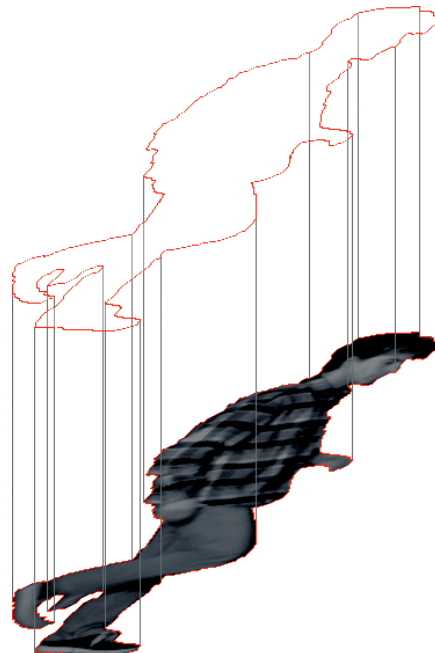


# Automatische Silhouetten-Extraktion aus Videosequenzen

Marco Nef  
Prof. Dr. A. Grün, Nicola D'Appuzzo

Semesterarbeit SS2001  
Institut für Geodäsie und Photogrammetrie  
Eidgenössische Technische Hochschule Zürich





# Zusammenfassung

Im Rahmen einer Semesterarbeit im Nebenfach Photogrammetrie wurde die Aufgabe gestellt, einen Algorithmus für die Extraktion von Silhouetten aus Videoströmen zu entwickeln und zu implementieren. Diese Schrift beschreibt den realisierten *MediPicture*-Algorithmus, eine robuste Methode, um mittels Hintergrundextraktion das sich im Vordergrund befindliche Objekt zu erkennen und anschliessend dessen Silhouette zu extrahieren.

# Abstract

It was the task of a semester thesis in the minor photogrammetry to develop and implement an algorithm which can extract silhouettes from video streams. This paper describes the *MediPicture* algorithm that was developed. It is a robust method that uses background extraction to detect the object in the foreground and to finally extract its silhouette.

# Danksagung

Ein herzlicher Dank geht an Nicola D'Appuzzo vom Institut für Geodäsie und Photogrammetrie an der Eidgenössischen Technischen Hochschule ETH in Zürich, Schweiz, der die Betreuung dieser Arbeit übernahm und mir gerne für Fragen zur Verfügung stand.

Ausserdem möchte ich den beiden Korrekturlesern Nicole Tobler und Claudio Nef einen grossen Dank spenden.

Marco Nef, Zürich, den 3. Juli 2001



# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Aufgabenstellung . . . . .	1
1.2	Übersicht . . . . .	2
<b>2</b>	<b>MediPicture-Algorithmus</b>	<b>3</b>
2.1	Das $W^4$ -System . . . . .	3
2.2	Hintergrund-Extraktion . . . . .	4
2.3	Filterung . . . . .	5
2.4	Silhouetten-Extraktion . . . . .	7
<b>3</b>	<b>Resultate</b>	<b>9</b>
3.1	SilExtractor . . . . .	9
3.2	Bewertung . . . . .	10
<b>4</b>	<b>Zusammenfassung</b>	<b>13</b>
4.1	Rückblick . . . . .	13
4.2	Ausblick . . . . .	13
	<b>Quelltexte des Algorithmus</b>	<b>15</b>
	<b>Bibliotheken</b>	<b>17</b>
	Image . . . . .	18
	MediPicture . . . . .	19
	Movie . . . . .	20
	Silhouette . . . . .	21
	<b>Literaturverzeichnis</b>	<b>22</b>
	<b>Index</b>	<b>25</b>



# 1

## Einleitung

Als Informatikstudent an der Eidgenössischen Technischen Hochschule ETH in Zürich habe ich die Verpflichtung, im Fachstudium ab dem fünften Semester ein Nebenfach zu wählen und dort eine bestimmte Anzahl Vorlesungen zu besuchen. Ausserdem muss man eine Semesterarbeit zu einem Themenbereich des Nebenfaches schreiben.

Vor rund zwei Jahren entschied ich mich für die Photogrammetrie als Nebenfach. Dieser Bereich faszinierte mich als Informatiker insbesondere wegen der interessanten algorithmischen Fragestellungen. Daneben war es für mich erstrebenswert, neben meiner Vertiefung in Computergraphik, die sich der Darstellung von graphischen Daten widmet, auch noch die „andere Seite“ des Spektrums in der graphischen Datenverarbeitung kennen zu lernen. Naheliegend wäre natürlich die *Computer Vision* gewesen, welche die Aquisition von Bildern behandelt. Die Photogrammetrie als Werkzeug der *Computer Vision* lockte mich als enger Teilbereich aber wesentlich mehr.

Die vorliegende Arbeit ist meine Semesterarbeit in Photogrammetrie. Das gewählte Thema hat allerdings keinen engen Bezug zur Kernkompetenz der Photogrammetrie, der Extraktion von dreidimensionalen Objektdaten aus Photographien. Vielmehr wird ein typisches Thema der *Computer Vision* behandelt, was aber nicht minder interessante Überlegungen notwendig macht.

### 1.1 Aufgabenstellung

Im Rahmen des Projektes REBOMO (Realistic Body Modeling from Video Sequences) [1, 2, 3, 4, 5, 6], eine Zusammenarbeit des Institutes für Geodäsie und Photogrammetrie an der ETH in Zürich und dem Computer Graphics LAB an der ETH in Lausanne, wird eine Person, die sich frei bewegt, gleichzeitig von drei CCD-Kameras gefilmt. Mit photogrammetrischen Methoden ist es möglich, aus den digitalisierten Bildsequenzen dreidimensionale Informationen sowohl des menschlichen Körpers als auch von dessen

Bewegung zu extrahieren. Für diesen Prozess ist es heutzutage erforderlich, manuell für jedes Einzelbild der Sequenz die Silhouette des menschlichen Körpers zu definieren.

