Tracking der artikularen Bewegung des Oberkörpers in Stereobildfolgen

Diplomarbeit

von

Julius Ziegler

30. September 2005

Betreuer:
Prof. Dr. Alexander Waibel
Dipl.-Inform. Kai Nickel
Dr.-Ing. Rainer Stiefelhagen
Ich versichere hiermit wahrheitsgemäß, die Arbeit selbständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderung entnommen wurde.

[Unterschrift]

Karlsruhe, den 30. September 2005
# Inhaltsverzeichnis

1 Einleitung ........................................ 7
   1.1 Motivation ........................................ 8
   1.2 Verwandte Arbeiten ................................ 8

2 Grundlagen ........................................ 11
   2.1 Stereobildverarbeitung ............................ 11
      2.1.1 Stereogeometrie ............................... 11
      2.1.2 Herstellen von Korrespondenzen .............. 12
      2.1.3 Vorverarbeitung und Rektifizierung .......... 13
      2.1.4 Nachbearbeitung der Disparitätenbilder .... 14
      2.1.5 Räumliche Kalibrierung mehrerer Kameras ... 14
   2.2 Kalmanfilterung .................................... 16
      2.2.1 Allgemeines Rahmenwerk ....................... 16
      2.2.2 Einfacher Kalmanfilter ....................... 19
      2.2.3 Erweiterter Kalmanfilter (EKF) ............... 19
      2.2.4 Unscented Kalman Filter (UKF) ............... 21
   2.3 Iterative Closest Point (ICP) .................... 25
   2.4 kd-Bäume .......................................... 26
   2.5 Kinematische Modellierung artikulärer Objekte .... 29

3 Trackingalgorithmus ............................... 31
   3.1 Segmentierung und weitere Vorverarbeitung ....... 31
      3.1.1 Vordergrundsegmentierung auf Disparitätenbildern .... 31
      3.1.2 Filterung der 3D-Daten ........................ 34
   3.2 Tracking artikulärer Objekte in Punktwolken ....... 35
      3.2.1 Systemmodell .................................... 36
      3.2.2 Messmodell und Generierung des Messvektors .... 37
      3.2.3 Ablauf der Filterung ........................... 41
      3.2.4 Kollisionsvermeidung ........................... 45

4 Experimente und Ergebnisse ....................... 47
   4.1 Datensammlung und Erstellung der Referenztrajektorien .... 47
   4.2 Parametersatz und Tuning ........................... 49
Kapitel 1
Einleitung


1.1 Motivation


Im Rahmen des CHIL-Projektes soll das System als Komponente in einem multimodalen Raum zum Einsatz kommen, dies ist ein Raum, der „intelligent“ auf das Verhalten seiner Benutzer reagiert. Auf der Basis des Tracking-Systems können neuartige Benutzeroberflächen zur Interaktion mit dem Raum und seinen Komponenten implementiert werden. Weiterhin soll der Raum in der Lage sein, ein vollständiges Protokoll der in ihm stattfindenden Ereignisse zu erstellen, hierzu gehören auch Bewegung, Körperhaltung und Gestikulation der Personen, die sich in ihm befinden.

1.2 Verwandte Arbeiten


Das in [Nie03] beschriebene Verfahren beispielsweise extrahiert basierend auf Farb- und Stereomerkmalen die Positionen von Kopf und Händen und kann durch
1.2. VERWANDTE ARBEITEN

ein nachgeschaltetes Klassifizierungssystem Zeigegesten erkennen.

Das System PFINDER (siehe [WADP97]) baut dynamisch ein zweidimensionales Modell einer Person auf. Es verfolgt und adaptiert hierzu die Statistik verschiedenfarbiger Bildregionen (Blobs) vor einem statischen Hintergrund und kann Kopf, Hände und Füße identifizieren.

