Korrespondenzpartnern verwendet: die Feature Deskriptor Distanz wird dabei lediglich durch die dreiteilige Distanz ersetzt.

In einem ersten Anlauf wurde versucht, die Distanzen mittels einer Gibbs-Verteilung in Gewichte umzuwandeln:

$$w_{i,\sigma(i,j)} = \frac{\exp\left(\frac{-c(p_1[i], p_2[\sigma(i,j)])}{T}\right)}{\sum_{i,j} w_{i,\sigma(i,j)}}$$

Hier stellte sich jedoch das Problem, wie die Temperatur T einzustellen ist: Wurde sie zu gross eingestellt, so waren die Gewichte zu uniform und gute Korrespondenzen konnten nicht genügend Einfluss ausüben. Wurde sie hingegen zu klein gewählt, so war die Verteilung zu diskriminativ, d.h. den guten Korrespondenzen wurde nun zu viel Gewicht zugemessen. Der erste Versuch dieses Problem zu lösen, orientierte sich am Simulated Annealing: gestartet wurde mit einem kleinen Wert für T, welcher in jeder Iteration vergrößert wurde bis ein gewisser Prozentsatz der Gewichte über einem gewissen Schwellenwert war.

Die so erreichten Resultate waren zwar besser, aber noch bessere Ergebnisse wurden mit folgender Methode erzielt: Ausgehend von einem hohen Wert für T und einer uniformen Verteilung der Gewichte, d.h. maximaler Entropie, wird die Temperatur in jeder Iteration solange verringert, bis die Entropie der Gewichte (als Wahrscheinlichkeitsverteilung interpretiert) um einen gewissen Prozentsatz von der maximalen Entropie abgenommen hat. Durch diesen

Schwellenwert der relativen Entropieabnahme ist es viel intuitiver zu steuern, wie diskriminativ die Verteilung sein soll. Gute Ergebnisse wurden erzielt, indem eine Entropieabnahme von 5% zugelassen wurde.

7.5 Schätzung der affinen Transformation

Bei der verwendeten Distanzfunktion müssen sowohl der Momentumterm als auch der geometrische Term initialisiert werden. Die Notwendigkeit des Initialisierens des Momentumterms liegt auf der Hand, beim geometrischen Term ist es von Vorteil, dass bereits eine einfache Transformation vorbestimmt wird und dann ausgehen von dieser die geometrische Abweichung zwischen zwei Korrespondenzpartnern berechnet wird. Ist eine solche einfache Transformation bestimmt, so ist es nahe liegend, den Momentumterm ebenfalls mit dieser Transformation zu initialisieren.

Als solche einfache Transformationen wurden affine Transformationen gewählt. Sie besitzen im zweidimensionalen lediglich sechs Unbekannte und lassen alle linearen Transformationen und Translationen zu. Es stellte sich jedoch heraus, dass selbst das Bestimmen einer solch stark eingeschränkten Transformation überhaupt nicht trivial ist.

Die Schätzung der affinen Transformation beruht lediglich auf den Distanzen zwischen den Feature Deskriptoren. Die Interest Points für die Schätzung bestehen bei beiden Bildern aus den Knoten eines uniformen Gitter, natürlich mit Ausfilterung nicht aussagekräftiger Feature Deskriptoren. Nun wird die Distanzmatrix der potentiellen Korrespondenzpartner bestimmt. Daraus werden zu jedem Punkt des ersten Patterns die k nächsten Nachbarn der Punkte im zweiten Pattern bestimmt.

Die Korrespondenzen werden wie zuvor beschrieben gewichtet. Die Koordinaten der Korrespondenzpartner gehen dann so gewichtet in die affine Transformationsschätzung ein.

Es wurde zudem eine Erweiterung der Gewichtungsmethode ausprobiert. Dieser Erweiterung liegt die Beobachtung zugrunde, dass Feature Deskriptoren in Bereichen im Bild, in welchen wenig Feature Deskriptoren liegen, stärker gewichtet werden sollen als solche in Bereichen mit vielen Feature Deskriptoren. Ein Mass für die räumliche Feature Deskriptor Dichte muss also gefunden werden. Eine einfache Möglichkeit zur Dichteschätzung besteht darin, als Dichte die Inverse des Mittelwertes der euklidischen Abstände der räumlich k -nächsten Nachbarn zu nehmen:

$$\rho(u_i, v_i) = \frac{1}{\frac{1}{n} \sum_{j=1}^k \| \begin{pmatrix} u_i \\ v_i \end{pmatrix} - \begin{pmatrix} u_{\sigma(i,j)} \\ v_{\sigma(i,j)} \end{pmatrix} \|}.$$

Um aus den Dichten Gewichte zu erhalten, nimmt man lediglich die Inverse der Dichte. Um Divisionen durch kleine Dichten zu vermeiden, wird zusätzlich ein Regularisierungsparameter α eingeführt. Die mittels relativer Entropieabnahme bestimmten Gewichte und die Dichte-basierten Gewichte werden als Konvexitätskombination zu den endgültigen Gewichten für die affine Transformationsschätzung kombiniert:

$$w_{i,\sigma(i,j)} = (1 - \beta) \frac{1}{\rho(u_i, v_i) + \alpha} + \beta w_{i,\sigma(i,j)}^{\text{entropy}}$$

Es stellte sich jedoch heraus, dass diese Erweiterung nicht gut funktionierte. Selbst wenn der Regularisierungsparameter α sehr klein gewählt wird, bleiben die dichte-basierten Gewichte sehr ähnlich und wenig diskriminativ. Darum wurde diese Erweiterung schliesslich fallen gelassen.

7.6 Iterationsphase

Nachdem alles Notwendige initialisiert ist, startet die Iteration. In jeder Iteration wird der Smoothing Parameter λ der Thin Plate Spline Approximation verkleinert und somit werden mächtigere und komplexere Transformationen zugelassen. Im Unterschied zur Schätzung der affinen Transformation werden im ersten Bild nun aber nicht mehr Interest Points, welche Knoten eines uniformen Gitters entsprechen, sondern Interest Points, die mittels einem Interest Point Detektor bestimmt werden, verwendet. Im zweiten Bild wird weiterhin ein uniformes Gitter verwendet.

7.6.1 Dreiteilige Distanzfunktion zwischen zwei Bildpunkten

Um die Korrespondenzen zu bestimmen wird eine Distanzfunktion zwischen zwei Punkten zweier Patterns p_1 und p_2 benötigt. Im Falle der affinen Transformationschätzung bestand diese Distanzfunktion lediglich aus der Distanz zwischen den Feature Deskriptoren an den zwei Punkten. Während der Iterationsphase wird jedoch eine komplexere

Distanzfunktion verwendet, welche drei Eigenschaften zwischen zwei Korrespondenzpartnern $p_1[i]$ und $p_2[\sigma(i,j)]$ berücksichtigt:

$$c(p_1[i], p_2[\sigma(i,j)]) = \alpha \cdot fdTerm(p_1[i], p_2[\sigma(i,j)]) + \beta \cdot mTerm(p_1[i], p_2[\sigma(i,j)]) + \gamma \cdot geomTerm(p_1[i], p_2[\sigma(i,j)])$$

wobei:

$$\alpha + \beta + \gamma = 1 \text{ und } \alpha \geq 0 \text{ und } \beta \geq 0 \text{ und } \gamma \geq 0$$

Der erste Term entspricht der Feature Descriptor Distanz zwischen den zwei Punkten, zum Beispiel der χ^2 Distanz. Der zweite und dritte entsprechen dem Momentumterm respektive dem geometrischen Term. Da die einzelnen Terme verschiedene Größenordnungen haben, werden sie jeweils einzeln normalisiert, d.h. der Mittelwert $E_{i,\sigma(i,j)}[\cdot]$ des Terms über alle Korrespondenzen wird abgezogen, danach wird durch die Standardabweichung $std_{i,\sigma(i,j)}[\cdot]$ dividiert. Dies ermöglicht eine gute Steuerung der einzelnen Terme durch die Faktoren der Konvexkombination.

