Universität Karlsruhe (TH)

Mehrpersonentracking mittels Farbe und Detektorkaskaden

Diplomarbeit am Institut für Theoretische Informatik
Prof. Dr.rer.nat. A. Waibel
Fakultät für Informatik
Universität Karlsruhe (TH)

von

Cand. inform.
Alexander Elbs

Betreuer:
Dipl.-Inform. Keni Bernardin
Dr.-Ing. Rainer Stiefelhagen
Prof. Dr.rer.nat. A. Waibel

Registrierungsdatum: 1. März 2005
Abgabedatum: 31. August 2005

Hiermit erkläre ich, die vorliegende Arbeit selbständig verfaßt und keine anderen als die angegebenen Literaturhilfsmittel verwendet zu haben.

Karlsruhe, den 31.8.05

A. Elbs

Alexander Elbs

Zusammenfassung

In dieser Arbeit wird ein Mehrpersonentracker vorgestellt. Er ist in der Lage, mehrere Personen innerhalb eines Raums zu erkennen und mit hoher Präzision und Robustheit zu tracken. Auf den Bildern der vier kalibrierten Kameras in den Ecken des Raums werden jeweils maximal drei verschiedene Objekterkenner (Haarkaskaden für Gesicht, Oberkörper und Ganzkörper) angewendet. Die Objekterkenner werden mit einem Bewegungsdetektor kombiniert, um Farbhistogramme zu initialisieren oder 2D-Tracks zu bestätigen. Die Farbhistogramme werden fehlerbereinigt und kontinuierlich adaptiert. Sie werden von einem farbbasierten Trackingalgorithmus, Meanshift verwendet, um Personen zu verfolgen. Die 2D-Tracks werden mit einer umfangreichen 3D-Fusionsalgorithmus zu 3D-Hypothesen kombiniert. Obwohl sowohl der Objekterkenner, als auch der Bewegungsdetektor für sich kein erfolgreiches Tracken ermöglichen, erreicht die Fusion der beiden mithilfe mehrerer Techniken eine hohe Erkennungsleistung. Der Tracker wurde auf mehreren Testsequenzen und Kombinationen von Objekterkennern evaluiert, wobei sich die Kombination der Haarkaskaden Oberkörper mit Ganzkörper als die durchschnittlich beste heraus gestellt hat. Es wird eine Trackinggenauigkeit von 39% bei einer Positionsabweichung von 17cm erreicht.

Danksagung

Hiermit möchte ich mich bei allen bedanken, die mir während meiner Arbeit geholfen haben. Insbesondere Keni Bernardin, Anja Wolf, Horst Wenske und Daniel Friedrich, die geduldig diese Arbeit gelesen und korrigiert haben. Weiterhin allen, die für die Testaufnahmen zur Verfügung standen.

Danke für Eure Hilfe!

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Motivation 1

1.2 Überblick 2

2 Stand der Forschung **3**

2.1 Techniken zur Initialisierung 3

2.2 Techniken zum Tracken 4

2.3 Handhabung von Verdeckung 5

3 Grundlagen **7**

3.1 Hintergrundsegmentierung 7

3.2 Farbhistogramme und ihre Verwendung 11

3.3 Triangulation und Korrespondenzproblem 14

3.4 Haarkaskaden 17

3.5 Trackingansätze 21

4 Entwickeltes Trackingsystem **25**

4.1 2D-Tracking mittels Farbhistogrammen 26

4.1.1 Initialisierung von Farbhistogrammen 27

4.1.2 Fehlerbereinigung von Farbhistogrammen 29

4.1.3 Adaption von Farbhistogrammen 33

4.1.4 Falsche Treffer 37

4.2 Erzeugung von 2D-Hypothesen 39

4.3 Erzeugung von 3D-Hypothesen 42

4.3.1 Korrespondenzfehler 42

4.3.2 Fehlerbehandlung 47

4.3.3 2D-Kreditsystem mit 3D-Wissen 49

4.3.4 Handhabung von Verdeckung 50

4.3.5 Der Hinweisgenerator 52
4.3.6 Triangulationsfehler 55
4.4 Ausgabehypothesen 56
4.5 Zusammenfassung der Techniken 59

5 Implementierung **61**
5.1 Allgemeines 61
5.2 Haarkaskaden 61
5.3 Design 61
5.4 Schwellwerte 64
5.5 Probleme bei der Aufnahme 65
5.6 Verwendung der Software 66

6 Experimente und Resultate **67**
6.1 Grundwahrheit und Fehlerarten 67
6.2 Zuordnung von Grundwahrheit zu Ausgabehypothese 68
6.3 Metriken 69
6.4 Ergebnisse auf Entwicklungsdaten 70
6.5 Details 72
6.6 Ergebnisse auf Testdaten 74

7 Zusammenfassung und Ausblick **77**
7.1 Zusammenfassung 77
7.2 Ausblick 78

Abbildungsverzeichnis

1 Beispielbilder für Bewegungsdetektoren 10

2 Beispielbild für eine Back Projection 12

3 Drei Kamerasichten auf ein Objekt. Die LoVs sind fehlerbehaftet 15

4 Kaskade: Sobald ein Knoten ablehnt, wird das Suchfenster verworfen (F), ansonsten wird fortgefahren (T) [18]. 18

5 Beispiel für Features [18] 19

6 Alle verwendeten Features [9]. 20

7 Ablauf einer Iteration eines Partikelfilters [7] 22

8 Back Projektionen eines normal eingelernten Farbhistogramms einer Person und verschiedener Bereinigungsmethoden 31

9 Inkrementelle Adaption von Farbhistogrammen 35

10 Beispiel für false positive Treffer der Gesichtskaskade 37

11 Normalisierung der Typen: Mittelpunkt des Ganzkörpers wird bei anderen Typen extrapoliert. 40

12 Fehlermöglichkeiten bei zwei oder drei aktiven 2D-Trackern . . 41

13 Als blaue Punkte eingezeichnete LoVs in CAM2 bis CAM4 für den Mittelpunkt der schwarz gekleideten Person aus CAM1 . . 44

14 Falsch gelöstes Korrespondenzproblem: P_1 bis P_4 stehen so, dass sich die LoVs in V schneiden und statt vier nur eine (virtuelle) Person V erkannt wird. 45

15 Korrespondenzproblem 54

16 Schematische Darstellung des Raumes 57

17 Grober Überblick über Design der Implementierung. Die Linien stellen eine Interaktion oder Enthaltungseigenschaft dar. . . 62

18 CAM1 zeigt ein neueres Bild als CAM2 bis CAM4 65

19 Verlauf des Trackens bei Aufnahme B mit Oberkörperhaarkaskade 73

1 Einleitung

1.1 Motivation

Für einen Menschen ist es leicht, mit einem Blick auf ein Bild viele verschiedene Informationen herauszufinden. Er kann erkennen, wieviele Personen sich in diesem Bild befinden, was diese gerade tun, wie sie angezogen sind und vieles mehr. Ein Computer kann mithilfe einer Kamera ebenfalls optische Informationen über seine Umwelt ermitteln. Allerdings ist für einen Computer ein Kamerabild zunächst nur eine geordnete Aufzählung von Farbwerten in einem bestimmten Farbraum. Er kennt noch keine weitergehenden Interpretationen dieses Kamerabildes. Das Extrahieren von Informationen aus Bildern oder Videosequenzen wird in der Informatik seit Dekaden erforscht. Dennoch sind die Fähigkeiten eines Computers denen eines Menschen in dieser Hinsicht weit unterlegen.

Die Anwendungsmöglichkeiten für einen "sehenden" Computer sind vielfältig. Sie reichen von beispielsweise Fahrerassistenzsystemen, die einen Autofahrer entlasten bis zu "intelligenten Räumen", die ihre Benutzer erkennen und unterstützen.

Am Institut für Theoretische Informatik an der Universität Karlsruhe wird an einem so genannten "intelligenten Raum" geforscht. Dieser Raum hat mehrere verschiedene optische und akustische Sensoren, die mit Computern verbunden sind. Für diesen Raum wurden bereits mehrere Algorithmen entwickelt oder angepasst:

- Ein Gestenerkenner [11], der ermittelt, in welche Richtung eine Person deutet.
- Ein System zur Ermittlung der Kopfposition eines Menschen [19], um herauszufinden, in welche Richtung die Aufmerksamkeit gerichtet ist.
- Ein Spracherkenner [8], der frei gesprochene Sprache versteht und in Text umwandeln kann.
- Personentracker [4, 22], die versuchen die Position von Personen innerhalb des Raums zu ermitteln.

Ein *Personentracker*, im weiteren *Tracker* genannt, soll zuverlässig Personen innerhalb des Raumes auffinden können, um diese dann zu verfolgen. Im Idealfall kennt ein Tracker zu jedem Zeitpunkt von allen Personen im Raum ihre 3D-Koordinaten mit hoher Präzision und verwechselt die Personen niemals.

Ein robust arbeitender Tracker kann dazu verwendet werden, weiterführende Algorithmen zuverlässiger auszuführen. Ein Gestenerkenner muss nicht mehr selbst das ganze Kamerabild nach Händen und Köpfen durchsuchen, sondern kann seine Suche auf die Positionen einschränken, die der Tracker liefert. Auf ähnliche Weise können anderen Algorithmen von den Positionsdaten eines Trackers profitieren.

Durch Kombination dieser grundlegenden Algorithmen sollte es möglich sein, komplexes Verhalten zu implementieren. Beispielsweise betritt eine Person den "intelligenten Raum". Sie wird von einem Tracker erkannt und verfolgt. Ein Gesichtserkenner identifiziert die Person und lädt angepasste Sprachdaten für den Spracherkenner. Eine bewegbare Kamera richtet sich auf den Kopf der Person, um mittels Lippenlesen die Spracherkennungsleistung zu erhöhen. Der Gestenerkenner findet heraus, dass sie auf einen Fernseher deutet und der Spracherkenner hört "ARD". Daraufhin wird dieser Bildschirm auf das entsprechende Fernsehprogramm umgeschaltet.

Deshalb ist ein robuster Mehrpersonentracker, der mehrere der Kameras verwendet, ein wichtiger Baustein für "intelligente Räume".

1.2 Überblick

Im folgenden Kapitel 2 wird der Stand der Forschung präsentiert. Auf dem Gebiet der Personenverfolgung gibt es viele Arbeiten mit zum Teil sehr unterschiedlichen Ansätzen. Daran schließt sich ein Kapitel 3 über grundlegende Algorithmen an, die zum Verständnis dieser Arbeit nötig sind. Das Hauptkapitel 4 diskutiert das Design des Trackers und in Kapitel 5 sind einige Details der Implementierung erläutert. Die Leistungsfähigkeit des Trackers wird im Kapitel 6 diskutiert. Darauf folgt eine Zusammenfassung mit einem Ausblick auf mögliche Verbesserungen und Erweiterungen.

2 Stand der Forschung

Bei einem Personentracker stellt sich immer das Problem der Initialisierung. Der Tracker muss zu einem Zeitpunkt beschließen, dass etwas Gesehenes eine Person darstellt, um diese dann zu verfolgen. Viele Arbeiten konzentrieren sich auf das eigentliche Trackingproblem und versuchen die Initialisierung möglichst einfach zu erledigen. Wenn ein Tracker eine Person erkennt und verfolgt, wird dies im Folgenden als *Track* bezeichnet. Im Idealfall gibt es für jede Person genau einen passenden Track.

2.1 Techniken zur Initialisierung

Eine einfache Möglichkeit ist *Manuelle Initialisierung*. Das hat den Vorteil, dass ein Tracker handoptimierte Daten verwendet, die nur wenige Fehler enthalten. Entsprechend ist der eigentliche Trackingvorgang etwas leichter. Während der Prototypphase kann diese Methode sinnvoll sein, um verschiedene Trackingalgorithmen zu testen. Für ein real einsetzbares System ist Manuelle Initialisierung allerdings unbrauchbar, da sie möglichst ohne Eingreifen eines Nutzers arbeiten sollen.

Eine Möglichkeit, das Initialisierungsproblem zu vereinfachen, ist die Verwendung von so genannten *Creation* und *Deletion Zones* [17, 2]. Hierbei gibt es im Kamerabild Bereiche mit unterschiedlicher Wahrscheinlichkeit für das Entstehen oder Löschen von Tracks. Ein Tracker für den Außenbereich definiert dazu den Rand des Bildes als solch eine Zone. Im Innenbereich werden hierfür häufig Türen benutzt, da nur dort Personen den Erfassungsbereich betreten oder verlassen können. Vorteil dieser Methode ist, dass die Entstehung von falschen Tracks leichter vermieden werden kann und bestehende Tracks nicht fälschlicherweise gelöscht werden.

Häufig werden *Bewegungsdetektoren* verwendet, die durch Analyse einer Bildsequenz Bereiche mit Bewegung finden. Eine weitere Möglichkeit der Initialisierung ist deshalb die Verwendung von zusammenhängenden Bewegungszonen. Falls solch eine Zone erkannt wird, die groß genug ist, um ein Mensch zu sein, wird davon ausgegangen, dass es sich tatsächlich um eine Person handelt. Nach diesem Grundprinzip initialisieren mehrere Systeme [23, 6, 17], wobei noch zusätzliche Kriterien verwendet werden:

- Das Pfinder-System [23] versucht den Umriss der einzulernenden Person zu erkennen.

- Das W4-System [6] lernt eine Zone ein, falls diese für mehrere aufeinander folgende Bilder stabil ist und sich nicht mit einer anderen Zone überlappt.
- Ein weiteres System [17] versucht mehrere Bewegungszonen mit einer wahrscheinlichsten Personenkonfiguration zu erklären.

Eine weitere Möglichkeit ist die Verwendung eines *generischen Farbmodells* [14] einer Person, um diese einzulernen.

2.2 Techniken zum Tracken

Für das eigentlich Tracken einer Personen gibt es viele verschiedene Strategien.

Pfinder [23] ist einer der bekanntesten Tracker. Er ist für das Tracken einer einzelnen Person vor einem relativ sauberen, statischen Hintergrund konzipiert. Es wird die räumliche Farbverteilung einer eingelernten Person benutzt, um im nächsten Bild deren Position zu bestimmen. Damit gehört Pfinder zur Klasse der Farbtracker. Pfinder modelliert die Person recht präzise und kann damit sogar Gestenerkennung durchführen.

Ein weiterer Tracker [14] versucht mit Gaußmischverteilungen die Farbwerte und deren räumliche Verteilung einer Person zu lernen. Besonderer Fokus wurde hier auf plötzliche starke Veränderungen gelegt. Beispielsweise scheint eine Person in einer Kameransicht größer zu werden, wenn sie nahe an diese Kamera herangeht. Die Adaption wird in diesem Fall temporär abgeschaltet, damit die Person weitergetrackt werden kann, wenn sie wieder weiter von der Kamera entfernt ist.

Das W4-System [6] ist ein Mehrpersonentracker, der versucht Bewegungszonen zu folgen. Wie auch der Pfinder funktioniert dieses System mit einer einzigen Kamera.

Ein weiteres System [2] benutzt eine Variante eines Partikelfilters. Zur optischen Erfassung wird eine Stereokamera verwendet; die Bewertungsfunktion verwendet Bewegungszonen. Zusätzlich kommen mehrere Mikrofone zum Einsatz, mit dem Sprachgeräusche im Raum geortet werden. Die optische und akustische Domänen sind kalibriert. Deswegen können beide Modalitäten mittels eines stochastischen Verfahrens zusammengeführt werden.

Ein weiteres System [17] ist ein Mehrpersonentracker für den Außenbereich. Er verwendet eine angepasste Variante eines *Partikelfilters*. Ein Partikelfilter

ist ein Standardverfahren im Trackingbereich und wird im Grundlagenkapitel 3.5 erläutert. Die Arbeitsweise eines Partikelfilters erfordert eine Bewertungsfunktion. Dieses System versucht die Kontur von Kopf und Schulter in Bewegungszonen zu erkennen und verwendet diese Information als Bewertungsfunktion.

Es gibt noch weitere gängige Ansätze, die auf Kalmanfilter [4] oder dem Meanshiftverfahren [13] basieren. Sie werden hier nicht weiter besprochen.

2.3 Handhabung von Verdeckung

Bei einem Einpersonentracker spielt Verdeckung eine geringe Rolle, da eine Person per Definition nicht von einer anderen verdeckt werden kann. Verdeckung durch Objekte wird entweder vermieden oder ignoriert. Ein Mehrpersonentracker dagegen sollte sich um diesen Fall kümmern, da sonst die Gefahr einer Verwechslung von Personen hoch ist.

Das W4-System modelliert die Texturen der getrackten Personen. Wenn es zu einer kurzfristigen Verdeckung kommt können diese auseinander gehalten werden.

Bei partikelfilterbasierten Systemen ist es üblich, den Verdeckungsfall implizit zu handhaben [17, 2]. Der Zustandsraum von Partikelfiltern kodiert dabei sowohl Position als auch Anzahl der Personen. Der Verdeckungsfall ist dabei ein normaler Zustand und wird nicht als Sonderfall gehandhabt.

3 Grundlagen

Damit das Hauptkapitel 4 verständlich ist, werden grundlegende Begriffe und Verfahren in diesem Kapitel erklärt. Im Einzelnen sind das:

- Vorder-/Hintergrundsegmentierung
- Farbhistogramme und ihre Verwendung
- Triangulation, Berechnung von 3D-Koordinaten aus 2D-Koordinaten und vice versa
- Korrespondenzprobleme und ihre möglichen Lösungen
- Haarkaskaden zum Auffinden von Objekten
- Trackingansätze, wie z.B. Kalman- und Partikelfilter

3.1 Hintergrundsegmentierung

Häufig interessiert man sich für Menschen oder andere Objekte, die sich in einem Bild befinden. Diese "interessanten" Stellen im Bild werden als Vordergrund, der Rest des Bildes als Hintergrund bezeichnet. Da die Unterscheidung Vordergrund/Hintergund sehr stark von der Fragestellung abhängt, ist es schwierig, eine einfache algorithmische Lösung zu finden. Im Fall von Trackern vereinfacht sich das Problem, da sich "interessante" Objekte häufig bewegen, z.B. Autos oder Personen. Versucht man, mit verschiedenen Verfahren Hintergrund und Vordergrund aufgrund von Bewegung zu unterscheiden, dann deckt sich der berechnete Vordergrund häufig relativ gut mit dem gewünschten Vordergrund. Man darf allerdings nie vergessen, dass es sich genau genommen um einen *Bewegungsdetektor* und nicht einen *Vordergrunddetektor* handelt.

In der Literatur wird häufig zwischen *Background Subtraction* und *Background Segmentation* unterschieden. Background Subtraction bezeichnet dabei einfache Verfahren, die versuchen, einen eingelernten Hintergrund vom aktuellen Bild pixelweise zu subtrahieren. In der Lernphase wird ein leerer Raum eingelernt. Es wird für jeden Pixel ein durchschnittlicher Farbwert errechnet. Nach der Lernphase wird ein zu untersuchendes Bild von diesem Mittelwertsbild subtrahiert. Pixel, die zum Hintergrund gehören, haben einen ähnlichen Farbwert wie das Mittelwertsbild und die Differenz ist deshalb nahe Null. Personen, die sich im Bild befinden, unterscheiden sich häufig stark

vom Hintergrund, den sie gerade verdecken, entsprechend ist die Differenz deutlich verschieden von Null. Mithilfe eines geeigneten Schwellwerts wird beim Differenzbild die Unterscheidung Vordergrund/Hintergrund vorgenommen. Alle Differenzen größer als der Schwellwert sind Vordergrund, der Rest Hintergrund.

Der Vorteil dieser Verfahren ist ihre leichte und damit schnelle Berechenbarkeit. Der Nachteil ist, dass sie nur wenig auf verändernde Helligkeit, Rauschen oder unsteten Hintergrund reagieren können. Die Verfahren aus der Klasse *Background Segmentation* versuchen diese Nachteile auszugleichen.

Da die Beleuchtung normalerweise nie konstant bleibt, sollte auch nach der eigentlichen Lernphase weiterhin adaptiert werden, um ein einigermaßen korrektes Modell des Hintergrunds zu haben. Allerdings kann es dabei leicht passieren, dass Personen als Hintergrund mitgelernt werden. Dadurch werden Personen, die bislang als Vordergrund klassifiziert wurden, als Hintergrund erkannt. Deshalb muss die Lernrate hoch genug sein, um auf langsame Veränderungen zu reagieren, darf aber nicht zu hoch sein, damit Personen noch als Vordergrund erkannt werden.

Der Hintergrundsegmentierer nach Stauffer [16], im folgenden *Stauffersegmentierer* genannt, ist ein komplexer Algorithmus und wird deshalb hier nicht im Detail erklärt. Im Gegensatz zur Background Subtraction wird nicht nur ein Mittelwert pro Pixel gespeichert, sondern jeder Pixel wird mit Gaußglocken modelliert. Die Anzahl der Gaußglocken pro Pixel wird durch einen Parameter bestimmt, der konstant und gleich für alle Pixel ist. Jede Gaußglocke speichert den Mittelwert und die Abweichung des Farbwerts von dem Pixel, den sie modelliert. Bei der Berechnung des Vordergrunds eines Bildes wird für jeden Pixel bestimmt, ob es eine passende Gaußglocke gibt. Nur bei Pixeln, die noch keine passende Gaußglocke haben, handelt es sich um Vordergrund. In der Adaptionsphase werden die Gaußglocken abhängig von der Lernrate modifiziert.

Es gibt Bildsequenzen, bei denen bestimmte Stellen des Bildes zwischen mehreren Farbwerten alternieren. Beispiele hierfür sind ein sich drehender Ventilator oder Wellenbewegungen eines Sees. Durch die Verwendung von mehreren Gaußglocken pro Pixel kann sich je eine auf einen der Farbwerte anlernen. Dadurch werden Stellen des Bildes, die sich zwar ändern, aber eigentlich zum Hintergrund gehören, richtig klassifiziert. Bei der Background Subtraction wird der Mittelwert der alternierenden Farbwerte berechnet und diese Stellen werden dann häufig als Vordergrund klassifiziert.

Solange eine Person sich bewegt, wird sie sowohl vom Stauffersegmentierer

als auch von der Background Subtraction korrekt als Vordergrund klassifiziert. Sobald sie stehen bleibt, werden durch die Adaption die Pixel an dieser Stelle des Bildes nach und nach auf die neuen Farbwert der Person justiert. Wenn die Person lange genug – abhängig von der Lernrate – unbeweglich verharrt, wird diese Stelle des Bildes als Hintergrund klassifiziert. Wenn sich die Person wieder bewegt, wird der Hintergrund aufgedeckt. Für die Background Subtraction ist dieser Hintergrund neu, da die Mittelwerte den Farbwerten der Person entspricht. Entsprechend wird der aufgedeckte Hintergrund fälschlicherweise als Vordergrund klassifiziert. Der Stauffersegmentierer kann durch die Verwendung von mehreren Gaußglocken den alten Hintergrund länger speichern und klassifiziert deshalb den aufgedeckten Hintergrund richtig.

Ein Nachteil von diesem Algorithmus ist ein höherer Rechenaufwand, unter anderem aufgrund der Verwendung vieler Gaußglocken anstatt von Mittelwerten bei der Background Subtraction.

Sowohl bei der einfachen Background Subtraction, wie auch bei dem Stauffersegmentierer kann durch nachgeschaltete Verfahren, sog. Morphologische Operatoren, die Qualität der Vordergrunderkennung gesteigert werden. Pixel, die als Vordergrund erkannt wurden, werden im folgenden als *Treffer* bezeichnet, Hintergrund entsprechend als *Nichttreffer*. Alle Operationen werden auf dem Ergebnis eines Bewegungsdetektors, nicht auf dem Originalbild ausgeführt.

Durch *Erosion* werden bestimmte Treffer zu Nichttreffern konvertiert. Für jeden Pixel wird ein 3x3 Rechteck betrachtet, dessen Mittelpunkt dieser Pixel ist. Es wird das Minimum über diese neun Pixel gebildet. Da ein Nichttreffer als 0 und ein Treffer als 1 behandelt wird, genügt ein einziger Nichttreffer in der Maske, damit das Minimum 0 ist. Das Ergebnis wird an der Stelle des Pixels gespeichert. Dadurch verschwinden vereinzelte Treffer und zusammenhängende Trefferflächen werden an den Rändern kleiner. Innerhalb von Trefferflächen werden keine Treffer gelöscht.