In [RFZ05] wird ein System vorgestellt, das dynamisch Modelle (appearance models) für das Aussehen der einzelnen Körperteile erstellt. Die Modelle werden in Einzelbildern der Sequenz erstellt, in denen ein Klassifikator eine Person in einer von wenigen möglichen Initialisierungsposen erkannt hat. Mittels der erstellten Modelle können die Körperteile dann in nachfolgenden Einzelbildern wiedergefunden werden.


Deutscher, Blake und Reid modifizieren den Partikelfilteralgorithmus, indem sie ihn nach Art des Simulated Annealing stufenweise ablaufen lassen. Sie erzielen hiermit eine Reduktion der benötigten Partikelanzahl und wenden diesen sogenannten Annealed Particle Filter in [DBR00] auf das Problem des artikularen Körpertrackings an.

Lee, Cohen und Jung reduzieren effektiv die Dimensionalität des Zustandsraumes, indem sie die Ergebnisse von Merkmalserkennern (Kopf, Hände, Körperrachse) in die Partikelsrepräsentation mit einbringen [LCJ02].


Auch [UF04] beschreibt ein Verfahren zum artikularen Tracking auf mit Stereokameras gewonnenen Daten. Die Einpassung des Modells in die 3D-Daten wird hier als Optimierungsproblem formuliert und mit einem Minimierungsverfahren


Kapitel 2

Grundlagen

Dieses Kapitel beschreibt bewährte Verfahren und mathematische Konzepte, die in das in dieser Arbeit entwickelte System eingeﬂossen sind.

2.1 Stereobildverarbeitung


2.1.1 Stereogeometrie

Abbildung 2.1 zeigt den Aufbau und die Geometrie einer idealisierten Stereokamera in Draufsicht. Die beiden Objektive bei \(L\) und \(R\) weisen die Eigenschaften einer Lochkamera auf. Wichtig ist weiterhin die Parallelität der optischen Achsen und die Koplanarität der Bildebenen beider Kameras. Geringfügige Abweichungen von dieser Idealgemetrie können durch Rektifizierung ausglichen werden (siehe Abschnitt 2.1.3).

Den Abstand \(b\) der Objektive nennet man die \(Aufnahmebasis\). Die Bildpositionen eines beobachteten Punktes \(P\) unterscheiden sich unter den beschriebenen Voraussetzungen nur in der \(x\)-Koordinate, für die linke Kamera sei diese \(x_L\), für die rechte \(x_R\). Ihre Differenz \(d = x_L - x_R\) nennt man (horizontale) Disparität (siehe [Jä97]). Mithilfe des Strahlensatzes ergibt sich für die Entfernung \(z\) die
Abbildung 2.1: Stereometrie. Zwei Kameras mit den Brennpunkten L und R und gleicher Brennweite f beobachten Punkt P. Aus der Differenz der Bildkoordinaten \( x_L \) und \( x_R \) (Disparität) lässt sich die Entfernung \( z \) berechnen (entnommen aus [Nie03]).

Beziehung

\[
z = \frac{f \cdot b}{d}
\]

Legt man ein rechtshändiges Koordinatensystem so in den Brennpunkt der rechten Kamera, das die \( z \)-Achse senkrecht auf der Bildebene steht und die \( x \)-Achse nach rechts zeigt, erhält man als Koordinaten von \( P \)

\[
x = \frac{x_R \cdot z}{f}
\]

\[
y = \frac{y_R \cdot z}{f}
\]

Die Koordinaten (relativ zur Kamera) von \( P \) lassen sich also auch aus der Disparität \( d \) und den Koordinaten \((x_R, y_R)\) seines Bildpunktes mittels (2.1), (2.2) und (2.3) vollständig rekonstruieren.