$$\begin{aligned} fdTerm(p_1[i], p_2[\sigma(i,j)]) &= \frac{\overline{fdTerm}(p_1[i], p_2[\sigma(i,j)]) - E_{i,\sigma(i,j)}[\overline{fdTerm}]}{std_{i,\sigma(i,j)}[\overline{fdTerm}]} \\ mTerm(p_1[i], p_2[\sigma(i,j)]) &= \frac{\overline{mTerm}(p_1[i], p_2[\sigma(i,j)]) - E_{i,\sigma(i,j)}[\overline{mTerm}]}{std_{i,\sigma(i,j)}[\overline{mTerm}]} \\ geomTerm(p_1[i], p_2[\sigma(i,j)]) &= \frac{\overline{geomTerm}(p_1[i], p_2[\sigma(i,j)]) - E_{i,\sigma(i,j)}[\overline{geomTerm}]}{std_{i,\sigma(i,j)}[\overline{geomTerm}]} \end{aligned}$$

Die Definitionen der einzelnen Terme sehen wie folgt aus:

$$\begin{aligned} \overline{fdTerm}(p_1[i], p_2[\sigma(i,j)]) &= dist_{\text{Feature Descriptor}}(\text{featureDescriptor}(p_1[i], \text{image}_1), \text{featureDescriptor}(p_2[\sigma(i,j)], \text{image}_2)) \\ \overline{mTerm}(p_1[i], p_2[\sigma(i,j)]) &= \frac{\|g_{\text{momentum}}^{(t)}(p_1[i]) - p_2[\sigma(i,j)]\|_2}{\|g_{\text{momentum}}^{(t)}(p_1[i]) - p_1[i]\|_2 + 1} \\ \overline{geomTerm}(p_1[i], p_2[\sigma(i,j)]) &= \|p_2[\sigma(i,j)] - A^* \begin{pmatrix} p_1[i] \\ 1 \end{pmatrix}\|_2 \end{aligned}$$

$dist_{\text{Feature Descriptor}}$: Distanzfunktion zwischen Feature Descriptoren
 A^* : während der Initialisierung gefundene affine Transformation

Der Momentumterm ist etwas komplizierter und verdient darum eine nähere Betrachtung. Wie gesagt hängt er von in den vorhergehenden Iterationen gefundenen Transformationen ab, somit also auch von der Zeit. Seine Aufgabe ist derjenigen des geometrischen Terms sehr ähnlich. Während der geometrische Term versucht, räumlich naheliegende Korrespondenzpartner zu finden, versucht der Momentumterm Korrespondenzpartner zu finden, die räumlich mit den zuvor gefundenen Transformationen in Einklang zu bringen sind. Denn mit jeder Iteration wird der Datentreue mehr Bedeutung zugemessen, was dazu führt, dass die zu optimierende Funktion feinere Details preisgibt und somit mehr lokale Minima aufweist. Um ein solches lokales Minimum möglichst zu vermeiden, wird darum eine Trägheit eingebaut, welche die zuvor berechneten, glatteren Transformation berücksichtigt. Die Idee ist, dass lokale Minima durch den Momentumterm verschmiert werden, aber trotzdem eine Konvergenz zum globalen Minima statt findet.

Formal kann das Gedächtnis des Momentumterms folgendermassen ausgedrückt werden.

$$\begin{aligned} g_{\text{momentum}}^{(t)}: UV \rightarrow XY: \quad g_{\text{momentum}}^{(t)}(u, v) &= \tau g_{\text{momentum}}^{(t-1)}(u, v) + (1 - \tau) g_{\text{momentum}}^{(t-1)}(u, v) \\ g_{\text{momentum}}^{(0)} &= A^* \end{aligned}$$

τ : backward time horizon

$g_{\text{momentum}}^{(t-1)}$: gefundene Thin Plate Spline Transformation im (t-1)-ten Iterationsschritt

In den Beispielen wurde der Backward Time Horizon auf 0.5 gesetzt, was einem ziemlich raschen „Gedächtnisverlust“ gleich. Interessant zu bemerken ist, dass für $\tau=0$ das Gedächtnis exakt der geschätzten affinen Transformation entspricht!

Die Divison im Momentumterm in der Distanzfunktion c geschieht, da es für einen bereits zuvor weit transformierten

Punkt wahrscheinlicher ist, dass eine erneute Transformation zu einer grösseren Änderung führt als für einen Punkt, der nur in seine unmittelbare Nähe transformiert wird. Durch diese Division wird der Thin Plate Spline Approximation exakt dies ermöglicht (die Addition von 1 verhindert eine Division durch 0).

Der Momentumterm ist eigentlich mehr dem zugrundeliegenden Algorithmus als der Distanz zwischen zwei Punkten in den Patterns zu zuordnen. Trotzdem wurde dieser Term der Einfachheit halber direkt in diese Distanzfunktion eingefügt.

Ein Detail zum Momentumterm ist noch erwähnenswert: $g_{momentum}^{(t)}$ wird als Matrix gespeichert, das Gedächtnis wird jedoch nicht für jede Koordinate einzeln gespeichert, sondern es wird jeweils der Mittelwert über einen ganzen Bildblock an diesem Block gespeichert. Dies führt zu einer zusätzlichen Regularisierung, sodass Ausreisser stabiler behandelt werden. Die Grösse eines solchen Blocks wurde identisch der Grösse eines Localized Histogram Patches gewählt.

7.6.1 Varianten der Anzahl und Gewichtung berücksichtigter Korrespondenzen

Da die Anzahl an Korrespondenzen gross werden kann, vor allem für grosses k und viele Punkte im ersten Pattern, stehen verschiedene Möglichkeiten offen, um nicht alle Korrespondenzen in die Thin Plate Spline Approximation einfließen zu lassen. Zudem stehen verschiedene Optionen zur Gewichtung der Korrespondenzen zur Verfügung:

- uniforme Gewichte: alle Korrespondenzen der k-nächsten Nachbarn werden uniform gewichtet.
- uniforme Gewichte, aber lediglich N beste Korrespondenzen: es gehen lediglich die N besten Korrespondenzen der k-nächsten Nachbarn in die Thin Plate Spline Approximation ein, diese werden ebenfalls uniform gewichtet.
- nonuniforme Gewichte: alle Korrespondenzen der k-nächsten Nachbarn werden gemäss ihrer Distanz gewichtet.
- nonuniforme Gewichte, aber lediglich N beste Korrespondenzen: es gehen lediglich die N besten Korrespondenzen der k-nächsten Nachbarn in die Thin Plate Spline Approximation ein, diese werden aber gemäss ihrer Distanz gewichtet.

Die Implementierung ermöglicht noch weitere Möglichkeiten, die aber eigentlich lediglich darum noch vorhanden sind, weil man in der zuvor verwendeten Matlab Thin Plate Spline Approximation keine Gewichte einfließen lassen konnte. Diese weiteren Möglichkeiten fügen dasselbe Korrespondenzpaar mehrmals in die Thin Plate Spline Approximation ein, um so dessen Gewicht zu erhöhen, wie in einem vorherigen Kapitel an einem Beispiel gezeigt wurde.

Diese Korrespondenzen gehen zusammen mit ihren Gewichten, welche für den Fall der nonuniformen Gewichtung analog zur affinen Transformationsschätzung mittels relativer Entropieabnahme aus den Distanzen bestimmt werden, in die Thin Plate Spline Approximation ein. Die Punkte des ersten Patterns im UV Bereich entsprechen dem Definitionsbereich der Thin Plate Spline Transformation, die Punkte des zweiten Patterns im XY Bereich dem Wertebereich.