Ziel der Semesterarbeit ist es, das aufwendige und mühsame manuelle Definieren von Silhouetten zu automatisieren. Dazu musste ein Algorithmus mit der entsprechenden Funktionalität entwickelt und implementiert werden.

Die Arbeit war arbeitstechnisch in zwei Teile unterteilt: In den ersten Wochen galt es, im Internet nach möglichen Lösungsansätzen zu suchen. Weil diese Problemstellung in der *Computer Vision* nicht neu ist, war es möglich, diverse Papers zu diesem Thema zu finden, welche in einer separaten Arbeit detailliert beschrieben wurden [7]. Anhand der Analyse von existierenden Arbeiten wurde eine Designentscheidung gefällt, welche sich auf [8] stützt.

## 1.2 Übersicht

Nach einer detaillierten Einführung und Beschreibung des *MediPicture*-Algorithmus im nachfolgenden Kapitel werden die erzielten Resultate aufgezeigt und einer kritischen Bewertung unterzogen. Schliesslich werden noch Methoden beschrieben, mit denen die Resultate weiter verbessert werden können.

# 2

## MediPicture-Algorithmus

Der *MediPicture*-Algorithmus wendet mehrere Verarbeitungsstufen an, um die Silhouette eines Objektes aus einem Bild zu extrahieren. In diesem Kapitel werden nach einer Beschreibung der Silhouetten-Extraktion im  $W^4$ -System die verschiedenen Schritte des *MediPicture*-Algorithmus detailliert in der Reihenfolge, wie sie auf ein Bild anzuwenden sind, beschrieben.

### 2.1 Das $W^4$ -System

Wie bereits in der Einleitung erwähnt, wurde als Grundlage für den zu entwickelnden Algorithmus zur automatischen Extraktion von Silhouetten aus Videosequenzen die in [8] beschriebene Methode verwendet. Das  $W^4$ -System ermöglicht die Erkennung und Verfolgung von Personen in einer Umgebung ausserhalb des Labors in Echtzeit (siehe auch [9, 10]). Es wurde an der Universität von Maryland, USA, entwickelt.

Die Voraussetzung, um mit diesem System überhaupt Silhouetten in Bildern erkennen zu können, ist eine Hintergrund-Extraktion (*background extraction*). Das ist eine Qualifizierung jedes einzelnen Pixels eines Bildes in eine der beiden Kategorien *Vordergrund* oder *Hintergrund*. Um dies erreichen zu können, wurde ein statistisches Modell der Hintergrundszene entwickelt.

Die Hintergrundszene wird dermassen modelliert, dass während einer Trainingsphase von mehreren Sekunden drei Werte ( $M, N, D$ ) für jedes Pixel berechnet und zugeteilt werden.  $M$  ist die minimale Intensität des entsprechenden Pixels, welche während der Trainingsphase beobachtet wurde,  $N$  ist entsprechend die maximale beobachtete Intensität und  $D$  ist der grösste Intensitätsunterschied eines Pixels, der während der Beobachtungszeit zwischen zwei aufeinanderfolgenden Bildern auftrat. Die so entstehende Repräsentation wird im folgenden *MediPicture* genannt. Sollte die nachfolgende Aufnahme für die Silhouetten-Extraktion über eine längere Zeit

erfolgen, so muss der Film in kürzere Teile zerlegt werden, für welche jeweils ein separates *MediPicture* berechnet wird. Dadurch können Varianzen in der Beleuchtung (z.B. Wolken, Sonnenstand) berücksichtigt werden.

Ein Pixel  $x$  des Bildes  $I$  ist nun genau dann ein Vordergrundpixel, also zum zu erkennenden Objekt gehörend, wenn mindestens eine der beiden Bedingungen 2.1 oder 2.2 erfüllt ist:

$$|M(x) - I(x)| > D(x) \quad (2.1)$$

$$|N(x) - I(x)| > D(x) \quad (2.2)$$

Bildlich ausgedrückt bedeuten diese Bedingungen, dass ein Pixel als Vordergrund erkannt wird, wenn die Differenz seiner Intensität zur maximalen respektive minimalen Intensität während der Beobachtungsphase grösser ist, als der grösste Intensitätsunterschied des besagten Pixels zwischen zwei aufeinanderfolgenden Bildern im Trainingsfilm. Nach erfolgter Hintergrund-Extraktion erscheint das prozessierte Bild in den zwei Farben schwarz für Hintergrund und weiss für Vordergrund.

Auf diese Art lässt sich eine schnelle und einfache Qualifizierung der Pixel eines Bildes realisieren. In Abbildung 2.1 wird hingegen das überraschenderweise höchst unbefriedigende Resultat gezeigt.



Abbildung 2.1: Ergebnis nach Anwendung der beschriebenen Hintergrund-Extraktion nach [8]

## 2.2 Hintergrund-Extraktion

Um erfolgreich die Silhouette eines Objektes aus einem Bild extrahieren zu können, ist es erforderlich, dass die Genauigkeit der Hintergrund-Extraktion sehr hoch ist. Dies ist dann der Fall, wenn nach der Anwendung des Algorithmus auf ein Bild die Grenzen zwischen Vorder- und Hintergrundbereichen scharf abgebildet sind und genau auf der Silhouette liegen. Ausserdem soll

möglichst wenig Rauschen zurückbleiben. Beide Voraussetzungen für den nächsten Schritt der Silhouetten-Extraktion sind in Abbildung 2.1 offensichtlich nicht erfüllt.

