

# **Personentracking über adaptive Hintergrund- und Differenzbildberechnung**

## **Studienarbeit im Studiengang Computervisualistik**

vorgelegt von

Christian Latsch

Betreuer: Prof. Dr.-Ing. Dietrich Paulus, Institut für Computervisualistik,  
Fachbereich Informatik  
Erstgutachter: Prof. Dr.-Ing. Dietrich Paulus, Institut für Computervisualistik,  
Fachbereich Informatik  
Zweitgutachter: Dipl.-Inf. Tobias Feldmann, Institut für Computervisualistik, Fach-  
bereich Informatik

Koblenz, im Mai 2007



## Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Richtlinien der Arbeitsgruppe für Studien- und Diplomarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden. ja  nein

Der Veröffentlichung dieser Arbeit im Internet stimme ich zu. ja  nein

Koblenz, den .....

Unterschrift



# Inhaltsverzeichnis

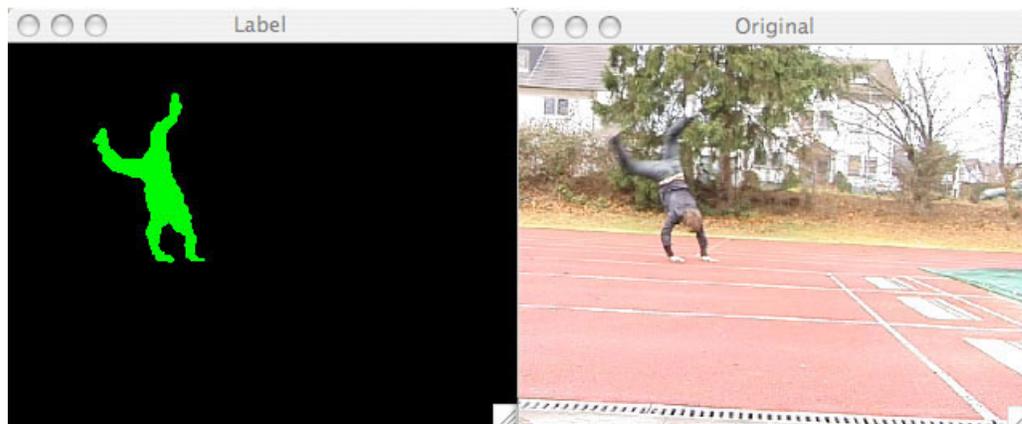
- 1 Einleitung** **7**
  - 1.1 Problemstellung und Ziele . . . . . 8
  - 1.2 Aufbau dieser Arbeit . . . . . 9
  
- 2 Stand der Wissenschaft** **11**
  - 2.1 Farbräume . . . . . 11
    - 2.1.1 Farbton . . . . . 12
    - 2.1.2 Helligkeit . . . . . 14
    - 2.1.3 Farbsättigung . . . . . 14
  - 2.2 Hintergrund- Vordergrunderstellung . . . . . 14
    - 2.2.1 Mixture of Gaussian . . . . . 15
  - 2.3 Entfernung von Schatten . . . . . 18
    - 2.3.1 Schattenerkennung im HSV-Farbraum . . . . . 19
  - 2.4 Region Growing . . . . . 20
  - 2.5 Aktive Konturen . . . . . 21
  - 2.6 Template Matching . . . . . 22
  
- 3 Eigener Ansatz** **25**

|          |  |           |
|----------|--|-----------|
| 3.1      | Aufbau des Systems . . . . .                 | 25        |
| 3.2      | Zusammenspiel der Module . . . . .           | 26        |
| 3.3      | Hintergrund- Vordergrunderstellung . . . . . | 28        |
| 3.3.1    | Probleme mit RGB . . . . .                   | 29        |
| 3.3.2    | Die Lösung mit HSV . . . . .                 | 29        |
| 3.3.3    | Weißer Wände . . . . .                       | 30        |
| 3.4      | Schattenentfernung . . . . .                 | 31        |
| 3.4.1    | Test auf Schatten . . . . .                  | 31        |
| 3.5      | Region Growing . . . . .                     | 33        |
| 3.6      | Konturerstellung . . . . .                   | 34        |
| 3.7      | Tracking . . . . .                           | 35        |
| <b>4</b> | <b>Experimente und Ergebnisse</b>            | <b>37</b> |
| 4.1      | Laufzeitanalyse . . . . .                    | 37        |
| 4.2      | Qualität der Kontur . . . . .                | 38        |
| 4.3      | Lernen des Hintergrundes . . . . .           | 41        |
| 4.4      | Entfernung von Schatten . . . . .            | 42        |
| 4.5      | Tracking . . . . .                           | 43        |
| <b>5</b> | <b>Zusammenfassung und Ausblick</b>          | <b>45</b> |
| 5.1      | Zusammenfassung . . . . .                    | 45        |
| 5.2      | Ausblick . . . . .                           | 46        |
|          | <b>Literaturverzeichnis</b>                  | <b>47</b> |

# Kapitel 1

## Einleitung

Diese Studienarbeit dient der Realisierung eines kostengünstigen, schnellen und zuverlässigen Personenverfolgungssystem, basierend alleine auf einer Kamera und einem Computer. Es bildet unter anderem die Grundlage zur Bewegungsanalyse von Sportlern.



**Bild 1.1:** Mögliches Szenario: Bewegungsstudien von Turnern.

## 1.1 Problemstellung und Ziele

Personenverfolgungssysteme bestehen oft aus teurer und meist an Personen befestigter Trackinghardware, die die Bewegungsfreiheit der Personen deutlich einschränkt. Durch die in den letzten Jahrzehnten angestiegene Rechenleistung der Computersysteme ist es möglich Bilddaten von digitalen Video-, Foto- oder Webkamas in Echtzeit auszuwerten. Dadurch erschließen sich neue Möglichkeiten, die eine Verfolgung von Personen auch ohne die störrige Trackinghardware erlauben.

Das macht eine schnelle und gute Interaktion mit dem Computer wesentlich einfacher und erschließt völlig neue Einsatzgebiete. Dazu zählen die Bewegungsanalyse von Sportlern oder Tieren, aber auch die Überwachung von Personen. Gerade hierbei gibt es ein weites Spektrum an Einsatzmöglichkeiten. Ob zur Überwachung des Privatgrundstückes oder zur automatischen Analyse von Menschenströmen. Auch eine automatisierte Verkehrszählung wäre dank des niedrigen Anschaffungspreises denkbar.

In dieser Arbeit soll ein System zum Verfolgen von Personen ausschließlich unter Zuhilfenahme einer Videokamera und eines Computers, also ohne Marker, entwickelt werden. Dazu werden Kamerabildströme einer sich bewegenden Person vor einem statischen Hintergrund aufgenommen. Dabei soll der Hintergrund automatisch erlernt werden, wenn Veränderungen im Hintergrund stattfinden. Befindet sich beispielsweise während der Aufnahme des Referenzbildes eine Person im Bild, wird diese Person verfolgt und das Referenzbild des Hintergrunds an den neu sichtbaren Stellen (die zuvor von der Person verdeckt waren) vervollständigt. Störende Schatten sollen ähnlich wie unter [TMM06] und [TM05] entfernt werden. Mit der Hilfe des Referenzbildes wird die Silhouette der sich bewegenden Person extrahiert und über die weiteren Bilddaten verfolgt. Dabei ist die Einhaltung von Glattheitsbedingungen der Silhouette und die Vermeidung von Sprüngen sowie eine glatte resultierende Trajektorie der Personenbewegung einzuhalten.

## **1.2 Aufbau dieser Arbeit**

Die Arbeit ist folgendermaßen aufgebaut: In Kapitel 2 wird der Stand der Technik dargestellt. Dazu werden die ausgewählten Algorithmen mit anderen Verfahren verglichen und näher erläutert. Im darauf folgenden Kapitel 3 wird die eigene Umsetzung, die entstandenen Probleme und ihre Lösungen beschrieben. Anschließend werden in Kapitel 4 die durchgeführten Experimente und ihre Ergebnisse präsentiert. Zum Schluss werden in Kapitel 5 die Ergebnisse zusammengefasst und ein Ausblick auf die möglichen Erweiterungen gegeben.



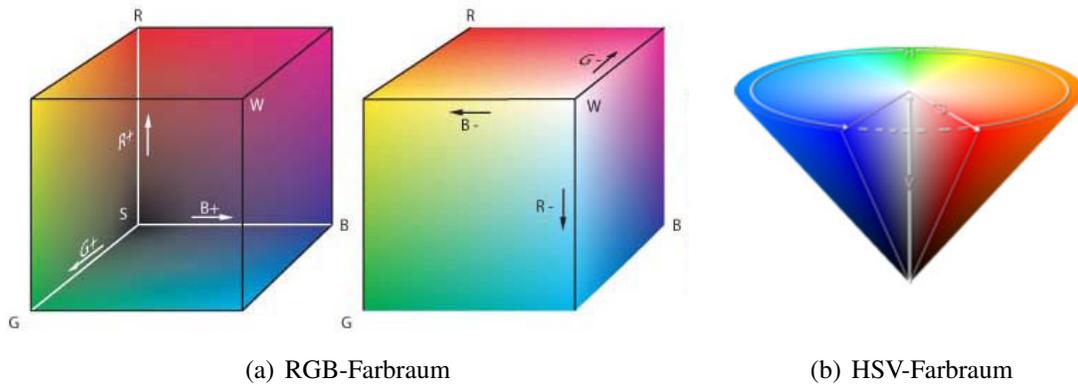
# Kapitel 2

## Stand der Wissenschaft

In diesem Kapitel werden die Grundlagen, sowie alle späteren verwendeten Algorithmen und deren heutigen Stand, die zur Realisierung dieses Systems notwendig waren, vorgestellt.

### 2.1 Farbräume

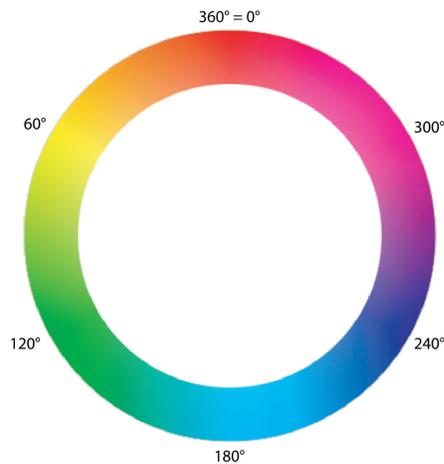
Ein Farbraum wird als Messraum für die einheitliche visuelle Wahrnehmung 'Farbe' definiert. Es existieren unterschiedliche Farbräume, die eine große Rolle in verschiedenen Anwendungsbereichen spielen. Anders als beim RGB-Farbraum (Bild 2.1(a)), bei dem die Farbe durch Mischung der einzelnen Farbkanäle (Rot, Grün und Blau) entsteht, wird beim HSV-Farbraum (Bild 2.1(b)) die Farbe mit Hilfe des Farbtons (engl. **hue**), der Farbsättigung (**saturation**) und der Helligkeit (**value**) definiert. Um diesen Farbraum in den später beschriebenen Verfahren verwenden zu können, müssen einige Eigenschaften beachtet werden, da es sonst zu falschen Ergebnissen führen kann. Im Folgenden wird im Detail auf die Besonderheiten der verschiedenen Kanäle eingegangen.