7.7 Distanz zwischen zwei Bildern

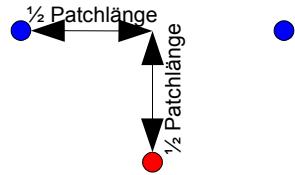
Die Distanzfunktion zwischen den zwei Bildern muss die Bilder selbst und die gefundene Transformation mit einbeziehen. Die hier verwendete Distanzfunktion ist sehr einfach gehalten, es wäre sicherlich interessant, komplexere Distanzfunktionen zu testen. Zum Beispiel könnte die Komplexität der gefundenen Transformation mit in die Distanzfunktion eingehen. Dies ist vor allem bei Thin Plate Splines interessant, da sie die Berechnung der Krümmung einfacher zulassen und Krümmung ein Mass für die Komplexität einer Transformation ist.

Die Distanzfunktion wird folgendermassen definiert:

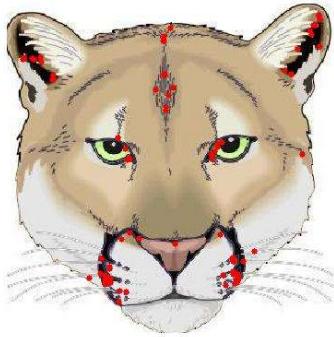
$$d(image_1, image_2) = \frac{1}{n} \sum_{i=1}^n dist_{\text{Feature Descriptor}} \left(featureDescriptor(p_1[i], image_1), featureDescriptor(g^*(p_1[i]), image_2) \right)$$

g^* bezeichnet die am Ende gefundene Thin Plate Spline Transformation. Die Distanzfunktion berechnet also den Mittelwert der Feature Desriptor Distanzen zwischen den Interest Points im ersten Bild und ihren transformierten Pendants im zweiten Bild.

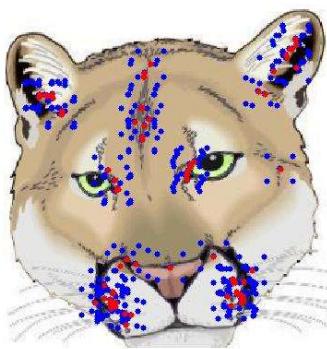
Zwei wichtige Details sind hervorzuheben: Erstens wird nicht nur an den transformierten Positionen ein Feature Desriptor bestimmt, sondern zusätzlich auch an diagonal verschobenen Positionen:



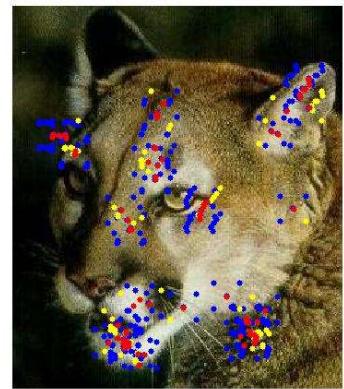
Von diesen fünf Feature Descriptoren wird schliesslich derjenige gewählt, welcher die kleinste Distanz zum Feature Descriptor an untransformierter Stelle im ersten Bild besitzt. Dies ergibt eine zusätzliche Regularisierung und vermindert so zum Beispiel den Effekt leicht falsch geschätzter Transformationen. Zweitens gehen Interest Points, welche nach der Thin Plate Spline Transformation ganz ausserhalb des zweiten Bildes zu liegen kommen, nicht in die Berechnung ein. Unter Berücksichtigung dieser Details ist obige Formel nicht mehr ganz korrekt, widerspiegelt aber dennoch die eigentliche Idee.



Erstes Bild mit Interest Points (rot)



Erstes Bild mit berechneter Thins Plate Spline Transformation vorwärts transformiert, transformierte Interest Points (rot) und ihre diagonal verschobenen Positionen (blau)



Zweites Bild mit transformierten Interest Points des ersten Bildes (rot), verschobenen Positionen (blau) und Punkten (gelb), an deren Stellen schliesslich Feature Descriptoren zur Distanzberechnung herangezogen wurden.

7.8 Ähnlichkeit zu [CR00]

Haili Chui und Anand Rangarajan beschreiben in [CR00] ein zu dem in dieser Semesterarbeit entwickelten verblüffend ähnliches Verfahren, das im folgenden Abschnitt näher gebracht werden soll.

7.8.2 Verfahren aus [CR00]

In [CR00] wird folgende Funktion über dieselben Unbekannten, wie sie in der vorliegenden Semesterarbeit gesucht werden, optimiert:

$$E(p, a, c) = \left(\sum_{i=1}^n \sum_{j=1}^m p_{i,j} \left\| \begin{pmatrix} x_i \\ y_i \end{pmatrix} - \underbrace{\begin{bmatrix} a_{ux} & a_{vx} & a_{tx} \\ a_{uy} & a_{vy} & a_{ty} \end{bmatrix}}_a \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} + \begin{pmatrix} c_{ix} \\ c_{iy} \end{pmatrix} \underbrace{U(\left\| \begin{pmatrix} u_i \\ v_i \end{pmatrix} - \begin{pmatrix} u_j \\ v_j \end{pmatrix} \right\|)}_{=K_{i,j}} \right\|^2 \right) \\ - \zeta \sum_{i=1}^n \sum_{j=1}^m p_{i,j} + T \sum_{i=1}^n \sum_{j=1}^m p_{i,j} \log(p_{i,j}) \\ + \lambda c^T K c + \xi \operatorname{trace}((a-I)^T(a-I))$$

wobei:

$$\forall j=1, \dots, m: \sum_{i=1}^{n+1} p_{i,j} = 1$$

$$\forall i=1, \dots, n: \sum_{j=1}^{m+1} p_{i,j} = 1$$

p_{ij} entspricht dem Gewicht einer soft-Korrespondenz welche den i -ten Punkt des ersten dem j -ten Punkt des zweiten Patterns mit Wahrscheinlichkeit p_{ij} zuordnet ($p_{i,\sigma(i,j)} = w_{i,\sigma(i,j)}$). p_{ij} ist also eine bedingte Wahrscheinlichkeit, dass, gegeben der i -te Punkt im ersten Pattern, dieser Punkt dem j -ten Punkt im zweiten Pattern zugeordnet wird. Darum wurde auch p anstelle der σ -Notation verwendet. Um Outlier zu berücksichtigen wird in den Nebenbedingungen auf $n+1$ respektive auf $m+1$ summiert.

Der erste Term ist ein gewichteter Datenterm, analog zu gewichteten Thin Plate Splines. Der zweite Term schützt vor Null-Zuordnungen. Der dritte Term entspricht der negativen Entropie. Er sichert Positivität der p_{ij} und steuert durch die Temperatur T , wie verlässlich die soft-Korrespondenzen sind. Der vierte Term ist der normale Thin Plate Spline Prior Term. Als letzter Term geht eine Bedingung an die affine Transformation ein. Dieser schützt vor unplausiblen Transformationen (zum Beispiel Bildspiegelungen).

Der Algorithmus funktioniert wie folgt. Es wird ebenfalls mehrmals iteriert, wobei in einer Iteration zuerst die Korrespondenzen, gegeben die vorherige geschätzte Transformation, gesucht werden, danach wird die Transformation gesucht, welche die zuvor bestimmten Korrespondenzpartner möglichst aufeinander abbildet. In jeder nachfolgenden Iteration werden die Parameter T , λ und ξ verkleinert. Dies führt einerseits zu komplexeren Thin Plate Spline Transformationen, da der Krümmungsterm weniger stark belastet wird, andererseits aber auch zu diskriminativeren Korrespondenzen, da durch Abnahme von T eine uniforme Verteilung weniger favorisiert wird. Leitet man die Funktion E für gegebene Transformation (a, c) nach den p_{ij} ab, so erhält man:

$$p_{i,j} = \exp \left(- \frac{\left\| \begin{pmatrix} x_j \\ y_j \end{pmatrix} - g^{(t-1)}(u_i, v_i) \right\|^2 - \zeta}{T} \right)$$

$g^{(t-1)}$: gefundene Thin Plate Spline Transformation im $(t-1)$ -ten Iterationsschritt

Um die Nebenbedingungen zu erfüllen wird die so genannte Sinkhorn Balancing Technik verwendet. Im zweiten Schritt wird dann der Least Squares Ansatz zum Finden der Thin Plate Spline Parameter (a, c) benutzt.