2.1.2 Herstellen von Korrespondenzen

Zur Ermittlung der Korrespondenzen kommt hier nach [Poi03] die sogenannte *sum-of-absolute-differences-correlation*-Methode zum Einsatz. Dieser Ansatz funktioniert folgendermaßen:

1. Für jedes Pixel \( p \) im Referenzbild (hier: rechte Kamera)
2.1. STEREOBILDVERARBEITUNG

(a) Wähle um \( p \) eine quadratische Region \( R_{ref} \) mit gegebener Seitenlänge \( m \) (Korrelationsmaske)

(b) Vergleiche \( R_{ref} \) mit einer Anzahl Regionen \( R_k \) im anderen Bild. \( R_k \)
    werden entlang der Zeile von \( p \) gewählt, mit einem Versatz \( d_k \), den
    man von \( d_{min} \) bis \( d_{max} \) laufen lässt.

(c) Wähle das \( R_k \) mit der besten Passung. Der zugehörige Versatz \( d_k \)
    ist die ermittelte Disparität.

Die Güte der Passung zweier Regionen wird mit folgender Formel ermittelt:

\[
\sum_{i=-\frac{m}{2}}^{\frac{m}{2}} \sum_{j=-\frac{m}{2}}^{\frac{m}{2}} |R_{ref}[x + i][y + j] - R_k[x + i + d][y + j]|
\]

Die Wahl der minimalen Diparität \( d_{min} \) begrenzt den nutzbaren Entfernungsbe-
reich nach hinten (von der Kamera weg), die maximale Disparität \( d_{max} \) schränkt
ihn nach vorne ein. Während Schritt 1 (b) lässt man \( d_k \) in einfachsten Fall in
Pixelschritten von \( d_{min} \) bis \( d_{max} \) laufen, man kann aber auch Intensitätswerte im
Subpixelbereich interpolieren und so die Genauigkeit des Verfahrens erhöhen. Die
mit diesem Verfahren erstellten Bilder werden Disparitätenbilder genannt (siehe
Abbildung 3.2 auf Seite 33 für ein Beispiel).

2.1.3 Vorverarbeitung und Rektifizierung

Die TRICLOPS-Bibliothek führt als Vorverarbeitung eine Glättung der Rohbilder
mit einem Tiefpassfilter durch. Hierdurch wird niedrigfrequentes Rauschen des
Bildsensors entfernt. Außerdem wird vermieden, das nachgeschaltete Rektifizie-
zung (siehe unten) Aliasing-Artefakte hervorbringt. Optional kann der Stereover-
arbeitung eine Kantendetektion vorgeschaltet werden. Dies ist aus zwei Grün-
den vorteilhaft: Erstens wird die Verstärkung der Bildsensoren beider Kameras in
einem TRICLOPS-System getrennt geregelt und so der Gesamtheiligkeit der Bil-
der angepasst. Hierdurch kann es Helligkeitsunterschiede zwischen den Kameras
geben, was bei einer rein intensitätsbasierten Ermittlung der Korrespondenzen
zu Fehlern führen kann. Die Kantenbilder sind gegenüber Helligkeitsschwankun-
gen invariant. Zweitens kann mithilfe der Kantenbilder eine Validierung der er-
mittelten Disparitätenbilder durchgeführt werden, dies wird in Abschnitt 2.1.4
beschrieben.

Echte Stereokameras weichen durch Ungenauigkeiten in der Optik und der
Ausrichtung der Kameras von der in 2.1.1 beschriebenen Idealgeometrie ab. So
führen Linsenobjektive gegenüber einer Lochkamera im Allgemeinen zu einer
leicht verzerrten und dezentrierten Abbildung. Rektifizierung nennt man den Pro-
zess, der diese störenden Effekte ausgleicht. Hierzu ist eine genaue Kalibrierung
der Kameras nötig. Die für diese Arbeit verwendeten Stereokameras sind vom
Hersteller vorkalibriert. Zu den durch die Kalibrierung ermittelten Rektifizierungsparametern gehören Brennweite, Linsenzentrierung, radialsymmetrische Verzerrung sowie das Aspektverhältniss der Bildelemente des Sensorschips, weiterhin die Aufnahmebasis und die Abweichung der optischen Achsen von der Parallelität. Der Hersteller garantiert eine Genauigkeit der Kalibrierung von 0.05 Pixel.