Bei der *Dilatation* wird dieselbe Maske wie bei der Erosion verwendet. Allerdings wird diesmal statt dem Minimum das Maximum gebildet. Es genügt also ein einziger Treffer innerhalb der Maske, um das Ergebnis zu einem Treffer zu machen. Wenn die Dilatation nach einer Erosion ausgeführt wird, bleiben vereinzelte Treffer verschwunden, aber Trefferflächen werden am Rand wieder aufgefüllt.

Effektiv werden durch Erosion/Dilitation Rauschen und andere Ungenauigkeiten entfernt, große Trefferflächen bleiben aber erhalten.

(a) Stauffersegmentierer (b) Background Subtraction

Abbildung 1: Beispielbilder für Bewegungsdetektoren

Wenn sich eine Person nicht oder nur geringfügig bewegt, wird in vielen Fällen zumindest das Personeninnere zu Hintergrund adaptiert. Dadurch kann es passieren, dass nur der Rand der Person als Vordergrund erkannt wird. Ein ähnliches Phänomen tritt auf, wenn die Person stellenweise Farbwerte mit hoher Ähnlichkeit zum Hintergrund hat. Hier werden ebenfalls lediglich Teile der Person als Vordergrund erkannt.

Um solche Phänomene auszugleichen, wird ein Konturensuchalgorithmus verwendet. Dieser versucht, einen zusammenhängenden Pfad zu finden und füllt das Innere mit Treffern aus. Dadurch kommt es zu einer in sich geschlossenen Trefferfläche, die dann genauer den Vordergrund der Person darstellt.

Ein weiterer Vorteil des Konturensuchalgorithmus ist, dass nicht nur einzelne Treffer bekannt sind, die zufällig eine zusammenhängende Fläche ergeben, sondern die Fläche an sich. Dadurch kann ein Minimum an Treffern pro Fläche definiert werden, ab der es sich um eine Person oder zumindest einen Teil davon handeln könnte.

In Abbildung 1(a) ist das Ergebnis des Stauffersegmentierers in Grün eingezeichnet. Die Person an der Tafel hat sich in den letzten paar Bildern bewegt. Ein Teil von ihr wird vom Stauffersegmentierer als Bewegung erkannt. Man sieht relativ deutlich, dass auch Stellen als Vordergrund klassifiziert werden, die in Wirklichkeit Hintergrund sind. In diesem Fall befand sich die Person an der Tür etwas weiter rechts und wurde beim Initialisieren des Stauffersegmentierers als Hintergrund eingelernt. Da die Person nach links zur Tür läuft, wird bislang unbekannter Hintergrund sichtbar, und der Stauffersegmentierer spricht darauf an. Damit es seltener zu falschen Klassifizierungen kommt, sollte einem Bewegungsdetektor ermöglicht werden, einen leeren Raum für einige Zeit einzulernen. Dann werden Personen, die den Raum betreten, besser als Vordergrund erkannt. In der Abbildung 1(b) sieht man ein Beispiel,

in dem die Vordergrund/Hintergrundanalyse besser funktioniert. Bei beiden Bildern wurde Erosion/Dilatation vorgenommen, allerdings keine Konturenanalyse.

3.2 Farbhistogramme und ihre Verwendung

Ein Farbhistogramm stellt die Häufigkeitsverteilung von Farbwerten dar. Zur Erklärung von Farbhistogrammen wird der Einfachheit halber statt dem RGB-Farbraum lediglich der Graustufenraum betrachtet. Jeder Pixel kann hierbei die Werte 0 (schwarz) bis 255 (weiß) annehmen. Werte dazwischen repräsentieren verschieden Graustufen. Ein Farbhistogramm besteht aus einer bestimmten Anzahl von *Töpfen*. Sei in diesem Beispiel die Anzahl der Töpfe drei. Dann entsprechen die Graustufenwerte 0 bis 84 dem Topf 0, 85-169 dem Topf 1 und 170-255 Topf 2. Beim Einlernen eines Farbhistogramms wird jeder Pixel des Bildes, von dem das Farbhistogramm berechnet werden soll, betrachtet. Jeder Wert eines Pixels kann surjektiv auf eine Topfnummer abgebildet werden. Der entsprechende Topf wird jeweils um eins erhöht. Wenn in dem Bild viele weißliche Graustufen vorkommen, ist Topf 2 größer als die anderen beiden. Im RGB-Farbraum wird analog verfahren, allerdings gibt es jetzt Töpfe in drei Dimensionen, anstatt nur in einer. Wenn man in jeder Dimension drei Töpfe haben möchte, ist die Gesamtzahl 27.

Damit verschiedene Farbhistogramme direkt vergleichbar sind, muss das zugrunde liegende Bild gleich viele Pixel haben. Außerdem muss die Anzahl der Töpfe in jeder Dimension übereinstimmen. Da häufig verschieden große Bilder benutzt werden, werden Farbhistogramme normiert. Durch eine Normierung hat die Summe aller Töpfe eine bestimmten gewünschten Wert. Wenn mit demselben Wert normiert wird, sind Farbhistogramme mit identischer Topfanzahl vergleichbar.

Back Projection ist ein Verfahren, bei dem ein Bild mithilfe eines Farbhistogramms bewertet wird. Es wird für jeden Pixel des Bildes ausgerechnet, wie wahrscheinlich es ist, dass er zum gegebenen Farbhistogramm gehört. Angenommen es wurde ein Farbhistogramm HIST von dem Bild einer Person berechnet. Diese Person habe ein rotes T-Shirt an. Die Back Projection eines anderen Bildes werde mithilfe von HIST berechnet. Entsprechend haben rote Region eine hohe Wahrschlichkeit zum Farbhistogramm HIST zu gehören. Zur Visualisierung der Back Projection wird im Folgenden der Rotkanal von RGB-Bildern manipuliert. Eine Wahrschlichkeit von 0.0 wird auf den Rotwert 0, eine Wahrschlichkeit von 1.0 auf 255 abgebildet. Andere Wahrscheinlichkeiten werden entsprechend linear abgebildet.

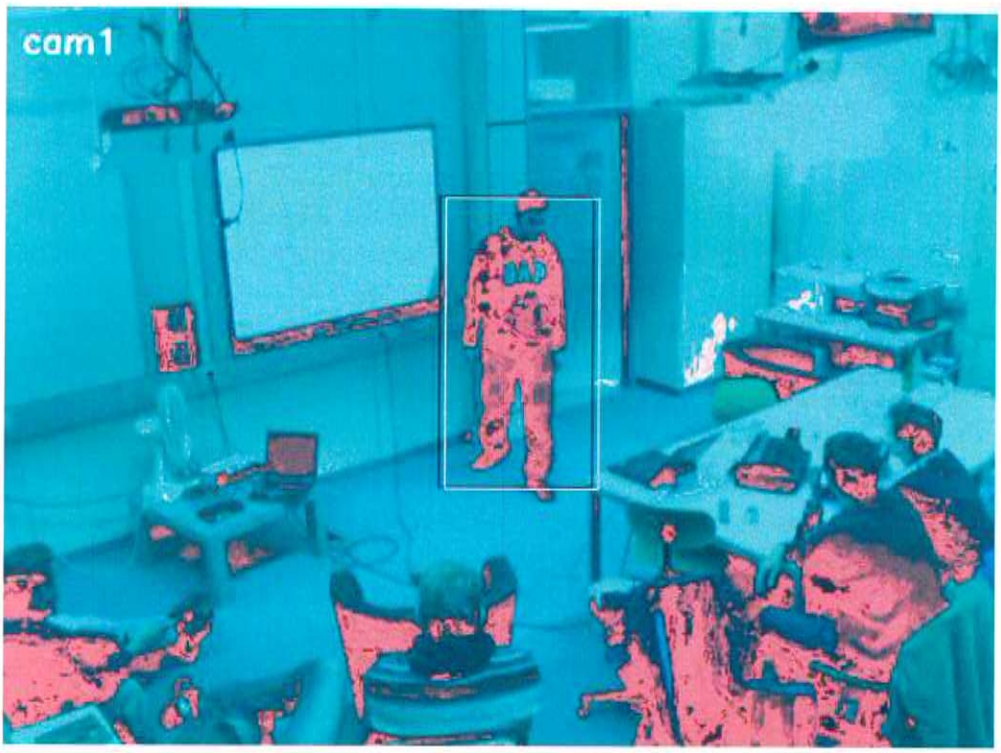


Abbildung 2: Beispielbild für eine Back Projection

In Abbildung 2 wurde das Farbhistogramm der Person eingelernt, die an der Tafel durch ein weißes Rechteck eingerahmt ist. Die Back Projection des Farbhistogramms über das gesamte Bild wurde im Rotkanal eingezeichnet, wie eben beschrieben. Die Person erscheint erwartungsgemäß in starken Rottönen. Die direkte Umgebung enthält fast kein Rot. Es gibt allerdings mehrere andere Regionen des Bildes, die sehr wohl Rottöne haben. Beispielsweise ist der schwarze Sessel rot gefärbt, da die eingelernte Person viele schwarze Farbtöne aufweist.

Im Idealfall wird das Farbhistogramm einer Person gebildet, die Back Projection berechnet und im Gesamtenbild gibt es genau ein Ballungszentrum an Rottönen, die Person. In der Praxis gibt es allerdings neben der Person weitere Regionen, die rot sind. Damit dennoch die Person möglichst eindeutig auffindbar bleibt, wurde ein Verfahren namens *Meanshift* [3] entwickelt.

Hier wird, ausgehend von einem initialen Suchfenster, versucht, das Suchfenster möglichst gut auf die rote Region anzupassen. In Abbildung 2 stellt das weiße Rechteck um die Person das Suchfenster dar. Es wird das Massezentrum innerhalb des Suchfensters berechnet, dann wird das Suchfenster auf das Zentrum verschoben. Falls eines der Abbruchkriterien zutrifft, wird abgebrochen, ansonsten eine weitere Iteration ausgeführt. Abbruchkriterien sind eine maximale Anzahl an Iterationen oder wenn sich das Suchfenster weniger als einen bestimmten Wert verschiebt.

Solange der Hintergrund sich farblich ausreichend von der Person unterscheidet, funktioniert dieses Verfahren. Bei einer zu hohen Farbähnlichkeit wird das Suchfenster häufig nicht auf der Person zentriert sein. Die Vorteile dieses Verfahrens sind die schnelle Berechenbarkeit im Vergleich zu beispielsweise Konturenerkennungsverfahren und die Robustheit gegen leichte Formänderungen oder partielle Verdeckung. Auch leichte Größenänderungen irritieren Meanshift nur wenig. Bei starken Größenänderungen sollte allerdings die Größe des Suchfensters entsprechend angepasst werden, da sonst das Suchfenster auf einen beliebigen Teil der Person zentriert wird.

Eine Erweiterung von Meanshift nennt sich *Camshift* [1]. Camshift ist für Videosequenzen entwickelt worden, die vorwiegend das Gesicht einer Person zeigen. Bei solchen Sequenzen kann Camshift neben der Bestimmung der Position die Lage des Kopfes berechnen. Ein Kopf kann gedreht, seitlich gekippt und gesenkt werden [15, Seite 35].

Neben Meanshift und Camshift kann mittels *Korrelation* [21] von Farbhistogrammen die neue Position eines Objekte berechnet werden. Hierbei wird von der vermuteten neuen Position des Objekts, die mittels Bewegungsvorhersage

ermittelt wurde, ein Farbhistogramm berechnet. Das neue Farbhistogramm und das alte Farbhistogramm werden miteinander korreliert. Das wird für mehrere der möglichen neuen Positionen wiederholt. Die Position mit dem besten Korrelationswert sei die berechnete neue Position des Objekts.

Dieses Verfahren hat den Vorteil, dass der Zwischenschritt Back Projection entfällt und dadurch Rechenzeit gespart wird. Je nachdem wieviele neue Positionen untersucht werden, kann diese Ersparnis durch das wiederholte Berechnen einer Korrelation deutlich aufgebraucht werden.

3.3 Triangulation und Korrespondenzproblem

Eine einzelne Kamera kann von einem Objekt lediglich eine Pixelposition innerhalb des Kamerabildes finden. Die tatsächliche Größe des Objekts kann nur schwer geschätzt werden, da ihre scheinbare Größe von der Entfernung von der Kamera abhängt.

Sobald mindestens zwei Kameras verwendet werden, kann mithilfe der *Triangulation* versucht werden, die räumliche Position eines Objekts zu bestimmen. Damit das möglich ist, müssen die extrinsischen Parameter der Kameras, kurz *Extrinsics*, bekannt sein. Die Extrinsics beschreiben die relative Position der Kameras und ihre Ausrichtung. Diese Daten werden durch eine *Kalibrierung* der Kameras gewonnen.

Ein weiteres Problem ist, dass Kamerabilder verzerrt sein können, beispielsweise durch Linsenfehler oder bauartbedingt, wie bei Bullaugenkameras. Eine perfekte Kamera liefert ein unverzerrtes Bild, allerdings gibt es solche Kameras in der Praxis nicht. Kamerabilder können allerdings korrigiert werden. Die Informationen, die dazu nötig sind werden als *Intrinsics* bezeichnet.

Es werde im Bild der ersten Kamera ein "interessanter" Punkt, beispielsweise der Mittelpunkt eines Objekts, gewählt. Im Bild von Kamera zwei wird der entsprechende Punkt ausgewählt. Durch die bekannte Position eines Punktes innerhalb eines Kamerabildes werden die möglichen Positionen des 3D-Punktes eingeschränkt. Durch die beiden zweidimensionalen Pixelkoordinaten der beiden Kamerabilder sind die Koordinaten des 3D-Punktes, bis auf einen Fehler, eindeutig bestimmt. Bei entsprechender Kalibrierung ist die Einheit der 3D-Koordinaten Millimeter. Bei der Verwendung von mehr als zwei Kameras funktioniert das Verfahren analog; man gewinnt dadurch aber keine weitere Information, lediglich der Fehler kann reduziert werden.

Umgekehrt können aus einem 3D-Punkt die 2D-Koordinaten dieses Punkts in einer gewünschten Kamera berechnet werden.

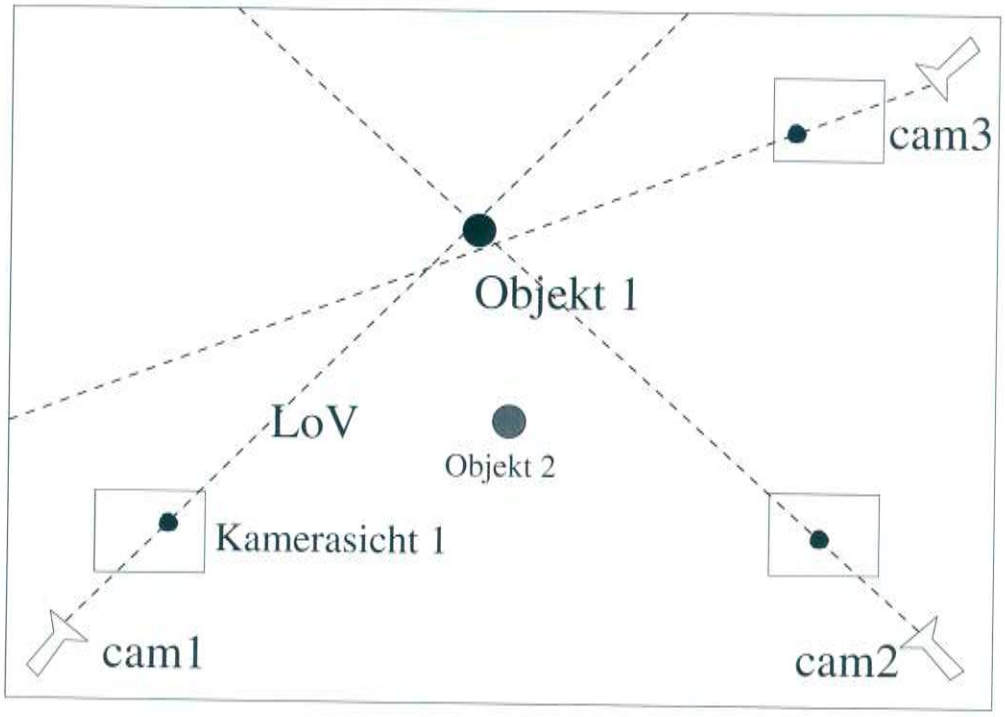


Abbildung 3: Drei Kamerasichten auf ein Objekt. Die LoVs sind fehlerbehaftet

In Abbildung 3 ist schematisch ein Raum von oben dargestellt. In drei Ecken des Raumes sind die Kameras CAM1 bis CAM3 angebracht, in der Mitte des Raumes befindet sich Objekt 1, dessen 3D-Position bestimmt werden soll. Die Kamera CAM1 habe Objekt 1 in ihrer zweidimensionalen Sicht auf den Raum gefunden. Dadurch sind die drei Freiheitsgrade, die der 3D-Punkt hat, auf einen Freiheitsgrad reduziert worden.

Der verbleibende Freiheitsgrad ist eine Gerade im Raum, auch *Line of View*, kurz *LoV* genannt. Kamera CAM2 habe den entsprechenden Punkt in seiner Sicht des Raums gefunden. Dadurch wird eine zweite LoV bestimmt. Diese beiden LoVs schneiden sich im Idealfall in einem Punkt im Raum. Dieser Punkt ist der gesuchte 3D-Punkt.

In der Praxis schneiden sich die LoVs nie exakt in einem Punkt, da die Genauigkeit in der physikalischen Welt begrenzt ist. Die Kalibrierung der Kameras hat eine begrenzte Genauigkeit und die Kamerabilder sind ebenfalls begrenzt genau, da die Auflösung und damit die Pixelgröße festgelegt ist. Dadurch schneiden sich die LoVs immer mit einem bestimmten Fehler. In der Abbildung wird das durch ein Dreieck, das von den LoVs gebildet wird, dargestellt.

Zwei aus verschiedenen Kameras ausgewählte Punkte, die demselben 3D-Punkt entsprechen, haben bei der Triangulation einen geringen Fehler, *Triangulationsfehler* genannt. Ein geringer Triangulationsfehler ist eine notwendige, aber nicht hinreichende Bedingung dafür, dass zwei Punkte demselben 3D-Punkt entsprechen. Es ist also möglich zwei Punkte auszuwählen, die eindeutig nicht demselben 3D-Punkt entsprechen, aber dennoch einen geringen Triangulationsfehler haben. Umgekehrt bedeutet ein hoher Triangulationsfehler, dass die beiden zugrunde liegenden Punkte nicht demselben 3D-Punkt entsprechen können.

Das *Korrespondenzproblem* ist vielschichtig und besteht u.a. darin, einen Pixel innerhalb eines Kamerabildes einem Objekt bzw. Punkt zuzuordnen.

Insbesondere bei mehreren Kameras mit mehreren Personen in einem Raum sollte für jede Kamera bekannt sein, wo diese Personen sich in den jeweiligen Kamerabildern befinden. Die Problematik wird durch Abbildung 3 veranschaulicht. Die Kamera CAM1 sieht Objekt 1 und rechts daneben Objekt 2. Die Kamera CAM3 dagegen sieht das Objekt 1 und links – nicht rechts – daneben Objekt 2. Die Reihenfolge von Objekten kann also je nach Perspektive variieren.

Sobald mehrere Personen in einem Bild sichtbar sind, kann es passieren, dass diese Personen verwechselt werden. Insbesondere wenn Personen einander

sehr nahe sind oder eine Person die andere verdeckt ist die Verwechslungsgefahr hoch.

Das W4-System [6] löst das Problem mittels einer Textur. Über einen längeren Zeitraum wird eine Textur je Person eingelernt, um bei einem Trackerfehler, wie einer Verdeckung, diese korrekt unterscheiden zu können.

Im Allgemeinen ist das Korrespondenzproblem schwierig zu lösen, deswegen gibt es eine Reihe von Strategien die Zuordnung vorzunehmen. Im Kapitel 3.3 über Triangulation wurde bereits erwähnt, dass ein geringer Triangulationsfehler eine notwendige Bedingung für die korrekte Lösung des Korrespondenzproblems ist. Obwohl die Bedingung nicht hinreichend ist, kann sie dennoch als Heuristik verwendet werden. Bei einer geringen Anzahl an Objekten ist die Irrtumswahrscheinlichkeit noch gering, steigt aber zunehmender Anzahl an.

Durch eine Beobachtung über einen längeren Zeitraum hinweg wird häufig eine falsche Zuordnung erkannt. Es ist unwahrscheinlich, dass verschiedene Personen langfristig Pfade wählen, die dauerhaft einen geringen Triangulationsfehler zur Folge haben.

Eine weitere Möglichkeit das Korrespondenzproblem zu lösen ist die Verwendung von Farbe oder Form. Beispielsweise wird normalerweise ein rotes Objekt in einer Kamera in den anderen Kameras ebenfalls rot erscheinen. Ähnlich kann es sich mit geometrischen Formen verhalten. Ein Objekt in einer Kamera kann eine vorhersagbare Form, beispielsweise durch Rotation eines Würfels, in den anderen Kameras haben.

Diese Methoden können je nach Aufgabenstellung das Problem vereinfachen oder nur wenig dazu beitragen. Beispielsweise ist Farbe und Form bei verschieden farbigen Klötzchen eine Hilfe, bei schwarzen Ameisen dagegen nicht.

3.4 Haarkaskaden

Eine *Haarkaskade* ist ein Objekterkenner für Bilder, wobei eine bestimmte Haarkaskade auf ein bestimmtes Objekt trainiert wurde. Ein Objekterkenner kann eine Person nicht identifizieren. In der ursprünglichen Veröffentlichung [18] wurde ein Gesichtserkenner vorgestellt, allerdings können auch andere Objekte trainiert werden.

Ein Bild kann potentiell an jeder Stelle ein Gesicht in vielen verschiedenen Größen darstellen. Deshalb gibt es ein *Suchfenster*, das sich über das Bild bewegt. Das Suchfenster wird bei jedem Durchlauf größer, bis alle Möglichkeiten

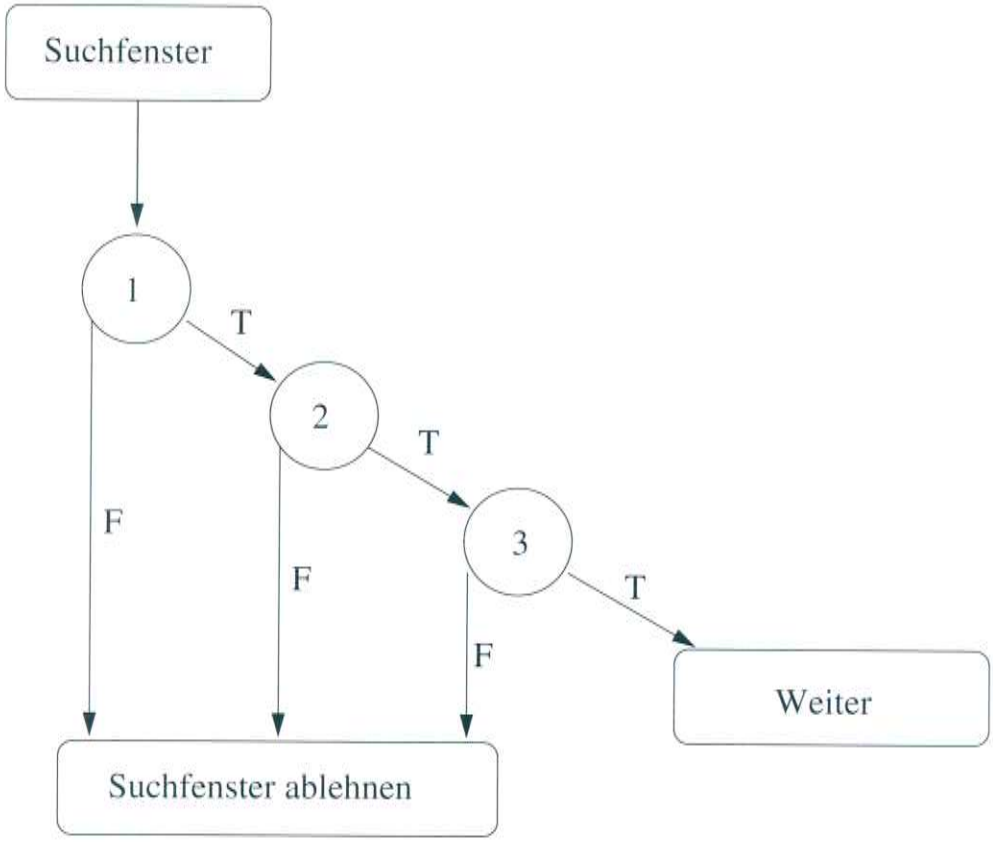


Abbildung 4: Kaskade: Sobald ein Knoten ablehnt, wird das Suchfenster verworfen (F), ansonsten wird fortgefahren (T) [18].

abgehandelt wurden. Selbst bei Bildern mit nur geringer Auflösung kommt es so zu mehreren Tausend Suchfenstern.