Eine wichtige Tatsache wird in [CR00] erwähnt: Soft-Korrespondenzen und die Thin Plate Spline Lösung sind zwar unabhängig betrachtet optimal, die Kombination von beiden aber nicht! Die gemeinsame Optimierung kann also in einem lokalen Minimum stecken bleiben.

7.8.2 Ähnlichkeiten und Unterschiede

Die Grobstruktur des Algorithmus ist mit dem in dieser Arbeit entwickelten Verfahren identisch. Es werden ebenfalls nacheinander die Korrespondenzen, gegeben die vorherige Transformation, dann die Transformation, gegeben die zuvor berechneten Korrespondenzen, bestimmt. Ebenfalls wird über den Smoothness Parameter der Thin Plate Spline iteriert. Zusätzlich werden die Korrespondenzen in jeder Iteration diskriminativer, da T verkleinert wird. Im Grenzfall für T gegen 0 erhält man sogar anstelle von diskreten Wahrscheinlichkeitsverteilungen eine Permutation, also eine harte 1-to-1 Korrespondenz. In der vorliegenden Arbeit hingegen bleibt die zulässige relative Entropieabnahme zur Gewichtsbestimmung über alle Iterationen hinweg konstant. Beide Verfahren setzen auf soft-Korrespondenzen, in [CR00] geht man allerdings noch einen Schritt weiter. Dort wird für jeden Punkt in einem Pattern, die m bzw. n

nächsten Nachbarn im anderen bestimmt. Es werden also Korrespondenzen in beide Richtungen bestimmt. Dies ist die natürliche Generalisierung von 1-to-1 Korrespondenzen, der Prozess erschwert sich jedoch, da nun Nebenbedingungen erfüllt werden müssen. Zudem werden die Matrizen viel grösser, da nun $m \cdot n$ anstatt $k \cdot n$ Korrespondenzen in die Berechnungen eingehen.

Die Funktion zur Bestimmung der Korrespondenzen mag zuerst unterschiedlich erscheinen. Doch mit Ausnahme des Feature Deskriptor Terms tauchen alle Terme der zuvor beschriebenen Distanz zwischen zwei Punkten auch in diesem Algorithmus auf. Für $\tau = 1$ entspricht der Momentumterm eigentlich der oben angegeben Funktion für p_{ij} . Ein Term für die affine Abbildung wird direkt im letzten Term der Zielfunktion E berücksichtigt. Dass keine Feature Deskriptor Distanzen auftauchen ist offensichtlich, denn in [CR00] wird das Problem der Patternttransformation unabhängig von zu Grunde liegenden Bildern angegangen.

Kapitel 8: Resultate

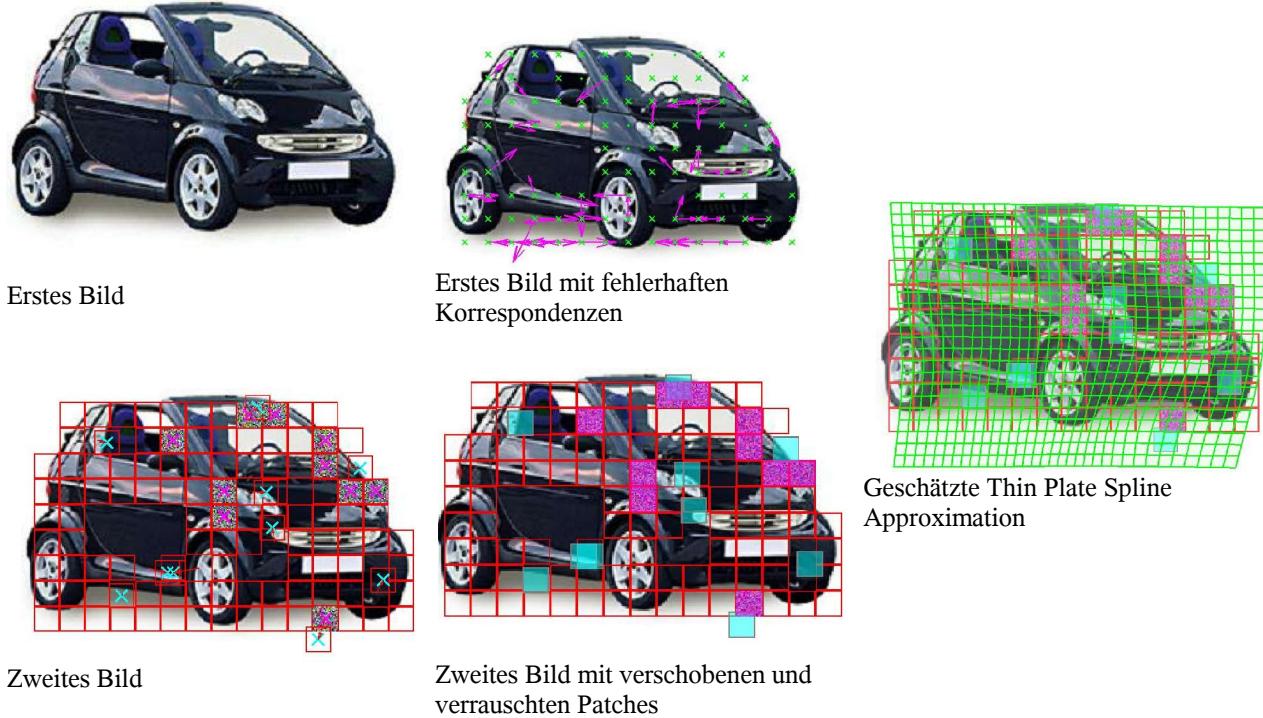
Die mit diesem Verfahren erzielten Resultate waren eher ernüchternd. Im Appendix sind einige Beispiele zu finden. Einige davon funktionierten gut, andere wiederum überhaupt nicht. In diesem Kapitel werden Versuche mit verschiedenen Typen von Bildern zusammengefasst.

8.1 Stabilität des Verfahrens

Um die Stabilität der weighted Thin Plate Spline Implementierung zu testen, wurden einige Tests durchgeführt. Ausgehend vom selben Bild mit denselben Interest Points wurden zum einen zufällig Patches ausgewählt und durch Rauschen ersetzt, zum anderen wurden zufällig Interest Points im XY-Bereich leicht verschoben. Die Interest Points wurden zuvor gefiltert, sodass lediglich genügend Information enthaltende Patches in den Algorithmus eingingen. Es wurden ebenfalls die k -nächsten Nachbarn bestimmt und gewichtet. Es wurde lediglich die Feature Deskriptor Distanz als Kriterium für die Korrespondenzsuche verwendet. Diese Korrespondenzen gingen dann in die gewichtete Thin Plate Spline Schätzung ein.

Die Ergebnisse dieser Versuche waren sehr viel versprechend. Die Stabilität der gewichteten Thin Plate konnte somit empirisch gezeigt werden. Denn die Korrespondenzen sollten in diesem Test sehr gut sein, da ja dieselben Bilder zugrunde liegen. Die Qualität der Thin Plate Spline Approximation kann somit also unabhängig von den Korrespondenzen untersucht werden.

Unten ist ein Beispiel zu sehen. In Cyan dargestellt sind die verschobenen Patches, in Magenta die verrauschten Patches. Mit Rot werden die Ränder der Patches dargestellt. Diese zusätzlichen Farbgebungen gingen natürlich nicht in die Berechnung ein, sie dienen lediglich der Visualisierung. Im mittleren oberen Bild deuten die magentafarbenen Pfeile falsche Korrespondenzen an. Ihre Orientierung gibt die Richtung des falschen Korrespondenzpartners an, ihre Länge das Gewicht dieser Korrespondenz in der Thin Plate Spline Berechnung.