**Bild 2.1:** Darstellung der beiden Farbräume RGB und HSV (Quellen: [Wik08b] und [Wik08a]).

### 2.1.1 Farbton

Der Farbton-Kanal ist ein zyklischer Kanal (siehe Bild 2.2), dessen Farbwerte nicht wie die des RGB-Kanals zwischen einem Minimum und Maximum definiert werden, sondern bei Erreichen des Maximums durch ein Überlauf wieder am Minimum ansetzen. Die Farben liegen also auf einem Farbkreis (z. B.  $0^\circ$  = Rot,  $120^\circ$  = Grün,  $240^\circ$  = Blau ).



**Bild 2.2:** Zyklischer Hue-Kanal.

Für die folgenden Algorithmen ist es notwendig Farben unterscheiden zu können. Für die Unterscheidung ist ein Maß für die Distanz zweier Farben nötig. Möchte man die Distanz zweier Farben bestimmen, reicht es nicht aus einfach den Betrag der Subtraktion zu bilden. Schaut man sich z. B. einen roten Ton mit  $350^\circ$  und einen Grünton mit  $100^\circ$  an, so ist die Distanz  $110^\circ$  und nicht  $250^\circ$ . Die Distanz  $d$  zweier Farben  $h_1$  und  $h_2$  wird daher definiert als:

$$d(h_1, h_2) = \min(360 - |h_1 - h_2|, |h_1 - h_2|) \quad (2.1)$$

### Mittelwert zweier Farben

Berechnet man den arithmetischen Mittelwert zweier Farbtöne so führt dies zu einem verfälschten Ergebnis, da es sich bei den Farbwerten um Winkel handelt. Deshalb muss der Mittelwert  $M\alpha$  gegebener Winkel  $\alpha_1, \dots, \alpha_n$  auf folgende Weise bestimmt werden. Für die Mittelwertberechnung zweier Farben ist  $n = 2$ .

$$M\alpha = \text{atan2}\left(\frac{1}{n} \sum_{j=1}^n \sin \alpha_j, \frac{1}{n} \sum_{j=1}^n \cos \alpha_j\right) \quad (2.2)$$

Dabei ist die *arctangents* Variante  $\text{atan2}$  für  $y \neq 0$  definiert als:

$$\text{atan2}(y, x) = \begin{cases} \varphi \cdot \text{sgn}(y) & x > 0 \\ \frac{\pi}{2} \varphi \cdot \text{sgn}(y) & x = 0 \\ (\pi - \varphi) & x < 0 \end{cases} \quad (2.3)$$

Wobei wenn der Winkel  $\varphi$  in  $[0, \pi/2)$  ist, gilt:  $\tan(\varphi) = \left|\frac{y}{x}\right|$ . Und  $\text{sgn}$  ist die Signum-Funktion.

Weiter gilt:

$$\text{atan2}(0, x) = \begin{cases} 0 & x > 0 \\ \text{nicht definiert} & x = 0 \\ \pi & x < 0 \end{cases} \quad (2.4)$$

### 2.1.2 Helligkeit

In diesem Kanal werden ausschließlich die Helligkeitsinformationen des Bildes gespeichert. Die Werte liegen dabei als Prozentwerte vor (z. B. 0% = keine Helligkeit, 100% = volle Helligkeit). Mit Hilfe dieser Information lassen sich Schatten, die irrtümlicher Weise bei der Segmentierung dem Vordergrund zugeordnet werden, erkennen und entfernen. Wenn sich im Bild nur die Helligkeit und nicht der Farbwert ändert, kann man davon ausgehen, dass es sich um einen Schatten handelt. Auch bei einer globalen Beleuchtungsänderung (z. B. Wolke vor Sonne), macht diese sich nur in der Helligkeit und nicht im Farbton bemerkbar.

### 2.1.3 Farbsättigung

Dieser Kanal dient der Speicherung der Sättigungsinformationen. Diese werden, wie zuvor beim Helligkeitskanal, als Prozentwerte (z. B. 0% = keine Sättigung, 100% = volle Sättigung) angegeben.

## 2.2 Hintergrund- Vordergrunderstellung

Um sich einen guten Überblick über die verschiedenen Ansätze der Hintergrund-Segmentierung zu verschaffen, empfehle ich [MG01]. Das besondere Augenmerk liegt dabei auf der sogenannten *background subtraction*-Methode. Hierbei wird der Vordergrund vom Hintergrund durch Differenzbildtechnik getrennt. Diese Methoden eignen sich bei nahezu statischen Hintergründen und nicht zu starken Veränderungen der Lichtverhältnisse, wie man es in Innenräumen vorfindet. Dabei muss die Kamera die gleiche Position beibehalten. Bei unruhigen Hintergründen, die zum Beispiel durch die ständigen Bewegungen der Blättern eines Baumes gestört werden, lohnt es sich auf andere Verfahren, welche auf eine statistische Auswertung des Bildes setzen, zurückzugreifen (vergleiche Tabelle 2.2).

Auf Grund der nicht immer gleichen Lichtverhältnisse fiel die Wahl auf die Verwendung einer Differenzbildtechnik (siehe [TMM06]), basierend auf einem Modell mit Mischungen von Gaußnormalverteilungen. Es handelt sich also um eine statistische Methode.

| Eigenschaften                       | Differenzbild | statistisches Differenzbild |
|-------------------------------------|---------------|-----------------------------|
| keine Kamerabewegung                | ja            | ja                          |
| leichte Kamerabewegung              | nein          | nein                        |
| statischer Hintergrund              | ja            | ja                          |
| Veränderungen im Hintergrund        | nein          | ja                          |
| gleiche Lichtverhältnisse           | ja            | ja                          |
| Veränderungen der Lichtverhältnisse | nein          | ja                          |

**Tabelle 2.1:** Die wichtigsten Unterschiede zwischen einfacher Hintergrundberechnung mit Differenzbildtechnik und mit *Mixtur Of Gaussian* erweiterten Differenzbildtechnik.

### 2.2.1 Mixture of Gaussian

Im Folgenden werden die Grundlagen zur Erstellungen eines *Mixture of Gaussian* Modells behandelt. Dabei wird das Verfahren nach [SG00] beschrieben. Er stellte fest, dass wenn jeder Pixel eines Bildes aus einer einzelnen Oberfläche und einer festen Lichtquelle resultiert, eine einfache Gaußverteilung zur Bestimmung der Pixelwerte ausreicht. Wenn sich nach einiger Zeit nur die Lichtquelle ändert, wäre eine einfache, adaptive Gaußverteilung pro Pixel ausreichend. Da aber in der Praxis oft mehrere Oberflächen an Stellen bestimmter Pixel vorkommen und dazu noch sich veränderende Lichtverhältnisse stoßen, wird eine adaptive Mischung von  $K$  Gaußverteilungen notwendig um dieses Problem zu lösen ( $K$  wird von [SG00] als eine Zahl von 3 bis 5 beschrieben). Dabei repräsentieren die verschiedenen Gaußverteilungen verschiedene Farben. Die Gewichtungparameter der Mischungen stehen für das Zeitverhältnis, das solche Farben in der Szene bleiben. Das bedeutet, dass Farben die wahrscheinlicher dem Hintergrund zu geordnet werden auch länger und statischer im Bild bleiben. Weiter wird beschrieben, dass statische sich nicht bewegend Objekte dazu tendieren, enge Cluster im Farbraum zu bilden. Anders sieht es bei sich bewegend Objekten aus, die aufgrund ihrer unterschiedlich reflektierenden Oberflächen während der Bewegung eher weite Cluster im Farbraum bilden. Das resultierende Maß wird *fitness* Wert genannt. Es wird ein selektives Aktualisierungsschema angewandt, welches das Arbeiten in Echtzeit ermöglicht und dem Modell erlaubt, sich den Änderungen der Lichtverhältnisse anzupassen.

### Online Mixture Model

Die Vergangenheit jedes Pixels  $\{X_1, \dots, X_t\}$  der Szene ist modelliert aus einer Mischung der  $K$  Gaußverteilungen. Die Wahrscheinlichkeit, das ein bestimmter Pixel den Wert von  $X_t$  zum Zeitpunkt  $t$  hat, wird definiert als

$$p(X_t) = \sum_{k=1}^K \omega_k \cdot \eta(X_t, \mu_{k,t}, \Sigma_{k,t}), \quad (2.5)$$

wobei  $\omega_k$  der Gewichtungparameter der  $k^{\text{ten}}$  Gaußkomponente zum Zeitpunkt  $t$ ,  $\mu_{k,t}$  und  $\Sigma_{k,t}$  der Mittelwert und die Kovarianzmatrix der  $k^{\text{ten}}$  Normalverteilung in der Mischung zum Zeitpunkt  $t$  ist. Dabei ist  $\eta$  eine Gauß-Funktion.

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X_t - \mu)^T \Sigma^{-1} (X_t - \mu)} \quad (2.6)$$

$K$  ist durch den Speicher und die Rechenleistung begrenzt auf eine Zahl von drei bis fünf. Der einfacheren und schnelleren Berechenbarkeit halber wird die Kovarianzmatrix in folgender Form angenommen.

$$\Sigma_{k,t} = \sigma_k^2 \mathbf{I} \quad (2.7)$$

Dabei ist  $\sigma^2$  die Varianz und  $\mathbf{I}$  die Identität. Nimmt man an, dass die roten, grünen und blauen Pixelwerte unabhängig voneinander sind und die gleiche Varianz besitzen, was jedoch nicht der Fall ist, lässt sich eine teure Matrixinvertierung auf Kosten der Genauigkeit sparen. Dadurch besteht die Möglichkeit, die aktuellen Werte jedes Pixels als Mischung von Gaußverteilungen zu charakterisieren. Generell wird ein neuer Pixelwert von einem der im Modell enthaltenen Komponenten des Modells repräsentiert und damit dazu benutzt es zu aktualisieren. Da für jeden Pixel ein Mischungsmodell angelegt wird, wäre ein exakter EM Algorithmus viel zu teuer, deshalb wird zur Aktualisierung des Modells eine *on-line K-means* Approximation benutzt.