Es gibt sogenannte *Knoten*, die entscheiden, ob ein Suchfenster das trainierte Objekt enthält oder nicht. Diese Knoten sind in einem degenerierten Entscheidungsbaum, *Kaskade* genannt, zusammengefasst. Eine solche Kaskade ist in Abbildung 4 dargestellt. Der erste Knoten fällt seine Entscheidung. Bei Ablehnung wird das Suchfenster verworfen. Bei Annahme fällt der nächste Knoten eine Entscheidung. Erst wenn alle Knoten annehmen, enthält das Suchfenster das trainierte Objekt. Sobald ein einziger Knoten ablehnt, wird das Suchfenster sofort verworfen.

Damit möglichst wenig Rechenzeit pro Suchfenster verwendet wird, sind die Knoten am Anfang des Entscheidungsbaum einfacher gebaut und lehnen Suchfenster, die mit hoher Wahrscheinlichkeit das trainierte Objekt nicht

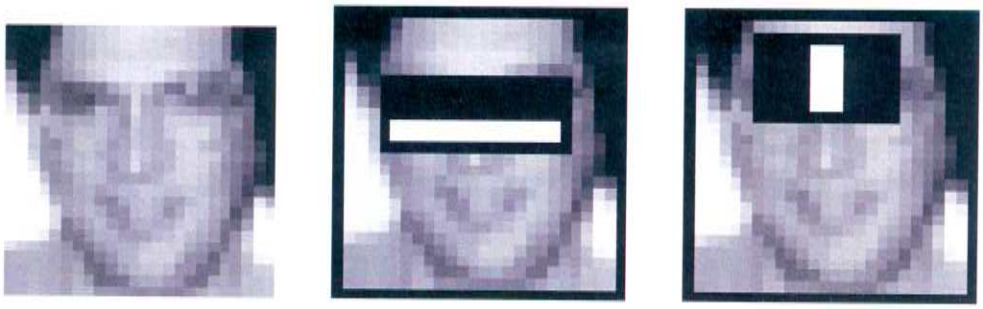


Abbildung 5: Beispiel für Features [18]

enthalten, ab. Allerdings machen diese Knoten häufig den Fehler, ein Suchfenster anzunehmen, welches das Objekt nicht enthält. Solche Suchfenster werden dann in tieferen, komplizierteren Knoten abgelehnt.

Jeder Knoten prüft ein oder mehrere einfache *Features*. In Abbildung 5 links ist ein Gesicht zu sehen. Im mittleren und rechten Gesicht ist jeweils ein Feature eingezeichnet. Die schwarz bzw. weiß markierten Pixel eines Features werden jeweils addiert. Die Differenz gibt den Helligkeitsunterschied von Augen zu Backen beim mittleren Bild und von Augen zu Nasenrücken beim rechten Bild an. Das funktioniert, weil Augen häufig dunkler als die Backen sind.

Mögliche Features werden in Abbildung 6 gezeigt. Die ursprüngliche Arbeit [18] kennt keine um 45 Grad gedrehten Features, sondern nur 1a), 1b), 2b) und ein hier nicht aufgeführtes. Diese zusätzlichen Features wurden in einer Erweiterung [9] eingeführt.

In der ursprünglichen Arbeit [18] hat beispielsweise eine Gesichtskaskade 38 Knoten mit insgesamt über 6000 Features. Im Schnitt müssen aber pro Suchfenster nur etwa 10 Features berechnet werden. Die Berechnung eines solchen Features kann trotz unterschiedlicher Größe der Suchfenster immer in konstanter Zeit durchgeführt werden. Dieses Verfahren nennt sich *Integral Image* [18, 9] und wird hier nicht genauer erklärt.

Aufgrund der Kaskade und der schnellen Berechnung von durchschnittlich wenigen Features müssen pro Suchfenster relativ wenige Rechenoperationen durchgeführt werden. Damit gehört dieser Objekterkenner zu den schnellsten, die es gibt.

Bevor das Trainieren des Objekterkenners erklärt wird, wird kurz das Verfahren *AdaBoost* [5] vorgestellt, da dieses benötigt wird. Mittels AdaBoost kann die Fehlerrate eines Lernalgorithmus stark reduziert werden. Als Ein-

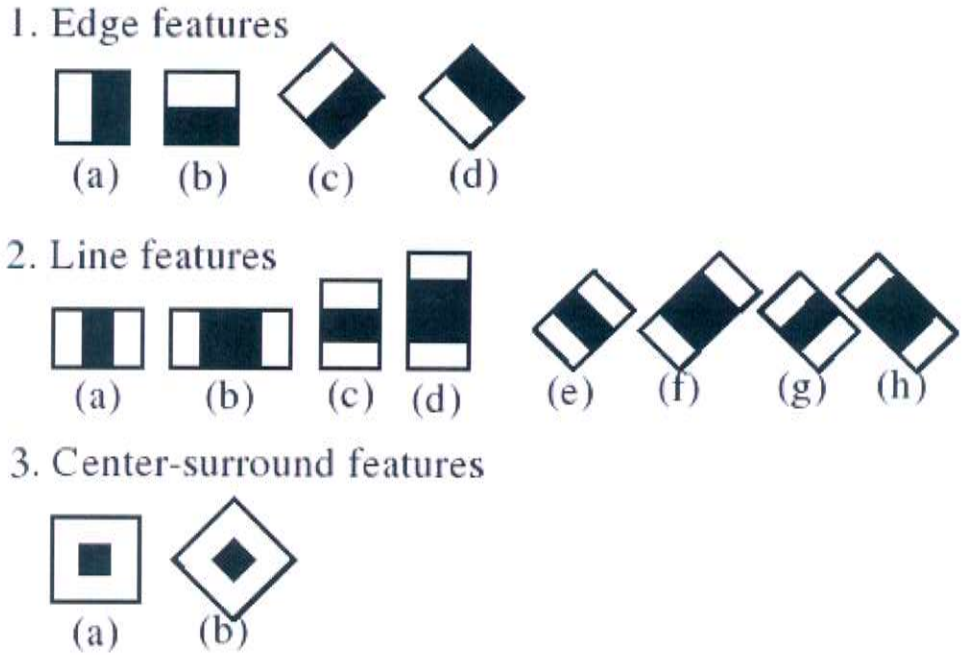


Abbildung 6: Alle verwendeten Features [9].

gabe benötigt man mehrere Beispiele mit richtiger Klassifizierung und den Lernalgorithmus. Dann lässt AdaBoost den Lernalgorithmus auf den Beispielen operieren. Auf die Beispiele, die falsch klassifiziert wurden, wird in der nächsten Runde verstärkt Wert gelegt, so dass sich der Lernalgorithmus auf die schwierigen Probleme konzentrieren muss. Nachdem alle Beispiele abgearbeitet wurden, werden alle so gewonnen Klassifizierer gewichtet zusammengefasst. Klassifizierer, die häufig richtig liegen, fließen entsprechend stärker ins Ergebnis ein. Auf diese Weise reduziert sich die Gesamtfehlerrate recht schnell, obwohl die Klassifizierer nur leicht besser als zufällig sein müssen. Unter welchen Voraussetzungen AdaBoost korrekt funktioniert, wird hier nicht dargelegt [5].

Es muss eine geeignete Auswahl aus den möglichen Features getroffen und der entsprechende Klassifizierer trainiert werden. Mittels AdaBoost wird in jeder Runde aus den über 180000[1] möglichen Features das mit der geringsten Fehlerrate ausgewählt. Entsprechend ist die Rundenanzahl von AdaBoost gleich der Featureanzahl. Wenn mehr Features verwendet werden, sinkt zwar die Fehlerrate, aber dafür steigt die Rechenzeit entsprechend an. Das ist auch

[1] Das gilt für ein 24x24 Bild mit den ursprünglichen Features [18].

der Grund, warum eine Kaskade verwendet wird. Daraus resultiert aber das Problem, dass man in mehrere Richtungen optimieren kann:

- Wie viele Knoten soll die Kaskade haben.
- Wie viele Features soll jeder Knoten haben.
- Wie hoch soll der Schwellwert pro Knoten sein.

Da der Suchraum relativ groß ist, ist es schwierig, das Optimum für diese Parameter zu finden. Der Suchraum wird deshalb mittels einer Heuristik eingeschränkt. Jeder Knoten wird solange trainiert, bis die gewünschte Rate an Detektion und false positives erreicht wird[2].

3.5 Trackingansätze

Zu bestimmten Zeitpunkten wird die Position eines Objekts gemessen. Solche Messungen sind immer fehlerbehaftet und sollten daher ausgebessert werden. Zudem wird für viele Zwecke eine Vorhersage in die Zukunft benötigt.

Es gibt viele verschiedene Verfahren, die das leisten, aber die gängigsten sind einfache Filter, die nur die Geschwindigkeit über einen bestimmten Zeitraum benutzen, um eine Vorhersage zu machen. Zudem gibt es etwas kompliziertere, die auch noch die Beschleunigung berücksichtigen und den vielleicht bekanntesten, den *Kalman-Filter* [20, 4]. Der Kalman-Filter erhält in jedem Schritt eine Messung und berechnet daraus eine neue Position. Er merkt sich intern einen Zustand, führt aber keinerlei Berechnungen über alle Messungen durch, sondern nur über die neueste. Er arbeitet also iterativ und deshalb ist der Aufwand pro Schritt gleichbleibend. Die erste Anwendung des Kalman-Filters diente der Flugbahnberechnung einer Apollo-Rakete.

Der Kalman-Filter wurde für lineare Systeme, wie beispielsweise die Apollo-Rakete, entwickelt und wird inzwischen in sehr vielen unterschiedlichen System eingesetzt [20]. Allerdings verhalten sich Menschen selten linear, da sie plötzlich stehen bleiben und die Richtung abrupt ändern können. Der Kalman-Filter benötigt einige Zeit, bis er sich an die neue Richtung angepasst hat, aber genau dafür wurde er entwickelt: Er soll fehlerhafte Messungen in einem linearen System ausgleichen können. Dennoch wird er für Personentracker verwendet, obwohl das nicht-lineare Systeme sind.

[2] Dafür gibt es neben dem Trainingssatz ein Validierungssatz.

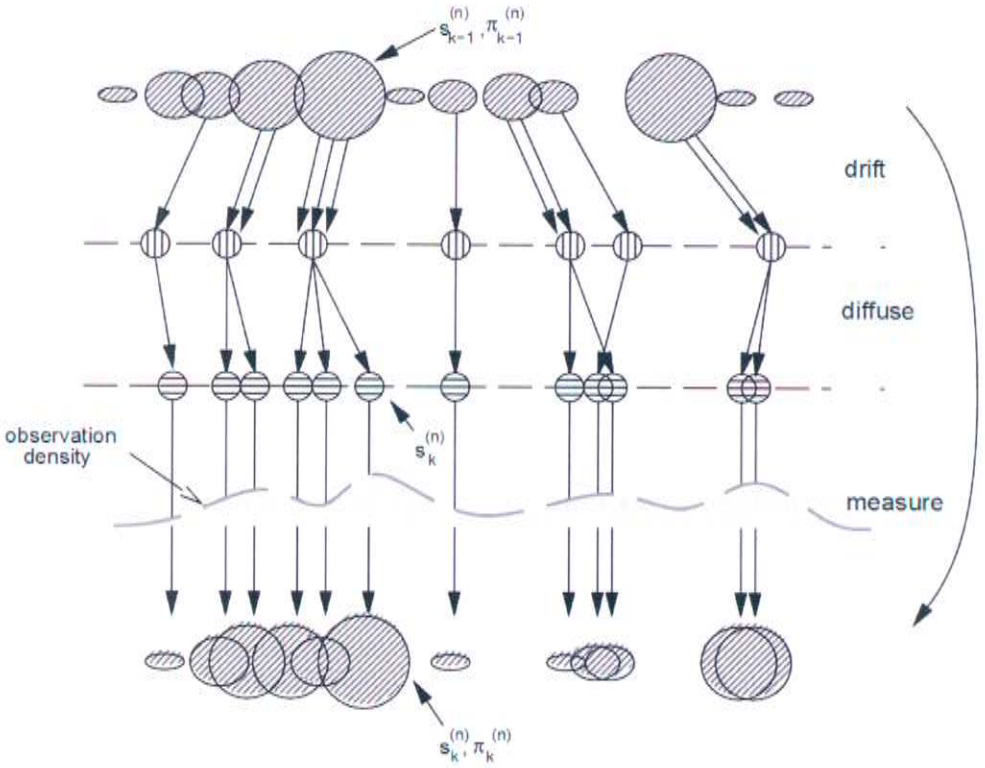


Abbildung 7: Ablauf einer Iteration eines Partikelfilters [7]

Neben Kalmanfiltern sind *Partikelfilter* eines der Standardverfahren im Trackingbereich. Sie werden dazu benutzt, Personen zu tracken. Die Aufenthaltswahrscheinlichkeit der Person wird durch eine Menge an diskreten Einzelhypothesen, den Partikeln, approximiert. Dazu wird in jedem Schritt eine Iteration des Partikelfilters durchgeführt, um die Hypothesen aktuell zu halten. Diese Erklärung lehnt sich stark an die Arbeit in [22, Seite 16] an. Eine detaillierte Erläuterung kann dort und in der Arbeit über *Condensation* [7], einer Variante eines Partikelfilters, genauer nachgelesen werden. Die Abbildung 7 zeigt eine Iteration:

Erzeugung Eine neue, noch ungewichtete Samplemenge wird aus der alten Partikelmenge erzeugt.

Bewegung Diese Partikel werden dann, getrieben durch das Bewegungsmodell, deterministisch bewegt und anschließend mit Rauschen überlagert.

Messung Die Gewichte werden durch das Beobachtungsmodell bestimmt und anschließend normiert.

Um schließlich den geschätzten Aufenthaltsort des Sprechers in 2D- oder 3D-Koordinaten – je nach Anwendung – zu bestimmen, wird der gewichtete Mittelwert der Einzelhypothesen gebildet.

Die Verwendung einer Variante von Partikelfiltern ist sehr gängig [2, 17, 22].

4 Entwickeltes Trackingsystem

Das Tracken von Personen oder anderen Objekten ist im allgemeinen Fall schwierig. Deswegen werden Tracker-Systeme mit bestimmten Randbedingungen entwickelt und für diese Einsatzszenarien optimiert. In dieser Arbeit soll ein Mehrpersonentracker für den Innenbereich, einem "intelligenten Raum", entstehen. Es soll keine Spezialhardware, wie Stereokameras, verwendet werden, sondern nur gewöhnliche Kameras. Es gibt vier Kameras, die in den Ecken des Raums statisch angebracht sind. Diese Kameras sind kalibriert (siehe 3.3). Das System soll automatisiert Personen erkennen und diese verfolgen. Eine genauere Kenntnis, beispielsweise Gestik, über die Personen ist nicht nötig. Echtzeitfähigkeit ist für diesen Prototypen kein direktes Ziel. Für die Entwicklung dieses Trackers standen mehrere Aufnahmen zur Verfügung, die als *Entwicklungsdaten* bezeichnet werden.

Solange eine Person nur in einer Kamera erkannt wurde, also mindestens ein *2D-Track* in einer einzigen Kamera existiert, wird diese Person auch *2D-Person* genannt. Sobald es mehrere 2D-Tracks in verschiedenen Kameras gibt, kann der Tracker diese zusammenfassen. Mehrere 2D-Tracks können also einen *3D-Track* bilden. Eine Person mit einem 3D-Track wird auch *3D-Person* genannt.

In jeder Kamerasicht wird unabhängig voneinander mittels eines Bewegungsdetektors der Vordergrund ermittelt. Zusätzlich analysieren bis zu drei verschiedene Haarkaskaden die Kamerabilder und finden rechteckige Zonen, in denen sich bestimmte Objekte (Ganzkörper, Oberkörper, Gesicht) befinden. Die Kombination aus Vordergrund und den Rechtecken wird verwendet, um ein Farbhistogramm zu initialisieren und ein initiales Suchfenster für das Trackingverfahren Meanshift zu erhalten. Damit Meanshift möglichst zuverlässig funktioniert, wird aus einer Auswahl an Bereinigungsverfahren zur Laufzeit das derzeit beste korrigierte Farbhistogramm ausgewählt. Zusätzlich wird mittels Adaption das Farbhistgramm kontinuierlich angepasst. Da mehrere verschiedene Haarkaskadentypen verwendet werden, müssen verschieden 2D-Tracks innerhalb einer Kamera fusioniert werden, wenn sie diesselbe Person tracken. Im Fehlerfall, wenn also die Tracks widersprüchliche Positionen liefern, werden die betroffenen Tracks gelöscht.

Die einzelnen 2D-Hypothesen der Kamerasichten werden mit einer 3D-Logik fusioniert. Hierbei wird als Kriterium, ob 2D-Tracks diesselbe Person repräsentieren primär ein geringer Triangulationsfehler verwendet. Ein weiteres Kriterium ist eine sinnvolle Begrenzung der Höhe des Körpermittelpunkts. Falls eines der Kriterien einen Fehlerfall impliziert, wird die 3D-Hypothese

verworfen, allerdings die dazugehörigen 2D-Tracks nicht gelöscht. Eine 3D-Hypothese kann verwendet werden, um Vorhersagen über die Position von 2D-Tracks in allen Kamerasichten zu machen. Dadurch wird eine geringe Trefferrate der Haarkaskaden ausgeglichen. Es werden also neue 2D-Tracks in inaktiven (bzgl. dieser 3D-Hypothese) Kameras eingelernt und bestehende 2D-Tracks in aktiven Kameras adaptiert. Falls mehrere 3D-Hypothesen bekannt sind, kann eine Verdeckung von Personen erkannt werden. Der verdeckte 2D-Tracker wird gelöscht und sobald keine Verdeckung mehr besteht durch den Hinweisgenerator wieder eingelernt. Mehrere der 3D-Hypothesen werden ggf. zu einer Ausgabehypothese zusammengefasst.

4.1 2D-Tracking mittels Farbhistogrammen

Die Farbverteilung von Personen ist ein charakteristisches Merkmal und wird deswegen häufig zur Verfolgung verwendet. Die Farbverteilung einer Person wird mittels eines Farbhistogramms gespeichert. Es gibt mehrere verschiedene Algorithmen, die mittels eines Farbhistogramms bzw. den Farbwerten einer Person diese verfolgen können (siehe 3.2):

Korrelation Das Farbhistogramm der Person wird mit dem Farbhistogramm der vermuteten neuen Position korreliert. Durch Maximierung der Korrelation über einer Menge an sinnvollen neuen Positionen wird eine der Positionen ausgewählt.

Meanshift Ausgehend von einem Suchfenster wird ein Massezentrum der Back Projection berechnet und das Suchfenster iterativ auf dieses Zentrum verschoben [3].

Camshift Eine Erweiterung von Meanshift speziell für das Tracken eines Kopfes und Bestimmung dessen relativer Orientierung [1].

Farbbasierter Partikelfilter Ein Partikelfilter, dessen Bewertungsfunktion Farbe verwendet [12].

Das Korrelationsverfahren hat das Problem, dass es nicht die Form der Person vorhersagen kann. Deshalb verwenden die untersuchten Farbhistogramme nicht nur Farbwerte der Person, sondern auch des Hintergrunds und unterscheiden sich deshalb teilweise stark vom gespeicherten Farbhistogramm. Dadurch wird das Verfahren unzuverlässig.

Das Meanshiftverfahren ist ein vergleichsweise schnell berechenbares Verfahren und kann Personen, die sich farblich scharf vom Hintergrund trennen zuverlässig verfolgen. Das Camshiftverfahren ist eine Erweiterung von Meanshift, ist allerdings für einen anderen Einsatzzweck optimiert worden. Die zusätzlichen Informationen, die Camshift ermittelt, werden für diesen Tracker nicht benötigt.

Der Partikelfilter ist für das Tracken von einzelnen Personen ausgelegt und kann deshalb unmodifiziert nicht verwendet werden. Eine genauere Evaluierung war mangels Zeit nicht möglich.

Von den betrachteten Verfahren eignet sich Meanshift am besten, da dieser schnell und robust Personen tracken kann. Meanshift ist ein zweidimensionales Verfahren, es kann also pro Kamerabild und Person verwendet werden. Es ist nicht in der Lage alle Kameras gleichzeitig zu berücksichtigen, wie beispielsweise ein Partikelfilter [22].

4.1.1 Initialisierung von Farbhistogrammen

Im letzten Kapitel wurde erwähnt, dass Farbhistogramme von Personen benötigt werden. Diese Farbhistogramme sollten möglichst die tatsächliche Farbverteilung der Person wiederspiegeln und insbesondere wenig Hintergrundfar-

ben enthalten. Farbhistogramme, bei denen dies gegeben ist, werden hier als *sauber* bezeichnet. Je sauberer ein Farbhistogramm ist, desto besser sind die Chancen, dass Meanshift die Person stabil tracken kann. Dazu ist es nötig, dass der Algorithmus, der die Person initial findet, unterscheiden kann, welche Pixel zu der Person gehören und welche nicht. Farbhistogramme, die nicht sauber sind, werden hier auch als *unrein* bezeichnet. Hier werden Bewegungsdetektion mit Haarkaskaden kombiniert, um möglichst saubere Farbhistogramme zu erhalten.

Es wurden zwei verschiedene Verfahren zum Auffinden von Bewegung bzw. Vordergrund untersucht:

Background Subtraction Ein gelernter Mittelwert pro Pixel wird pixelweise vom Bewertungsbild subtrahiert. Ein statischer Schwellwert unterscheidet Vordergrund von Hintergrund.

Stauffersegmentierer Komplexes System zur Bewegungsdetektion. Siehe 3.1 und [16].

Zusammen mit den nachgeschalteten Bereinigungsverfahren (siehe 3.1) kann eine Person vom Hintergrund getrennt werden. Hierbei hat das einfachere Verfahren Background Subtraction die besseren Ergebnisse geliefert. Personen wurden häufig vollständig als Vordergund erkannt. Der Stauffersegmentierer kann zwar besser Schattenwurf als Hintergrund klassifizieren, allerdings erkennt er häufig die Person nicht vollständig. Der Stauffersegmentierer hat mehrere Parameter, die komplex korreliert sind. Eine Veränderung an einem Parameter kann die Bedeutung der anderen Parameter verändern. Das ist einer der Gründe, warum mit dem Stauffersegmentierer keine zufriedenstellenden Resultate erzielt wurden.

Ein Bewegungsdetektor liefert also mit einer gewissen Wahrscheinlichkeit Pixel, die zu einer Person gehören. Allerdings ist hierbei unbekannt, zu welcher Person sie gehören. In Kapitel 3.1 wurde bereits der Unterschied zwischen einem Bewegungs- und einem Vordergrunddetektor erklärt. Zu Vordergrund zählen in diesem Fall per Definition ausschließlich Personen. Die Ausgabe eines Bewegungsdetektors deckt sich zu einem hohen Grad mit dem definierten Vordergrund, allerdings nicht vollständig. Angenommen es gäbe einen Detektor, der immer genau den Vordergrund liefert, dann wäre ein Großteil dieser Arbeit überflüssig. Es gibt durchaus Tracker [6], bei denen der Ansatz, Bewegungszonen zu folgen funktioniert. Damit Vordergrund zuverlässiger einer Person zugeordnet werden kann, wird ein Objekterkenner benötigt. Hier werden als Objekterkenner Haarkaskaden verwendet.