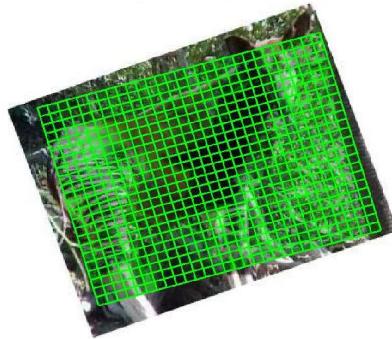


8.2 Künstliche Beispiele

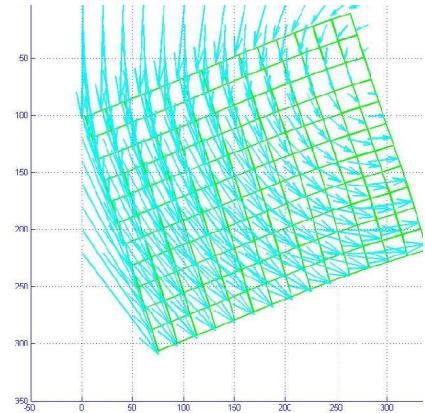
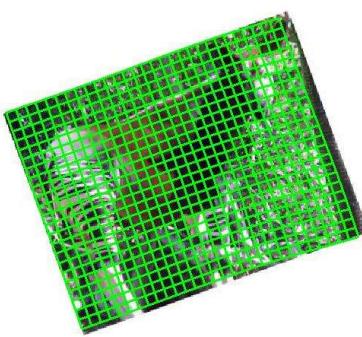
Bevor das entwickelte Verfahren an unterschiedlichen Bildern derselben Kategorie getestet wurde, wurden Versuche mit künstlich transformierten Bildern durchgeführt. Das zweite Bild war dabei eine Translation, Rotation oder Scherung des ersten Bildes. Auch hier zeigten sich gute Resultate. Aber es muss ebenfalls bemerkt werden, dass die Korrespondenzen hier sehr gut waren, denn es lagen wiederum dieselben Bilder zugrunde.



Erstes Bild



Affine Transfomationsschätzung


 Zweites Bild: erstes Bild um 20° Thin Plate Spline Approximation rotiert


Geschätzte Transformation als Vektorfeld dargestellt.

8.3 Bilder aus der Caltech101 Datenbank

Das eigentliche Ziel ist natürlich die Objekterkennung. Hier sahen die Resultate allerdings nicht mehr so viel versprechend aus. Das Verfahren funktionierte überhaupt nicht robust. Eine gute Schätzung für die affine Transformation ist essentiell, doch häufig wurde schon in der Initialisierungsphase eine schlechte oder gar ganz falsche affine Transformation geschätzt.

Zudem liefern die k-nächsten Nachbarn manchmal schlechte Korrespondenzen. Es ist nämlich möglich, dass viele Punkte des ersten Patterns als Korrespondenzpartner im zweiten Pattern denselben Punkt besitzen. Das folgt aus der Tatsache, dass keine Nebenbedingungen an die Korrespondenzen wie zum Beispiel in [CR00] gestellt werden. Dadurch verliert man Vorwissen. Denn es ist ja so, dass Objekte derselben Kategorie meist dieselbe Anzahl eines bestimmten Features aufweisen, zum Beispiel besitzen Autos meist vier Räder, wovon allerdings meist nur zwei sichtbar sind.

Die zuletzt ausgegebene Distanz zwischen den zwei Bildern konnte nicht für eine robuste nächste-Nachbar Klassifizierung verwendet werden: diese Distanzen waren zu stark zufällig. Das mag vielleicht auch daran liegen, dass eine eher einfach Bilddistanzfunktion verwendet wird. Die durchgeführten Versuche liefen so ab, dass zufällig ein Bild aus einer beliebigen Kategorie der Caltech101 Bilddatenbank als erstes Ausgangsbild gewählt wird. Danach werden zufällig drei Bilder derselben Kategorie und drei Bilder von beliebigen anderen Kategorien als zweites Ausgangsbild gewählt. Dem ersten Ausgangsbild wird dann die Kategorie jenes der sechs zweiten Ausgangsbilder zugeordnet, welches die kleinste Distanz liefert. Dies entspricht einer nächster-Nachbar Klassifizierung mit sechs Testbildern, drei davon in derselben Kategorie, drei in beliebig anderen Kategorien.

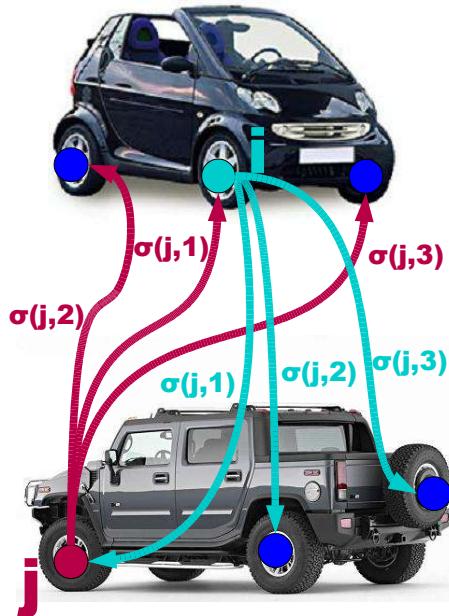
Die Klassifizierung war aber meist zufällig, die Distanzen waren alle sehr nahe beieinander. Auch wenn manchmal eine korrekte Klassifizierung statt fand, war dies noch lange nicht als Erfolg zu werten. Denn es konnte beobachtet werden, dass manchmal bei einer korrekten Klassifizierung die Distanz zwar klein war, die gefundene Transformation aber sehr schlecht war: das transformierte Bild stimmte überhaupt nicht mit dem zweiten Bild überein! Für eine solche schlechte Transformation sollte die Distanz eigentlich hoch sein, war sie aber nicht immer.

Um ein besseres Bild über das Verfahren angewendet auf Bilder der Caltech101 Datenbank zu vermitteln, sind im Appendix einige Beispiele aufgeführt.

Kapitel 9: Schlussfolgerung und weiterführende Ideen

Thin Plate Splines sind eine gute Wahl zur Transformationsschätzung. Die Resultate lassen den Schluss zu, dass das Problem am Bestimmen von möglichst guten Korrespondenzen liegt!

Die Idee, die k-nächsten Nachbarn als Korrespondenzen zu verwenden, ist sicherlich ein viel versprechender Weg. Denn dadurch wird man nicht gezwungen, alles-oder-nichts Entscheidungen zu treffen, wie es bei harten 1-to-1 Korrespondenzen der Fall ist. Andererseits ist die Anzahl der Korrespondenzen im Vergleich zu den soft 1-to-1 Korrespondenzen, wie sie in [CR00] verwendet werden, viel geringer. K-nächste Nachbar Korrespondenzen stellen also einen guten Kompromiss zwischen harten und soft 1-to-1 Korrespondenzen dar. Soft Korrespondenzen implizieren aber eine gewichtete Form der Least Squares Thin Plate Spline Approximation, welche in dieser Arbeit erfolgreich hergeleitet wurde. Aber in der Form, wie k-nächsten Nachbar Korrespondenzen in dieser Arbeit verwendet werden, führen sie zu schlechten Ergebnissen. Ich vermute, dass es unvermeidbar ist, Nebenbedingungen wie in [CR00] zu berücksichtigen. Zudem sollten die Korrespondenzen nicht nur in eine Richtung gehen, sondern auch vom anderen Bild zurück. Das heisst, dass die k-nächsten Nachbarn jedes Punktes im ersten Pattern, aber auch die k-nächsten Nachbarn jedes Punktes im zweiten Pattern bestimmt werden sollten. Dies wurde in dieser Arbeit vernachlässigt.



Anstatt die Gewichte der Korrespondenzen über alle Korrespondenzpaar zu normalisieren, ist es zudem von Vorteil, die Gewichte lediglich bedingt auf den einen Punkt zu normalisieren, so wie es in [CR00] gemacht wird. Das erfordert aber eine spezielle Technik, zum Beispiel die ebenfalls in [CR00] benutzte Sinkhorn Balancing Technik. Durch solche Gewichte, oder äquivalent eben auch bedingte Wahrscheinlichkeiten, wird die gewichtete Least Squares Thin Plate Spline Approximation voll ausgenutzt. Eine solche Erweiterung birgt meiner Meinung nach das grösste Potential.