### Background Model Estimation

Der Hintergrund  $B$  des jeweiligen Pixel setzt sich aus den  $K$  Gaußnormalverteilungen zusammen, deren Gewichte  $\omega^2$  über den Schwellwert  $T$  reichen. Dazu werden sie sortiert mit der höchsten *fitness*  $\omega/\sigma$  an erster Stelle der Reihe nach addiert, bis sie den Wert des Schwellwertes überschreiten (siehe Gleichung 2.8). Alle restlichen Komponenten, die noch nicht mit einbezogen wurden, gehören nicht zum Hintergrund. Ist der Hintergrund definiert, wird der neue Pixelwert mit jeder Komponente, wieder beginnend mit der Ersten, auf Ähnlichkeit verglichen. Das bedeutet, ist er weniger als eine Standardabweichung von 2,5 von einer Komponente entfernt, wird nur diese aktualisiert (Gleichungen 2.9 - 2.11). Handelt es sich bei der aktualisierten um eine Komponente des Hintergrundes, befindet sich der aktuelle Pixelwert auch im Hintergrund. Handelt es sich dagegen um eine Komponente, welche nicht zum Hintergrund gehört, wird diese auf gleiche Weise aktualisiert, der Pixel jedoch wird zum Vordergrund gezählt. Kann ein Pixel nicht erfolgreich auf Ähnlichkeit getestet werden, ist also von allen  $K$  Gaußnormalverteilungen mit einer Standardabweichung von mehr als 2,5 entfernt, wird die letzte Komponente, es handelt sich dabei um die Komponente mit der niedrigsten Gewichtung, aus dem Model entfernt und eine neue hinzugefügt. Die neue Komponente erhält als Normalverteilung den Wert des Mittelwertes  $\mu$  des aktuellen Pixels, eine hohe Varianz  $\sigma^2$  und ein niedriges initiales Gewicht  $\omega$ .

Der Hintergrund  $B$  wird berechnet als

$$B = \underset{b}{\operatorname{argmin}} \left( \sum_{k=1}^b \omega_k > T \right). \quad (2.8)$$

Die Aktualisierung eines Gewichtes zum Zeitpunkt  $t$  kann folgendermaßen beschrieben werden:

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t-1} + \alpha(M_{k,t}), \quad (2.9)$$

wobei  $\alpha$  die Lernrate ist und  $M_{k,t}$  genau 1 für das passende Model und 0 für alle folgenden. Nach dieser Approximation sind die Gewichte wieder normalisiert.  $1/\alpha$  beschreibt

die Zeitkonstante, welche die Veränderung festlegt. Die restlichen Parameter der zu aktualisierenden Komponente werden folgend angeglichen:

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho X_t \quad (2.10)$$

$$\sigma_t^2 = (1 - \rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t), \quad (2.11)$$

wobei

$$\rho = \alpha\eta(X_t | \mu, \sigma_k) \quad (2.12)$$

der Lernfaktor zur Anpassung der aktuellen Normalverteilung ist. Ein großer Vorteil dieser Methode ist, wenn ein Objekt lange genug an einer Stelle steht und dadurch in den Hintergrund aufgenommen wird, wird das alte Hintergrundmodell nicht zerstört. Die Originalfarbe des Hintergrundes bleibt solange im Modell enthalten, bis sie sich an der  $K$  wahrscheinlichsten Stelle im Modell befindet und durch eine neue Farbe ersetzt wird. Befindet sich also ein Objekt lange genug an einer Stelle und wird in den Hintergrund aufgenommen, bewegt sich dann aber wieder fort, existiert die Originalfarbe noch mit den selben  $\mu$  und  $\sigma^2$  aber einem niedrigeren  $\omega$  und es dauert nicht lange, bis sie wieder in den Hintergrund aufgenommen wurde.

## 2.3 Entfernung von Schatten

Um die Silhouette einer sich bewegenden Person optimal zu extrahieren ist es nötig alle Störungen zu erkennen und zu entfernen. Dazu zählen Schatten der Person, die die Form der Silhouette entscheidend verändern können. Personen und Schatten haben beide die Eigenschaft, das sie sich bewegen. Somit wird die Person und fälschlicherweise auch der Schatten dem Vordergrund zugeteilt. Die Lösung dieses Problems wird unter [CGP<sup>+</sup>01] beschrieben. Dazu wird der HSV Farbraum verwendet. Führt man sich vor Augen, dass Schatten eine Verdunkelung des Hintergrundes sind, gilt es nur noch diese Bereiche zu erkennen und nicht mehr dem Vordergrund zu zu ordnen.

### 2.3.1 Schattenerkennung im HSV-Farbraum

Vergleicht man das Originalbild mit dem Hintergrund, wird man feststellen, dass an den Schattenstellen des Originalbildes folgende Bedingungen gelten:

1. Der Farbwinkel (hue) des Schattens entspricht etwa dem des Hintergrundes.
2. Die Sättigung (saturation) des Schattens ist geringer als die des Hintergrundes.
3. Die Dunkelstufe (value) des Schattens ist größer als die des Hintergrundes.

Um alle Punkte  $S(x, y)$  bei denen es sich um Schatten handelt erkennen zu können, müssen alle oben genannten Bedingungen 1-3 zu treffen.

Bei der Farbwinkelkomponente sollte die absolute Differenz zwischen dem Vordergrundpixel  $I_{hue}(x, y)$  und dem Hintergrundpixel  $B_{hue}(x, y)$  im Schattenbereich unter einem bestimmten Schwellwert  $\tau_{hue}$  liegen.

$$|I_{hue}(x, y) - B_{hue}(x, y)| < \tau_{hue} \quad (2.13)$$

Der nächste Schritt ist die Überprüfung der Sättigung eines Punktes  $I_{sat}(x, y)$  des Vordergrundes mit dem Punkt  $B_{sat}(x, y)$  des Hintergrundes, wobei die Sättigung des Vordergrundes an der Schattenstelle geringer als die des Hintergrundes sein muss. Die Differenz der Punkte muss also unter einem gewissen Schwellwert  $\tau_{sat}$  liegen.

$$(I_{sat}(x, y) - B_{sat}(x, y)) < \tau_{sat} \quad (2.14)$$

Man überprüft die Helligkeit eines Punktes  $I_{value}(x, y)$  an der Stelle  $(x, y)$  des aktuellen Bildes mit der des Punktes  $B_{value}(x, y)$  des Hintergrundes. Mit Hilfe der Variablen  $\beta < 1$ , kann verhindert werden, dass durch Bildrauschen Pixel fälschlicherweise als Schatten erkannt werden. Die Variable  $\alpha$  beschreibt die Stärke der Lichtquelle. Je stärker die Lichtquelle, umso niedriger muss  $\alpha$  gewählt werden.

$$\alpha < \frac{I_{value}(x, y)}{B_{value}(x, y)} < \beta \quad (2.15)$$

Um die Schatten  $S(x, y)$  zu erkennen, werden alle drei Bedingungen vereinigt. Man erhält also:

$$S(x, y) = \begin{cases} 0 & \text{if } |I_{hue}(x, y) - B_{hue}(x, y)| < \tau_{hue} \\ & \wedge (I_{sat}(x, y) - B_{sat}(x, y)) < \tau_{sat} \\ & \wedge \alpha < \frac{I_{value}(x, y)}{B_{value}(x, y)} < \beta \\ 1 & \text{otherwise} \end{cases} \quad (2.16)$$

## 2.4 Region Growing

Ziel des Region Growing Verfahrens ist es homogene Bildelemente zu Regionen zu verschmelzen, indem Pixel auf Gleichheit miteinander verglichen werden. Dabei wird das Bild zuerst in Bereiche (Zellen) geteilt, anschließend mit z. B. einem initialen Startpixel gestartet und alle Nachbarpixel auf Gleichheit getestet. Sind sich Pixel ähnlich, also liegt die Differenz des zu testenden Wertes des Pixel und dem Mittelwert des Segmentes unter einem gewissen Schwellwert, wird der Pixel zum Segment hinzugefügt. Dadurch wachsen Segmente. Kann kein Pixel mehr hinzugefügt werden, wird ein neuer, noch unzugeordneter Pixel als Startpixel ausgewählt und der Prozess fängt wieder von vorne an. Dies geschieht so lange, bis jeder Pixel zu einer Region gehört.

Ein Ansatz von [SC80] ist dabei Raster abzutasten, neue Regionen zu benennen, aber auch zusammenhängende Regionen zusammenzuführen. Um die einzelnen Regionen zu finden, betrachtet man nicht nur jede Zeile eines Bildes einzeln, sondern auch die jeweils vorherige Zeile. Dabei wird nach dem Start- und Endpunkt einer Region in der aktuellen Zeile überprüft, welche Farbe sie enthält, wurde sie schon mal berechnet oder nicht, handelt es sich um eine neue oder alte Region oder ist sie eine Brücke zwischen schon existierenden Regionen. Es existieren viele mögliche Varianten, welche man [SC80] entnehmen kann.

## 2.5 Aktive Konturen

Um die Konturen der Silhouette zu beschreiben, wird auf aktive Konturen zurückgegriffen. Aktive Konturen werden in der Bildverarbeitung auch ‘Snakes’ genannt und wurden zum ersten Mal von [KWT88] vorgestellt. Sie dienen der Beschreibung von Objektkonturen mit Hilfe parametrischer Kurven und basieren auf der Minimierung einer Energiefunktion. Eine Modellkontur der Snake wird manuell initialisiert und sollte möglichst nahe an der gesuchten Kontur anliegen.

Abhängig von ihrer Form und Lage im Bild wird die Energie der Modellkontur schrittweise verändert, bis sie ein lokales Minimum erreicht und somit der Kontur des gesuchten Objektes entspricht. Es werden Apriori-Informationen dazu verwendet, die Modellkontur an die gesuchte Stelle zu schieben.

Snakes werden von [KWT88] als energieminimierende, parametrische geschlossene Kurve beschrieben, gelenkt von externen Kräften. Die mit der Snake verbundene Energiefunktion  $E$  ist

$$E = E_{\text{int}} + E_{\text{ext}}, \quad (2.17)$$

wobei  $E_{\text{int}}$  die interne und  $E_{\text{ext}}$  die externe Energie darstellt.  $E_{\text{int}}$  berechnet sich aus den geometrischen Eigenschaften der Form der Kontur und beschreibt vereinfacht die Glätte.  $E_{\text{ext}}$  berechnet sich aus dem Bildinhalt im Bezug zur Lage der Kontur. Das Ziel der Snake ist es, die Form mit der geringsten Energie einzunehmen.

$p_1, \dots, p_n$  sei eine diskrete Repräsentation einer Snake, welche eine Sequenz von Punkten auf einer Bildebene ist. Die interne Energie ist eine Summe aus den Kräften der Kontinuität  $E_{\text{cont}}$  und der Krümmung  $E_{\text{curv}}$ .

$$E_{\text{int}} = E_{\text{cont}} + E_{\text{curv}} \quad (2.18)$$

Mit den Kontinuitätskräften  $E_{\text{cont}}$  wird der Abstand zwischen benachbarten Punkten verringert. Dadurch schrumpft die Modellkontur. Durch die Krümmungskräfte  $E_{\text{curv}}$  wird die Modellkontur geglättet. Dadurch wird ein Oszillieren der Modellkontur verhindert.