Ein Objekterkenner, wie beispielsweise eine Haarkaskade (siehe 3.4), kann Personen detektieren. Sie können allerdings Personen nicht identifizieren. Gängige Haarkaskadentypen sind hierbei:

Ganzkörper Dieser Typ erkennt vollständige Personen.

Oberkörper Brust und Kopf werden von diesem Typ erkannt.

Gesicht Dieser erkennt ein Gesicht und den restlichen Kopf.

Eine Haarkaskade liefert bei einer Detektion ein Rechteck, innerhalb dessen sich das jeweilige Objekt befindet. Sie kann nicht unterscheiden, welche Pixel zum Objekt gehören und welche nicht. Der Bewegungsdetektor liefert vorwiegend Pixel, die zu Personen gehören, kann aber nicht unterscheiden zu welcher Person ein Pixel gehört.

Deshalb werden beide Verfahren kombiniert. Die Haarkaskade liefert eine Person oder einen Teil von ihr. Gleichzeitig liefert ein Bewegungsdetektor innerhalb des Rechtecks Vordergrundpixel. Die Wahrscheinlichkeit, dass diese Pixel vorwiegend zur Person gehören ist hoch. Das Personenfarbhistogramm wird über die Farbwerte berechnet, die als Vordergrund klassifizert sind und sich innerhalb des Rechtecks befinden. Es wird ein sauberes Farbhistogramm einer Person eingelernt werden, falls die Vordergrundklassifizierung richtig arbeitet. Das Suchfenster für Meanshift wird auf die maximale Ausdehnung des Vordergrunds innerhalb des Rechtecks reduziert.

Eine Heuristik, die das Erstellen oder Löschen von Tracks nur in bestimmten Bereichen vorsieht wurde **nicht** verwendet (siehe 2.1). Eine solche Heuristik ist wenig sinnvoll, da Personen verloren werden können oder die Haarkaskade es möglicherweise nicht schafft, im Türbereich Personentracks zu erkennen, sondern erst später.

4.1.2 Fehlerbereinigung von Farbhistogrammen

Wie im letzten Kapitel 4.1.1 über die Initialisierung von Farbhistogrammen erwähnt wurde, kann es passieren, dass Pixel, die nicht zur Person gehören, mitgelernt werden. Der Bewegungsdetektor liefert mit hoher Wahrscheinlichkeit Vordergrund, der zu einer Person gehört. Allerdings wird eine sich öffnende Tür oder ein verschobener Stuhl ebenfalls als Bewegung erkannt. Schattenwurf wird je nach Stärke ebenfalls fälschlicherweise als Vordergrund[3] klassifiziert.

[3]Zur Erinnerung: Vordergrund soll per Definition ausschließlich Personen sein.

Deshalb werden bei der Initialisierung häufig Farbwerte mitgelernt, die in Wirklichkeit zum Hintergrund oder zu einer anderen Person gehören.

In Abbildung 8(a) ist eine Person abgebildet, die soeben von einer Oberkörperkaskade erkannt und für die entsprechend ein Farbhistogramm eingelernt wurde. Das Resultat des Bewegungsdetektors ist grün eingezeichnet. Die Person wird fast vollständig als Vordergrund erkannt, allerdings ist die Tür im Hintergrund und etwas Schatten ebenfalls dabei. Mithilfe des eingelernten Farbhistogramms wird eine Back Projection (siehe 3.2) berechnet und die Wahrscheinlichkeit[4] im Rotkanal des Bildes eingezeichnet, wie in Abbildung 8(b) dargestellt. Das Rechteck, das durch einen Haarkaskadentreffer erzeugt wurde ist in der Abbildung mit "3 Upper" bezeichnet. Hierbei ist "3" die interne Trackerindex der Person und "Upper" die Kurzbezeichnung für Oberkörper. Deutlich ist zu sehen, dass viele Bereiche des Bildes einen hohen Rotanteil haben. Das Meanshiftverfahren wird also Schwierigkeiten haben, die Person stabil zu verfolgen, da sich die Person in der Back Projection nicht deutlich vom Hintergrund abhebt. Deswegen sollte das Farbhistogramm von Fehlern bereinigt werden.

Das eingelernte Personenfarbhistogramm HIST1 gibt die Wahrscheinlichkeit an, mit der ein Farbwert zur Person gehört [10, Seite 8]:

$$P(x|Person) = \text{HIST1}_+(x)$$

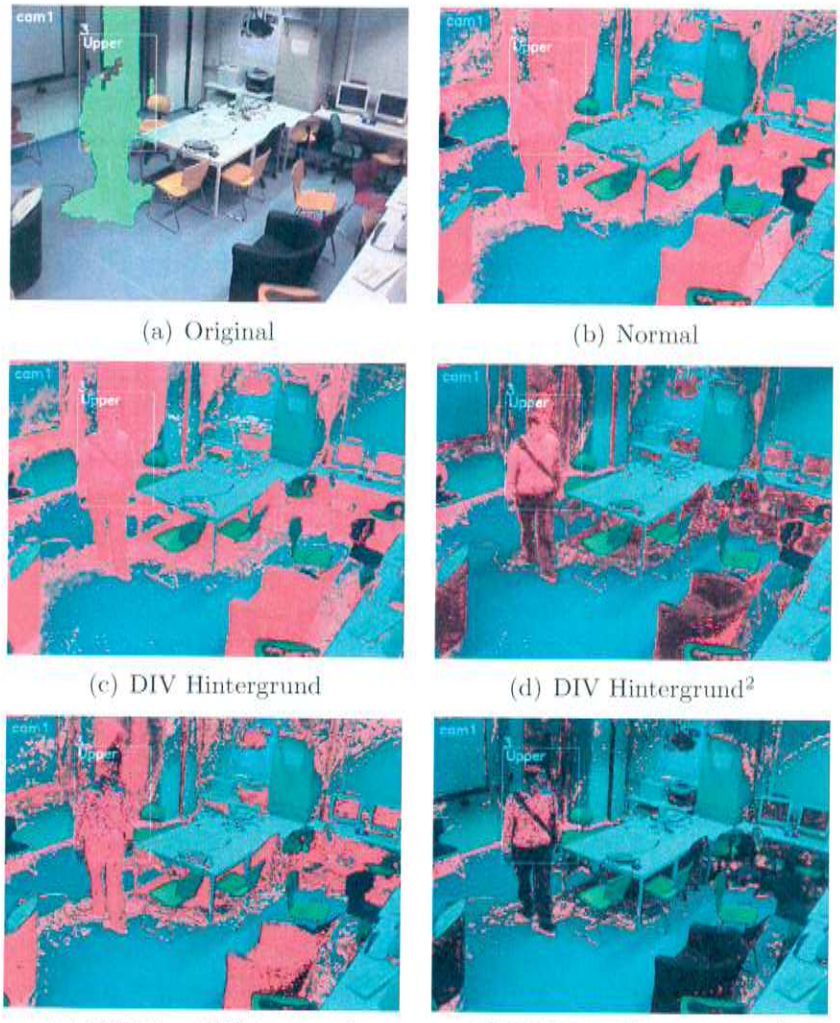
Hierbei sei x ein Farbwert und das + normiere das Farbhistogramm auf 1. Ein Farbhistogramm, das über alle Nichtpersonenpixel eingelernt wurde sei das Hintergrundfarbhistogramm HIST2. Mithilfe der Regel von Bayes ergibt sich:

$$\frac{\text{HIST1}_+(x)}{\text{HIST2}_+(x)} > S$$

Wobei S ein Schwellwert ist. Falls der Schwellwert überschritten wird, wird die Farbe x als zur Person gehörig klassifiziert. Da hier Meanshift eingesetzt wird, ist eine solche Klassifizierung unnötig; die Wahrscheinlichkeiten werden direkt verwendet. Durch die Farbhistogrammdivision haben Farbwerte, die sowohl in der Person, als auch im Hintergrund vorkommen eine geringere Wahrscheinlichkeit, als ähnlich häufige Farbwerte, die nur in der Person vorkommen.

Das Farbhistogramm HIST1 beschreibe u.a. die Farbe grün. Es gibt also einen oder mehrere Töpfe, die die Farbe grün repräsentieren. Diese Töpfe haben einen relativ großen Wert im Vergleich zu den restlichen Töpfen des

[4]Hohe Wahrscheinlichkeit entspricht hohem Rotanteil. Details werden in Kapitel 3.2 erklärt.



(a) Original (b) Normal

(c) DIV Hintergrund (d) DIV Hintergrund^2

(e) DIV Rand*Hintergrund (f) DIV Rand*Hintergrund^2

Abbildung 8: Back Projektionen eines normal eingelernten Farbhistogramms einer Person und verschiedener Bereinigungsmethoden

Farbhistogramms. Es gebe ein zweites Farbhistogramms HIST2, das ebenfalls die Farbe grün darstellt. Bei der Division von HIST1 durch HIST2 wird jeder Topf aus HIST1 durch den korrespondierenden Topf aus HIST2 geteilt. Die Sonderfälle, wie beispielsweise Division durch Null, werden durch geeignete Extremwerte, also Minimal- und Maximalwert des Topfs aufgefüllt. Wenn ein Topf in HIST2 einen großen Wert hat, ist das Ergebnis der Division kleiner, als wenn der Wert klein ist. Ein Topf in HIST1 wird folglich stark abgeschwächt, wenn er stark in HIST2 vertreten sind. HIST1 wird durch die Division um die Farben aus HIST2 *bereinigt*. Das resultierende Farbhistogramm wird wieder normiert.

In Abbildung 8(c) ist eine Back Projection zu sehen, bei der das Farbhistogramm der Person durch das Farbhistogramm des gesamten Bildes geteilt wurde. Das Farbhistogramm des gesamten Bildes beschreibt vorwiegend die Farben des Hintergrunds, da Personen nur einen kleinen Teil des Bilds ausmachen. Es wird deshalb als *Hintergrundfarbhistogramm* bezeichnet. In diesem Beispiel hat sich im Vergleich zum Originalhistogramm nur wenig verbessert.

Wenn ein Farbhistogramm potenziert wird, verstärken sich die stark vertretenen Töpfe, die anderen werden abgeschwächt. In Abbildung 8(d) wurde durch das Hintergrundfarbhistogramm HIST_+^2 geteilt[5]. Diesmal ist ein deutlich geringerer Rotanteil zu sehen. Prinzipiell kann jeder Exponent, nicht nur 2 verwendet werden. Damit aber aus wenigen sinnvollen Alternativen ausgewählt werden kann, wird für das Hintergrundfarbhistogramm nur 1 und 2 verwendet.

Wenn eine Person eingelernt werden soll, ist das Rechteck, das von den Objektdetektoren geliefert wird, meist etwas zu groß und wird daher auf die maximale Ausdehnung der enthaltenen grünen Pixel reduziert. Dieses häufig leicht kleinere Rechteck und das Originalrechteck haben links und rechts einen Rand, der mit hoher Wahrscheinlichkeit nicht zur Person gehört. Der Rand oben und unten wird ignoriert. Nun wird ein Farbhistogramm dieses Randes berechnet, im weiteren als *Randfarbhistogramm* bezeichnet. Dieses Farbhistogramm beschreibt also Pixel, die sich nahe bei der Person befinden, aber nicht zu ihr gehören. Das Personenfarbhistogramm werde durch das Randfarbhistogramm geteilt. Dadurch kann es von leichten Verunreinigungen durch den Hintergrund bereinigt werden. Die Abbildungen 8(c) und 8(e) bzw. 8(d) und 8(f) zeigen die Auswirkungen eines Randfarbhistogramms.

Pauschal lässt sich keine Aussage treffen, welche dieser Bereinigungsmöglichkeiten die beste ist, da das u.a. von der Stärke der Verunreinigung ab-

[5]Das Normieren des Farbhistogramms wird nach dem Potentieren ausgeführt.

zuhängen scheint. Das Kriterium zur Beurteilung der Qualität einer Methode liefert Meanshift. Das Personenfarbhistogramm wird eingelernt, wenn der Objekterkenner behauptet, dass sich genau dort eine Person bzw. ein Teil von ihr befindet. Wenn das Meanshiftverfahren an dieser Stelle durchgeführt wird, sollte die neue Position nur geringfügig von der alten abweichen. Folglich wird für alle Bereinigungsmethoden probehalber das Meanshiftverfahren angewendet, um herauszufinden welche Methode die kleinste Abweichung aufweist.

Tabelle 1: Abweichung der Position nach Meanshift

Methode	Abstand	Iterationen[6]
Original	33.24	25
Hintergrund	25.18	25
Hintergrund^2	7.62	3
Rand*Hintergrund	26.00	11
Rand*Hintergrund^2	22.36	4

Für das Beispiel in Abbildung 8 wurden die Werte, die in Tabelle 1 aufgelistet sind, ermittelt. Hier wurde also die Methode "Hintergrund^2" gewählt, da sich hier das Zentrum nur 7.62 Pixel bewegt hat. Die anderen Methoden haben eine deutlich höhere Verschiebung. Ein weiteres Indiz, für die Zuverlässigkeit der Methode, ist die niedrige Iterationsanzahl[6] von Meanshift. Das Auswahlverfahren berücksichtigt allerdings nur die Abweichung, da das einfacher ist, als zwei verschiedene Werte zu betrachten. Ab einem bestimmten Schwellwert für den Abstand wird entschieden, dass kein stabiler Track eingelernt werden kann und der Track wird verworfen.

4.1.3 Adaption von Farbhistogrammen

Ein Farbhistogramm kann aus verschiedenen Gründen verbesserungsbedürftig sein. Die Beleuchtungseigenschaften eines Raumes können sich verändern, das Farbhistogramm kann verunreinigt sein oder die Person sich gedreht haben, so dass jetzt andere Farben sichtbar sind. Um solchen langsamen Änderungen gerecht zu werden, müssen die Personenfarbhistogramme adaptiert werden.

[6]Maximal erlaubte Anzahl ist 25.

Beim Initialisieren wird eine neue Person an einer neuen Position gelernt. Bei der Adaption soll das Farbhistogramm einer bestehenden Person verändert werden. Dazu ist es notwendig, einen Haarkaskadentreffer einer bestehenden Person zuzuordnen.

Damit solch eine Zuordnung möglich ist, wird zunächst die Überlappung zweier Rechtecke untersucht. Hierbei gibt es zwei Richtungen, wie eine Überlappung stattfinden kann. Rechteck R_1 überlappt R_2 oder umgekehrt. Eine Überlappung berechnet sich wie folgt:

$$v(R_x, R_y) = \frac{R_x \cap R_y}{R_x}$$

Hierbei bezeichne R die Fläche des Rechtecks.

Sei R_2 deutlich kleiner als R_1, dann hat die Überlappung $v(R_2, R_1)$ beispielsweise den Wert 90% und die andere Richtung $v(R_1, R_2) = 5\%$. Es kann also durchaus relevant sein, in welcher Richtung eine Überlappung betrachtet wird.

Damit ein Haarkaskadentreffer, also ein Rechteck, einer bestehenden Person zugeordnet werden kann, muss die Überlappung mit dem Personenrechteck in beiden Richtungen hoch sein. Falls sich die Überlappungsgrade beider Richtungen stark unterscheiden, gibt es einen starken Größenunterschied zwischen den beiden Rechtecken und die Zuordnung ist nicht sinnvoll.

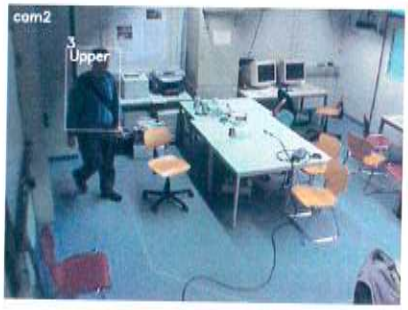
Falls innerhalb des Haarkaskadentreffers ausreichend viel Vordergrund erkannt wurde, wird das Farbhistogramm adaptiert. Dazu wird ein neues Farbhistogramm H_{neu} eingelernt. Jeder Topf wird gewichtet mit dem entsprechenden Topf des bisherigen Farbhistogramms H_{t-1} addiert:

$$H_t = (1 - \alpha)H_{t-1} + \alpha H_{neu} \text{ mit } 0.0 \leq \alpha \leq 1.0$$

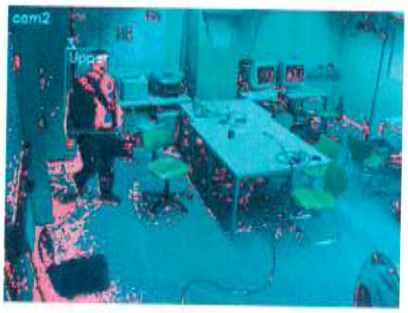
Das resultierende Farbhistogramms H_t sei das adaptierte. Es wurde der Adaptionsfaktor α verwendet. Ein Wert nahe 1.0 führt zu einer starken, nahe 0.0 zu einer geringen Adaption.

Rapide Veränderungen, beispielsweise eine Person mit einem roten Pullover zieht sich eine schwarze Jack an, können auf diese Weise häufig nicht abgefangen werden. Der Tracker verliert diese Person, da das Farbhistogramm nun vollkommen falsch ist, und lernt die Person möglicherweise später neu ein.

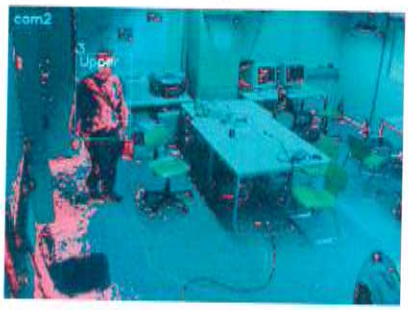
In Abbildung 9 wird ein Beispiel gezeigt, in dem die Adaption des Farbhistogramms eine Verbesserung bringt. In 9(a) ist die Person dargestellt, in 9(b)



(a) Original

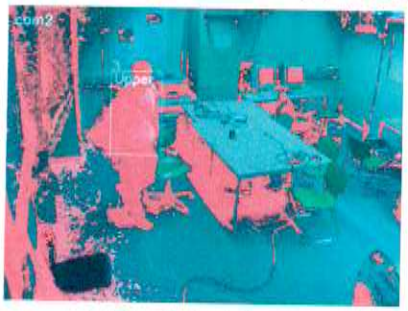


(b) Back Projection



(c) Nach 1. Adaption

(d) Nach 2. Adaption



(e) Nach 3. Adaption

Abbildung 9: Inkrementelle Adaption von Farbhistogrammen

die Back Projection des dazugehörigen Farbhistogramms, in 9(c) bis 9(e) das Ergebnis nach ein- bis dreimal Anpassen.

Eine Anpassung des Farbhistogramms muss aber nicht immer eine Verbesserung darstellen, sondern kann leicht auch das Farbhistogramm verschlechtern, also die Verunreinigung erhöhen. Deshalb ist eine Entscheidung nötig, wann eine Adaption erfolgreich ist. In Kapitel 4.1.2 wurde das Meanshiftverfahren verwendet, um verschiedene Fehlerbereinigungsstrategien zu bewerten. Diesmal ist das Meanshiftverfahren nicht anwendbar. Ein nochmaliges Durchführen von Meanshift an der selben Stelle mit demselben Farbhistogramm führt häufig zu einer Abweichung von 0.0. Ursache hierfür ist das Abbruchkriterium von Meanshift. Die Iterationen werden solange durchgeführt, bis sich das Suchfenster weniger als eine vorgegebene Entfernung bewegt. Weitere Iterationsdurchläufe ändern daran nichts. Nachdem das Farbhistogramm probehalber adaptiert wurde und Meanshift wieder gestartet wird, kommt es meist zu einer geringen Abweichung von wenigen Pixeln. Deswegen wird ein anderes Maß benötigt.

Damit Meanshift optimal tracken kann, wird angestrebt, dass die Person rot (in der Darstellung der Back Projection) und der Rest des Bildes oder zumindest die nähere Umgebung der Person möglichst nicht Rot ist. Es soll also eine möglichst scharfe Abgrenzung der Person zur unmittelbaren Umgebung statt finden. Aus dieser Überlegung ergibt sich direkt ein einfaches Bewertungsverfahren.

Es wird der Durchschnitt der Back Projection innerhalb des Trackrechtecks berechnet, dann der Durchschnitt eines Rahmens um dieses Rechteck. Das Verhältnis beider Durchschnitte sei das Maß.

Hier die genaue Definition. Sei

$$P = \{(x, y) \mid x_1 \leq x \leq x_2 \wedge y_1 \leq y \leq y_2\}$$

ein Rechteck, das die Person trackt. Sei weiterhin

$$R = \{(x, y) \mid x_1 - \frac{x_2 - x_1}{2} \leq x \leq x_2 + \frac{x_2 - x_1}{2} \wedge y_1 - \frac{y_2 - y_1}{2} \leq y \leq y_2 + \frac{y_2 - y_1}{2}\}$$

ein Rechteck, welches etwas größer als das Personenrechteck ist. Dann wird das Maß durch:

$$m = \frac{\frac{1}{\#(R \setminus P)} \sum_{(x,y) \in R \wedge (x,y) \notin P} back(x, y)}{\frac{1}{\#P} \sum_{(x,y) \in P} back(x, y)}$$

bestimmt, wobei # die Anzahl der Pixel innerhalb von P bzw R bezeichne. Die Funktion $back(x, y)$ liefere den Wert der Back Projection (siehe 3.2)

Abbildung 10: Beispiel für false positive Treffer der Gesichtskaskade

an der Stelle (x, y) für das Farbhistogramm der Person. Je kleiner m ist, desto schärfer unterscheidet sich P von seiner Umgebung. Wenn nach einer Adaption m sinkt, wird diese Adaption übernommen, ansonsten ignoriert.

Die 3. Adaption in Abbildung 9(e) wird übrigens abgelehnt, da m nicht sinkt, sondern steigt.

4.1.4 Falsche Treffer

Bei den Haarkaskaden gibt es zwei Kenngröße für fehlerhaftes Verhalten. Die eine nennt sich *false negative*, hier wird ein Suchfenster abgelehnt, welches das Objekt enthält. Bei einer hohen false-negative-Rate findet die Kaskade immer weniger Objekte und wird dadurch unbrauchbar. Die andere Sorte Fehler wird *false positive* genannt. Hierbei wird ein Suchfenster angenommen, welches das Objekt nicht enthält.

In Abbildung 10 ist ein Beispiel für dieses Verhalten zu sehen. Von den vier Treffern ist nur einer richtig, die anderen nicht.

Die Haarkaskade liefere, wie eben im Bild gesehen, ein false positive Objekt. Der Bewegungsdetektor liefere dort gleichzeitig Vordergrund. Unter diesen Umständen wird ein *falscher Track* erzeugt. Eine weitere Möglichkeit für falsche Tracks sind *instabile Tracks*. Hierbei verfolgt ein Track anfangs korrekt eine Person, aber ab einem bestimmten Zeitpunkt nicht mehr. Beispielsweise wird statt der schwarzen Person ein schwarzer Sessel getrackt.

Damit die Anzahl an false positives der Haarkaskade reduziert wird, wird eine *Größenheuristik* verwendet. Die minimale und maximale Größe eines Rechtecks, das von einer Haarkaskade erzeugt wird, ist definiert. Zusätzlich muss die Haarkaskade nicht alle Suchfenster (siehe 3.4) analysieren und spart dadurch Rechenzeit. Die Größenheuristik definiert außerdem einen Bereich von sinnvollen Seitenverhältnissen der Rechtecke. Der konkrete Wert dieser Konstanten wurde durch die Entwicklungsdaten empirisch ermittelt.

Der Bewegungsdetektor erkennt nicht immer die Person vollständig als Vordergrund. Außerdem kann ein false positive der Haarkaskaden falschen Vordergrund enthalten. Im ersten Fall wird die Person "einseitig" eingelernt. Da nur ein Teil der Person Vordergrund ist, ist die Gefahr groß, dass nur bestimmte Farbwerte der Person gelernt werden und nicht das ganze Spektrum. Im zweiten Fall wird ein falscher Track erzeugt, der nur wenig Vordergrund hat. Damit solche Szenarien nicht zu einem Einlernen eines Tracks führen, gibt es eine *Vordergrundheuristik*. Diese Heuristik fordert, dass das Verhältnis von Vordergrundpixel zu Gesamtpixel innerhalb des Rechtecks einen bestimmten Wert überschreitet. Dadurch wird die Anzahl an falschen oder instabilen Tracks reduziert. Der Nachteil ist, dass manchmal auch richtige Tracks nicht eingelernt werden. Deshalb muss der Schwellwert für diese Heuristik geeignet gewählt werden.