Da die in [8] beschriebene Methode keine brauchbaren Ergebnisse liefert, in der Theorie hingegen funktionieren sollte (und es im  $W^4$ -System auch in der Praxis zu tun scheint, wie die im erwähnten Paper gezeigten Abbildungen belegen), muss eine Modifikation der Bedingungen 2.1 und 2.2 vorgenommen werden.

Ein Blick auf Abbildung 2.1 zeigt, dass zu viele Bildpunkte als Vordergrund definiert werden (weisse Bereiche). Ausserdem zeigt eine genauere Untersuchung der berechneten *MediPicture*-Daten, dass der maximale Intensitätsunterschied zwischen zwei Bildern für die meisten Pixel sehr klein oder gar null ist. Schon die kleinste Abweichung von der durchschnittlichen Beleuchtung resultiert deshalb in einer eventuell falschen Zuteilung des betroffenen Pixels zum Vordergrund. Solche minimale Abweichungen der Pixelintensitäten innerhalb eines Films von mehreren Sekunden sind sehr wahrscheinlich und können verschiedene Ursachen haben: Ungenügende Farbgenauigkeit des CCD-Sensors in der Kamera, Änderungen der Beleuchtung, aber auch eine unzureichend fixierte Platzierung der Kamera, was zu einem Ruckeln führt, welches ebenso als grösserer Intensitätswechsel wahrgenommen wird.

Das Ziel muss folglich sein, die Menge dieser Vordergrundpixel zu verkleinern. Die im *MediPicture*-Algorithmus angewandte Erweiterung der Bedingungen 2.1 und 2.2 ist das Hinzufügen einer Toleranz  $tol$ , welche den zum Vordergrund gehörenden Intensitätsbereich verkleinert:

$$|M(x) - I(x)| > |D(x) + tol| \quad (2.3)$$

$$|N(x) - I(x)| > |D(x) + tol| \quad (2.4)$$

In Tests mit verschiedenen Videosequenzen hat sich gezeigt, dass eine Toleranz  $tol = 7$  gut geeignet ist, um für die Weiterverarbeitung brauchbare Zwischenergebnisse zu erzielen. Die Toleranz muss aber unbedingt für jede Anwendung manuell abgeschätzt und eingestellt werden. Das in Abbildung 2.2 gezeigte Resultat ist für die Weiterverarbeitung wesentlich besser geeignet als Abbildung 2.1.

## 2.3 Filterung

Nach der Hintergrund-Extraktion ist es notwendig, das Bild zu filtern, um Rauschen zu entfernen. Wird dies unterlassen, so können die unter Umständen noch zahlreich vorhandenen Hintergrundelemente, welche fälschlicherweise als Vordergrund qualifiziert wurden, das Resultat der Silhouetten-Extraktion verfälschen. Ebenso können natürlich auch Pixel als Hintergrund qualifiziert worden sein, die aber (für das menschliche Auge offensichtlich)

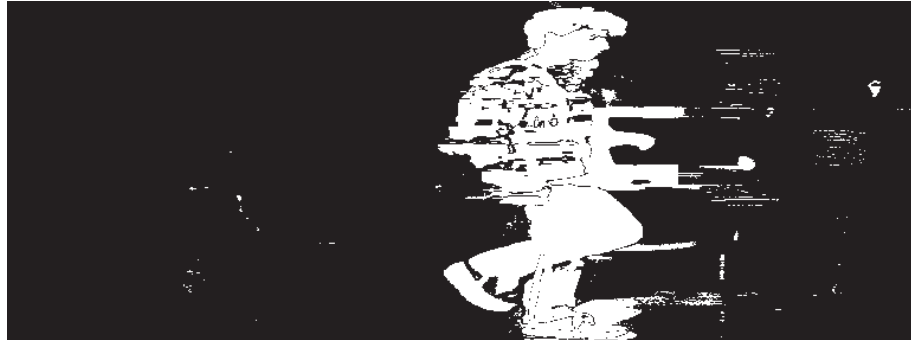


Abbildung 2.2: Ungefiltertes Ergebnis nach Anwendung der Hintergrund-Extraktion mit Toleranz  $tol = 7$

in einem Vordergrundbereich enthalten sind. Ein typisches Beispiel für viel Rauschen ist der mehrheitlich schwarze Hintergrund in der linken oberen Ecke von Abbildung 2.1, der durch weisse Vordergrundpixel „aufgelockert“ wird.

Ein einfacher Filterkernel (siehe Abbildung 2.3) summiert die mit der Filtermatrix gewichteten Nachbarpixel des gerade betrachteten Bildelements. Der Bildpunkt selber wird mit zweifachem Gewicht zur Summe hinzuaddiert. Angenommen wird dabei, dass die Hintergrundpixel den Wert 0, die Vordergrundelemente den Wert 1 haben. Erreicht oder überschreitet die berechnete Summe einen zu definierenden Grenzwert  $minSupport$ , so ist der Bildpunkt ein Vordergrundelement, ansonsten ein Hintergrundelement.

1	1	1
1	2	1
1	1	1

Abbildung 2.3: Gewichte der Zellen eines einfachen Filterkernels

Die zum Filterkernel gehörende Auswertung sieht in C-Syntax ausgedrückt folgendermassen aus:

$$val = \left( \sum_{i=1}^9 I(x_i) \geq minSupport \right) ? Vordergrund : Hintergrund$$

wobei die  $x_i$  die im Filter betrachteten Pixel des Bildes  $I$  sind.