Die externe Energie wird in [Kass88] beschrieben als

$$E_{\text{ext}} = E_{\text{img}} + E_{\text{con}}, \quad (2.19)$$

wobei  $E_{\text{img}}$  die Bildenergie und  $E_{\text{con}}$  die Energie anderer Zwänge ist. Somit lässt sich die Energie jedes Punktes beschreiben als

$$E_i = \alpha_i E_{\text{cont},i} + \beta_i E_{\text{curv},i} + \gamma_i E_{\text{img},i}, \quad (2.20)$$

wobei  $\alpha, \beta, \gamma$  die Gewichte der Energien sind. Die Gesamtenergie der Snake ist die Summe aller Punkte  $E_i$ . Ein großes  $\alpha$  verteilt die Snakepunkte nach einem gleichen Abstand, ein großes  $\beta$  stumpft den Winkel zwischen Ecken mehr ab und ein großes  $\gamma$  reagiert sensitiver auf die Bildenergie, als auf die Einhaltung der Glattheit.

## 2.6 Template Matching

Das Verfolgen der Person, um Informationen über ihren Standort zu erhalten, wird mit einem Template Matching durchgeführt. Darunter versteht man die Technik, den Bildinhalt einer Schablone, das so genannte Template, in einem Bild zu finden. Dafür wird die zuvor erstellte Schablone Stück für Stück über das Bild geschoben und mit dem darunter liegenden Ausschnitt verglichen. Dabei wird an jeder Position die Ähnlichkeit mit Hilfe eines Ähnlichkeitsmaßes berechnet. Es existieren verschiedene Verfahren der Ähnlichkeitsbestimmung. Folgend werden relevante Verfahren beschrieben.

Das Ursprungsbild besteht aus  $W \times H$  Pixel und das Template aus  $w \times h$  Pixel. Der Ursprung des Bildes befindet sich in dessen linken oberen Ecke. Das Template darf nicht über die Begrenzung des Bildes hinauslaufen. Es existieren daher  $(W - w + 1) \times (H - h + 1)$  Positionen, an die das Template verschoben werden kann. Der Wert jedes Punktes  $(x, y)$  beschreibt die Ähnlichkeit zwischen Template und Bildausschnitt des Ursprungsbildes mit der oberen linken Ecke bei  $(x, y)$  und der unteren rechten Ecke bei  $x+w-1, y+h-1$ . Die Bestimmung der Ähnlichkeit kann auf unterschiedliche Weise berechnet werden. Einen guten Überblick über diese Varianten erhält man unter [Fel05]. Im Folgenden wird eine

Auswahl dieser dargestellt.

Summe der quadratischen Abstände:

$$S(x, y) = \sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} [T(x', y') - I(x + x', y + y')]^2,$$

wobei  $I(x, y)$  der Wert des Bildpunktes an der Stelle  $(x, y)$ , während  $T(x, y)$  der Wert des Templatepixel an der Stelle  $(x, y)$ . Beide Bilder müssen die gleichen Intensitäten aufweisen, daher reagiert es sehr sensibel auf Ausreißer.

Summe der absoluten Abstände:

$$S(x, y) = \frac{\sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} [T(x', y') - I(x + x', y + y')]^2}{\sqrt{\sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} T(x', y')^2 \sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} I(x + x', y + y')^2}}. \quad (2.21)$$

Da die bei diesem Verfahren die absoluten Abstände verwendet werden, ist es weniger Empfindlich als das erste Verfahren.

Kreuzkorrelation:

$$C(x, y) = \sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} T(x', y') I(x + x', y + y'). \quad (2.22)$$

Hierbei werden die Abstände der Grauwerte aller Pixel zum Mittelwert aller Pixel eines Bildes berechnet.

Normalisierte Kreuzkorrelation:

$$\tilde{C}(x, y) = \frac{\sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} T(x', y') I(x + x', y + y')}{\sqrt{\sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} T(x', y')^2 \sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} I(x + x', y + y')^2}}. \quad (2.23)$$

Dieses Verfahren ist unempfindlich gegen Helligkeitsschwankungen, da im Gegensatz zur Kreuzkorrelation durch die Standardabweichung beider Bilder dividiert wird.

Korrelationskoeffizient:

$$R(x, y) = \sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} \tilde{T}(x', y') \tilde{I}(x + x', y + y'), \quad (2.24)$$

wobei  $\tilde{T}(x', y') = T(x', y') - \bar{T}$ ,  $\tilde{I}(x + x', y + y') = I(x + x', y + y') - \bar{I}(x, y)$  und dabei  $\bar{T}$  für den Durchschnittswert der Pixel des Templateraster steht und  $\bar{I}(x, y)$  für den Durchschnittswert der Pixel des aktuellen Fensters aus dem Bild.

Normalisierter Korrelationskoeffizient:

$$\tilde{R}(x, y) = \frac{\sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} \tilde{T}(x', y') \tilde{I}(x + x', y + y')}{\sqrt{\sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} \tilde{T}(x', y')^2 \sum_{y'=0}^{h-1} \sum_{x'=0}^{w-1} \tilde{I}(x + x', y + y')^2}}. \quad (2.25)$$

# Kapitel 3

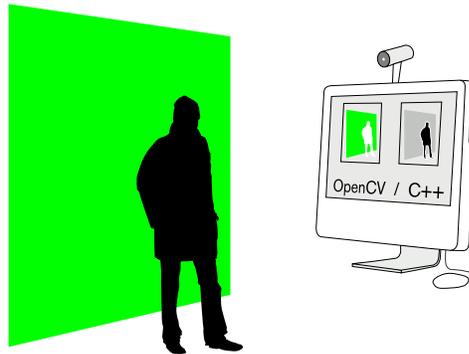
## Eigener Ansatz

Im Folgenden wird die Realisierung des Systems beschrieben. Beginnend mit dem Aufbau, in dem alle Teile und ihr Zusammenspiel des Systems kurz beschrieben werden, folgen anschließend nähere Details zu den Implementierungen dieser einzelnen Module.

### 3.1 Aufbau des Systems

Der Aufbau des Systems besteht im Wesentlichen aus zwei Komponenten. Einem Rechner auf dem die Software installiert ist und einer handelsüblichen FireWire-Kamera, welche mit dem Rechner verbunden ist, mit der eine Person vor einem statischen Hintergrund betrachtet wird.

Die Software wurde unter C++ und der Zuhilfenahme der OpenCV Bibliothek implementiert. In den folgenden Kapiteln werden die einzelnen Module, in die das System gegliedert wurde und welche Rolle OpenCV dabei spielt, beschrieben.



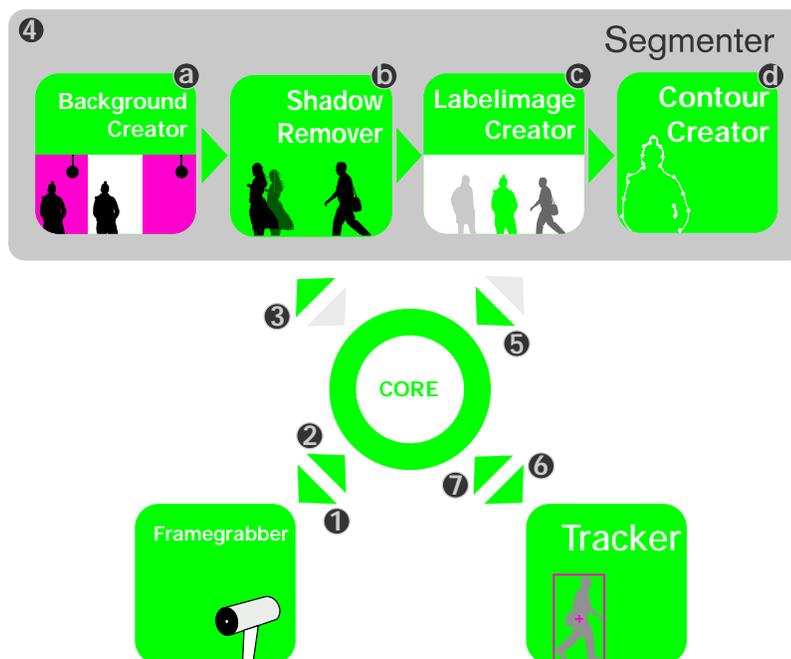
**Bild 3.1:** Komponenten des Systems. Diese Abbildung zeigt ein mögliches Aufbauszenario.

## 3.2 Zusammenspiel der Module

Das System besteht aus der Zusammensetzung verschiedener Module. Jedes Modul ist für die Bearbeitung einer Aufgabe verantwortlich. Dies hat den Vorteil, dass gegebenenfalls Komponenten gegen Andere ausgetauscht werden können, ohne dabei bleibende Komponenten verändern zu müssen. Ein einzelner Durchlauf des Systems kann wie folgt beschrieben werden.

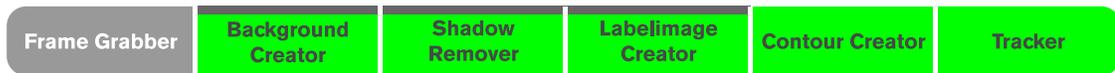
1. Das *Core Modul* fordert einen neuen Frame vom *Framegrabber Modul* an.
2. Der *Framegrabber* liest den aktuellen Frame der Kamera ein und schickt ihn zurück an das *Core Modul*.
3. Nach Erhalt des Frames gibt das *Core Modul* zwecks Segmentierung den Frame weiter zum *Segmenter Modul*.
4. Diesem Modul untergeordnet sind die eigentlich für die Segmentierung verantwortlichen Module.
  - (a) Zuerst erhält das *Background Creator Modul*, zur Lösung des Vordergrundes vom Hintergrund, danach der
  - (b) *Shadow Remover*, zur Entfernung unerwünschter Schatten, dann der

- (c) *Lable Image Creator*, zur Regioneneinteilung und zum Schluss der
  - (d) *Contour Creator*, zur Konturerstellung, den Frame.
5. Das *Segmenter Modul* kann dem *Core Modul* nun die Informationen gefundener Personen zurückliefern.
  6. Sind nützliche Informationen angekommen, schickt der *Core* diese weiter an das *Tracker Modul*.
  7. Das *Tracker Modul* versucht anhand der erhaltenen Informationen gefundene Personen zu verfolgen und schickt diese Daten wieder zurück zum *Core*.
  8. Der *Core* trägt jetzt alle gewünschten Informationen in das Bild ein und stellt es auf dem Bildschirm da. Danach beginnt der Prozess von vorne.