Wenn Haarkaskaden verwendet werden, die eine hohe false-positive-Rate haben, werden zwar die zu trackenden Personen schnell gefunden, aber es entstehen potentiell auch viele falsche Tracks. Es kann sogar passieren, dass die Erkennungsleistung sinkt, wenn beispielsweise falsche und richtige Tracks zusammengefasst werden.

Dem gegenüber haben Haarkaskaden mit einer niedrigen false-positive-Rate den Vorteil, dass sie vorwiegend nur Objekte finden, die es wirklich gibt. Der Preis dafür ist normalerweise, dass die false-negative-Rate hoch ist, also nur selten ein Objekt gefunden wird. Das führt dazu, dass Personen nicht oder erst später gefunden werden.

Die beiden beschriebenen Heuristiken versuchen die Auswirkungen von false positive Treffer zu reduzieren. Es gibt noch eine weitere Heuristik, die aber

erst im Kapitel 4.3 über die 3D-Logik erklärt werden kann.

Damit falsche Tracks wieder gelöscht werden, wird ein einfaches *Kreditsystem* verwendet. Dieses Kreditsystem wird ausschließlich für 2D-Personen verwendet. Eine Person bekommt bei der Initialisierung ein bestimmtes Guthaben. Nach jedem Bild wird der Person eine Einheit abgezogen. Sobald 0 erreicht wird, wird die Person mit allen Tracks gelöscht. Falls die Person von einer Haarkaskade bestätigt wird (siehe 4.1.3), erhöht sich ihr Guthaben um einen bestimmten Wert, bis ein definiertes Maximum erreicht ist.

Auf diese Weise sollten falsche Tracks wieder verschwinden, während echte Personen ab und zu bestätigt werden. Bei 3D-Personen wird dieses Kreditsystem nicht verwendet, da diese Personen als "glaubwürdiger" eingestuft werden. Eine 3D-Person hat in verschiedenen Kameras stabile 2D-Tracks. Ein einzelner 2D-Track kann instabil oder falsch sein, aber die Chancen sind geringer, dass mehrere 2D-Tracks gleichzeitig falsch sind.

Das Kreditsystem wird zwar verwendet, um 2D-Tracks zu löschen, aber es wird durchaus 3D-Wissen benutzt. Die 3D-Logik wird allerdings erst in Kapitel 4.3 erklärt. Dadurch kann ein Teil des Kreditsystems erst nach Einführung von diesen Konzepten beschrieben werden. Entsprechend gibt es weiterführende Informationen in Kapitel 4.3.3.

4.2 Erzeugung von 2D-Hypothesen

Eine einfache Möglichkeit, die Anzahl an Haarkaskadentreffern zu erhöhen, ist, mehrere verschiedene Haarkaskaden gleichzeitig laufen zu lassen. In Kapitel 4.1.1 wurde bereits erwähnt, dass es die Haarkaskadentypen Ganzkörper, Oberkörper und Gesicht gibt. Jede dieser drei Haarkaskaden kann beliebig an- oder abgeschaltet werden. Die meisten Heuristiken in diesem Kapitel beschäftigen sich mit dem Fall, dass alle drei Haarkaskaden aktiv sind.

Im Idealfall gibt es pro Person und Kamera drei Haarkaskadentreffer. Innerhalb einer Kamera müssen folglich diese verschiedenen 2D-Tracks zu einer 2D-Person zusammengefügt werden.

Da die Positionen der Haarkaskadentypen nicht direkt verglichen werden können, müssen sie normalisiert werden. Als Referenz wird der Mittelpunkt, der Bauchnabel des Ganzkörpertyps, benutzt. Bei den anderen Typen wird entsprechend die Position des Bauchnabels extrapoliert. Abbildung 11 zeigt schematisch die Extrapolation. Beim Oberkörper entspricht der untere Rand der Höhe des Ganzkörpermittelpunkts. Beim Gesicht wird ein vielfaches der Rechteckhöhe verwendet, um den gesuchten Punkt zu extrapolieren.

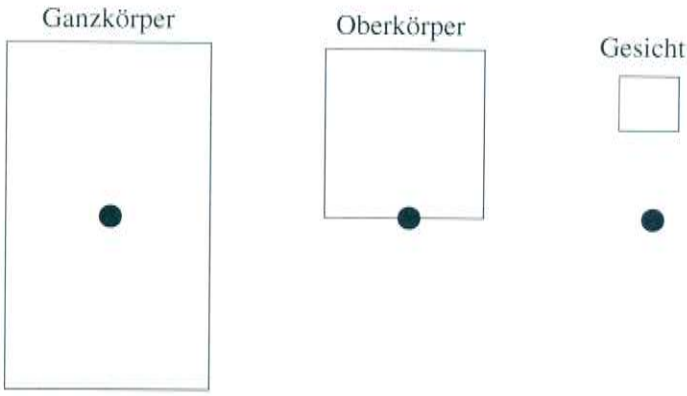


Abbildung 11: Normalisierung der Typen: Mittelpunkt des Ganzkörpers wird bei anderen Typen extrapoliert.

Jetzt sind die Typen vergleichbar und das Zusammenfügen der 2D-Tracks wird möglich. Falls der euklidische Abstand der normalisierten Positionen kleiner als ein bestimmter Schwellwert ist, wird davon ausgegangen, dass die beiden Typen zur ein und derselben Person gehören. Die beiden bislang seperaten 2D-Tracks werden zu einer (2D-)Person zusammengefügt. Wenn im weiteren Verlauf die 2D-Tracks eine ähnliche Bahn haben, gehören sie wahrscheinlich wirklich zu derselben Person.

Angenommen, eine Person wird innerhalb einer Kamera von zwei 2D-Tracks verfolgt, beispielsweise einem Gesichtstrack und einem Ganzkörpertrack. Solange beide Tracks Positionen liefern, deren Abstand gering genug ist, funktioniert das Tracken der Person. Aber es kann vorkommen, dass beispielsweise der Gesichtstracker instabil ist, die weiße Wand trackt, und sich dabei immer weiter von der Person entfernt. Die beiden Tracks laufen also auseinander.

Eine Möglichkeit auf dieses Fehlerszenario zu reagieren ist das Auftrennen der Person. Die Person besteht aus zwei oder drei 2D-Tracks. Beim Auftrennen werden ein oder zwei weitere Personen erstellt und je ein 2D-Track in diese neuen Personen transferiert. Nach dem Auftrennen existieren also bis zu drei Personen mit jeweils einem 2D-Track.

Eine weitere Möglichkeit, auf diesen Fehlerfall zu reagieren, ist Ausmitteln der normalisierten Koordinaten. Dabei werden die normalisierten Koordinaten addiert und durch die Anzahl geteilt. Die Tracks werden an diese neue Position verschoben. Bei einer leichten Trackerungenauigkeit oder einem kurzzeitigem Versagen eines Trackers ist diese Methode anwendbar. Allerdings in dem eben beschriebenen Szenario wird der fehlerhafte Gesichtstracker den

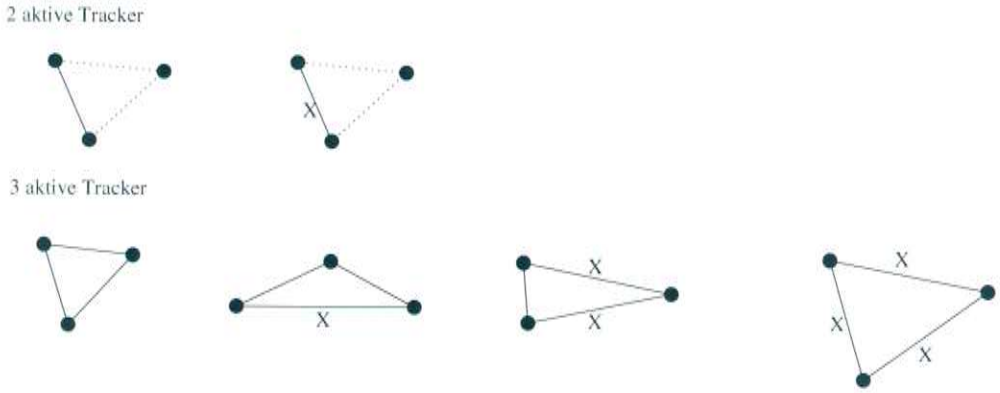


Abbildung 12: Fehlermöglichkeiten bei zwei oder drei aktiven 2D-Trackern

funktionierenden Ganzkörpertracker durch das Ausmitteln zu stark verschieben. Dadurch kann der Ganzkörpertracker ebenfalls instabil werden, da das initiale Suchfenster für Meanshift falsch ist.

Aus diesem Grund ist ein Ausmitteln zwar kurzfristig und für kleine Fehler sinnvoll, aber, wie die Entwicklungsdaten belegen, im großen und ganzen schädlich.

Der Fall "widersprüchliche 2D-Tracks" muss folglich anders behandelt werden. Ein Problem ist, dass der Tracker nicht ohne Weiteres entscheiden kann, welche der 2D-Tracks falsch tracken. Neben dem oben beschriebenen Fehlerszenario kann auch ein anderen Fall vorliegen. Der Gesichtstracker wurde richtig initialisiert und trackt stabil, aber er gehört in Wirklichkeit zu einer anderen Person. Das Zusammenfügen mit dem Ganzkörpertrackers war also ein Irrtum.

Da der Tracker nicht wissen kann, ob ein Track falsch ist oder eine andere Person trackt, hat er keine Möglichkeit angepasst zu reagieren. Ein falscher Track ist störend und sollte nach Möglichkeit entfernt werden. Ein richtiger Track, der entfernt wird, kann durch einen Haarkaskadentreffer wieder eingelernt werden. Deshalb ist es sinnvoller Tracks im Fehlerfall zu löschen, anstatt sie lediglich aufzutrennen.

Sobald es drei Tracks gibt, kann durch Mehrheitsentscheidung herausgefunden werden, welcher fehlerhaft ist. Solch ein Track wird entsprechend gelöscht. Wenn der fehlerhafte Track nicht bestimmbar ist, müssen möglicherweise alle Tracks gelöscht werden.

In Abbildung 12 sind alle Fehlermöglichkeiten zusammengefasst. Die Punkte stellen die normalisierte Position des jeweiligen 2D-Tracks dar, die Linien den

Abstand der Punkte. Linien, die mit einem X gekennzeichnet sind, haben einen zu großen Abstand. Wenn nur zwei Tracker aktiv sind, ist die Lage eindeutig: Entweder haben die zwei aktiven Tracker eine kleinen Abstand (linkes Bild, oben) oder einen zu großen (rechtes Bild, oben). Der inaktive Tracker wurde mit gestrichelter Linie eingezeichnet.

Wenn drei Tracker aktiv sind, gibt es vier Möglichkeiten. Im ersten Fall (erstes Bild, unten) sind alle Abstände in Ordnung, kein Tracker muss gelöscht werden. Im zweiten Fall ist ein Abstand (zweites Bild, unten, Linie mit X markiert) zu groß, der linke und rechte Tracker werden gelöscht. Im dritten Fall wird der rechte Tracker und im vierten werden alle gelöscht.

Die normalisierte Position von einem oder mehreren 2D-Tracks einer Kamera, die zu einer Person gehören, wird ausgemittelt. Diese ausgemittelte Position wird als Korrespondenzpunkt für die nachgeschaltete 3D-Logik verwendet.

Der hier vorgestellte Tracker arbeitet hierarchisch: Zuerst werden in jeder Kamera die 2D-Tracks mit der oben Beschriebenen Logik Korrigiert Und erst danach wird 3D-Logik verwendet, um das Gesamtergebnis zu verbessern. Man kann einwenden, dass es doch sinnvoll wäre im Fall "widersprüchliche 2D-Tracks" die anderen Kameras zu fragen, um zu entscheiden, welcher Track fehlerhaft ist und welcher nicht. Allerdings muss man zuerst die anderen Kameras (2D-)korrigieren, die dann möglicherweise auch eine 3D-Frage haben.

Eine hierarchische Arbeitsweise reduziert die Komplexität der Gesamtlogik für die Korrektur aller 2D-Tracks einer Person.

4.3 Erzeugung von 3D-Hypothesen

Wie zu Beginn bereits erwähnt, werden Personen, die einen 3D-Track haben als *3D-Personen* bezeichnet. Es gibt also in mindestens zwei Kameras je mindestens einen 2D-Track, der dieser Person zugeordnet ist. Erst bei 3D-Personen können 3D-Koordinaten bestimmt werden. Dazu müssen allerdings die Kameras kalibriert sein. Genauers ist im Kapitel 3.3 erklärt.

4.3.1 Korrespondenzfehler

Wie werden 3D-Personen erstellt? Angenommen es gibt eine 2D-Person P_1 in CAM1 und eine weitere P_2 in CAM2. Diese beiden 2D-Personen könnten in Wirklichkeit diesselbe Person sein. Das Problem ist, dass der Tracker das herausfinden muss. Eine Möglichkeit dieses sogenannte Korrespondenzproblem (siehe 3.3) zu lösen, ist die Verwendung des Triangulationsfehlers. Da-

zu wird der normalisierte (siehe 4.2) Mittelpunkt der Personen verwendet, um 3D-Koordinaten auszurechnen. Ein geringer Triangulationsfehlers ist eine notwendige, allerdings nicht hinreichende Bedienung, dass es sich wirklich um diesselbe Person handelt.

Zu dem Zeitpunkt, an dem eine Person neu erstellt wird, also eine 2D-Person mit einem aktiven Tracker, wird mit allen möglichen Kandidaten trianguliert. Hierbei wird angenommen, dass die anderen Personen im Moment präzise getrackt werden und deshalb kleine Fehler bei der Triangulation mit Personen, die zusammengehören und große Fehler bei der Triangulation mit Personen, die nicht zusammengehören auftreten. Diese Annahme ist allerdings nicht immer richtig.

Falls also P_1 und P_2 einen kleinen Triangulationsfehler haben, werden diese beiden zu einer neuen 3D-Person zusammengefügt. Da, wie gesagt, diese Bedienung nur eine notwendige ist, können auch verschiedene Personen fälschlicherweise zusammengefügt werden.

Beispielsweise in diesem Szenario: Im Raum befinden sich zwei Personen. In Kamera CAM1 wird eine der Personen erkannt und entsprechend eine neue Person mit einem einzigen aktiven 2D-Track erzeugt. Die LoV (siehe 3.3) dieser Person werde in CAM2 eingezeichnet. Es handelt sich dabei um alle Punkte, bei denen die Triangulation mit dem normalisierten Mittelpunkt der Person aus CAM1 einen geringen Fehler liefert. Es befinden sich die Mittelpunkte beider Personen darunter. Jetzt wird die andere Person in CAM2 erkannt und fälschlicherweise mit der in CAM1 erkannten Person zusammengefügt.

In Abbildung 13 ist ein leicht komplizierteres, aber ähnliches Szenario dargestellt. Der Mittelpunkt der schwarz gekleideten Person in CAM1 und die blauen Punkte in den restlichen Kameras haben jeweils einen geringen Triangulationsfehler. Besonders gut sichtbar ist, dass in CAM2 sowohl die schwarz gekleidete als auch die blau gekleidete Person einen ähnlich geringen Triangulationsfehler haben. In CAM3 ist die schwarze Person nicht sichtbar, aber die LoV schneidet eine rot gekleidete Person. In CAM4 gibt es keine Probleme, da andere Personen ausreichend weit von der LoV entfernt sind.

Folgendes Beispiel klingt vielleicht konstruiert, ist aber mit drei Personen bei den Entwicklungsdaten vorgekommen. Vier verschiedene Personen stehen so im Raum verteilt, dass alle LoVs perfekt zusammenpassen. In Abbildung 14 sind die Person P_1 bis P_4 mit ihren LoVs eingezeichnet, wobei CAM1 genau einen aktiven Tracker auf P_1, CAM2 genau einen aktiven auf P_2, usw. haben.

Diese vier 2D-Personen werden mittels Triangulation zur 3D-Person V zu-

Abbildung 13: Als blaue Punkte eingezeichnete LoVs in CAM2 bis CAM4 für den Mittelpunkt der schwarz gekleideten Person aus CAM1

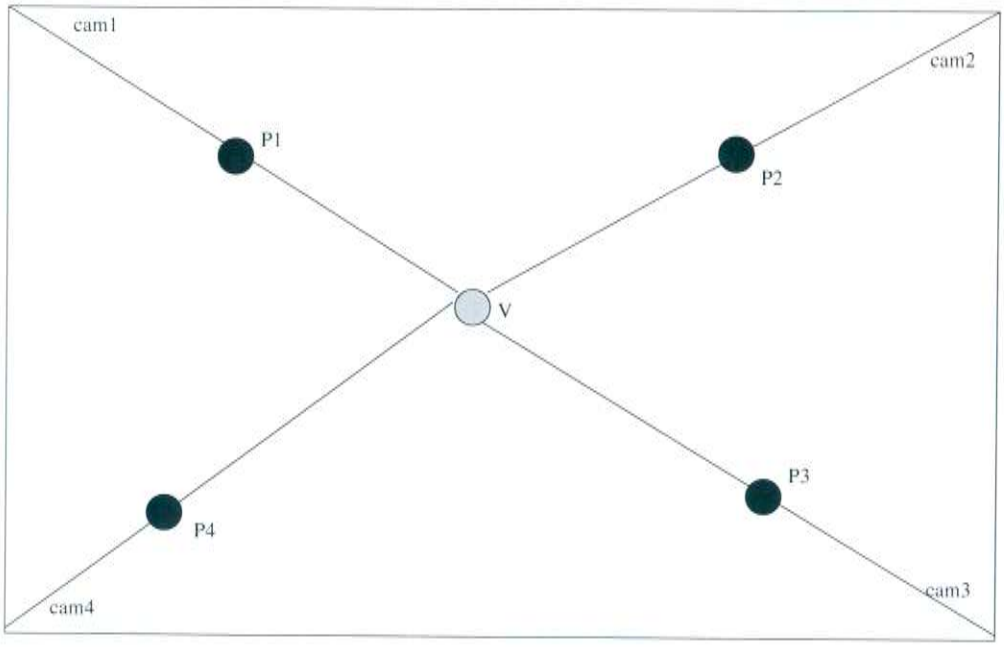


Abbildung 14: Falsch gelöstes Korrespondenzproblem: P_1 bis P_4 stehen so, dass sich die LoVs in V schneiden und statt vier nur eine (virtuelle) Person V erkannt wird.

sammengefügt, die scheinbar in der Mitte des Raumes steht. Solche Konfigurationen haben normalerweise nur kurz bestand, denn sobald eine Person sich bewegt, steigt der Triangulationsfehler an.

Mit zunehmender Anzahl an Personen in einem Raum kann es also immer leichter zu Verwechslungen kommen, wenn nur der Triangulationsfehler benutzt wird, um das Korrespondenzproblem zu lösen.

Eine Möglichkeit zur Verbesserung der Heuristik, ist die Verwendung zusätzlicher Kriterien. Beispielsweise wäre es denkbar, die Farbhistogramme der fraglichen 2D-Tracks zu analysieren. Es wird versucht herauszufinden, wie ähnlich die Farbhistogramme sind, damit eine schwarz gekleidete Person nicht mit einer blau gekleideten zusammengefügt wird. Allerdings sind Farbhistogramme häufig fehlerbehaftet, da die automatische Initialisierung beispielsweise Hintergrund mitlernt. Zudem können sie nicht über Kameragrenzen vergleichbar sein, falls der Weißabgleich der Kameras nicht identisch ist.

Das Hauptproblem ist aber, dass Farbhistogramme spezifisch auf eine angelernte Person angepasst sind und nur schwer auf andere Personen übertragen

werden können. Es ist sowohl schwierig eine positive ("Personen passen zusammen") als auch eine negative ("Personen passen nicht zusammen") Aussage zu treffen. Deshalb müssten für dieses zusätzliche Kriterium niedrige Schwellwerte gewählt werden, damit das Zusammenfügen zusammengehörender Personen nicht verhindert wird.

Diese Heuristik wird deshalb **nicht** verwendet.

Der Mittelpunkt einer Person kann sich nur innerhalb einer bestimmten Höhe befinden. Die Schwellwerte (siehe Kapitel 5.4) müssen dabei so gewählt werden, dass richtige Personen nicht ausgeschlossen werden. Die untere Grenze ist durch sitzende Personen, bei denen der Tracker leicht nach unten verrutscht ist, festgelegt. Die obere ist durch stehende Personen, bei denen der Tracker etwas zu klein ist und sich weit oben befindet, bestimmt. Die Spezialfälle werden ignoriert, dass eine Person etwa extrem groß ist, auf einen Tisch klettert, sich auf den Boden legt oder ähnliches.

Zusätzlich wird im laufenden Betrieb jede 3D-Person ständig geprüft, ob sie diese Bedingung verletzt. Falls sie es tut, wird versucht, den 2D-Track zu finden, der am meisten zu einer hohen Abweichung der z-Koordinate[7] zum Erwartungswert beiträgt. Dieser Track wird dann deaktiviert. Deaktivieren macht in diesem Fall mehr Sinn als abspalten, da dieser Track offenbar nicht mehr richtig funktioniert, also instabil oder falsch ist.

Bei nur zwei aktiven Kameras wird die Person aufgetrennt, da offenbar das Korrespondenzproblem falsch gelöst wurde.

Auf diese Weise können Korrespondenzfehler reduziert werden. Insbesondere wenn Haarkaskaden mit hoher false-positive-Rate verwendet werden, sorgt diese Heuristik dafür, dass viele falsche 3D-Tracks nicht entstehen können.

Insbesondere in folgendem Beispiel führt die Höhenheuristik zu besserem Trackingverhalten: Eine Person wird mit allen vier Kameras getrackt. In einer Kamera wandere der bislang stabile 2D-Track nach oben. Die Höhenheuristik deaktiviert diesen Tracker. Im nächsten Bild werde ein neuer, bereinigter Tracker eingelernt und zu dieser Person hinzugefügt. Die Person kann zuverlässig weitergetrackt werden.

3D-Personen werden als deutlich wertvoller bzw. glaubwürdiger als 2D-Personen eingestuft. Deshalb wird regelmäßig geprüft, ob es für eine 3D-Person eine 2D-Person gibt, die passen würde. Falls das eindeutig ist, also eine 2D-Person genau einer 3D-Person zugeordnet werden kann, werden diese beiden

[7] Die Koordinaten der Kameras sind so kalibriert, dass sich x und y in der Ebene befinden und z senkrecht nach oben gerichtet ist.

zusammengefügt. Falls ein einzelner 2D-Track für mehrere 3D-Personen passend wäre, wird vorsichtshalber kein Zusammenfügen angestoßen.

Das hat den Vorteil, dass einzelne 2D-Tracks von 3D-Personen "aufgesammelt" werden und somit nicht mehr für eine potentiell falsche Triangulation zur Verfügung stehen. Zudem kann sich die Anzahl an 2D-Tracks erhöhen, die eine 3D-Hypothese unterstützen.

Angenommen, die Entscheidung, die Personen zusammenzufügen, war falsch. Der aufgesammelte 2D-Track ist also entweder instabil oder trackt eine andere Person. In diesem Fall wird er ganz regulär, wie im nächsten Kapitel 4.3.2 beschrieben, wieder aus der 3D-Person entfernt.

4.3.2 Fehlerbehandlung

Angenommen mehrere 2D-Tracks bilden eine 3D-Person. Dann kann es dazu kommen, dass der Triangulationsfehler mit der Zeit recht groß wird. Ursache für dieses Problem ist, dass in mindestens einer Kamera ein 2D-Track instabil oder falsch ist. Dadurch passen die 2D-Tracks nicht mehr richtig zu einem 3D-Track zusammen.