In der Praxis hat sich ein Grenzwert  $minSupport = 5$  bewährt, welcher dazu führt, dass jedes Vordergrundpixel mit weniger als drei Nachbarn, die ebenfalls zum Vordergrund gehören, dem Hintergrund zugewiesen wird.

Ebenso kann ein Hintergrundpixel mit mindestens fünf Nachbarn im Vordergrund selber in den Vordergrund gerückt werden.

Ein weiterer Effekt der Filterung besteht in der Glättung hochfrequenter Randbereiche des zu extrahierenden Objektes. Unter der Annahme, dass die meisten realen Objekte (auch der Mensch) eine eher niederfrequente Oberfläche haben, ist dieser Nebeneffekt sehr willkommen.

## 2.4 Silhouetten-Extraktion

Als letzter Schritt des *MediPicture*-Algorithmus kommt die eigentliche Silhouetten-Extraktion an die Reihe. Das für diesen Schritt zur Verfügung stehende Bild, das Resultat der beiden oben beschriebenen Schritte, besteht aus klar getrennten Vorder- und Hintergrundbereichen. Ausserdem wurden mit der Filterung das Rauschen entfernt und die Objektgrenze geglättet. Das zu extrahierende Objekt ist deshalb von Auge als klar vom Hintergrund abgegrenzt erkennbar.

Um die Silhouette im Bild auffinden zu können, ist es hilfreich, die Objektgrenze speziell zu bezeichnen. Ein Bildelement an der Grenze eines Objektes zeichnet sich dadurch aus, dass es selber zum Vordergrund gehört, aber mindestens einen Nachbarn im Hintergrund hat. Nach einer solchen Qualifizierung der Pixel auf der Objektgrenze können diese Bildpunkte im Uhrzeiger- oder Gegenuhrzeigersinn der Reihe nach eingesammelt und der Silhouette zugeführt werden, bis wieder der Startpunkt erreicht wird. Dadurch ergibt sich immer eine geschlossene Silhouette.

Ein Problem, das dabei auftreten kann, ist in Abbildung 2.4 dargestellt: Bei der Verfolgung der Objektgrenze (grün) trifft der Algorithmus an eine Stelle, welche eine Schlaufe bildet (1). Da dieses Problem für den Algorithmus zu diesem Zeitpunkt noch nicht erkennbar ist, wird die Schlaufe vorerst durchlaufen, worauf irgendwann nur noch Pixel angetroffen werden, welche bereits zur Silhouette gehören (2). Um dies feststellen zu können, müssen alle besuchten Grenzpunkte als solche markiert werden.

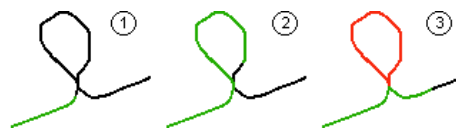


Abbildung 2.4: Schlaufe in der Objektgrenze

An dieser Stelle muss nun eine Entscheidung getroffen werden, wie fortgefahren werden soll. In der Implementierung zu dieser Arbeit wird eine solche Schlaufe wieder aus der Silhouette entfernt (rot) und anschliessend die Objektgrenze weiterverfolgt (3). Diese Entscheidung ist deshalb plau-

sibel, weil eine Fläche, welche nur über eine dünne Linie von der Breite eines Pixels mit dem Objekt verbunden ist, kaum zu einem realen Objekt gehören kann. Selbstverständlich sind andere Entscheidungen möglich und können ebenfalls zu guten Ergebnissen führen.

Damit schliesslich eine möglichst kompakte Silhouette resultiert, wird empfohlen, bei jeweils drei aufeinander folgenden und auf einer Geraden liegenden Silhouettenpunkten den mittleren zu entfernen. Dies kann so lange gemacht werden, bis keine drei aufeinander folgende Punkte mehr auf einer Geraden liegen.

Abbildung 2.5 präsentiert das Schlussresultat der Silhouetten-Extraktion präsentiert. Eine Diskussion folgt im nächsten Kapitel.



Abbildung 2.5: Extrahierte Silhouette

# 3

## Resultate

Nachdem nun bekannt ist, wie man Silhouetten aus Videosequenzen extrahieren kann, werden in diesem Kapitel die erreichten Resultate demonstriert. Vor der eigentlichen Diskussion der Resultate wird eine Übersicht über die Funktionen der Test- und Demonstrations-Applikation *SilExtractor* gegeben, mittels welcher alle in dieser Arbeit präsentierten Resultate realisiert wurden.

### 3.1 SilExtractor

Um den entwickelten und implementierten *MediPicture*-Algorithmus testen zu können, wurde im Rahmen dieser Semesterarbeit zusätzlich das umfangreiche Test- und Demonstrations-Programm *SilExtractor* entwickelt. Die Applikation visualisiert die einzelnen Schritte des Algorithmus, um das Verständnis für den Ablauf und die möglichen Probleme zu fördern. Im Folgenden eine Übersicht über die Funktionalität des Programms:

- Laden von Filmen, bestehend aus Einzelbildern im Sun-Rasterformat
- Navigation innerhalb von Filmen
- Bestimmung von nicht in die Berechnung des *MediPictures* einzubeziehenden Bereichen
- Berechnung des *MediPictures* sowie Darstellung der drei berechneten Teilbilder
- Speichern und Laden von *MediPictures*
- Hintergrund-Extraktion
- Filterung
- Silhouetten-Extraktion
- Speichern der berechneten Silhouette