**Bild 3.2:** Gliederung eines Ablaufes. Diese Grafik beschreibt das Zusammenspiel der einzelnen Module.

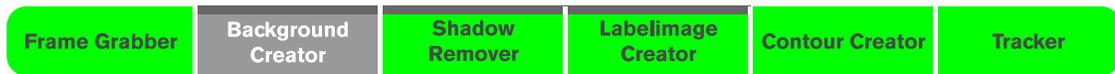
Bild 3.3 dient zur Orientierung und beschreibt den Aktuellen Stand im System, beginnend mit dem Einlesen des Frames durch den *Framegrabber*. Es wird jeweils am Anfang der folgenden Kapitel aktualisiert.



**Bild 3.3:** Aktuelle Position in der Pipeline.

### 3.3 Hintergrund- Vordergrunderstellung

Um den Vordergrund vom Hintergrund zu lösen, wurde das in Kapitel 2.2 vorgestellte *Mixture Of Gaussian* Verfahren verwendet, da dieses bereits in einer Implementierung [KB01] vorlag.



**Bild 3.4:** Aktuelle Position in der Pipeline.

Als Eingabe übergibt man der Funktion den aktuellen Frame (Bild 3.5(a)). Daraufhin wird ein Hintergrund (Bild 3.5(b)) durch Mischung verschiedener Gaußverteilungen erstellt (siehe Kapitel 2.2.1). Nach einigen Durchläufen sollte ein passendes Hintergrundmodell erstellt worden sein. Bewegt sich nun ein Objekt durch das Bild, wird anhand dieses Modells das sich bewegende Objekt als Vordergrund erkannt und in einem binären Vordergrund (Bild 3.9(a)) gespeichert. Hierbei werden Pixel, die dem Vordergrund zugeordnet werden, weiß markiert, während der Hintergrund schwarz ist. Finden Veränderungen im Hintergrund statt, werden diese erkannt und nach einer gewissen Zeit aktualisiert. Zur Konfiguration stehen verschiedene Variablen zur Verfügung. Die zwei wichtigsten sind dabei der Lernfaktor  $\alpha$ , also die Dauer, bis Veränderung in das Hintergrundmodell aufgenommen werden und der Schwellwert  $T$ , aus welchen Gaußverteilungen der Hintergrund besteht. Nach ersten Tests mit der originalen Implementierung war jedoch klar, dass man

in Innenräumen, wo häufig durch Schatten Abdunklungen größerer Bereiche des Hintergrundes stattfinden, nicht ausreichend gute Ergebnisse erhält.



(a) Aktueller Frame der Kamera. (b) Visualisierung des Hintergrundmodells im HSV-Farbraum. (c) Berechneter binärer Vordergrund.

**Bild 3.5:** Darstellung des Hintergrundmodells und des Vordergrundes.

### 3.3.1 Probleme mit RGB

Veränderungen der Lichtverhältnisse wirken sich im RGB-Farbraum auf alle drei Kanäle aus. Das bedeutet, verdunkelt sich die Wand, so handelt es sich um eine ganz neue Farbe, die erst noch dem Hintergrundmodell zugefügt werden muss. Da die Schatten oft nicht konstant an einer Stelle sind und auch nur zeitweise auftreten, werden sie als Bewegung detektiert. Handelt es sich dann noch um viele kleine verschiedene Schattenbereiche, muss das Hintergrundmodell an zu vielen Stellen aktualisiert werden, was eine Weiterverarbeitung der Bilder unmöglich macht, da außer Rauschen nichts zu erkennen ist. Abhilfe schafft die Konvertierung der Bilder vom RGB- in den HSV-Farbraum.

### 3.3.2 Die Lösung mit HSV

Betrachtet man Veränderungen des Hintergrundes durch Schatten bei HSV-Bildern, tritt die Veränderung der Lichtverhältnisse nur im Helligkeits- und Sättigungskanal auf. Der Farbkanal bleibt weitestgehend unbeeinflusst. Um diesen Vorteil auszunutzen, musste das bestehende *Mixture Of Gaussian* -Verfahren an den HSV-Farbraum angepasst werden, unter Beachtung der in Kapitel 2.1.1 beschriebenen Eigenschaften von HSV. Die Abfrage

der Distanz sieht dann folgendermaßen aus:

---

```
float DistanceHue(float h1, float h2) {  
  
    int betrag = sqrt(square(h1 - h2));  
    if(betrag < (180 - betrag))    return betrag;  
    else return (180 - betrag);  
}
```

---

**Listing 3.1:** Ausschnitt aus der `cvbgfb_gaussmix.cpp`. Berechnung der Farbdistanz im HSV-Farbraum.

Auch die Berechnung des Mittelwertes wurde angepasst, damit bei der Aktualisierung der Mittelwerte keine falschen Werte eingetragen werden und somit Farben ungleich schnell in den Hintergrund aufgenommen werden.

### 3.3.3 Weiße Wände

Die der größten Probleme im HSV-Farbraum entstehen bei weißen und schwarzen Wänden. Bei diesen Farbwerten ist der Farbvektor nicht eindeutig definiert, da er schon bei minimalem Rauschen seine Richtung ändert. Dadurch kann mit dem Modell, das lediglich den Farbwinkel betrachtet, kein vernünftiges Hintergrundmodell aufgebaut werden. Ein Distanzvergleich auf einzig den Farbwerten macht daher keinen Sinn. Die Werte des Sättigungs- und Helligkeitskanal dagegen sind bei Farben im Graustufenbereich entweder sehr hoch oder sehr niedrig. Befindet sich zum Beispiel eine weiße Wand im Hintergrund, ist daher anzunehmen, dass die Helligkeit sehr hoch und die Sättigung sehr niedrig in den Bereichen der weissen Wand sein muss. Daher wird an all diesen Stellen im Hintergrund die Farbdistanz nicht auf der Basis des Farbkanals berechnet, sondern nur auf der Grundlage des Helligkeits- und des Sättigungskanals. Wann diese unterschiedlichen Berechnung durchgeführt werden, entscheiden veränderbare Schwellwerte.

## 3.4 Schattenentfernung

Zur Entfernung störender Schatten, die bewegende Objekte oft begleiten, wurde das in Kapitel 2.3 vorgestellte Verfahren verwendet. Das *Shadow-Remover Modul* bekommt als Eingabe das binäre Vordergrundbild, den aktuellen Frame und das aktuelle Hintergrundbild übergeben. Dabei handelt es sich bei dem aktuellen Frame und Hintergrundbild immer noch um HSV-Bilder.



**Bild 3.6:** Aktuelle Position in der Pipeline.

Um nicht jeden Pixel des Vordergrundbildes auf die Eigenschaften eines Schattens überprüfen zu müssen, führt man die Berechnung nur bei allen weißen Pixel des Vordergrundbildes durch. Ob es sich bei einem der Pixel dann um einen Schattenpixel handelt, wird wie im Folgenden beschrieben, errechnet.

### 3.4.1 Test auf Schatten

Der mögliche Schatten wird in allen drei Kanälen überprüft. Dabei sollten an Schattenstellen des aktuellen Frames der Farbton gleich, die Dunkelstufe höher und die Sättigung niedriger als an den schattenfreien Stellen des Hintergrundes sein.

#### Farbkanal

Um falsche Werte bei der Differenz des Frame- und des Hintergrundpixels auszuschließen, muss die Rechnung auf der in Kapitel 2.1.1 beschriebene Weise durchgeführt werden. Dabei ist zu beachten, dass der Farbkanal des HSV-Farbraums in OpenCV nicht wie normalerweise von  $0^\circ$  bis  $360^\circ$ , sondern nur bis  $180^\circ$  reicht, bis er wieder wieder bei  $0^\circ$  beginnt. Die absolute Differenz der beiden Punkte muss unter dem Schwellwert  $\tau_{hue}$  liegen (vergleiche Gleichung 2.13). Der Standardwert wurde nach Tests mit  $\tau_{hue} = 0,5$  festgelegt.

#### Sättigung

Bei der Sättigung muss die Differenz der Pixel unter dem Schwellwert  $\tau_{sat}$  liegen (verglei-

che Gleichung 2.14). Dieser wurde mit  $\tau_{sat} = 1$  nach verschiedenen Tests als Standardwert festgelegt.

### Helligkeit

Das Ergebnis der Division der beiden Punkte muss zwischen den beiden Schwellwerten  $\alpha$  und  $\beta$  liegen (vergleiche Gleichung 2.15). Dabei wurde  $\alpha = 0.3$  und  $\beta = 0.8$  nach verschiedenen Tests als Standardwerte festgelegt.

Sind alle drei Tests erfolgreich, so wird der weisse Pixel an dieser Stelle durch einen schwarzen Pixel ersetzt. Dadurch wird dieser Pixel in den späteren Berechnungen nicht mehr berücksichtigt.



(a) Originalaufnahme.

(b) Binarisierung des Vordergrun-  
des mit Schatten.

(c) Binarisierung des Vordergrun-  
des ohne Schatten.

**Bild 3.7:** Ergebnis der Schattenentfernung.

---

```
laufe über das ganz Bild
```

```
    wenn der Hintergrund weiß ist
```

```
        wenn der Farbton der beiden Hue-Kanäle gleich ist
        und wenn die Sättigung des Kamerabildes niedriger als die des
           Hintergrundes ist
        und wenn die Helligkeit des Kamerabildes niedriger als die des
           Hintergrundes ist
```

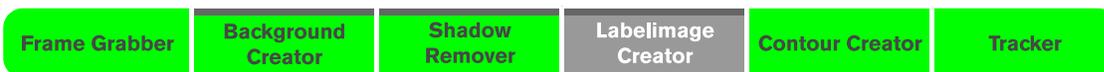
```
            dann färbe den Pixel schwarz
```

---

**Listing 3.2:** Ausschnitt aus dem Shadow Remover Modul. Schattendetektion in HSV.

## 3.5 Region Growing

Durch die Einteilung des binären Vordergrundbildes in einzelne Regionen (Blobs), welches vom *Segmenter* an das *Labelimage Creator Modul* weitergereicht wird, erhält man die Möglichkeit, bewegende Objekte von einander zu trennen, um später einzeln auf sie zuzugreifen zu können. Für diesen Zweck stellt OpenCV eine *Region Growing*-Implementierung bereit.



**Bild 3.8:** Aktuelle Position in der Pipeline.

Der Algorithmus untersucht das Vordergrundbild nach der in Kapitel 2.4 beschriebenen Methode. Um Rauschen und Markierung des Hintergrundes als Blob zu verhindern, werden alle Regionen größer bzw. kleiner bestimmter Schwellwerte ignoriert. Anschließend werden die verbliebenen Regionen verschiedenfarbig eingezeichnet, wobei die größte gefundene Region grün eingefärbt wird. Am Ende stehen ein RGB-Bild, indem alle existierenden Blobs eingezeichnet sind und ein Vektor, in dem die Boundingboxinformationen der einzelnen Labels gespeichert sind, zur Verfügung. Bei der verwendeten Auflösung von  $320 \times 240$  Pixel wurde die Mindestgröße der Regionen auf 500 Pixel und die Maximalgröße auf 40000 Pixel gesetzt, da sonst der Hintergrund als größte Region detektiert wird.