Um gute Korrespondenzen zu finden, egal welcher Art, ist eine gut modellierte Distanzfunktion nötig. In dieser Arbeit, aber auch in vielen anderen, basiert diese Distanzfunktion um Korrespondenzen zu finden lediglich auf Summanden, die von zwei Punkten abhängen. Eine interessante Erweiterung wird in [BBM05] vorgeschlagen. Dort wird eine Funktion modelliert, welche von Paaren von Korrespondenzpaaren abhängig ist, also von vier Punkten: jeweils zwei im ersten und jeweils zwei im zweiten Pattern. Dadurch können Zusammenhänge zwischen Korrespondenzpaaren mit eingebracht werden. So gehen in [BBM05] geometrische Zusammenhänge zwischen Korrespondenzpaaren in die Korrespondenzsuche ein. Es werden harte 1-to-1 Korrespondenzen gesucht, was zu einem Integer Quadratic Programming Problem führt, welches mittels einer Heuristik gelöst wird. Es wäre sicherlich interessant, diesen Ansatz auf soft k-nächste Nachbarn Korrespondenzen zu erweitern.

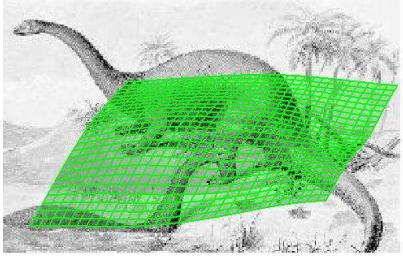
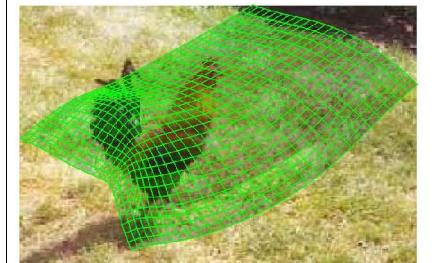
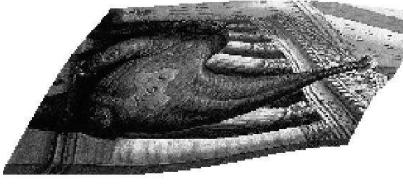
Wie schon im vorigen Kapitel bemerkt, ist die Schätzung einer guten affinen Transformation essentiell. Doch dies ist bereits für sich ein sehr schweres Problem. Vielleicht würde sich die Schätzung der affinen Transformation in die Thin Plate Spline Schätzung einbauen lassen, denn affine Transformationen sind ja ein Spezialfall der Thin Plate Spline

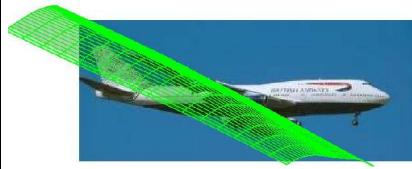
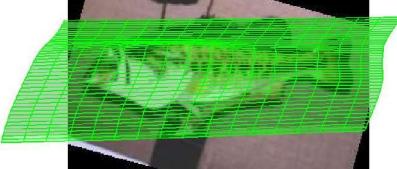
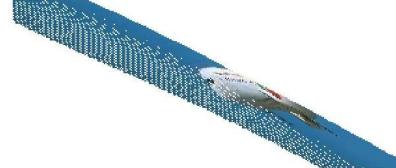
Transformationen. Principal Warps finden im Moment keine Anwendung. Aber vielleicht könnten sie sich zur „Dimensionsreduzierung“ bei soft Korrespondenzen als nützlich erweisen. Die verwendete Distanzfunktion zwischen zwei Bildern ist sehr einfach gehalten. Ich vermute, dass auch da Verbesserungsmöglichkeiten verborgen liegen. Denn diese Distanz wird schliesslich bei der nächsten-Nachbar Klassifizierung verwendet.

Appendix A: Beispiele

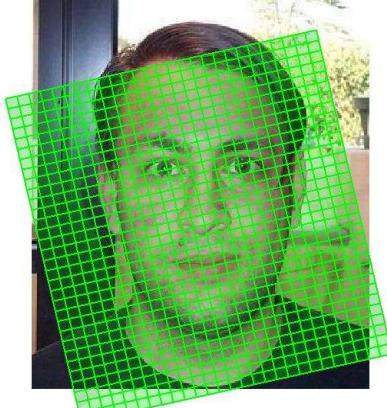
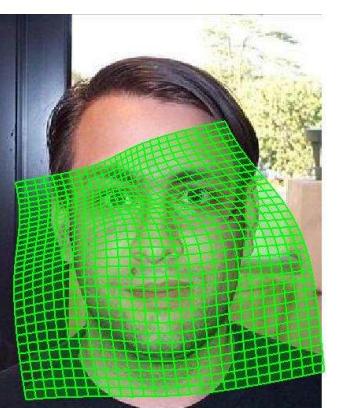
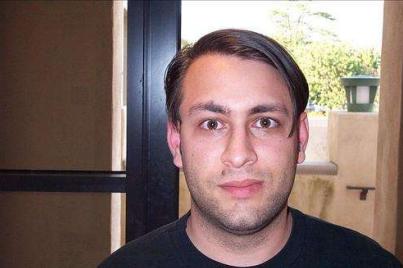
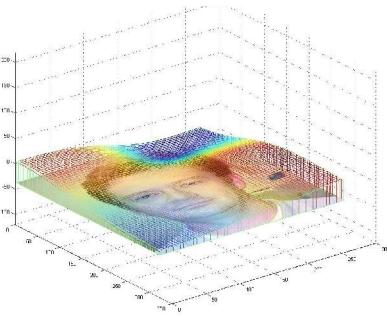
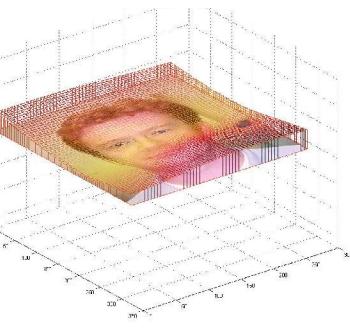
In diesem Appendix befinden sich einige Beispiele der Objektklassifizierungsmethode. Alle Beispiele sind nicht subjektiv ausgesucht worden, sondern alle wurden nacheinander berechnet und hier aufgeführt. Es wurden also keine speziell guten oder schlechten Beispiele herausgesucht, sondern um einen eher repräsentativen Eindruck der Funktionsweise des Algorithmus zu erhalten, wurde eine zufällige Auswahl getroffen. Alle Beispiele wurden mit den Standardwerten für die Parameter berechnet, die Anzahl nächster Nachbarn pro Punkt betrug drei.

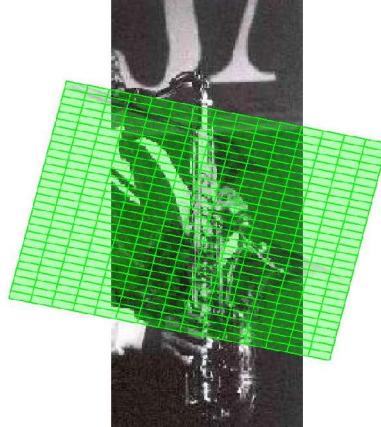
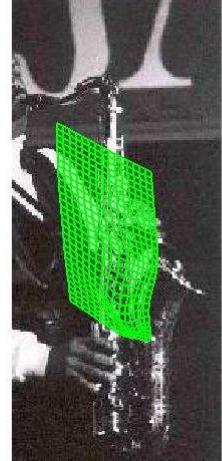
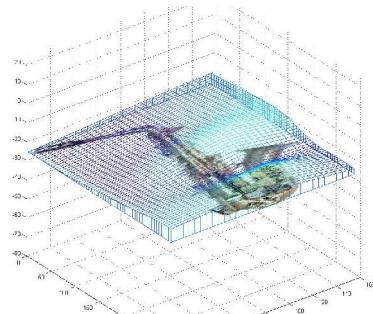
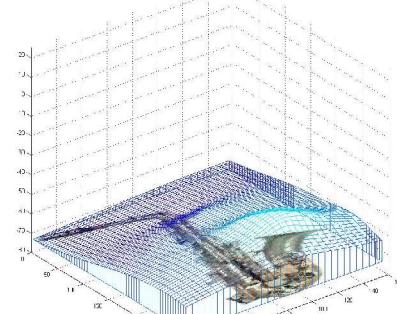
Zuerst folgen zwei 1-nächster Nachbar Klassifizierungsbeispiele. Dabei wurde ein Bild zufällig aus einer zufälligen Kategorie der Caltech101 Bilddatenbank ausgewählt und sowohl mit drei zufällig gewählten Bildern derselben Klasse, als auch mit drei zufällig gewählten Bildern von anderen Klassen verglichen. Dargestellt sind jeweils der nächste Nachbar derselben Kategorie und der nächste Nachbar der drei fremden Kategorien.