Die Funktion zur Bestimmung von nicht in die Berechnung des *MediPictures* einzubeziehenden Bereichen wurde deshalb implementiert, weil sich unter den zur Verfügung stehenden Testdaten keine Videosequenz ohne sich bewegende Objekte, also nur mit Hintergrund, befand. Eine solche Sequenz ist für die Berechnung eines brauchbaren *MediPictures* als Trainingsphase Voraussetzung (siehe Kapitel 2.1). Mittels dieser Funktion wird die Möglichkeit gegeben, für jedes einzelne Frame des Filmes einen Bereich, in dem sich die bewegten Objekte befinden, als inaktiv zu markieren. Dieser Bereich wird anschliessend nicht in die Berechnung des *MediPictures* einbezogen. Dabei ist aber unbedingt darauf zu achten, dass jedes Pixel des Bildes in mindestens zwei Frames des Filmes ausserhalb eines solchen Bereiches ist. Die Qualität des *MediPictures* steigt mit der Anzahl Frames, in denen die einzelnen Pixel in die Berechnung einbezogen werden.

Um den Aufwand für die Bestimmung der inaktiven Bereiche in Grenzen zu halten, werden bei Vorhandensein solcher Bereiche nur diejenigen Bilder der Videosequenz in die Berechnung einbezogen, für welche diese definiert wurden. Damit ist es nicht notwendig, für jedes Frame einen inaktiven Bereich festzulegen, und es kann eine Auswahl der zu verwendenden Frames getroffen werden.

## 3.2 Bewertung

Im Folgenden wird das in Abbildung 3.1 gezeigte Ergebnis diskutiert:

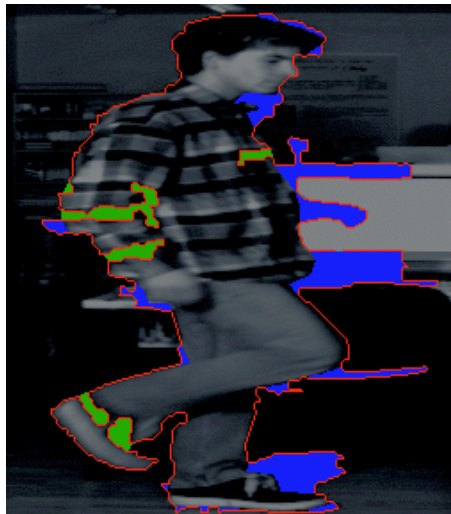


Abbildung 3.1: Probleme bei der Extraktion von Silhouetten unter schwierigen Bedingungen. False positives sind blau, false negatives grün dargestellt

Auf den ersten Blick scheint das Resultat nicht sehr zu überzeugen. Die blau markierten Bereiche entsprechen *false positives*, die grünen Flächen *false negatives*. Erstere sind Bereiche, die zusätzlich zu der wirklich zum extrahierten Objekt gehörenden Fläche dem Objekt zugesprochen wurden, letztere wurden fälschlicherweise vom Objekt entfernt. Offensichtlich wurde die Silhouette der abgebildeten Person nicht exakt extrahiert.

Vor einer weiteren Diskussion des Resultates soll ein Blick auf die verwendeten Testdaten geworfen werden. Diese wurden nicht unter Laborbedingungen aufgenommen. Ganz im Gegenteil fehlte eine gute Beleuchtung der Szene, es gab starke Schattenwürfe und die sich bewegende Person trug ein kariertes Hemd, von welchem einige Teilflächen eine sehr ähnliche Farbe hatten wie der Hintergrund. Zusätzlich war die zu extrahierende Person auch nicht die einzige im Raum, die sich bewegte.

Unter Berücksichtigung dieser Datenlage können die farbig markierten Bereiche schnell den Problemen zugeordnet werden: Die *false positives* entsprechen den Schatten, besonders gut erkennbar auf dem Boden und dem Bildschirm im Hintergrund. Die *false negatives* wurden durch die vom Hintergrund kaum unterscheidbare Farbe (Textur) einiger Bereiche des Hemdes und der Socken verursacht. Speziell zu bemerken ist ausserdem der blaue Bereich unterhalb des Kopfes der extrahierten Person. Dabei handelt es sich um eine zweite Person, die sich im Hintergrund befindet und sich auf dem Film ebenfalls bewegt.

Damit ist es möglich, Voraussetzungen zu benennen, die für eine robuste Silhouetten-Extraktion erfüllt sein sollten:

- Schatten durch geeignete Beleuchtung <sup>1</sup> minimieren
- Trainingsphase unter denselben Beleuchtungsbedingungen wie die eigentliche Aufnahmephase
- Klarer Kontrastunterschied zwischen der Hintergrundszene und dem zu extrahierenden Objekt, insbesondere entlang der Silhouette
- Zu extrahierendes Objekt besitzt grösste Silhouette aller sich bewegenden Objekte

Wenn noch weitere sich bewegende Objekte in der Szene vorhanden sind, so ist das solange kein Problem, wie sich die Silhouetten der einzelnen Objekte nicht berühren. Im Berührungsfall werden die beiden Objekte im Algorithmus als ein einziges erkannt, woraus eine *siamesische* Silhouette resultiert.

Weil der Algorithmus jeweils nur die grösste entdeckte Silhouette ausgibt, werden Texturen auf der Kleidung einer Person erst dann zu einem Problem,

---

<sup>1</sup>Erreichbar durch mindestens zwei helle Beleuchtungsquellen, welche alle Seiten des Objektes beleuchten.

wenn sie im Bereich der Silhouette keinen genügenden Intensitätsunterschied zur Hintergrundszene bilden. Manchmal erzwingen Texturen kleinere Silhouetten im Innern des Objektes, welche aber, da nicht mit der Haupt-Silhouette verbunden, zu klein sind und deshalb automatisch wieder entfernt werden. Dieser Effekt kann sehr gut mit der Beispiel-Applikation beobachtet werden.