Eine mögliche Strategie, die schon von der Behandlung von 2D-Tracks (siehe 4.2) bekannt ist, nennt sich Ausmitteln. Aus allen 2D-Tracks wird mittels Triangulation ein 3D-Punkt ausgerechnet. Dieser ist natürlich fehlerbehaftet, aber er wurde so berechnet, dass der Abstand zu allen beteiligten LoVs möglichst gering ist. Es handelt sich hierbei also um einen ausgemittelten Wert. Dieser 3D-Punkt wird nun in die Kameras zurückprojiziert und alle 2D-Tracks werden entsprechend verschoben. Diese Strategie hat den Vorteil, dass ein kurzfristiges Versagen eines 2D-Tracks in einer einzelnen Kamera relativ gut ausgeglichen wird.

Eine schwarz gekleidete Person laufe an einem schwarzen Sessel vorbei. Je nach Kameraperspektive wird ein 2D-Tracker versuchen, den Sessel und die Person gleichzeitig zu tracken. Der Track bleibe in einer Kameraperspektive beim Sessel hängen. Aus einem anderen Blickwinkel, bei dem die Person den Sessel verdeckt, kommt es zu einem geringeren Fehler. Durch das 3D-Ausmitteln wird nun der 2D-Tracker, der den Sessel zu tracken versucht, zurück auf die Person verschoben.

Allerdings kann ein Trackerfehler auch andere Gründe haben. Beispielsweise ein 2D-Track ist instabil und beginnt die Wand anstelle der Person zu tracken. Dann kann es leicht passieren, dass die anderen, korrekt arbeitenden Tracker durch das Ausmitteln von der Person "weggeschoben" werden. Das initale

Suchfenster für Meanshift wird also schlecht gewählt und Meanshift kann deshalb die Person verlieren.

Ein weiterer unangenehmer Fall ist das bereits erwähnte falsch gelöste Korrespondenzproblem. Hier wurden 2D-Tracks von verschiedenen Personen fälschlicherweise einer Person zugeordnet. Wenn sich nun diese Personen bewegen, wird der Triangulationsfehler normalerweise immer größer. Durch das Ausmitteln wird eine falsche 3D-Hypothese für längere Zeit aufrecht erhalten. Zudem können hier ebenfalls korrekt arbeitende 2D-Tracker so weit verschoben werden, dass sie anfangen etwas anderes zu tracken.

Aus diesem Grund ist es sinnvoll, eine andere Strategie zu verfolgen. Falls der Triangulationsfehler zu groß wird (siehe 5.4), dann ist offenbar die Hypothese falsch. Es sollte die Hypothese anstelle der 2D-Tracker korrigiert werden.

Es wird versucht, eine minimale Anzahl an Trackern zu entfernen, um den Triangulationsfehler maximal zu senken. Das sollte normalerweise der fehlerhafte Tracker sein. Im Idealfall sind alle Kameras aktiv, also normalerweise vier. Probehalber wird jede Kamera komplett deaktiviert. Eine der Konfigurationen an drei aktiven Kameras hat einen minimalen Triangulationsfehler. Angenommen dieser Fehler ist bereits ausreichend gering, dann ist das Auftrennen der Person erfolgreich. Die Person wird in zwei aufgeteilt. Die eine Person hat alle 2D-Tracker, die zu der Drei-Kamera-Konfiguration gehören, und die andere die restlichen 2D-Tracker der vierten Kamera.

Falls das nicht zum Erfolg führt, könnte man alle möglichen Zwei-Kamera-Konfigurationen untersuchen, allerdings kommt dieser Fall bei den Entwicklungsdaten sehr selten vor. Deshalb wird das Problem iterativ gelöst. Im ersten Schritt wird aus einer Person mit vier Kameras einer Person mit drei. Falls im nächsten Bild der Fehler immer noch zu groß ist, wird die Person um eine weitere Kamera reduziert. Sobald nur noch zwei Kameras aktiv sind, führt ein Auftrennen dazu, dass eine 3D-Person in zwei 2D-Personen zerfällt. Dadurch geht zwar eine 3D-Person verloren, da es sich aber um einen Fehlerfall handelt, ist es nicht sinnvoll diese 3D-Person aufrecht zu erhalten. Mittels der Haarkaskade werden bekanntlich (siehe 4.1.1) ständig neue 2D-Tracks erstellt. Deswegen wird eine 2D-Person, die wirklich eine Person trackt, wahrscheinlich wieder zu einer 3D-Person werden.

Im letzten Abschnitt wurde das Auftrennen von Personen beschrieben. Die eine Hälfte des Auftrennens ist häufig eine 3D-Person, die andere immer eine 2D-Person. Anstatt diese 2D-Person weitertracken zu lassen könnte man sie auch löschen. Allerdings kann es verschiedene Gründe haben, warum ein Auftrennen nötig ist. Einer davon ist, dass der Track falsch ist. Diesen Fall

erledigt das Kreditsystem (siehe 4.1.4) nach einiger Zeit, für die anderen Fälle ist ein Löschen des Tracks die falsche Lösung. Deshalb ist das Auftrennen von Personen die geschickteste Vorgehensweise.

Neben dem Auftrennen gibt es eine weitere Möglichkeit, wie eine Person 2D-Tracker oder eine ganze Kamera, also alle 2D-Tracks dieser Kamera, verlieren kann. Sobald eine Person sich soweit bewegt hat, dass sie in einer Kamera nicht mehr sichtbar ist, wird sie aus dieser Kamera entfernt. 2D-Tracks, die zu einem großen Teil die Kamera verlassen haben, müssen also deaktiviert werden.

Angenommen eine Personen legt einen Weg zurücklegt, durch den sie nach und nach aus allen Kameras entfernt wird. In der Zwischenzeit werden auch keine neuen Tracks eingelernt. Dann kann es passieren, dass diese Person nicht mehr getrackt werden kann.

4.3.3 2D-Kreditsystem mit 3D-Wissen

In Kapitel 4.1.4 wurde das Kreditsystem eingeführt. Dort wird erklärt, wie 2D-Personen nach einiger Zeit gelöscht werden. Der Teil des Kreditsystems, der 3D-Wissen verwendet wird hier erläutert.

Alle Tracker einer 3D-Person bekommen pro Bild eine Einheit Guthaben, bis das vordefiniertes Maximum erreicht ist. Ein langes Bestehen als 3D-Person ist ein Indiz dafür, dass es diese Person wirklich gibt. Falls ein 2D-Track aus einer 3D-Person entfernt wird, wird eine neue 2D-Person, die nur diesen einen aktiven 2D-Track hat, erstellt. Diese neue Person hat im Vergleich zu einer normal entstandenen Person mehr Guthaben und kann deshalb länger bestehen, bevor sie gelöscht wird. Dieses Verhalten ist sinnvoll, da es sich bei diesem 2D-Track mit hoher Wahrscheinlichkeit um einen funktionierenden Track handelt. Das Kreditsystem soll falsche Tracks löschen und richtige behalten.

Es stellt sich nun die Frage, warum ein Track aus einer Person abgespaltet wurde. Beispielsweise hat eine Kamera einen temporären Trackerfehler. Der Track in Kamera CAM2 sei zum Beispiel im Moment nicht mehr auf der Person, sondern deutlich daneben. Er verfolgt statt der schwarz gekleideten Person den schwarzen Sessel, der bis vor kurzem von der Person verdeckt war. Die Tracks in den anderen Kameras seien nur geringfügig falsch, da die Person den Sessel aufgrund der anderen Kameraperspektiven noch vorwiegend verdeckt.

Durch den Trackerfehler in CAM2 wird entschieden, dass ein Track abgespal-

tet werden muss. Nun wird der Tracker ermittelt, bei dessen Entfernung aus der Person der Triangulationsfehler minimiert wird. Das sollte natürlich der Track in CAM2 sein. Allerdings kann es passieren, dass sich die leichten Fehler der anderen Tracks aufheben bzw. verstärken, so dass der Track in CAM1 als Ursache vermutet wird. Es werde also der Track in CAM1 abgespaltet. Im nächsten Bild bleibt der Track in CAM2 weiterhin auf dem Sessel und diesmal wird er korrekt abgespaltet. Nun passt der vorhin abgespaltete Track aus CAM1 wieder zu der 3D-Person, da der Triangulationsfehler gering ist. Er wird entsprechend hinzugefügt. Es kann passieren, dass der Track in CAM2 im späteren Verlauf weiterhin den Sessel trackt und die Person vollständig verliert. Alternativ "erholt" sich der Track und verfolgt wieder die Person. Da beide Fälle vorkommen können, ist es nicht klar, ob es besser ist, abgespalteten Tracks Guthaben zu geben oder zu nehmen. In den verwendeten Entwicklungsdaten ist der Fall, dass ein Track entfernt und wieder hinzugefügt wird häufig vorgekommen. Falsche Tracks haben zudem häufig die Eigenschaft, dass sie schnell das Bild verlassen und deshalb gelöscht werden oder dass sie an einer Positionen verharren. Diese Positionen ist häufig für eine Person nicht zulässig. Deshalb legen die Entwicklungsdaten letztendlich den Schluss nahe, dass es sinnvoller ist, einem abgespalteten Track das Guthaben zu lassen, welches während der Zeit als Teil einer 3D-Person erworben wurde.

4.3.4 Handhabung von Verdeckung

Bereits in 4.1.3 wurde definiert, was Überlappung bedeutet. Zwei Rechtecke können sich gegenseitig überlappen. Hierbei ist die Richtung wichtig, da je nach Richtung der Grad der Überlappung unterschiedlich sein kann.

Im folgenden werden nur Überlappungen zwischen Rechtecken desselben Haarkaskadentyps betrachtet.

Bei einer Verdeckung ist eine Person aus einer Kameraperspektive nicht vollständig sichtbar. Ursache für eine Verdeckung kann ein Gegenstand im Raum sein. Das wird vom Tracker nicht erkannt, da er kein Modell des Raums hat, sondern nur von Personen. Sobald eine Person eine andere Person verdeckt, ist es prinzipiell möglich, dass der Tracker dies erkennt. Voraussetzungen hierfür ist, dass die beiden Personen korrekt getrackt werden. Sobald die Person, die näher an der fraglichen Kamera ist die andere Person verdeckt, wird es zu einer Überlappung der entsprechenden Rechtecke des Trackers kommen.

Das heißt, dass Personen sich verdecken können, ohne dass es zu einer Über-

lappung kommt, da die entsprechende 2D-Tracker falsch oder nicht vorhanden sind. Oder es können sich Tracker überlappen, ohne dass die Personen sich verdecken.

Der Tracker kann Überlappung einfach ermitteln, aber ob wirklich eine Verdeckung vorliegt, kann er nur mit hoher Wahrscheinlichkeit vermuten.

Angenommen ein Tracker einer 2D-Person überlappt sich mit dem Tracker einer anderen 2D-Person. Die Überlappung habe in beiden Richtungen einen gewissen Schwellwert überschritten. Dadurch wird entschieden, dass die beiden Tracker diesselbe Person beschreiben und ein Zusammenfügen wird angestoßen.

Dabei werden die Rechtecke in Größe und Position ausgemittelt und die Farbhistogramme sollten geeignet zusammengeführt werden. Die einfache Möglichkeit ist hierbei, zufällig eines der beiden auszuwählen. Eine etwas bessere ist, mit dem in Kapitel 4.1.2 angesprochenen Verfahren das passendere Farbhistogramm auszuwählen. Eine weitere Möglichkeit wäre der Versuch, die Gemeinsamkeiten der Farbhistogramme in ein gemeinsames neues zusammenzuführen. Durch die automatische Anpassung (siehe 4.1.3) von Farbhistogrammen wird ständig eine inkrementelle Adaption durchgeführt. Deshalb ist selbst die erste Möglichkeit häufig vollkommen ausreichend.

Angenommen ein Tracker einer 2D-Person überlappt sich mit einem Tracker einer 3D-Person. Dieser Fall wird analog zum vorigen Fall gehandhabt.

Beim Zusammenführen der beiden Personen wird darauf geachtet, dass der Index der 3D-Person erhalten bleibt. Dadurch wird die Namensstabilität einer 3D-Person erhöht. Das ist wichtig, damit Vertauschungen von Personen möglichst vermieden werden.

Falls sich Tracker von zwei 3D-Personen überlappen, wird davon ausgegangen, dass eine Verdeckung statt findet. Es befinden sich also an einer Stelle zwei Personen, deren Silhouetten mindestens zum Teil verschmolzen sind, wobei die eine Person die andere zumindest aus dieser Kamerasicht partiell verdeckt.

Dadurch ergibt sich das Problem, dass ein Tracker von einer Person zu einer anderen überspringen könnte. Die Gefahr ist besonders groß, wenn die beiden Personen ähnlich gekleidet sind. Normalerweise sind auch die Bewegungszonen verschmolzen, es ist also sehr schwierig geworden, die beiden Personen zu unterscheiden. Da 3D-Koordinaten bekannt sind, kann der Abstand der Personen von der Kamera berechnet werden. Sollten sich die Abstände recht deutlich, also größer als ein Schwellwert, unterscheiden, dann wird der verdeckte Tracker deaktiviert.

Häufig sind die anderen Kameras aktiv und können die Position der verdeckten Person interpolieren. Sobald die Verdeckung nicht mehr besteht, also das geschätzte Rechteck eine geringe Überlappung hat, könnte dieser Tracker wieder aktiviert werden.

Das Problem hierbei ist, dass sobald eine Verdeckung in einer Kamera auftritt, die Chancen gut sind, dass auch in einer anderen Kamera eine Verdeckung statt findet. Noch komplizierter wird es, wenn mehrere Personen sich gegenseitig verdecken. Aus diesem Grund sollte erst bei einer relativ hohen Überlappung ein Verdeckungszustand erkannt werden. Andererseits kann ein verdeckter Tracker instabil werden, da er nur noch einen geringen Teil der Person sieht. Wie bei vielen Schwellwerten ist auch hier der "richtige" Wert von mehreren Faktoren abhängig und wird durch die Entwicklungsdaten empirisch ermittelt.

4.3.5 Der Hinweisgenerator

In Kapitel 4.2 wurde erklärt, dass es mehrere verschiedene Haarkaskadentypen gibt, die mithilfe einer Normalisierung vergleichbar gemacht wurden.

Prinzipiell kann jeder Haarkaskadentyp in jeden anderen konvertiert werden. Allerdings gibt es immer eine leichte Ungenauigkeit, so dass schon das Extrapolieren des Bauchnabels unpräzise werden kann. Deshalb ist ein Konvertieren der Haarkaskadentypen **nicht** zu empfehlen.

Sobald eine 2D-Person zu einer 3D-Person wird, sind die 3D-Koordinaten bekannt. Diese können in alle Kameras zurückprojiziert werden. Dadurch ist der ausgemittelte Mittelpunkt der Personen in aktiven Kameras bekannt. Zusätzlich kann für inaktive Kameras eine Vorhersage gemacht werden, wo sich eine Person befinden müsste. Ein Vorhersage für aktive oder inaktive Kameras wird als *Hinweis* bezeichnet.

Hier tritt das Problem auf, dass Personen partiell verdeckt sein können. Angenommen eine Person wird durch Gesichtshaarkaskaden getrackt. Daraus kann gefolgert werden, dass das Gesicht in den anderen Kameras meist sichtbar ist. Allerdings ist die Folgerung, dass die komplette Person sichtbar ist deutlich riskanter, da Personen beispielsweise sitzen können.

Aus diesem Grund werden nur gleiche Typen kombiniert. Ein Gesicht in einer Kamera und ein Ganzkörper in einer anderen werden also nicht benutzt, um Vorhersagen zu treffen. Ein weiterer Grund ist das Ungenauigkeitsproblem, das schon im 2D-Fall eine Konvertierung der Typen verhindert.

Gleiche Haarkaskadentypen in verschiedenen Kameras können dagegen ver-

gleichsweise problemlos benutzt werden, um diesen Typ in anderen Kameras vorherzusagen.

Angenommen eine 3D-Person hat zwei aktive Ganzkörpertracker in CAM1 und CAM2. Bekanntlich benutzen die Tracker Rechtecke. Es sollen nun die Rechtecke für die restlichen Kameras berechnet werden.

Es werden die Koordinaten der linken oberen Ecke des ersten Rechtecks und des zweiten Rechtecks verwendet, um 3D-Koordinaten auszurechnen. Dies werde analog für die restlichen Ecken durchgeführt. Aus diesen vier 3D-Koordinaten kann für jede Kamera ein Rechteck durch Rückprojektion berechnet werden.

Die resultierenden Rechtecke können richtig sein, allerdings ist dieser Ansatz falsch. In Abbildung 15 ist zu erkennen, dass die Eckpunkte der Rechtecke keineswegs miteinander korrespondieren. Die Abbildung ist eine schematische Sicht des Raumes von oben. Aus Sicht von CAM1 hat das Objekt (der große schwarze Punkt in der Mitte) einen linken Rand (beschriftet mit 1) und einen rechten (2). Allerdings sind aus Sicht von CAM2 der linke (3) und rechte (4) Rand des Objekts andere Punkte. Wenn einfach die linken bzw. rechten Punkte benutzt werden, um 3D-Koordinaten auszurechnen, wurde das Korrespondenzproblem offensichtlich falsch gelöst.

Da dieser erste Ansatz nicht zuverlässig funktioniert, wird eine andere Strategie verfolgt.

Es wird jeweils der Mittelpunkt des Rechtecks als Referenzpunkt benutzt, um den Mittelpunkt der zu interpolierenden Rechtecke zu berechnen. Für Breite und Höhe der neuen Rechtecke wird eine 3D-Standardgröße gewählt. Die Höhe ist unabhängig von der Perspektive der Kamera, die Breite dagegen schon, da Personen nicht zylinderförmig sind. Diese Standardgrößen werden in 2D-Pixelkoordinaten der einzelnen Kameras umgerechnet. Da die Entfernung des Mittelpunkts von der jeweiligen Kamera bekannt ist, ist das möglich. Die Rechtecke werden auf die maximale Ausdehnung des Vordergrunds innerhalb des Rechtecks reduziert. Deshalb ist eine leicht zu groß gewählte Standardgröße akzeptabel.

Bei Kameras, in denen es noch keinen Tracker des vorhergesagten Typs gibt, wird ein neues Lernen eines Trackers angestoßen. Diese Hinweise unterscheiden sich nicht von Treffern des entsprechenden Haarkaskadentyps. Es wird also ein Tracker, wie in 4.1.1 beschrieben, angelernt. Falls dieser Tracker zu der 3D-Person passt, die den Hinweis ausgelöst hat, wird der Tracker, wie in 4.3.1 beschrieben, hinzugefügt.

Bei Kameras, in denen schon ein Tracker aktiv ist, wird der Hinweis zur Ad-

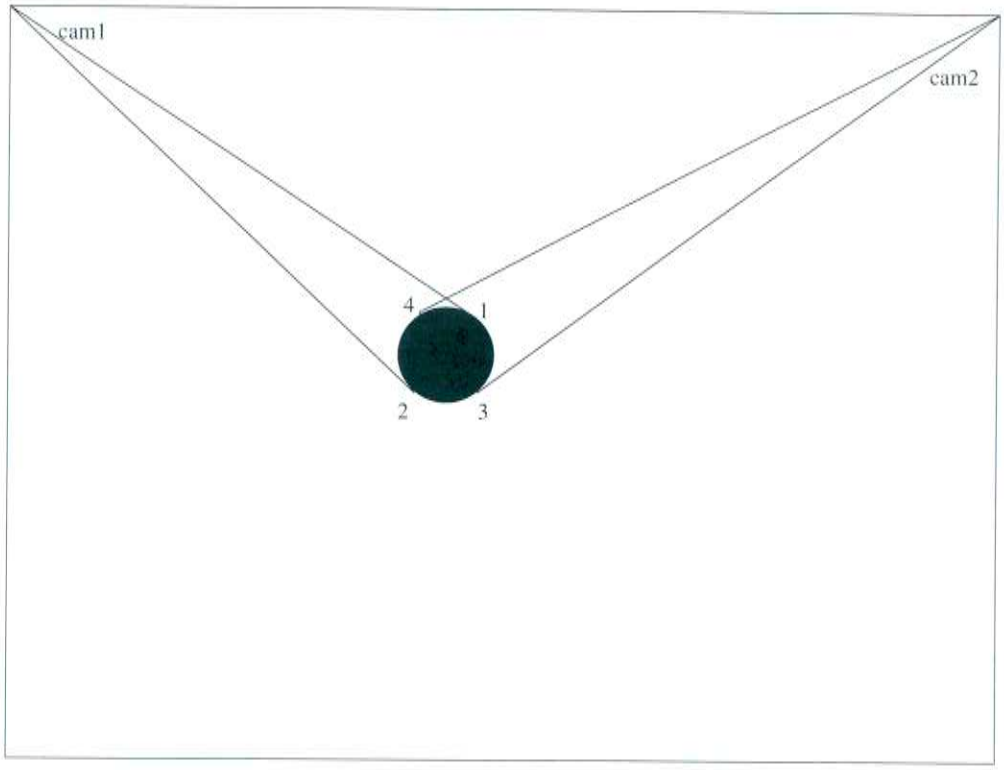


Abbildung 15: Korrespondenzproblem

aption (siehe 4.1.3) verwendet. Normale Hinweise werden benutzt, um neue Tracker zu erzeugen. Diese Hinweise allerdings haben eine hohe Überlappung mit einem bestehenden Tracker, da er den Hinweis mit ausgelöst hat. Bei diesen Hinweisen wird die zugehörige Personennummer gespeichert, damit er sicher dem richtigen Tracker zugeordnet wird.

Auch hier kann es passieren, dass ein Hinweis eine Überlappung mit einer anderen Person hat. Da mit Hinweisen die Farbhistogramme adaptiert werden, ist es im Überlappungsfall sicherer den Hinweis zu ignorieren. Ansonsten sind die Chancen gut, ein Farbhistogramm mit einer anderen Person zu verunreinigen.

Neben der geringen Überlappung gibt es eine weitere Heuristik, die Hinweise verwirft. Falls der Triangulationsfehler der 3D-Person hoch ist, werden keinerlei Hinweise erzeugt, da hier die interpolierte Position leicht falsch sein kann.

Der Hinweisgenerator ist ein wichtiger Bestandteil des Trackers. Sobald eine Person in lediglich zwei Kameras erkannt wurde, wird sofort versucht die restlichen Kameras zu aktivieren. Während des normalen Trackens kann ein einzelne Kamera durchaus einen Tracker haben, der instabil wird. Dieser wird normalerweise entfernt und durch einen neu angelernten Tracker ersetzt. Größenänderungen oder Farbänderungen werden durch die automatische Adaption zügig ausgeglichen.

Deswegen führt der Hinweisgenerator dazu, dass der Tracker sich schnell von vielen Fehlerarten erholen kann.

4.3.6 Triangulationsfehler

Häufig befinden sich Tracker nicht ganz korrekt auf einer Person. Insbesondere der Oberkörpertyp trackt häufig nicht Oberkörper und Kopf, sondern Oberkörper ohne Kopf und dafür mit etwas mehr Bauch.

Ursache hierfür ist die Arbeitsweise von Meanshift. Ein Farbhistogramm eines Oberkörpers (mit Kopf) erzeugt normalerweise in der Back Projection einen roten[8] Ober- und Mittelkörper, der Kopf ist häufig nur leicht gerötet. Zudem hat ein Kopf eine kleine Fläche im Vergleich zum Rest des Körpers. Deshalb kann das Massezentrum, welches Meanshift sucht leicht nach unten verschoben sein.

Damit dieses Verhalten nicht zu sehr ins Gewicht fällt, wird bei der Berech-

[8]Hoher Rotanteil entspricht hoher Wahrscheinlichkeit.

nung des Triangulationsfehlers statt dem normalen euklidischen Abstand

$$residual = \sqrt{x^2 + y^2 + z^2}$$

eine abgeschwächte Variante

$$residual = \sqrt{x^2 + y^2 + \frac{1}{2}z^2}$$

verwendet.

4.4 Ausgabehypothesen

Ein Problem, das insbesondere bei der Gesichtskaskade auftritt, ist das mehrfache Tracken einer Person. Es kann passieren, dass beispielsweise zwei 3D-Personen in evtl. unterschiedlichen Kameras diesselbe Person tracken. Der Triangulationsfehler ist hierbei zu groß, als dass diese 3D-Personen zu einer zusammengefasst werden können.