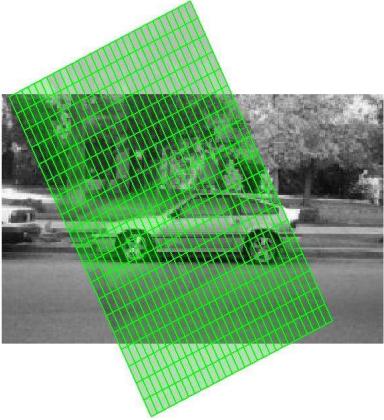
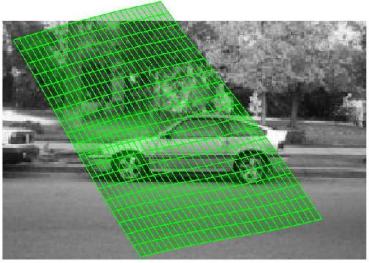
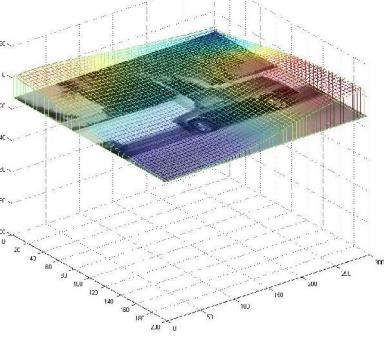
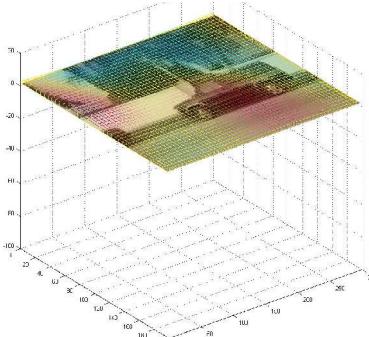
		
Erstes Bild	Nächstes Bild derselben Kategorie	Nächstes Bild der drei fremden Kategorien
Distanz	1.0101	1.0339
Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes		
Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild		

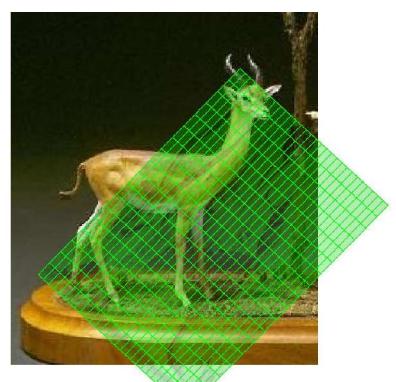
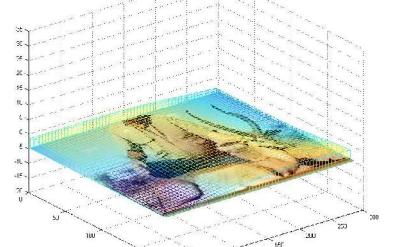
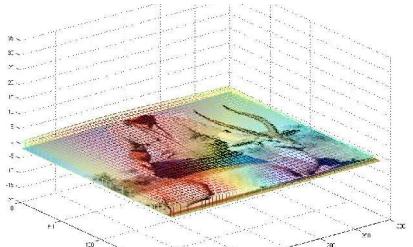
		
Erstes Bild	Nächstes Bild derselben Kategorie	Nächstes Bild der drei fremden Kategorien
Distanz	1.2735	1.4656
Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes		
Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild		

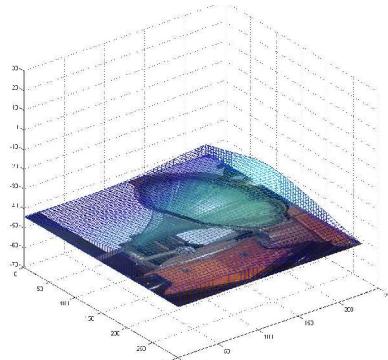
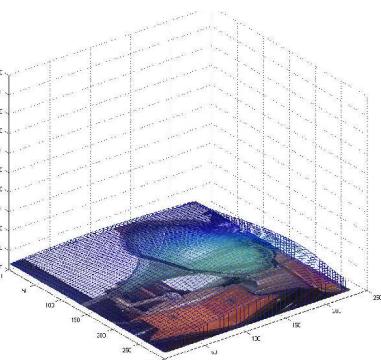
Nun folgen ein paar Beispiele wobei die beiden Bilder wiederum zufällig ausgewählt wurden, aber von derselben, zufällig gewählten Kategorie her stammen. Dargestellt sind neben der geschätzten affinen Transformation der Initialisierungsphase, der geschätzten Thin Plate Spline Transformation und dem vorwärts transformierten ersten Bild auch die Stärke des nicht-linearen Anteils der gefundenen Thin Plate Spline Transformation, jeweils einmal für die x- und einmal für die y-Thin Plate Spline. Für die letzteren beiden Plots wurde der affine Teil der Thin Plate Spline nicht berücksichtigt und der nicht-lineare Teil der x- und y-Thin Plate Spline als z-Koordinate dargestellt.

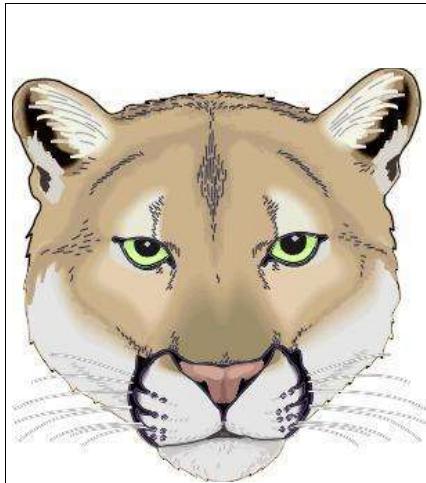
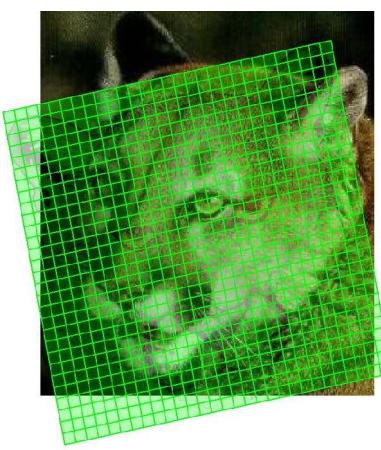
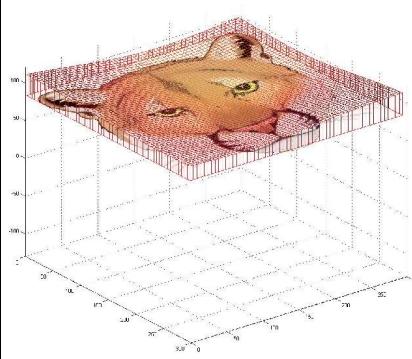
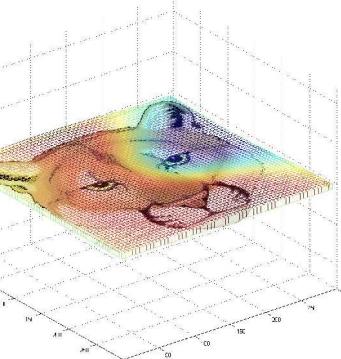
		
Erstes Bild	Zweites Bild mit als Gitter angedeuteter affiner Transformation des ersten Bildes	Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes
		Distanz = 0.76671
Zweites Bild	Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild	
	 x-Thin Plate Spline als Funktion von Z.	 y-Thin Plate Spline als Funktion von Z.