Wird in Abbildung 3.1 die Silhouette an Hinterkopf und Rücken der extrahierten Person betrachtet, so ist das Resultat trotz der schwierigen Testdaten sehr gut. Darum darf an dieser Stelle bemerkt werden, dass die beschriebene Methode bei guter Datenlage sehr robust ist. Ausserdem ist der Algorithmus sehr einfach, was eine äusserst schnelle Ausführung der Implementierung ermöglicht.

# 4

## Zusammenfassung

### 4.1 Rückblick

Wie im Kapitel 3.2 erläutert, zeichnet sich der *MediPicture*-Algorithmus unter geeigneten Bedingungen als sehr robust aus und liefert exzellente Ergebnisse. Da die Ausführung äusserst schnell ist, besteht sogar die Möglichkeit, die Implementierung in Realtime-Systemen zu verwenden.

Im Ziel-Projekt *REBOMO* wird hingegen keine Realtime-Ausführung benötigt. Ganz im Gegensatz dazu werden die Daten zuerst aquiriert und anschliessend in einer Nachbearbeitung behandelt. Die benötigten Aufnahmen werden unter Laborbedingungen erstellt, wodurch auch sämtliche Bedingungen an Beleuchtung und Hintergrundszenerie problemlos erfüllt werden können.

Bei genügendem Kontrast zwischen dem zu extrahierenden Objekt und dem Hintergrund ist es auch möglich, ausserhalb des Labors, also in der natürlichen Umwelt, Aufnahmen zu machen und anschliessend mit dem *MediPicture*-Algorithmus zu prozessieren. Damit in diesem Fall keine Schatten das Resultat verfälschen, ist es von Vorteil, die Aufnahmen bei bewölktem Himmel zu erstellen, weil dann Schatten gar nicht erst entstehen. Dabei ist aber speziell auf möglichst ähnliche Bedingungen für die Trainingsphase und die eigentliche Aufnahmephase zu achten, insbesondere auf Wolken und Sonnenstand.

### 4.2 Ausblick

Obwohl mit dem beschriebenen Algorithmus ein robustes Verfahren geliefert wird, ist es denkbar, dass die zu beachtenden Bedingungen an die Szenerie nicht vollständig erfüllt werden können. In diesem Fall ist es notwendig, weitere Verfahren zu entwickeln, welche das Resultat verbessern können.

Bereits in der zu dieser Arbeit gehörenden Literaturrecherche [7] wurde auf solche Möglichkeiten eingegangen. Im Folgenden sollen die Ideen noch einmal kurz aufgenommen werden.

Das schwierigste auftretende Problem dürfte die Schattenbildung sein. Das hängt mit der Tatsache zusammen, dass der Schatten einer Person auf dem Boden immer von den Füßen ausgeht und damit direkt mit der Silhouette der Person verbunden ist. Wird der Schatten während der Hintergrund-Extraktion dem Vordergrund zugewiesen, so gehört er auch sofort zur Silhouette des Objektes.

In der Literatur werden Methoden beschrieben, um dieses Problem zu lösen. Das ebenfalls zum  $W^4$ -System gehörende *Ghost* [9, 8, 10] ist in der Lage, mittels eines statistischen Modells des menschlichen Körpers Körperteile zu erkennen und zu benennen. Dadurch entsteht die Möglichkeit, von den Füßen ausgehende Schatten zu detektieren und anschliessend aus der Silhouette zu entfernen.

Eine weitere Methode wird in [11] beschrieben. Dabei handelt es sich um ein System, welches den optischen Fluss (*optical flow*) eines Films beobachtet und im Speziellen Rotationen des Vektorfeldes detektiert. Unter der Annahme, dass nur wenige Körperteile wie Arme und Beine schwingen, der übrige Körper sich hingegen mehr oder weniger linear bewegt, ist es auf diese Art und Weise möglich, besagte Körperteile zu erkennen und damit auch die Schatten zu identifizieren. Es sei allerdings angemerkt, dass solche *flow-based* Methoden sehr rechenintensiv sind.

Weil das Gebiet der *Computer Vision* sehr umfangreich und populär ist, dürfte es möglich sein, weitere Ansätze zu finden, wie die Resultate des *MediPicture*-Algorithmus noch verbessert werden können. Da der entwickelte Algorithmus aber den in der Semesterarbeit gestellten Anforderungen genügt, wird hier nicht mehr weiter darauf eingegangen.

# Quelltexte des Algorithmus

Die Quelltexte (Sourcen) des in dieser Arbeit beschriebenen Algorithmus und die Beispielanwendung SilExtractor sind auf dem Internet frei verfügbar. Es handelt sich dabei um eine Archivdatei im zip-Format, welche die Programmcodes in C++ enthält. Getestet wurde die Ausführbarkeit mit SuSE Linux 7.1. Voraussetzung sind die GDK/GTK-Bibliotheken <sup>1</sup>, welche in allen aktuellen Linux-Distributionen enthalten und automatisch installiert sind, wenn der Fenstermanager GNOME <sup>2</sup> vorhanden ist.

Folgende Adresse führt zu den erwähnten Programmtexten:

<http://www.shima.ch/papers/>

Unter derselben Adresse sind ausserdem diese Arbeit wie auch die Literaturrecherche [7] als PDF-Dokument verfügbar.