Zur Lösung dieses Problems wird eine sogenannte *Ausgabeperson* erstellt, die eine oder mehrere 3D-Personen enthalten kann. Dabei wird ausschließlich der Abstand in der Ebene berücksichtigt, also $\sqrt{x^2 + y^2}$. Wenn eine neue 3D-Person erzeugt wird und diese sich in der Nähe einer Ausgabeperson befindet, wird sie dieser zugeordnet. Umgekehrt können 3D-Personen verschwinden oder zerfallen, wobei die Ausgabeperson nicht gelöscht wird, solange mindestens eine 2D-Person übrig bleibt.

Dadurch erhöht sich die Namensstabilität. Eine Person wird erheblich länger mit demselben Index getrackt und es kommt dadurch zu deutlich weniger Vertauschungen. Selbst wenn eine Ausgabeperson alle 2D- oder 3D-Personen verliert, wird sie ein paar Bilder (inaktiv) an der letzten bekannten Position beibehalten. Sobald eine neu 3D-Person in der Nähe entsteht, wird diese Ausgabeperson wiederverwendet. Dieses Verfahren verringert also die Anzahl Vertauschungen durch die erhöhte Namensstabilität.

Zusätzlich gibt es weniger false positive Personen, da 3D-Personen, die in Wirklichkeit diesselbe Person tracken, häufig zu einer zusammengefasst werden.

In Abbildung 16 ist das Ausgabefenster des Trackers zu sehen. Es stellt eine schematische Ansicht des Raumes von oben dar. Die Kameras werden mit ihren (echten) Koordinaten eingezeichnet. Das große grüne Rechteck ist eine Schätzung der Wände des Raumes. Zur Orientierung ist in blau die Position des Tisches eingezeichnet. Von 3D-Personen werden die x- und y-Koordinaten

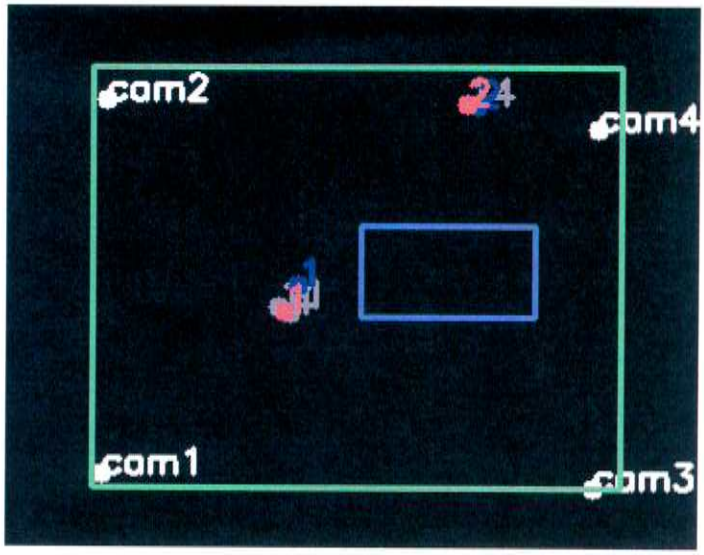


Abbildung 16: Schematische Darstellung des Raumes

verwendet, um ihre Position im Raum als kleine Kreise mit Personennummer einzuzeichnen. Da es sich um eine Draufsicht von oben handelt, werden die z-Koordinaten einer Person ignoriert. In der Abbildung sind zur Zeit zwei Personen aktiv, die mit einer roten 1 bzw. 2 markiert werden. Person 2 befindet sich in der Nähe der Tür. In blau ist die letzte bekannte Grundwahrheit (siehe Kapitel 6.1) eingezeichnet.

Die grauen Artefakte sind die Positionen von 3D-Personen, die von der Ausgabeperson (rot, siehe weiter unten) und der Grundwahrheit (blau) übermalt wurden.

Blau Letzte bekannte Grundwahrheit

Grau Position der 3D-Person

Rot Position der Ausgabeperson

Die Wände stellen eine obere Grenze dar. Deswegen kann es niemals vorkommen, dass sich Personen außerhalb des Raumes befinden. Falls also der Tracker eine 3D-Person liefert, die sich angeblich außerhalb des Raumes befindet, kann es sich nur um einen Irrtum handeln. Solch eine Person wird sofort komplett gelöscht.

Der Tisch ist nur zur Orientierung des Betrachters eingezeichnet und wird nicht verwendet. Es ist nicht möglich, dass sich 3D-Personen innerhalb des

Tisches befinden, da Personen auf dem Tisch aufgrund der Höhenheuristik (siehe 4.3.1) nicht getrackt werden.

Allerdings kann es durch leichte Fehler eines 2D-Tracks passieren, dass sich eine Person leicht innerhalb des Tisches zu befinden scheint. Der Tisch ist nicht im Raum fixiert, da er sich beispielsweise in späteren Aufnahmen an einer anderen Stelle befindet. Zudem wird er nicht automatisch vom Tracker erkannt. Deshalb wird diese Art des Weltwissens nicht verwendet.

4.5 Zusammenfassung der Techniken

Der Tracker verwendet viele Techniken. Die wichtigsten werden hier noch einmal zusammengefasst dargestellt:

Kreditsystem 2D-Personen werden, falls sie keine Bestätigung erhalten nach einiger Zeit gelöscht. (siehe 4.1.4, 4.3.3)

Zusammenfügen von 2D-Tracks 2D-Tracks verschiedener Typen, die Nahe beieinander sind, werden zusammengefasst. (siehe 4.2)

2D-Überlappung 2D-Tracks gleichen Typs, die sich überlappen werden zusammengefasst. (siehe 4.3.4)

Löschen von 2D-Tracks Falls zusammengefasste 2D-Tracks auseinander laufen werden sie evtl. gelöscht. (siehe 4.2)

Zusammenfügen zu einer 3D-Person Beim Einlernen eines 2D-Tracks wird ein Zusammenfügen mit einem anderen 2D-Track durchgeführt, falls der Triangulationsfehler gering ist. Dadurch entsteht eine 3D-Personen. (siehe 4.3.1)

Auftrennen von 3D-Tracks Falls der Triangulationsfehler zu groß wird, wird ein geeigneter 2D-Track aus der Person entfernt. (siehe 4.3.2)

Höhenheuristik Die Höhe des Bauchnabels wird sinnvoll begrenzt und reduziert dadurch Korrespondenzfehler. (siehe 4.3.1)

Aufsammeln 2D-Personen werden zu 3D-Personen hinzugefügt, falls möglich und eindeutig. (siehe 4.3.1)

3D-Verdeckung Tracker von verdeckten 3D-Personen werden deaktiviert. (siehe 4.3.4)

Hinweisgenerator Durch Interpolation der 3D-Koordinaten werden Hinweise zum Neulernen und Adaptieren von Personen erzeugt. (siehe 4.3.5)

Ausgabeperson 3D-Personen können zusammengefasst werden, falls sie diesselbe Person tracken. Dies bringt eine Erhöhung der Namensstabilität. (siehe 4.4)

5 Implementierung

Die Implementierung ist ein Prototyp, der das eben beschriebene Design umsetzt. Ein großer Teil des Arbeitsaufwands ist natürlich in die Implementierung geflossen, allerdings werden hier nur die wichtigsten Informationen erwähnt.

5.1 Allgemeines

Der Tracker wurde in `C++` unter `GNU/Linux` entwickelt und hat etwa 6000 Codezeilen[9]. Hierbei wurde die umfangreiche Grafikbibliothek `opencv 0.9.6` verwendet. Aus ihr wurden u.a. die Implementierung von Haarkaskaden, Meanshift und Farbhistogramme verwendet. Weitere Algorithmen wie der Stauffersegmentierer, Triangulation und das Laden der Testdaten stammen aus dem cvs repository des Instituts. Eine rudimentäre GUI zur Steuerung des Trackers wurde mithilfe von `Qt3` realisiert.

5.2 Haarkaskaden

Obwohl die Haarkaskade zu den schnellsten Objekterkennern gehört, benötigt sie dennoch viel Rechenzeit. Die Implementierung in [18] erreicht bei einem Bild von 384x288 mit einem 700 MHz Pentium III 15fps. Bei diesem Tracker werden Bilder mit einer Auflösung von 640x480 und vier Kameras verwendet. Zudem läuft nicht nur eine Haarkaskade sondern bis zu drei. Der Rechenaufwand ist also etwa um den Faktor 30 größer. Deshalb erreicht ein einzelner Prozessor[10] keine Echtzeit (15fps).

Die Implementierung der Haarkaskaden aus `opencv 0.9.6` wurde erweitert, um das minimale und maximale Suchfenster angeben zu können. Neben einer leichten Steigerung der Geschwindigkeit wurden damit falsche Haarkaskadentreffer, die eine falsche Größe haben, reduziert: Die Ganzkörperkaskade scheint eine Vorliebe für Colaflaschen zu haben.

5.3 Design

In Abbildung 17 werden die verwendeten Klassen und ihre Interaktion vereinfacht dargestellt. Die wirkliche Interaktion ist erheblich komplexer, würde

[9] laut SLOCCount von David A. Wheeler
[10] Pentium 4, 2.8GHz

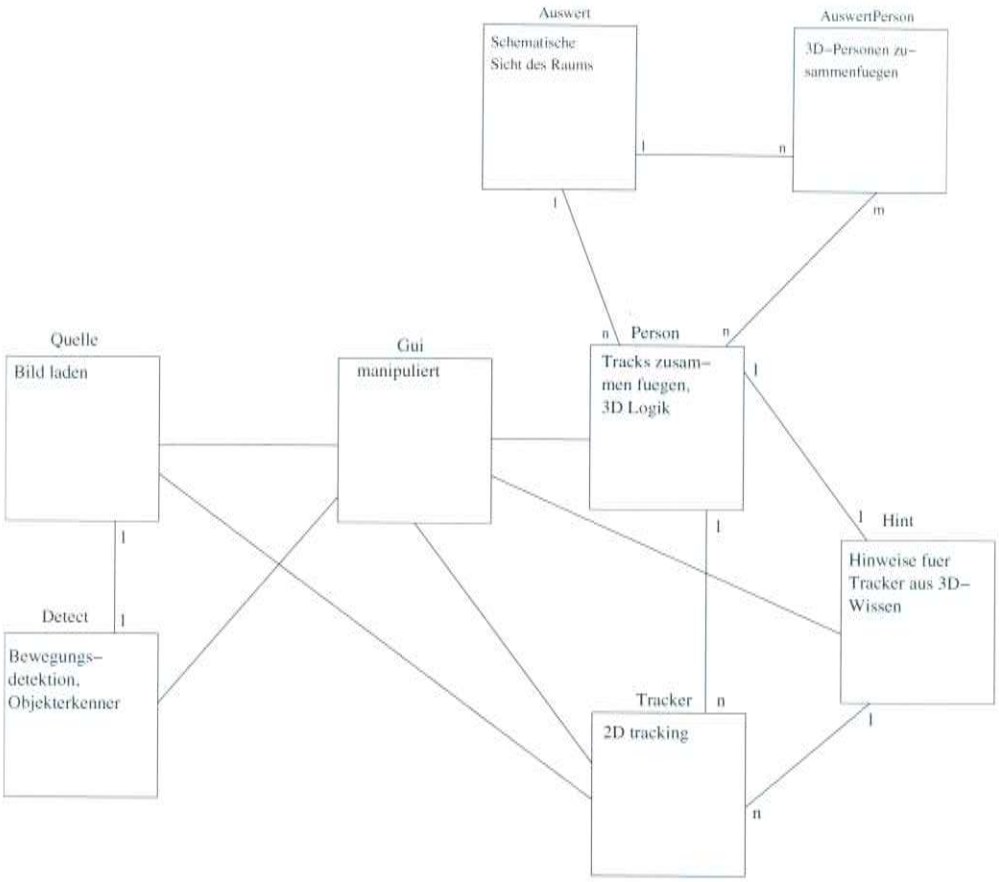


Abbildung 17: Grober Überblick über Design der Implementierung. Die Linien stellen eine Interaktion oder Enthaltungseigenschaft dar.

aber das Diagramm unübersichtlich machen.

Quelle Es gibt ein Quellenobjekt für jede Kamera. Im verwendeten Raum normalerweise vier. Das Quellenobjekt sorgt dafür, dass das aktuelle Bild der Kamera eingelesen wird und kameraspezifische Parameter, wie Kalibrierungsdaten, gespeichert werden. Die anderen Objekte können dadurch einfach auf die Bilddaten zugreifen.

Detect Jedes Quellenobjekt hat sein Detectobjekt, das mehrere Algorithmen auf die Bilddaten anwendet. Dazu gehören ein Bewegungsdetektor (siehe 3.1), sowie mehrere Objekterkenner (siehe 3.4).

Person Sobald ein 2D-Track gefunden wird, wird ein Personenobjekt angelegt. Im Personenobjekt ist die ganze 2D- und 3D-Logik, wie beispielsweise die Behandlung von Tracks (siehe 4.2, 4.3) oder das Kreditsystem (siehe 4.1.4) implementiert.

Tracker Für alle potentiellen 2D-Tracks einer Personen wird je ein Trackerobjekt angelegt, also bei vier Kameras und drei Haarkaskaden insgesamt zwölf Stück. Das jeweilige Trackerobjekt führt das eigentliche 2D-Tracken mittels Meanshift (siehe 3.2) durch. Das Objekt speichert alle dafür nötigen Daten, wie beispielsweise das Farbhistogramm.

Hint Jedes Personenobjekt hat ein Hintobjekt, das versucht 2D-Tracks mittels Triangulation für fehlende Kameras vorherzusagen, sobald eine Person 3D-Koordinaten hat (siehe 4.3.5).

Auswert Das Auswertobjekt stellt den Raum schematisch dar und zeichnet die Position von 3D-Personen und der Grundwahrheit ein (siehe 4.4).Das Auswertobjekt sorgt für die Erstellung und Löschung von Ausgabepersonen (in der Implementierung AuswertPerson genannt) und z.B. die Einhaltung der Wände.

GUI Das GUIobjekt stellt ein paar Möglichkeiten zur Manipulation der Objekte zur Verfügung. Dadurch können bestimmte Situationen getestet werden, um beispielsweise Schwellwerte entsprechend justieren zu können.

Eine detaillierte Beschreibung aller Klassen und Funktionen liefert die Referenzdokumentation, die mittels `doxygen 1.4.3` erstellt wurde. Da diese über 90 Seiten umfasst, wird sie hier nicht mitgeliefert.

5.4 Schwellwerte

Es gibt an vielen Stellen des Programms Schwellwerte, aufgrund derer eine Entscheidung getroffen wird. Die meisten wurden in die Datei `config.h` als Konstanten ausgelagert. Ein Auswahl wichtiger Schwellwerte liefert die folgende Tabelle.

Tabelle 2: Schwellwerte

Name	Wert	Beschreibung
`threshold_merge_2d`	50	Zusammenfügen von 2D-Tracks im Umkreis von x Pixeln zulassen
`threshold_split_2d`	75	Und ab x wieder auftrennen
`threshold_merge_3d`	160.0	Falls Triangulationsfehler kleiner, Zusammenfügen erlaubt
`threshold_split_3d`	200.0	Ab hier wieder auftrennen
`threshold_overlap_same`	60%	Falls gegenseitige Überlappung größer als x, dann ist es dasselbe Rechteck
`threshold_overlap_caption`	40%	Ab diesem Wert einer Überlappung nicht einlernen, außer groß genug für Gleichheit
`threshold_runaway`	30.0	Mindestabstand von Meanshift beim Einlernen des Tracks, damit Track gelöscht wird
`threshold_credit_start`	30	Ein neuer Track bekommt soviel Kredit
`threshold_credit_max`	50	Obergrenze für Kredit eines Tracks
`threshold_center_min`	150.0	Der Bauchnabel darf nicht tiefer sinken (in mm)
`threshold_center_max`	1250.0	Und nicht höher als dieser Wert

Abbildung 18: CAM1 zeigt ein neueres Bild als CAM2 bis CAM4

5.5 Probleme bei der Aufnahme

Da jede Kamera an einen eigenen Rechner angeschlossen ist, ist die richtige Synchronisation wichtig. Wie in Beispielbild 18 erkennbar, ist das Bild in CAM1 neuer als in den anderen Kameras. Die rot gekleidete Person sitzt in CAM1 bereits, während sie in den anderen Kameras noch steht. Solange reines 2D-Tracken durchgeführt wird, spielt das noch keine Rolle. Sobald 3D-Koordinaten berechnet werden, können korrespondierende Punkte einen großen Triangulationsfehler haben. Deshalb kann es passieren, dass insbesondere eine laufende Person aufgetrennt wird, obwohl sie korrekt getrackt wird.

Ein weiteres Problem ist hohe Last auf einem oder mehreren der Server. Dadurch kommt es zu Aussetzern bei der Bildaufnahme. Es sieht dann so aus, als ob eine Person eine zeitlang stehen bleiben und dann plötzlich "springen" würde. Dadurch steigt die Chance, dass ein 2D-Tracker eine Person verliert. Die Gesamterkennungsrate des Systems sinkt.

Bei der Aufnahme sollte also darauf geachtet werden, dass solche Fehler möglichst nicht auftreten. Das Synchronisationsproblem kann nachträglich manuell behoben werden. Bei fehlenden Bildern muss die Aufnahme noch einmal durchgeführt werden.

5.6 Verwendung der Software

Ein typischer Aufruf der Trackersoftware sieht folgendermaßen aus:

```
./tracker -c ./calib_05/ -p /data/videos/Cut_2005-04-15_A/ \
-l 10 -r -u
```

Die Parameter `-c` und `-p` müssen immer benutzt werden, da hier die richtigen Kamerakalibrationsdaten und eine gespeicherte Videosequenz angegeben wird. Mit `-u` wird die Oberkörperhaarkaskade angeschaltet. Ein Übersicht über einige wichtige Parameter gibt folgende Tabelle:

Tabelle 3: Kommandozeilenparameter des Trackers

Parameter		Beschreibung
-h	–help	Hilfetext anzeigen
-c	–calpath	Pfad zu Kalibrierungsdaten
-p	–campath	Pfad zu Videodaten
-a	–cams	Anzahl verwendeter Kameras (1 bis 4)
-r	–run	Niemals anhalten, wie z.B. bei neue Person erstellen
-s	–start	Starte bei Framenummer #
-l	–learn	Die ersten x Frames sind ein leerer Raum, damit der Bewegungsdetektor lernen kann
-f	–face	Gesichtskaskade verwenden
-u	–upper	Oberkörperkaskade verwenden
-b	–full	Ganzkörperkaskade verwenden
-t	–truth	Mit ground truth eine perfekte Oberkörperkaskade simulieren

6 Experimente und Resultate

Mithilfe von eigens aufgenommenen Entwicklungsdaten wurde der Tracker entwickelt und optimiert. Zusätzliche Testdaten dienen der Verifikation, dass die eingesetzte Methode funktioniert. Der Tracker wird mit bestimmten Parametern auf diesen Aufnahmen ausgeführt. Im Wesentlichen kann hierbei die Verwendung von Haarkaskaden variiert werden. Die Testdaten sind Aufnahmen im "intelligenten Raum" des Instituts mit drei bis vier Personen. Damit der Bewegungsdetektor einen leeren Raum lernen kann, ist bei allen Aufnahmen der Raum am Anfang leer und die Personen kommen durch die Tür herein. Der Tracker stellt für jedes Bild der Aufnahme eine Hypothese auf, wieviele Personen er gerade an welcher Stelle gefunden hat.

6.1 Grundwahrheit und Fehlerarten

Alle Aufnahmen sind durch einen Menschen markiert worden. Die so gewonnenen Kopfpositionen werden als *Grundwahrheit* bezeichnet. Die Position aller Menschen im Raum ist somit hochpräzise und vertauschungssicher bekannt. Daran muss sich ein Tracker messen. Da es viel Arbeit wäre, jedes Bild der Aufnahmen zu markieren, wurde nur etwa jedes 15. Bild markiert. Ein Tracker könnte also jedes 15. Bild richtige Daten liefern und dazwischen ständig falsche Hypothesen aufstellen – das ist aber recht unwahrscheinlich. In allen Aufnahmen ist die Framerate 15fps. Es gibt also eine Grundwahrheit pro Sekunde.

Die Grundwahrheit besagt, dass sich Person P_1 an einer bestimmten Position befindet, da sie beispielsweise durch die Tür hereingekommen und in mindestens zwei Kameras sichtbar ist. Falls der Tracker die Person ebenfalls erkannt hat und 3D-Koordinaten dafür berechnet, gibt es immer eine Abweichung zur Grundwahrheit. Diese wird wie folgt berechnet:

$$abstand = \sqrt{(x_{truth} - x_{tracker})^2 + (y_{truth} - y_{tracker})^2}$$

Es handelt sich also um den euklidischen Abstand der Position, die der Tracker ermittelt hat, zur Grundwahrheit, wobei nur die Entfernung in der Ebene berücksichtigt wird.

Falls der Tracker die Person nicht erkannt hat, handelt es sich um einen sogenannten *miss*. Ein solches Verpassen von Personen kann pro Person aufgeschlüsselt werden, da bekannt ist, welche Person nicht gefunden wurde.

Es kann auch der umgekehrte Fall auftreten, dass mehr Personen gefunden

werden, als es in Wirklichkeit gibt. Solche überzähligen Personen sind wie ein miss ebenfalls ein Trackerfehler und werden als *false positive* bezeichnet.

Eine weitere Fehlermöglichkeit ist das Vertauschen von Personen, auch *mismatch* genannt. Hier wird einer Person vom Tracker plötzlich ein anderer Index zugewiesen. Angenommen, es werde gerade eine Person getrackt und diese habe den Index 5 zugewiesen bekommen. Nun kommt es zu einem Trackerfehler und die Person 5 wird gelöscht. Ab jetzt handelt es sich pro Grundwahrheit um einen miss, da der Tracker keine Person mehr liefert. Der Tracker lerne die Person neu ein und weise ihr den Index 6 zu. Es werden keine misses mehr gezählt, dafür aber eine Vertauschung. Bei einer Verwechslung von zwei Personen, wenn also bei P_1 mit Index 5 und P_2 mit Index 8 vom Tracker die Indizes vertauscht werden, werden zwei Vertauschungen gezählt, je eine für P_1 und P_2.

6.2 Zuordnung von Grundwahrheit zu Ausgabehypothese

Bei einem Einpersonentracker ist die Bewertung der Trackerleistung relativ einfach. Entweder hat der Tracker die Person gefunden, falls es sie gibt, oder nicht. Es kann also der Abstand zwischen der Grundwahrheit und der Hypothese des Trackers berechnet werden und das Bewertungsproblem ist gelöst.

Bei einem Mehrpersonentracker ist es nicht offensichtlich, welche Grundwahrheit (kurz: gPerson) mit welcher Ausgabehypothese (kurz: tPerson) korrespondiert. Zudem sollte ab einem bestimmten Abstand davon ausgegangen werden, dass der Tracker die Person nicht mehr trackt. Wenn der Tracker beispielsweise zwei Meter neben der eigentlichen Person einen Track liefert, hat er die Person verloren.

Es wird daher folgender Algorithmus verwendet:

Zuordnung finden Versuche für gPersonen eine passende tPerson zu finden. Es werden alle Möglichkeiten der Zuordnung betrachtet, bei dem der Abstand einen bestimmten Schwellwert unterschreitet. Das Minimum ist die erste Zuordnung von gPerson zu tPerson. So wird weiter verfahren, bis entweder alle gPersonen eine passende tPerson gefunden haben oder gPersonen leer ausgehen, weil in der Zuordnungsmatrix keine Möglichkeiten mehr übrig sind.

Auswertung In der Bewertungsphase wird für alle Personen-Hypothesen-Paare der aktuelle Abstand ausgerechnet und gespeichert. Es wird wei-

terhin gezählt, welche gPersonen leer ausgegangen sind (misses) und welche tPersonen überzählig sind (false positives).

Zuordnung lösen Jetzt werden die Daten des nächsten markierten Zeitpunkts geladen. Das ist etwa 15 Bilder später. Zuerst wird geprüft, ob es Paare gibt, deren Abstand den Schwellwert überschreitet. Diese Zuordnungen werden gelöscht.

Nächste Iteration Danach kommt wieder die erste Phase, die versucht, jeder freien gPerson eine passende tPerson zuzuordnen. Sollte sich der Index einer tPerson gegenüber dem letzten bekannten Index einer gPerson geändert haben, handelt es sich um eine Vertauschung.