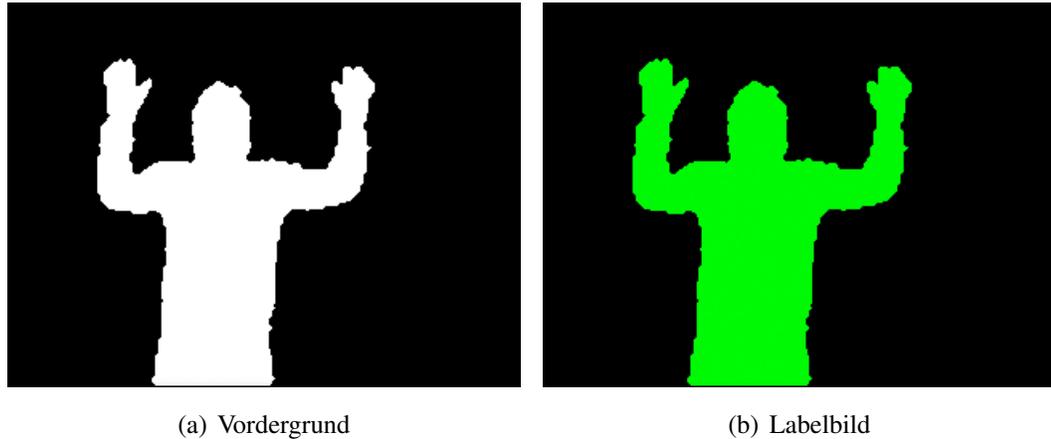
```
blob_image = foreground_image;

int blobArea_min = 500;
int blobArea_max = 40000;
int blobThreshold = 100;

// extract the blobs using a threshold
blobs = CBlobResult( blob_image, NULL, blobThreshold, true );

// discard the blobs with less area than blobArea_min pixels
blobs.Filter( blobs, B_INCLUDE, CBlobGetArea(), B_GREATER, blobArea_min );

// discard the blobs with more area than blobArea_max pixels
blobs.Filter( blobs, B_INCLUDE, CBlobGetArea(), B_LESS, blobArea_max );
```



**Bild 3.9:** (a) ist das aktuelle Vordergrundbild und (b) das in Regionen unterteilte Labelbild.

```

CBlob blobWithBiggestPerimeter;
blobs.GetNthBlob( CBlobGetPerimeter(), 0, blobWithBiggestPerimeter );
blobWithBiggestPerimeter.FillBlob( label_image, CV_RGB( 0, 255, 0 ));

```

**Listing 3.3:** Ausschnitt aus dem Labelimage Creator Modul. Blobs erstellen und einfärben.

## 3.6 Konturerstellung

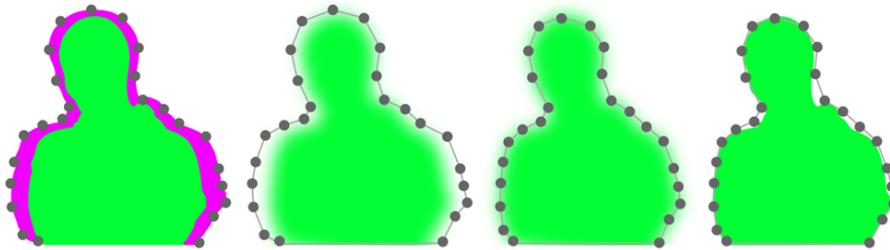
Das *Contour Creator Modul* bekommt ein in einzelne Label unterteiltes Binärbild und einen Vektor mit den Informationen der Boundingbox der einzelnen Labels vom *Core Modul* geschickt. Durch die Verwendung der Boundingboxen muss nicht das ganze Bild, sondern nur der unmittelbare Bereich der Blobs untersucht werden.



**Bild 3.10:** Aktuelle Position in der Pipeline.

Um eine brauchbare Silhouette aus dem binären Labelbild zu erhalten, wird die Funktion ‘cvSnakeImage’ benutzt. Dafür werden zuerst alle gefundenen Blobs des Vordergrundes

mittels Delitation mit einer  $3 \times 3$  Maske vergrößert und anschließend auf die vergrößerten Blobs ein Kantendetektor angewendet, der alle gefundenen Kanten in eine Liste schreibt. Dadurch befinden sich alle Punkte der Liste nicht genau auf, sondern leicht außerhalb der originalen Vordergrundblobs. Diese Liste dient als Initialisierung für die Snake. Das bedeutet, dass die Snake sehr nahe, aber nicht genau auf der zu findenden Kontur geboren wird. Zusätzlich wird ein Gaußfilter auf die originalen Blobs angewandt. Der entstandene Gradient vereinfacht es der Snake, sich an die Kontur anzupassen. Die Punkteliste der finalen Snake wird im nächsten Frame wiederverwendet. Sie dient als Startpunkteliste der neuen Snake und der Kantendetektor muss nicht angewendet werden. Erst nach zwanzig Frames wird die Snake wieder neu initialisiert. Somit wird die aufwendige Initialisierung der Snake verkürzt, was sich positiv auf die Bildwiederholrate auswirken sollte.



**Bild 3.11:** Initialisierung der Snake. Diese Abbildung zeigt die einzelnen Schritte zur Generierung der Snake.

## 3.7 Tracking

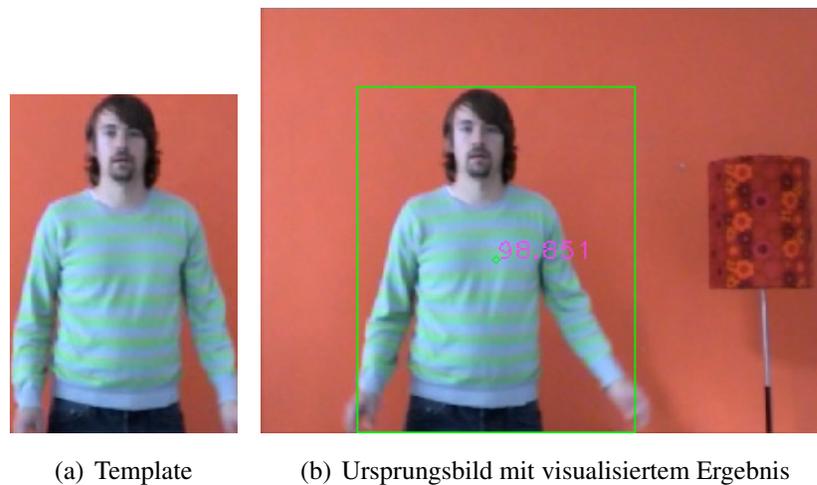
Zur Realisierung der Personenverfolgung und Erhaltung einer glatten Bewegungstrajektorie wurde ein einfaches Template Matching implementiert. Das Tracking Modul bekommt als Eingabebild das aktuelle Vordergrundbild, als auch das Vordergrundbild des letzten Frames.

Als Schablone dient dabei das Vordergrundbild des letzten Frames. Dieses wird mit Hilfe der im Frame zuvor durch das *Labelimage Creator Modul* ermittelten Boundingboxinformationen beschnitten (siehe Bild 3.13(a)). Diese Schablone wird im aktuellen Vor-



**Bild 3.12:** Aktuelle Position in der Pipeline.

dergrundbild (Ursprungsbild) mit der unter Kapitel 2.6 beschriebenen Methode versucht, wieder zu finden. Um diesen Vorgang zu beschleunigen werden sowohl das Template als auch das Ursprungsbild herunter skaliert. An der wahrscheinlichsten Stelle, an der die Schablone dem Ursprungsbild entspricht, wird zur Erkennung die Boundingbox und der Schwerpunkt der Schablone sowie die Trefferquote in Prozent in das originale Kamerabild gezeichnet (vergleiche Bild 3.13(b)). Der aktuelle Schwerpunkt und der des letzten Frames werden dabei noch mit einer Linie verbunden, falls die Trefferquote des Templates über einem gewissen Schwellwert liegt. Wird das Template nicht wiedergefunden, wird auch die Linie der Bewegungstrajektorie wieder aus dem Bild entfernt.



**Bild 3.13:** Ergebnis des *Templatematchings*.

# Kapitel 4

## Experimente und Ergebnisse

Die Schwerpunkte der Tests liegen auf einer Laufzeitanalyse und einer Beurteilung der Qualität der gefundenen Kontur. Anschließend wird die Lernfähigkeit des Hintergrundes und die Schattenentfernung getestet. Alle Versuche werden auf einem 2,16 GHz Intel Core 2 Duo mit 3GB Arbeitsspeicher durchgeführt. Bei der benutzten Kamera handelt es sich um eine Sony DCR-TRV900E. Dabei ist zu erwähnen, dass das Eingangsformat der Kamera zur Verwendung vom Betriebssystem umgewandelt wird, was zusätzlich Rechenleistung kostet. Somit kann es mit anderen Kameras zu leicht veränderten Ergebnissen kommen.

### 4.1 Laufzeitanalyse

Im Wesentlichen hängt die Bildwiederholrate von der Segmentierung ab. Darunter versteht man die Hintergrunderstellung, Schattenentfernung und Label-Erstellung. Dazu kommt noch die Generierung der Snake und das Verfolgen der gefundenen Objekte. Da das Hintergrundmodell mehr Zeit in Anspruch nimmt, wenn es aktualisiert werden muss, wird am Anfang jedes Testszenarios eine Veränderung des Hintergrundes erzwungen, indem bei der Initialisierung eine Person im Bild steht. Alle Testvideos werden vom Band der DV-Kamera eingespielt und können somit wie die Live-Bilder der Kamera bearbeitet werden, wobei zusätzlich die Wiederholbarkeit der Tests gewährleistet ist.

| Operation   | Zeit  |
|---|-------|
| Frame einlesen (Frame Grabber Modul)                  | 40ms  |
| RGB zu HSV Konvertierung (Core)                       | 2ms   |
| Hintergrundmodel erstellen (Background Creator Modul) | 120ms |
| Schatten entfernen (Shadow Remover Modul)             | 3ms   |
| Regionen verlinken (Labelimage Creator Modul)         | 11ms  |
| Snake generieren (Contour Extractor Modul)            | 21ms  |
| Objekt verfolgen (Template Matching Modul)            | 30ms  |
| Bilder darstellen (Core)                              | 3ms   |
| Gesamt  | 230ms |

**Tabelle 4.1:** Laufzeitanalyse mit mittleren Ausführungszeiten von 4 Videos nach jeweils 300 Frames. In Klammern steht das dafür verantwortliche Modul. Die Bildgröße beträgt  $320 \times 240$  Pixel.