---

<sup>1</sup> GIMP Drawing Kit, GIMP Toolkit (beide unterliegen der LGPL-Lizenz)

<sup>2</sup> GNU Network Object Model Environment



# Bibliotheken

Auf den folgenden Seiten werden die Schnittstellen zu den C++ Klassen des *MediPicture*-Algorithmus beschrieben. Aus Platzgründen kann in dieser Arbeit allerdings keine detaillierte Beschreibung erfolgen, was insbesondere bedeutet, dass nur gerade die notwendigsten Klassenmethoden aufgeführt sind. Ebenso sind die Definitionen in einer Art Pseudo-C++ geschrieben, die jedoch als Schnellreferenz genügen sollte. Für genauere Informationen sei auf die frei verfügbaren Quelltexte verwiesen (siehe Seite 15), welche detailliert kommentiert sind.

Die folgenden Klassen werden beschrieben:

- Image
- MediPicture
- Movie
- Silhouette

Die entsprechenden Headerfiles sind mit den Klassennamen identisch. Das den Quelltexten beiliegende und ebenfalls frei verfügbare Demonstrationsprogramm *SilExtractor* zeigt den Gebrauch sämtlicher Klassen und demonstriert insbesondere die Funktionalität des *MediPicture*-Algorithmus.

## Image

Die Image-Klasse stellt ein Bild-Objekt zur Verfügung, welches erlaubt, Bilddateien im SUN-Rasterformat einzulesen. Neben den für die Bildmanipulation notwendigen Methoden sind in dieser Klasse auch die Kernfunktionalitäten der Silhouettenextraktion enthalten: Hintergrundextraktion, Filterung und Silhouettenerkennung.

```
#include <gtk/gtk.h>
#include "MediPicture.h"
#include "Silhouette.h"

class Image
{ Image(*filename);
  Image(width,height);
  Image(width,height,color);

  int getHeight();
  int getWidth();
  byte get(x,y);
  byte get(index);
  set(x,y,val);
  set(index,val);
  clear(color);

  copyTo(*destination);
  copyFrom(*source);

  show(*drawable,*gc);
  extractBackground(*mediPic,tolerance);
  filter();
  Silhouette *calcSilhouette(*drawable,*gc);
};
```

## MediPicture

Die MediPicture-Klasse implementiert die Berechnung des *MediPictures*. Als Argument muss der zu verwendende Film übergeben werden, wobei ein besonderes Dateinamenformat für die den Film zusammensetzenden Bilder zu beachten ist:

*Dateiname.Framenummer*

*Framenummer* bezeichnet dabei die Position des Bildes innerhalb des Films. Das Argument `numWidth` bestimmt die Anzahl Ziffern, mit der die Frame-Nummern der zum Film gehörenden Bilder dargestellt werden. Heisst zum Beispiel eine Bilddatei `walk.001`, so ist `numWidth=3`. Ein weiteres Argument `restr` ermöglicht die Zuweisung von nicht in die Berechnung einzubeziehenden Bereichen für jedes Bild. Dies ist nützlich, wenn keine Aufnahmen mit statischem Hintergrund vorhanden sind, um die Bereiche mit Bewegung auszuschliessen. Es ist aber wichtig, dass jedes Pixel in mehreren Frames Hintergrundinformation enthält.

Die Berechnung des *MediPictures* wird bereits im Constructor der Klasse gestartet. Alternativ kann ein gespeichertes *MediPicture* geladen werden. Die drei Einzelbilder für die minimale und maximale Intensität und den maximalen Intensitätsunterschied zwischen zwei aufeinanderfolgenden Bildern können einzeln betrachtet werden.

```
typedef byte MediIntensity;

class MediPicture
{ MediPicture(*movie,numWidth);
  MediPicture(*movie,numWidth,*restr);
  MediPicture(*filename);

  store(*filename);
  MediIntensity getMinimum(x,y);
  MediIntensity getMaximum(x,y);
  MediIntensity getMaxDifference(x,y);
  MediIntensity getMinimum(index);
  MediIntensity getMaximum(index);
  MediIntensity getMaxDifference(index);

  showMinimum(*drawable,*gc);
  showMaximum(*drawable,*gc);
  showMaxDifference(*drawable,*gc);
};
```

## Movie

Die Movie-Klasse repräsentiert ein Objekt, welches die im Zusammenhang mit Filmen benötigte Funktionalität bietet (Anzeigen von Bildern, Navigieren im Film und Abspielen des Films). Ein Film besteht aus je einem Image-Objekt (siehe Seite 18) für jedes Frame, wobei jeweils nur das aktuelle Frame geladen ist.

Die Funktionen `getSource` und `getDigits` liefern die für die MediPicture-Klasse (siehe Seite 19) benötigten Parameter `movie` und `numWidth`.

```
#include "Image.h"
#include "MediPicture.h"

class Movie
{ Movie(*filename,numWidth);

    firstImage();
    lastImage();
    bool nextImage();
    bool prevImage();
    int getWidth();
    int getHeight();
    int getActImage();
    int length();

    setActImageInactiveArea(x1,x2);
    setActImageInactiveArea(InactiveArea *inactArea);
    deleteActImageInactiveArea();
    deleteInactiveAreas();
    ImageRestriction* getInactiveAreas();

    showFrame(*drawable,*gc,showInactiveArea);
    play(*drawable,*gc,showInactiveArea);
    playFromHere(*drawable,*gc,showInactiveArea);

    string *getActImageFilename();
    string *getSource();
    int getDigits();
};
```