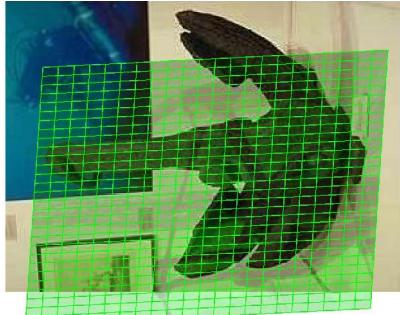
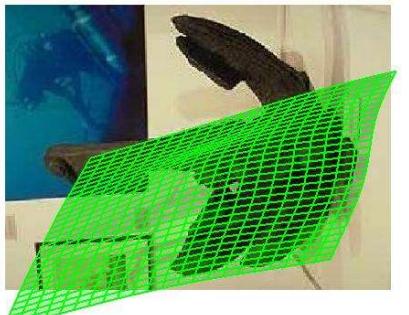
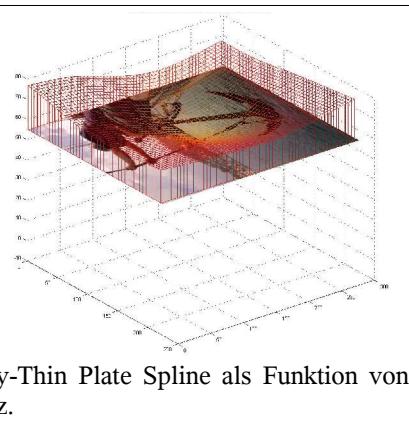
 <p>Erstes Bild</p>	 <p>Zweites Bild mit als Gitter angedeuteter affiner Transformation des ersten Bildes</p>	 <p>Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes</p>
 <p>Zweites Bild</p>	 <p>Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild</p>	<p>Distanz = 0.76671</p>
	 <p>x-Thin Plate Spline als Funktion von z.</p>	 <p>y-Thin Plate Spline als Funktion von z.</p>

 <p>Erstes Bild</p>	 <p>Zweites Bild mit als Gitter angedeuteter affiner Transformation des ersten Bildes</p>	 <p>Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes</p>
 <p>Zweites Bild</p>	 <p>Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild</p>	<p>Distanz = 0.92885</p>
	 <p>x-Thin Plate Spline als Funktion von z.</p>	 <p>y-Thin Plate Spline als Funktion von z.</p>

 <p>Erstes Bild</p>	 <p>Zweites Bild mit als Gitter angedeuteter affiner Transformation des ersten Bildes</p>	 <p>Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes</p>
 <p>Zweites Bild</p>	 <p>Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild</p>	<p>Distanz = 0.71729</p>
	 <p>x-Thin Plate Spline als Funktion von z.</p>	 <p>y-Thin Plate Spline als Funktion von z.</p>

 <p>Erstes Bild</p>	 <p>Zweites Bild mit als Gitter angedeuteter affiner Transformation des ersten Bildes</p>	 <p>Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes</p>
 <p>Zweites Bild</p>	 <p>Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild</p>	<p>Distanz = 0.86971</p>
	 <p>x-Thin Plate Spline als Funktion von Z.</p>	 <p>y-Thin Plate Spline als Funktion von Z.</p>

 <p>Erstes Bild</p>	 <p>Zweites Bild mit als Gitter angedeuteter affiner Transformation des ersten Bildes</p>	 <p>Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes</p>
 <p>Zweites Bild</p>	 <p>Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild</p>	<p>Distanz = 1.0566</p>
	 <p>x-Thin Plate Spline als Funktion von z.</p>	 <p>y-Thin Plate Spline als Funktion von z.</p>

		
<p>Erstes Bild</p>	<p>Zweites Bild mit als Gitter angedeuteter affiner Transformation des ersten Bildes</p>	<p>Zweites Bild mit als Gitter angedeuteter Thin Plate Spline Transformation des ersten Bildes</p>
		<p>Distanz = 1.2046</p>
<p>Zweites Bild</p>	<p>Mittels gefundener Thin Plate Spline vorwärts transformiertes erstes Bild</p>	 <p>x-Thin Plate Spline als Funktion von z.</p> <p>y-Thin Plate Spline als Funktion von z.</p>

Quellenverzeichnis

- [B87] I. Biederman
Recognition-by-components: A theory of human image understanding
Psychological Review, 94(2), pp. 115-147, 1987
- [B89] F. L. Bookstein
Principal Warps: Thin-Plate Splines and the Decomposition of Deformations
IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 11, no. 6, June 1989
- [BBM05] A. C. Berg, T. L. Berg, J. Malik
Shape Matching and Object Recognition using Low Distortion Correspondence
IEEE Computer Vision and Pattern Recognition (CVPR'05) – Volume 1, pp. 26-33, 2005
- [BM01] A. C. Berg, J. Malik
Geometric Blur for Template Matching
IEEE CVPR, pp. 607-614, 2001
- [BMP02] S. Belongie, J. Malik, J. Puzicha
Shape Matching and Object Recognition Using Shape Contexts
Technical Report UCB//CSD00-1128, UC Berkeley, 2001
- [CBC01] J.C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McClallum
T. R. Evans
Reconstruction and Representation of 3D Objects with Radial Basis Functions
SIGGRAPH 2001
- [CR00] H. Chui, A. Rangarajan
A New Algorithm for Non-Rigid Point Matching
Proceedings of CVPR, June 2000
- [EPP00] T. Evgeniou, M. Pontil, T. Poggio
Regularization Networks and Support Vector Machines
Advances in Computational Mathematics 13 (2000) 1, pp. 1-50.
- [FFP04] L. Fei-Fei, R. Fergus, P. Perona
Learning generative visual models from few training examples: an incremental
Bayesian approach tested on 101 object categories
IEEE CVPR 2004, Workshop on Generative-Model Based Vision, 2004
- [G00] J. Gallier
Curves and Surfaces in Geometric Modeling: Theory and Algorithms
Morgan Kaufmann Publishers, 2000
- [GJP95] F. Girosi, M. Jones, T. Poggio
Regularization Theory and Neural Networks Architectures
Neural Computation, vol. 7, number 2, pp. 219-269, 1995
- [KSJ00] E. R. Kandel, J. H. Schwartz, T. M. Jessel
Principles of Neural Science, 4th Edition
ISBN: 0838577016, McGraw-Hill Medical, 2000
- [L03] I. Laptev
Computational Vision and Active Perception Laboratory (CVAP)
Dept. of Numerical Analysis and Computer Science
KTH, SE 100 44 Stockholm, Sweden

- [MFTM01] D. Martin, C. Fowlkes, D. Tal, J. Malik
A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics
Proc. 8th International Conference Computer Vision, Volume 2, pp. 416-423, 2001
- [MS05] K. Mikolajczyk, C. Schmid
Scale and Affine invariant interest point detectors.
IJC V 1(60):63-86, 2004.
- [OB05] B. Ommer, J. M. Buhmann
Object Categorization by Compositional Graphical Models
Energy Minimization Methods in Computer Vision and Pattern Recognition, LNCS 3757, pp. 235-250, Springer, 2005
- [T86] A. Treisman
Features and Objects in Visual Processing
Sci Am 255(5), pp. 114-125, 1986
- [VH99] R. C. Veltkamp, H. Hagedoorn
State of the Art in Shape Matching
Technical Report UU-CS-1999-27, Utrecht, 1999
- [VOC05] M. Everingham and others
The PASCAL Object Recognition Database Collection: The VOC 2005 Database:
Dataset 1 and Testset 2
<http://www.pascal-network.org/challenges/VOC/databases.html>, 2005