In Pseudocode sieht das so aus:

```
while( daten ) {
  loeseZuordnung();
  findeZuordnung();
  auswertung();
}
zeigeZusammenfassung();
```

Die Abweichung einer tPerson zu einer gPerson ist beim "Zuordnung lösen" mit einem Schwellwert von 500.0 mm begrenzt.

6.3 Metriken

Um die verwendeten Metriken zur Bewertung der Trackerleistung einzuführen, seinen folgende Variablen definiert:

m_T Die Anzahl der misses von allen Personen.

v_T Die Anzahl der Vertauschungen (mismatches) von allen Personen.

f Die Anzahl an überzähligen Personen (false positives).

g_i Die Anzahl der Zeitpunkte, an der P_i laut Grundwahrheit sichtbar ist.

g_T Die Summe aller g_i: $\sum_i g_i$

s_i Die Anzahl der Zeitpunkte von P_i, für welche eine Zuordnung der Trackerhypothese zur Grundwahrheit existiert.

s_T Die Summe aller s_i: $\sum_i s_i$

d_{it} Der Abstand der Trackerhypothese zur Grundwahrheit von Person P_i zum Zeitpunkt t. Misses werden nicht berücksichtigt.

Anhand dieser gemessenen Zahlen lassen sich zwei einfache Metriken zur Trackerbewertung definieren, die *tracking precision* und die *tracking accuarcy*. Die *tracking precision* ist der gewichtete Durchschnitt des Abstands aller Personen zu ihren Hypothesen, an Zeitpunkten, an denen eine Zuordnung möglich ist. Er kann folgendermaßen aus den eben definierten Variablen ausgerechnet werden:

$$trackingprecision = \frac{\sum_{i,t} d_{it}}{s_T}$$

Das Verhältnis der *misses* werden relativ zur Grundwahrheit angegeben. Gleiches gilt für *mismatches* und *falsepositives*.

$$misses = \frac{m_T}{g_T}$$

$$mismatch = \frac{v_T}{g_T}$$

$$falsepositive = \frac{f}{g_T}$$

Die *tracking accuracy* wird in Prozent angegeben.

$$trackingaccuracy = 1 - \frac{m_T + v_T + f}{g_T}$$

6.4 Ergebnisse auf Entwicklungsdaten

Es gibt mehrere verschiedene Haarkaskadentypen:

full Eine Ganzkörperhaarkaskade.

upper Eine Oberkörperhaarkaskade.

faceA Eine Gesichtshaarkasade.

faceB Eine weitere Gesichtshaarkaskade, die speziell für den "intelligenten Raum" trainiert wurde. Vorgabe war hierbei, dass die false-negative-Rate möglichst gering sein soll. Diese Haarkaskade findet also relativ oft ein Gesicht, allerdings ist die false-positive-Rate relativ hoch.

truth Hierbei handelt es sich um **keine** Haarkaskade, sondern die Grundwahrheit wird verwendet, um eine Oberkörperhaarkaskade zu simulieren. Die Größe des Rechtecks wird, wie in 4.3.5 beschrieben, mittels einer Standardgröße geschätzt. Mittels dieser "Haarkaskade" kann also abgeschätzt werden, wie gut der Tracker funktionieren würde, falls die benutzten Haarkaskaden sehr zuverlässig arbeiten. Da die Grundwahrheit nur jedes 15. Bild angegeben wurde, hat diese "Haarkaskade" eine Rate von 1 Treffer pro Sekunde.

Die Haarkaskaden können fast beliebig kombiniert werden, deswegen gibt es bis zu 11 Durchläufe pro Aufnahme. Unabhängig davon läuft natürlich die truth-Methode, da sie nur der Referenz dient.

Es gibt drei Aufnahmen (A, B und C), die für die Entwicklung und Optimierung des Trackers verwendet wurden. Die Aufnahmen haben die Länge von 2:05, 1:30 und 1:16 Minuten.

In der Tabelle 4 wird jeder mögliche Durchlauf dargestellt, wobei alle Aufnahmen gewichtet zusammengefasst sind. Der Durchlauf mit der besten tracking accuracy ist markiert.

Tabelle 4: Zusammenfassung pro Methode

	tracking precision	misses	mismatch	f.p.	tracking accuracy
full	151	41.9%	0.5%	5.1%	52.5%
upper	185	5.6%	1.5%	39.4%	53.6%
faceA	196	39.4%	1.2%	15.9%	43.6%
faceB	200	17.1%	4.1%	72.0%	6.9%
full+upper	**168**	**5.6%**	**2.4%**	**36.1%**	**55.9%**
full + faceA	161	30.1%	1.9%	33.6%	34.5%
full + faceB	181	14.0%	4.7%	62.5%	18.8%
upper + faceA	187	4.6%	2.5%	48.8%	44.1%
upper + faceB	170	16.6%	4.6%	96.1%	-17.2%
full + upper + faceA	205	11.9%	2.6%	56.6%	28.9%
full + upper + faceB	186	13.7%	4.4%	115.7%	-33.8%
truth	168	5.7%	0.5%	5.4%	88.3%

Bei den verschiedenen Aufnahmen hat sich heraus gestellt, dass es keine Haarkaskade gibt, die immer die beste ist. Die *full*-Haarkaskade erreicht eine hohe Präzision von bis zu 14cm. Allerdings dauert es relativ lange, bis die Haarkaskade überhaupt eine Person erkennt. Deswegen ist die miss-Rate

vergleichsweise hoch.

Die *upper*-Haarkaskade arbeitet am zuverlässigsten. Sie findet relativ schnell die Personen und hat eine geringe false-positive-Rate. Allerdings ist das in den Ergebnissen nicht so deutlich zu sehen, da die false-positive-Rate des Trackers je nach Aufnahme hoch sein kann.

Die Haarkaskade $faceA$ hat ähnlich wie $full$ eine hohe miss-Rate, da $faceA$ nur relativ selten ein Gesicht findet. Die false-positive-Rate ist vergleichsweise moderat. Da die Grundwahrheit den Mittelpunkt des Gesichts markiert, ist zu erwarten, dass die Gesichthaarkaskade die beste Präzision liefert. Allerdings ist das nicht der Fall, im Gegenteil, diese Haarkaskade hat mit z.T. 22cm die schlechteste Präzision. Da ein Gesicht häufig nicht stabil im Gesicht der betreffenden Person bleibt, sondern im Oberkörperbereich wandert, ist diese relativ schlechte Präzision verständlich.

Die andere Gesichtshaarkaskade $faceB$ liefert sehr oft ein Gesicht, allerdings handelt es sich hierbei häufig um einen false positive. Der Tracker kann zwar viele der false positives der Haarkaskade korrekt entfernen, aber er liefert mit dieser Haarkaskade dennoch eine hohe false-positive-Rate. Eine hohe false-positive-Rate einer Haarkaskade führt also zu einer höheren false-positive-Rate des Trackers, wie deutlich in den Ergebnissen zu sehen ist.

Bei den Kombinationen der Kaskaden schneidet $full + upper$ häufig relativ gut ab. In den einzelnen Aufnahmen ist diese Kombinationen zwar nie die beste, aber sie liefert immer gute Ergebnisse. Die Kombination $upper + faceA$ ist ähnlich beständig, hat aber sowohl eine schlechtere Präzision als auch Genauigkeit.

Alle Haarkaskaden liefern eine durchschnittliche Abweichung von etwa 15 bis 20cm, was ein guter Wert ist. Deutlich ist zu sehen, dass die optimale *truth*-"Haarkaskade" eine hohe Genauigkeit liefert. Die $full + upper$ Methode hat eine vergleichbare Abweichung, aber eine höhere false-positive-Rate. Deswegen ist die Genauigkeit etwas schlechter.

6.5 Details

Abbildung 19 ist eine genaue Darstellung des Trackingverhaltens. Die x-Achse gibt die Bildnummer der Videodaten an, die linke y-Achse die Abweichung eines Tracks von der Grundwahrheit in mm. Die rechte y-Achse die Anzahl an misses bzw. false positives. Vertauschungen sind nicht eingezeichnet.

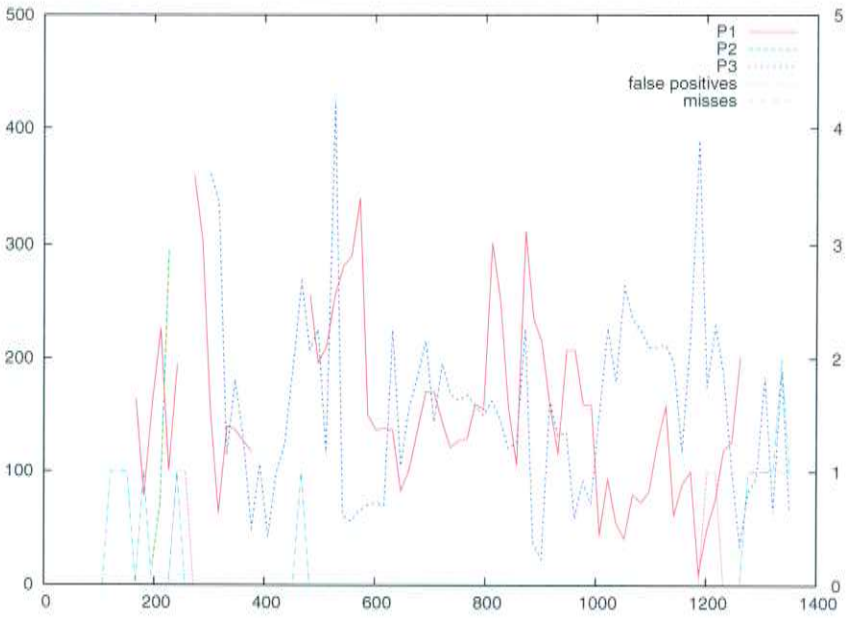


Abbildung 19: Verlauf des Trackens bei Aufnahme B mit Oberkörperhaarkaskade

Die grüne Kurve stellt die Person P_2 dar. Diese betritt den Raum und verlässt zielstrebig den Erfassungsbereich von drei Kameras. Deswegen wird sie nur am Anfang bei etwa Bild #200 getrackt.

Die rote und blaue Kurven sind die Personen P_1 und P_3. Diese beiden Personen sind fast die gesamte Aufnahme über sichtbar. Bei etwa Bild #160 betritt P_1 bereits den Raum, ist aber noch nicht erkannt. Deshalb ist die miss-Kurve (hellblau) auf 1 bis die Person erkannt wird. Bei etwa Bild #240 verschwindet P_2 aus dem Bild, aber der Tracker löscht die Person erst kurze Zeit später. Entsprechend ist dort die die false-positives-Kurve (rosa) auf 1.

P_1 setzt sich auf ein Sofa und ist nur noch in zwei Kameras sichtbar. Bei etwa Bild #400 wird P_1 in eine der beiden Kameras von P_3 verdeckt. Dadurch kommt es zu der Unterbrechung der roten Kurve. Bald darauf ist P_1 wieder sichtbar, es kommt zu einem kurzen miss und sie wird wieder getrackt.

Daraufhin setzen sich die beiden Personen an einen Tisch und sind in allen Kameras gut sichtbar. Entsprechend werden die beiden Personen stabil getrackt. Danach stehen sie auf und laufen im Raum umher, wobei sie sich zeitweise gegenseitig verdecken. Deshalb kommt es am Ende der Aufnahme zu Fehlern beim Tracken.

6.6 Ergebnisse auf Testdaten

Die Entwicklung des Trackers wurde mithilfe der Aufnahme A bis C durchgeführt. Um zu testen, wie gut der Tracker mit anderen, nicht optimierten Aufnahmen zurecht kommt, gibt es weitere Aufnahmen. Die Aufnahmen D bis F, die Testdaten, sind im selben Raum, aber mit anders gekleideten Personen aufgenommen worden. Es sind häufig vier Personen gleichzeitig sichtbar. Die Aufnahmen haben eine Länge von 1:22, 1:34 und 1:51 Minuten.

In der folgenden Tabelle wurden die Aufnahmen D bis F mit den Tracker analysiert. Hierbei wurden nicht alle möglichen Haarkaskadenkombinationen verwendet, sondern nur eine Auswahl. Bei den Entwicklungsdaten haben *full* und *upper* relativ gut abgeschnitten. Die Kombination *full* + *upper* war durchschnittlich am besten. Der Vollständigkeit halber ist auch *full*+*upper*+ *faceA* aufgeführt. Insbesondere die *faceB*-Methode wird weggelassen, da sie sich als unbrauchbar herausgestellt hat.

Tabelle 5: Zusammenfassung pro Methode

	tracking precision	misses	mismatch	f.p.	tracking accuracy
full	144	71.2%	0.8%	4.9%	23.1%
upper	171	26.8%	2.4%	26.2%	44.6%
full+upper	**169**	**29.8%**	**2.7%**	**28.9%**	**38.6%**
full + *upper* + *faceA*	199	19.5%	3.4%	80.3%	-3.2%
truth	163	24.0%	2.2%	23.4%	50.4%

Die tracking precision ähnelt stark der Abweichung, die bei den Entwicklungsdaten aufgetreten ist. Die tracking accuracy ist durchgängig schlechter geworden. Hierbei fällt auf, dass vor allem die miss-Rate gestiegen ist. Offenbar hat der Tracker größere Probleme, Personen zu tracken, wenn sich mehrere verdecken. Zudem zeigt sich hier, dass die simulierte Oberkörperhaarkaskade *truth* zwar perfekt weiß, wo die Person ist, aber die Schätzung des Oberkörpers Verdeckung nicht berücksichtigt. Aus diesem Grund ist die Genauigkeit bei Aufnahme D gering.

Bei den Entwicklungsdaten war *full* + *upper* die beste Methode, dicht gefolgt von *upper*. Bei den Testdaten dagegen ist *upper* marginal besser als *full* + *upper*. Im Gegensatz zu den Entwicklungsdaten ist die *truth*-Methode nicht wesentlich besser. Der limitierende Faktor scheint also nicht optimal angepasste Schwellwerte und Heuristiken zu sein. Die Fehler der Haarkaskaden spielen nur eine untergeordnete Rolle.

Die Testdaten unterscheiden sich von den Entwicklungsdaten in mehreren Details. Der Tisch ist anders angeordnet, es sind immer vier Personen sichtbar, anstatt zwei bis drei. Da die Leistung des Trackers auf den Aufnahmen vergleichbar zu der Leisung auf den Entwicklungsdaten ist, ist der verwendete Ansatz für den Tracker offenbar sinnvoll.

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

Der in dieser Arbeit vorgestellte Tracker kann mehrere Personen, in den Experimenten bis zu vier, mit hoher Präzision und geringer Fehlerrate tracken.

Erreicht wird das durch die Kombination von zwei Methoden, einem Objekterkenner und einem Bewegungsdetektor. Das eigentliche Tracken wird auf Farbmerkmalen mittels Meanshift durchgeführt. Zusätzlich gibt es mehrere Techniken, welche die Fehler, die das System macht, stark reduzieren. In Kapitel 4.5 ist eine Zusammenfassung aller wichtigen Verfahren gegeben.

Es wurde viel Wert auf eine möglichst zuverlässige automatische Initialisierung gelegt. Die Initialisierung sorgt zusammen mit mehreren Bereinigungstechniken für eine geringe false-positive-Rate. Die verwendeten Haarkaskaden finden schnell eine Person, die den Raum betritt. Dadurch ist die miss-Rate hervorragend. Eine weitere Fehlerart ist die Vertauschung. Durch den Einsatz von Techniken, wie die Heuristik zur Namensstabilität, wird diese Fehlerart stark reduziert.

Damit Meanshift präzise eine Person verfolgen kann, ist es nötig, dass das dazugehörige Farbhistogramm möglichst gut die Person repräsentiert. Dazu wird zur Laufzeit aus einer Menge von verschiedenen Bereinigungsverfahren das derzeit beste automatisch ausgewählt und verwendet. Zudem werden Farbhistogramme ständig adaptiert, wenn dies gefahrlos durchführbar ist.

Dieses System ist eine Mischung aus einem 2D- und einem 3D-Tracker. Das eigentliche Verfolgen findet auf reiner 2D-Basis mittels Meanshift statt, aber die gewonnen Informationen werden so kombiniert, dass das System 3D-Koordinaten der Personen, sog. Ausgabehypothesen kennt. Hierbei wird zweistufig verfahren. Zuerst werden 2D-Hypothesen innerhalb einer Kamera fusioniert, falls diese dieselbe Person verfolgen. Danach werden die Hypothesen mehrerer Kameras intelligent zu 3D-Hypothesen kombiniert, falls diese dieselbe Person verfolgen.

Der Tracker funktioniert zwar schon mit nur einer einzigen Kamera, liefert dann aber keine 3D-Koordinaten. Eine hohe Trackerleistung wird erst durch die Verwendung der komplexen 3D-Logik erreicht.

In den Experimenten wurde eine Genauigkeit von durchschnittlich 56%, bei manchen Aufnahmen bis zu 84% erreicht. Eine hohe Genauigkeit bedeutet eine geringe Fehlerrate für misses, mismatches und false positives. Gleichzeitig wurde eine eindrucksvolle durchschnittliche Abweichung von unter 17cm

erreicht. Bei den Testdaten, die zur Verifikation dienen, sinkt die Genauigkeit auf durchschnittlich 39%, die Abweichung bleibt gleich. Mit einer simulierten, sehr leistungsfähigen Haarkaskade sind 88% und 17cm erreichbar.

7.2 Ausblick

Es gibt mehrere Stellen des Systems, die durch sorgfältige Optimierung weiter verbessert werden können. Im einzelnen sind das folgende Punkte:

Farbbasierter Partikelfilter: In [12] wird nachgewiesen, dass ein Partikelfilter besser tracken kann als Meanshift. Möglicherweise sind diese Ergebnisse auf diesen Tracker übertragbar.

Bewegungsdetektor: Derzeit wird eine einfache Background Subtraction verwendet. Der Stauffersegmentierer oder ein anderes Segmentiererverfahren sollte so justiert werden können, dass er bessere Resultate als der derzeitige Bewegungsdetektor liefert.

Echtzeit: Die Haarkaskaden haben einen immensen Rechenbedarf, so dass der Tracker auf einem einzelnen Rechner ca. einen Faktor 30 langsamer als Echtzeit ist. Durch mehrere Maßnahmen sollte es möglich sein Echtzeit zu erreichen. Jede Kamera ist derzeit an einen eigenen Rechner angeschlossen. Deswegen könnte man den Tracker auf diese vier Rechner verteilen. Zudem ist es unnötig, dass die Haarkaskaden Suchfenster über das ganzen Bild legen. Eine Reduktion auf Bereiche, die vom Bewegungsdetektor als Vordergrund erkannt werden, ist vollkommen ausreichend.

Farbhistogramme wiederverwenden: Derzeit werden Farbhistogramme gelöscht, wenn eine Person eine Kamera verlässt. Sobald die Person wieder in der Kamera sichtbar ist, könnte versucht werden, ein altes Farbhistogramm wiederzuverwenden. Dadurch kann der Tracker diese Kamera schneller wieder benutzen.

3D-Kreditsystem: Falls alle Personen den Raum verlassen, kann es passieren, dass weiterhin Ausgabehypothesen existieren, die niemals gelöscht werden. Deshalb sollten auch 3D-Hypothesen, die nicht durch eine Haarkaskade bestätigt werden, irgendwann entfernt werden.

Deletion Zone Falls eine Person in Richtung Tür geht und die Bewegungsvorhersage vermuten lässt, dass die Person den Raum verlässt, sollte

über ein Löschen der Person nachgedacht werden. Das ergänzt sich evtl. gut mit einem 3D-Kreditsystem.

Verdeckung Die Handhabung von Verdeckung ist einfach gehalten, könnte allerdings deutlich umfangreicher gestaltet werden, um auf komplexe Verdeckungsszenarien zu reagieren.

Literatur

[1] Gary R. Bradski, *"Computer Vision Face Tracking For Use in a Perceptual User Interface"*. Microcomputer Research Lab, Santa Clara, CA, Intel Corporation, Intel Technology Journal, 1998.

[2] Neal Checka, Kevin Wilson, Vibhav Rangarajan, Trevor Darrell, *"A Probabilistic Framework for Multi-modal Multi-Person Tracking"*, Proceedings of Workshop on Multi-Object Tracking, 2003.

[3] Dorin Comaniciu and Peter Meer, *"Mean Shift: A Robust Approach Toward Feature Space Analysis"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 5, May 2002.

[4] Dirk Focken, *"Vision-based 3-D Tracking of People in a Smart Room Environment"*. Diplomarbeit, Interactive Systems Laboratories Universität Karlsruhe (TH), Nov. 2002.

[5] Yoav Freund and Robert E. Shapire, *"Experiments with a New Boosting Algorithm"*. Machine Learning: Proceedings of the Thirteenth International Conference, 1996.

[6] Ismail Haritaoglu, David Harwood and Larry S. Davis, *"W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People"*. Third Face and Gesture Recognition Conference, pp. 222–227, 1998.

[7] Michael Isard and Andrew Blake, *"Condensation – conditional density propagation for visual tracking"*. International Journal of Computer Vision 29(1), pp. 5–28, 1998.

[8] Jan Kratt, *"Audio-Visuelle Spracherkennung auf großem Vokabular"*. Diplomarbeit, Institut fur Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe (TH), März 2004.

[9] Rainer Lienhart and Jochen Maydt, *"An Extended Set of Haar-like Features for Rapid Object Detection"*. IEEE ICIP 2002, Vol. 1, pp. 900–903, Sep. 2002.

[10] Kai Nickel, *"3D-Tracking von Gesicht und Händen mittels Farb- und Tiefeninformation"*. Studienarbeit, Institut fur Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe (TH), Mai 2002.

[11] Kai Nickel, *"Erkennung von Zeigegesten basierend auf 3D-Tracking von Kopf und Händen"*. Diplomarbeit, Institut fur Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe (TH), März 2003.

[12] Katja Nummiaro, Esther Koller-Meier and Luc Van Gool, *"A Color-based Particle Filter"*. Image and Vision Computing, 2002.

[13] Fatih Porikli and Oncel Tuzel, *"Human Body Tracking by Adaptive Background Models and Mean-Shift Analysis"*. IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, März 2003.

[14] Yogesh Raja, Stephen J. McKenna, Shaogang Gong, *"Tracking and Segmenting People in Varying Lighting Conditions using Colour"*. Proceedings of the 3rd. International Conference on Face & Gesture Recognition, pp. 228, 1998.

[15] Edgar Seemann, *"Estimating Head Orientation with Stereo Vision"*. Diplomarbeit, Interactive Systems Labs, Universität Karlsruhe (TH), November 2003.

[16] Chris Stauffer and W.E.L. Grimson, *"Adaptive background mixture models for real-time tracking"*. CVPR, 1998.

[17] Hai Tao, Harpreet Sawhney and Rakesh Kumar, *"A Sampling Algorithm for Tracking Multiple Objects"*. Proceedings of the International Workshop on Vision Algorithms: Theory and Practice, pp. 53–68, 1999.

[18] Paul Viola and Michael Jones, *"Rapid Object Detection using a Boosted Cascade of Simple Features"*. Accepted Conference On Computer Vision And Pattern Recognition, 2001.

[19] Michael Voit, *"Visuelle Schätzung der horizontalen Kopfdrehung in Multikameraumgebungen"*. Diplomarbeit, Institut fur Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe (TH), Dezember 2004.

[20] Wikipedia, *"Kalman-Filter"*, http://de.wikipedia.org/wiki/Kalman-Filter, http://en.wikipedia.org/wiki/Kalman_filter, Juli 2005.

[21] Wikipedia, OpenCV, *"Korrelation"*, http://de.wikipedia.org/wiki/Korrelation, opencv-0.9.6/doc/ref/opencvref_cv.htm#decl_cvCompareHist, Juli 2005.

[22] Christian A. Wojek, *"Visuelle Personenverfolgung mit Partikelfiltern"*. Studienarbeit, Institut fur Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe (TH), April 2004.

[23] Christopher Wren, Ali Azarbayejani, Trevor Darrell, Alex Pentland, *"Pfinder: Real-Time Tracking of the Human Body"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 19, no 7, pp. 780–785, July 1997.