Tabelle 4.1 zeigt, dass gut ein Drittel der Leistung zur Erstellung des Hintergrundmodells beansprucht wird. Mit einem Gesamtergebnis von etwa 3,5 Bildern pro Sekunde ist das System nicht wirklich für die Verfolgung von Bewegungsabläufen geeignet, wenn es sich um Echtzeitbilder einer Kamera handelt. Alternativ können die Bilder jedoch von der Festplatte eingelesen werden, dabei wird jedes Bild bearbeitet und das Ergebnis in einem neuen Videofilm gespeichert, welcher sich dann in Echtzeit anschauen lässt.

## 4.2 Qualität der Kontur

Da eine objektive Beurteilung der Kontur nur schwer bis gar nicht durchführbar ist, wird die qualitative Einstufung der errechneten Konturen durch eine subjektive Beurteilung durchgeführt. Dabei werden diese in drei Klassen unterteilt.

- Gut

Die Kontur entspricht, bis auf kleinste Abweichungen, der des Benutzers.

- **Brauchbar**  
Es treten kleinere Fehler auf. Die Kontur wirkt unsauber.
- **Schlecht**  
Teile der Kontur sind abgetrennt bzw. durchlöchert oder Bereiche des Hintergrundes werden zur Kontur hinzugefügt.

Eine allgemeine Beurteilung der Konturqualität ist nicht möglich, da zu viele Parameter und ihrer Kombinationsmöglichkeiten dabei eine Rolle spielen. Deshalb wurden repräsentativ verschieden Szenen ausgewählt, die das Funktionieren, aber auch die Schwachstellen der Konturgenerierung zeigen. Gerade weil sich die Konturerzeugung in gewissen Fällen schwieriger gestaltet, es aber nicht einfach zu entscheiden ist, wie viele von diesen Situationen wie lange in den Testvideos vorkommen sollten, wird das Ergebnis unter Berücksichtigung der oben genannten Güteklassen, solcher Szenen beschrieben, ohne jedoch auf prozentualen Anteil der einzelnen Güteklassen einzugehen. Die meisten Tests wurden vor weißen Wänden durchgeführt, da zum einen der Anteil an weißen Wänden sehr hoch ist und zum anderen das Hintergrundmodell (wie in Kapitel 3.3.3 beschrieben) dabei die größten Probleme aufweist. Außerdem wurde die Snake, nicht wie unter Kapitel 3.6 beschrieben, nach jedem Durchgang neu initialisiert. Dies hatte zum einen eine erhebliche Steigerung der Qualität und zum anderen eine kaum merkbare Verringerung der Leistung zur Folge.

Bei normalem Tageslicht vor weißem Hintergrund (Bild 4.1(a) bis Bild 4.1(c)) mit verschiedener Kleidung getestet ist die Qualität der erstellten aktiven Kontur mit gut bis brauchbar zu bewerten. Bei eher schlechten Lichtverhältnissen (Bild 4.1(d)) sind die Konturen mit brauchbar zu bewerten. Sind Hintergrund- und Kleidungsfarbe, wie in Bild 4.1(c) gezeigt, jedoch sehr ähnlich, liefert die Segmentierung schlechte Ergebnisse.

Eigentlich sollten leichte Lichtveränderungen durch das Hintergrundmodell kompensiert werden. Da jedoch bei weißen Wänden während der Trennung des Vordergrundes vom Hintergrund nur die Helligkeit und die Sättigung der Bilder in Betracht gezogen werden, macht sich das negativ auf die Lichtempfindlichkeit des Modells bemerkbar. Dazu zählen nicht nur Lichtveränderungen, sondern auch der Weißabgleich oder die Blendensteuerung der Kamera, auf die man nicht immer Einfluss nehmen kann.



**Bild 4.1:** Verschiedene Farben vor weißem Hintergrund.

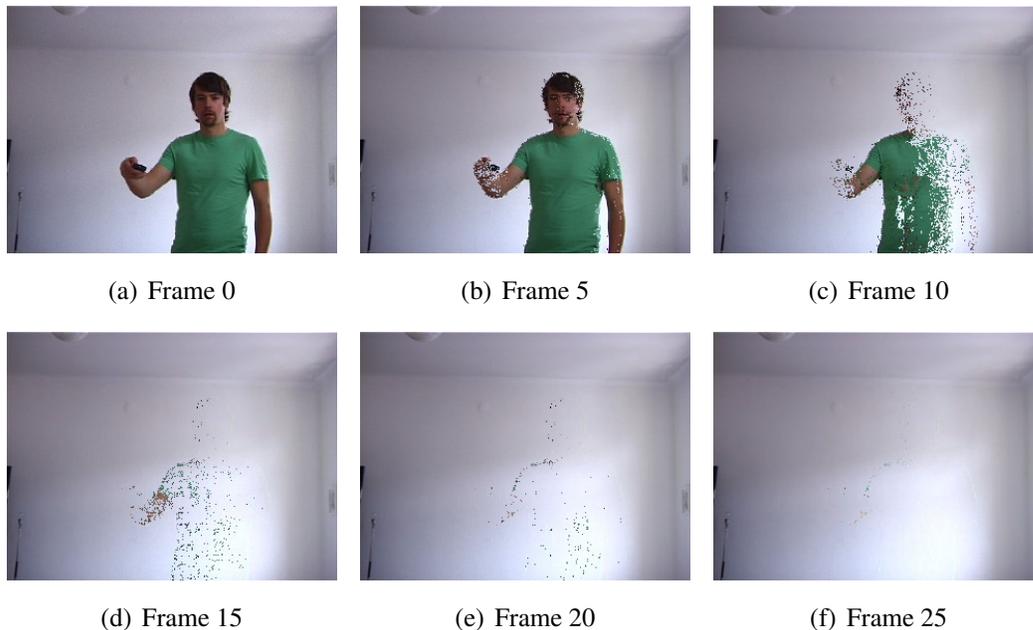
Nimmt die zu segmentierende Person stark konkave Formen an, wie es Bild 4.2 zeigt, kann es passieren, dass die Snake nicht eng genug anliegt und dadurch unbrauchbare Ergebnisse erzielt werden. Die ermittelte Kontur des Blobs dagegen liegt genau auf der des Objektes.



**Bild 4.2:** Vergleich von aktiven Kontur und Kontur der Blobs.

## 4.3 Lernen des Hintergrundes

Die Lernfähigkeit des Hintergrundmodells ist ein großer Vorteil gegenüber der normalen Differenzbildtechnik. Dadurch können aber auch unbeabsichtigt Personen in den Hintergrund aufgenommen werden. Mit diesem Test soll die Lernfähigkeit des Hintergrundmodells getestet werden. Dafür wurde am Anfang der Sequenz eine Person im Bild positioniert (Bild 4.3(a)), die den Hintergrund während der Initialisierung verdeckt, sich aber direkt aus dem Bild bewegt. Die Aktualisierung sollte nicht mehr als 50 Bilder in Anspruch nehmen, um die Segmentierung der Person nicht durch ein falsches Hintergrundmodell zu stören.



**Bild 4.3:** Verlauf der Aktualisierung des Hintergrundmodells.

Bei 4.3(a) handelt es sich um den Initialisierungsframe. Da die Person sich nach diesem Frame direkt aus dem Bild bewegt und die Gewichtung der Gaußkomponenten des Hintergrundmodells noch nicht so hoch ist, sieht man schon bereits nach wenigen Frames (Bild 4.3(b) und Bild 4.3(c)) starke Veränderung im Hintergrundmodell. Die letzten Überreste der Person (Bild 4.3(f)) haben keine negativen Auswirkungen mehr auf die Segmen-

tierung.

Bleibt die Person allerdings bei der Initialisierung des Hintergrundes länger an der selben Stelle stehen, wird mehr Zeit benötigt bis diese wieder aus dem Hintergrund entfernt wird. Solange die Person noch teilweise im Hintergrund sichtbar ist, kommt es zu Störungen bei der Segmentierung. Das gleiche gilt auch für Personen, die längere Zeit an der gleichen Position ausharren und dadurch in den Hintergrund aufgenommen werden.

## 4.4 Entfernung von Schatten

Zur Überprüfung des Schattenentfernen-Algorithmus wurden Szenen mit verschiedenen Schattensituationen nachgestellt. Ein Schatten zählt als entfernt, wenn er die Kontur der Person nicht verändert (siehe Bild 4.4), das bedeutet, auch wenn ein Schatten nicht komplett entfernt wurde, aber die Reste des Schattens in keiner Weise die Segmentierung beeinträchtigen, wird er als 'Entfernt' gewertet.

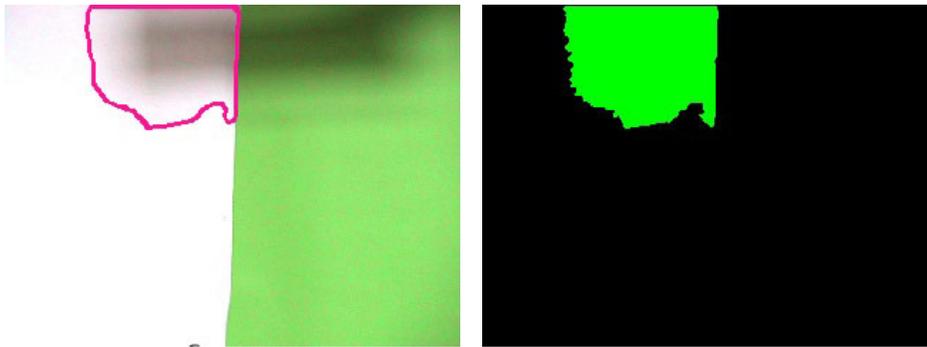


**Bild 4.4:** Kontur ohne Schatten.

Handelt es sich um Schatten vor farbigen Wänden, wird in den meisten Fällen der Schatten restlos entfernt, so dass keine Beeinträchtigung der Kontur stattfindet (vergleiche Bild 4.5). Die Größe des Schattens hat dabei keinen negativen Einfluss. Wird jedoch der Farbton des Hintergrundes vom Schatten verändert, kommt es zu Problemen, da wie in Kapitel 3.4 beschrieben, der gleiche Farbton ein Kriterium zur korrekten Entfernung ist. Das geschieht

wenn z. B. der Schatten den Hintergrund so stark abdunkelt, dass dadurch der Farbton nicht mehr richtig erkannt wird oder wenn der Farbton direkt vom Schatten beeinflusst wird.

Bei weißen Wänden jedoch spielt das unter Kapitel 3.3.3 Beschrieben Problem eine große Rolle. Denn Schatten werden nur entfernt, wenn der Farbton ähnlich ist, dies ist bei einem springenden Farbvektor allerdings nicht der Fall. So kann in vielen Fällen der Schatten nicht richtig entfernt werden (siehe 4.5).