## Silhouette

Mittels der Silhouetten-Klasse werden Silhouetten-Objekte dargestellt. Solche Objekte werden als Resultat der Silhouetten-Extraktion vom Image-Objekt (siehe Seite 18) zurückgeliefert. Neben der Möglichkeit, eine Silhouette in eine Datei zu speichern und von dort auch wieder zu laden, werden ausserdem Methoden für die Darstellung und die Vereinfachung zur Verfügung gestellt. Bei der Vereinfachung werden Punkte aus der Liste entfernt, welche auf einer Geraden liegen.

```
#include <gtk/gtk.h>

struct silPoint
{ int x,y;
};

const silPoint EOS={-1,-1};

class Silhouette
{ Silhouette();
  Silhouette(*filename);

  addPoint(x,y);
  removePoint();
  getPoint(*point);
  firstPoint(*point);
  firstPoint();
  nextPoint(*point);
  nextPoint();
  int getIndex();
  int getNumPoints();

  simplify();
  show(*drawable,*gc);
  store(*filename);
};
```



# Literaturverzeichnis

- [1] Pascal Fua, Armin Grün, Ralf Plänklers, Nicola D'Apuzzo, Daniel Thalmann. Human Body Modeling and Motion Analysis from Video Sequences. *International Archives of Photogrammetry and Remote Sensing, Hakodate, Japan, 1998, Vol. 32 (B5), pp 866-873, 1998.*
- [2] Nicola D'Apuzzo, Ralf Plänklers, Pascal Fua, Armin Grün, Daniel Thalmann. Modeling Human Bodies from Video Sequences. *Videometrics VI, Part of IS&T/SPIE's Symposium on Electronic Imaging '99, San Jose, California, 1999.*
- [3] Ralf Plänklers, Pascal Fua, Nicola D'Apuzzo. Automated Body Modeling from Video Sequences. *Proc. IEEE International Workshop on Modelling People (mPeople), Corfu, Greece, 1999.*
- [4] Nicola D'Apuzzo, Ralf Plänklers. Human Body Modeling from Video Sequences.
- [5] Nicola D'Apuzzo, Ralf Plänklers, Pascal Fua. Least Squares Matching Tracking Algorithm for Human Body Modeling. *International Archives of Photogrammetry and Remote Sensing, Amsterdam, The Netherlands, 2000.*
- [6] Nicola D'Appuzzo. Motion Capture by Least Squares Matching Tracking Algorithm. *AVATARS2000, 30.11.-1.12.2000, Lausanne, Switzerland, 2000.* Institute of Geodesy and Photogrammetry, ETH Zürich, Switzerland.
- [7] Marco Nef. Automatic Extraction of Silhouettes from Video Sequences: Literature Research and Design Decision. 2001. Semester thesis in Photogrammetry, Institut für Geodäsie und Photogrammetrie, ETH Zürich, Switzerland.
- [8] Ismail Haritaoglu, David Harwood, Larray S. Davis. W<sup>4</sup>: Who? When? Where? What? A Real Time System for Detecting and Tracking People. *3. International Conference on Face and Gesture Recognition, April 14-16, 1998, Nara, Japan.* Computer Vision Laboratory, University of Maryland, USA.

- [9] Ismail Haritaoglu, David Harwood, Larray S. Davis. *Ghost: A Human Body Part Labeling System Using Silhouettes*. *14th International Conference on Pattern Recognition, August 16-20, 1998, Brisbane, Australia*. Computer Vision Laboratory, University of Maryland, USA.
- [10] Ismail Haritaoglu, David Harwood, Larray S. Davis. *W<sup>4</sup>S: A Real-Time System for Detecting and Tracking People in 2 $\frac{1}{2}$ d*. Computer Vision Laboratory, University of Maryland, USA.
- [11] Toru Tamaki, Tsuyoshi Yamamura, Noboru Ohnishi. *Extraction of Human Limb Regions and Parameter Estimation based on Curl of Optical Flow*. *Proceedings of the Fourth Asian Conference on Computer Vision (ACCV2000)*, II:1008–1013 (2000,1), 2000.

# Index

- Algorithmus
  - Bewertung, 10
  - MediPicture, 5
  - Quelltexte, 15
  - W<sup>4</sup>-System, 4
- Aufnahmephase, 11, 13
- Beleuchtung, 11
- Bewölkung, 13
- Computer Vision, 1
- Computergraphik, 1
- false negative, 11
- false positive, 11
- Filterung, 5
  - Filterkernel, 6
  - Minimaler Support, 6
- GDK/GTK, 15
- Ghost, 14
- GNOME, 15
- Hintergrund-Extraktion, 3, 4
  - Probleme, 5
- inaktive Bereiche, 10
  - Aufwand, 10
- Internet, 15
- Körperteile, 14
- Klassen
  - Image, 18
  - MediPicture, 19
  - Movie, 20
  - Silhouette, 21
- Kontrast, 11
- Laborbedingungen, 13
- MediPicture, 3
  - Algorithmus, 3
  - Intensität, 3
  - Qualität, 10
- Nachbearbeitung, 13
- Objektgrenze, 7
  - Schlaufe, 7
- optical flow, 14
- Photogrammetrie, 1
- Probleme
  - Schatten, 13
  - Schattenwurf, 11
  - Textur, 11
- Quelltexte, 15
  - Headerfiles, 17
- Realtime, 13
- REBOMO, 1, 13
- Robustheit, 12
- SilExtractor, 9, 17
- Silhouette
  - siamesische, 11
  - Vereinfachung, 8
- Silhouetten-Extraktion, 7
  - Voraussetzungen, 4
- Sonnenstand, 13
- Sourcen, 15
- Test-Applikation, 9
- Testdaten, 11
- Testprogramm, 17

Toleranz, 5

Trainingsphase, 3, 10, 11, 13