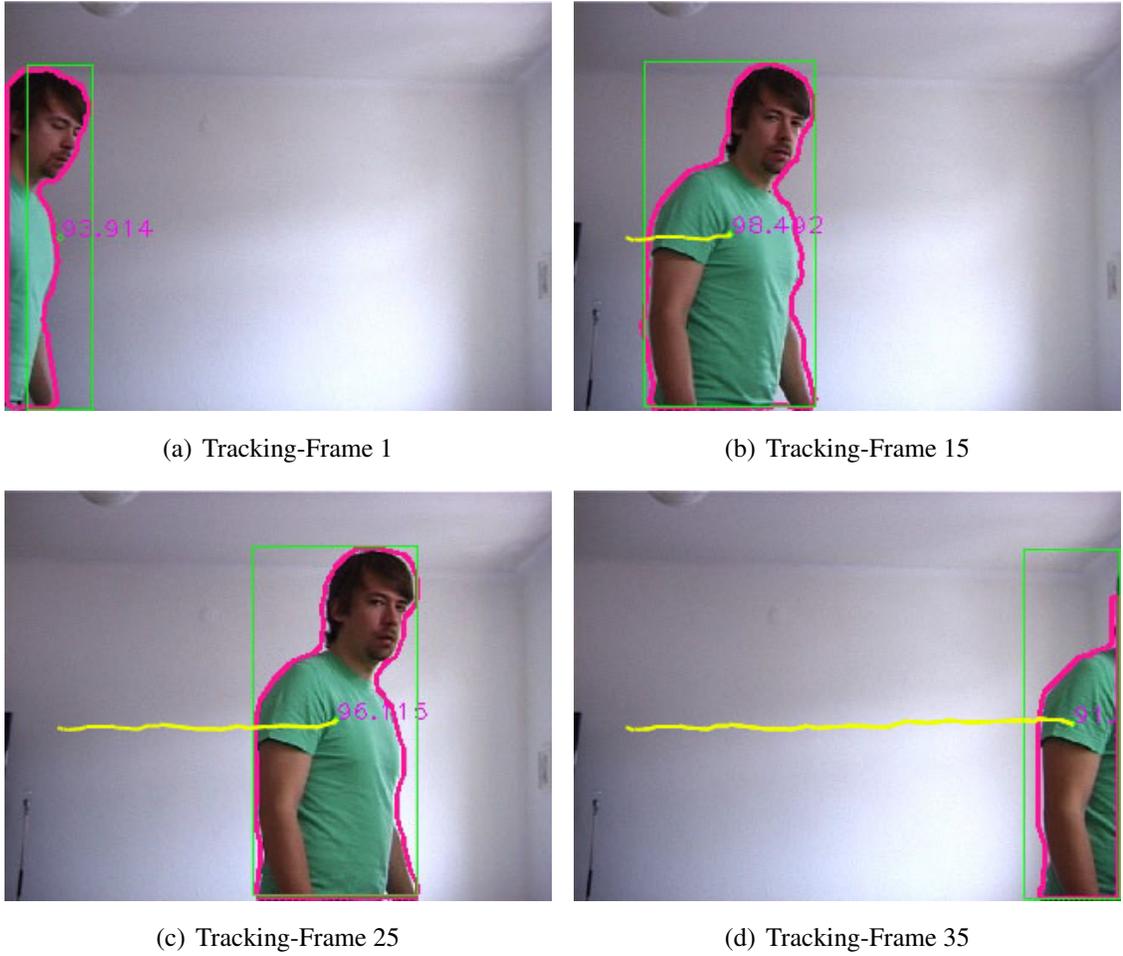
(a) Schatten vor weißer und farbiger Wand.

(b) Detektierter Blob.

**Bild 4.5:** Entfernung des Schattens mit anschließender Blob Generierung.

## 4.5 Tracking

Das Trackingverfahren wurde auf alle Testvideos angewandt. Es ist abhängig von der Qualität der erkannten Blobs. Befinden sich keine Störungen im Vordergrund, funktioniert das Wiederfinden der Person gut. Treten dagegen Störungen oder zusätzliche, sich bewegende Objekte auf, kann es passieren, dass das falsche Objekt verfolgt wird. Die Bewegungstrajektorie der verfolgten Personen kann als glatt (Bild 4.6) beschrieben werden. Einzig in Situationen, wenn sich die Person nahe an der Kamera befindet und dann z. B. durch Heben der Hand die Boundingbox vergrößert wird, kommt es zu leichten Sprüngen in der Trajektorie, denn durch diese Bewegung wird der Schwerpunkt der Person gehoben.



**Bild 4.6:** Verlauf des Trackings mit Trajektorie.

# Kapitel 5

## Zusammenfassung und Ausblick

Innerhalb dieses Kapitels werden zusammenfassend die erreichten Ergebnisse dargestellt. Im drauf folgenden Abschnitt wird ein kurzer Ausblick über nützliche Erweiterungen gegeben, die aufgrund des zeitlich vorgeschriebenen Rahmens keinen Platz in dieser Studienarbeit fanden, es aber dennoch wert sind hier erwähnt zu werden.

### 5.1 Zusammenfassung

Die zentralen vorher festgesteckten Eigenschaften zur Realisierung des Personverfolgungssystem waren:

- Die Segmentierung des Vordergrundes vom Hintergrund mit Hilfe der Differenzbildtechnik, wobei der Hintergrund lernfähige Eigenschaften besitzen sollte.
- Schatten der zu verfolgenden Personen sollten dabei erkannt und entfernt werden.
- Die Silhouette der Person musste unter Beachtung von Glattheit der Kontur bestimmt werden.
- Die gefundene Kontur sollte über die weiteren Bilddaten verfolgt werden, wobei auf eine glatte Bewegungstrajektorie Wert gelegt wurde.

Um den Vordergrund vom Hintergrund zu trennen und somit sich bewegende Personen segmentieren zu können, wurde ein 'Mixture of Gaussian' Modell auf die Verwendbarkeit des HSV-Farbraums erweitert. Somit war es möglich störende Schatten, unter Berücksichtigung der Eigenschaften des HSV-Farbkanals, zu entfernen. Es wurden zusammenhängende Regionen verbunden und somit konnte mittels aktiver Kontur die Silhouette der Person bestimmt werden. Diese wurde mit einem beschleunigten Template-Matching über die weiteren Bilddaten verfolgt und die Bewegungstrajektorie der Person wurde dargestellt. Somit wurden alle zuvor gesteckten Ziele erreicht.

## 5.2 Ausblick

**Verfolgung mehrerer Personen** Während des aktuellen Standes ist es nur bedingt bis gar nicht möglich mehr als eine Person zu verfolgen. In manchen Situationen ist das leider nicht ausreichend und somit müsste das System erweitert werden. Dies könnte durch eine Veränderung der Verfolgungsstrategie geschehen.

**Verwendung auf anderen Plattformen mit anderer Hardware** Dieses System wurde ausgiebig auf einem Rechner unter Mac OS X 10.4.11 mit vergleichbaren Kameras getestet. Da es unter C++ und OpenCV programmiert wurde, sollte es auch unter Linux und Windows getestet werden. Eine lauffähige Mac OS X und Linux Version liegen der Studiarbeit bei. Auch der Vergleich verschiedener Kameras, von der günstigen Webcam bis zur teuren Hightech-Kamera wäre interessant.

**Erweiterung der Personenerkennung** Ein Erweiterung der Personenerkennung um ercheinungsbasiertem Modell, wie es unter [TMM06] beschrieben wir, wäre von großem Nutzen. Dadurch könnten die einzelnen Gliedmaße des menschlichen Körpers erkannt und verfolgt werden. Zur Zeit wird jedes Objekt, welches sich vor der Kamera bewegt, verfolgt.

**Verbesserung der Schattenentfernung** Da die Entfernung von Schatten bei weißen Wänden Schwierigkeiten aufweist, wäre an dieser Stelle eine Lösung des Problems wünschenswert.

# Literaturverzeichnis

- [CGP<sup>+</sup>01] CUCCHIARA, R. ; GRANA, C. ; PICCARDI, M. ; PRATI, A. ; SIROTTI, S.: Improving shadow suppression in moving object detection with HSVcolor information. In: *Proceedings of the Fourth International IEEE Conference on Intelligent Transportation Systems* (2001), 8, S. 334–339
- [Fel05] FELDMANN, Tobias: *Kombination verschiedener Ähnlichkeitsmaße für die 2-D/3-D Registrierung von Röntgenbildern*, Universität Koblenz-Landau, www.uni-koblenz.de, Diplomarbeit, 2005. – Diplomarbeit teilweise extern betreut von Dr. Frank Deinzer bei Siemens Medical Solutions, Forchheim.
- [KB01] KAEWTRAKULPONG, P. ; BOWDEN, R.: *An Improved Adaptive Background Mixture Model for*. 4 2001
- [KWT88] KASS, Michael ; WITKIN, Andrew ; TERZOPOULOS, Demetri: Snakes: Active contour models. In: *International Journal of Computer Vision* 1 (1988), 1, Nr. 4, 321–331. <http://dx.doi.org/10.1007/BF00133570>
- [MG01] MOESLUND, Thomas B. ; GRANUM, Erik: A Survey of Computer Vision-Based Human Motion Capture. In: *Computer Vision and Image Understanding: CVIU* 81 (2001), Nr. 3, 231–268. [citeseer.ist.psu.edu/moeslund01survey.html](http://citeseer.ist.psu.edu/moeslund01survey.html)
- [SC80] SNYDER, W.E. ; COWART, A.E.: An Iterative Approach to Region Growing. In: *ICPR80*, 1980

- [SG00] STAUFFER, Chris ; GRIMSON, W. Eric L.: Learning Patterns of Activity Using Real-Time Tracking. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, <http://www.computer.org/tpami/> 22 (2000), Nr. 08, 747-757. <https://www.uni-koblenz.de/~agas/Pool/Stauffer2000LPO.pdf>. ISBN 0162-8828
- [TM05] THOME, Nicolas ; MIGUET, Serge: A Robust Appearance Model for Tracking Human Motions. In: *AVSS (IEEE International Conference on Advanced Video and Signal-Based Surveillance)*, 2005
- [TMM06] THOME, Nicolas ; MERAD, Djamel ; MIGUET, Serge: Human Body Part Labeling and Tracking Using Graph Matching Theory. In: *AVSS '06: Proceedings of the IEEE International Conference on Video and Signal Based Surveillance*. Washington, DC, USA : IEEE Computer Society, 2006. – ISBN 0-7695-2688-8, 38
- [Wik08a] WIKIPEDIA: *HSV-Farbraum*. 01 2008. – <http://de.wikipedia.org/wiki/HSV-Farbraum>
- [Wik08b] WIKIPEDIA: *RGB-Farbraum*. 01 2008. – <http://de.wikipedia.org/wiki/RGB-Farbraum>