2.1.4 Nachbearbeitung der Disparitätenbilder


2.1.5 Räumliche Kalibrierung mehrerer Kameras


Zur Kalibrierung der Kameras wurde auf Routinen der Softwarebibliothek OPENCV ([INT01]) zurückgegriffen. Der Ablauf des Verfahrens ist folgendermaßen: Ein Schachbrettmuster mit bekannter Dimensionierung wird so positioniert,

Abbildung 2.3: Fusion der 3D-Daten. Die weißen und schwarzen Punkte stammen von zwei Stereokameras, die sich ungefähr gegenüberstehen und eine Person mit ausgestreckten Armen beobachten. Durch die räumliche Kalibrierung konnten die Punkte aus den lokalen Kamerakoordinatensystemen in ein gemeinsames Weltkoordinatensystem transformiert werden.

### 2.2 Kalmanfilterung


#### 2.2.1 Allgemeines Rahmenwerk

Der Kalmanfilter ist ein Algorithmus zur Zustandsschätzung eines Systems, das sich nur durch indirekte und verrauschte Messungen beobachten lässt. Er bedient sich hierzu eingebrachten Wissens um die Dynamik des Systems und Annahmen über die Größe von Messfehlern. Man modelliert das System als zeitdiskretes, nichtlineares System mit stochastischen Störungen (siehe auch Abbildung 2.4):
\[ x[k+1] = f(x[k], u[k]) + v[k] \] (2.4)

\( k \) ist der diskrete Zeitindex. Die \textit{Systemgleichung} 2.4 beschreibt das Verhalten des Systems, indem sie den neuen Zustand \( x[k+1] \) funktional mit dem vorigen Zustand \( x[k] \) in Beziehung setzt, unter Berücksichtigung der gegenwärtigen Ansteuerung des Systems, \( u[k] \). Die Ansteuerung \( u[k] \) ist bei vielen Systemen unbekannt oder nicht vorhanden, so auch bei dem von uns betrachteten System „menschlicher Körper“. Sie wird uns im Rahmen dieser Arbeit nicht mehr begegnen. Die Funktion \( f(\cdot, \cdot) \) nennt man \textit{Systemmodell}. Die Vorhersage des Systemmodells ist mit einem additiven Fehler, dem \textit{Systemrauschen} \( v[k] \), belegt. Der Zustandsvektor \( x[k] \) selbst ist unbekannt, man erhält aber zu diskreten Zeitpunkten Messungen \( z_k \), die mit dem Zustand über folgende Gleichung in Zusammenhang stehen:

\[ z[k] = h(x[k]) + w[k] \] (2.5)

Das \textit{Messmodell} \( h(\cdot) \) beschreibt den funktionalen Zusammenhang zwischen dem Zustand \( x[k] \) und dem zugehörigen Messwert \( z[k] \). Die Messung ist mit dem Fehler \( w[k] \), \textit{Messrauschen} genannt, behaftet.

Der Kalmanfilter schätzt, wie noch gezeigt wird, die Statistik zweiter Ordnung eines Systems, also Mittelwert und Standardabweichung des Zustandes.

In realen Systemen muss man von Ungenaigkeiten bei der Beschreibung bzw. Modellierung ausgehen, beispielsweise, wenn eines der Modelle vereinfacht wurde, um es mathematisch fassbar zu machen. Die Rauschprozesse \( v[k] \) und \( w[k] \) fassen alle Abweichungen vom Modellverhalten zu additivem Rauschen zusammen. Sie werden beim klassischen Kalmanfilter, wie auch bei der hier verwendeten Variante UKF als weißes, unkorreliertes Rauschen, dessen Erwartungswert verschwindet, modelliert. Sei also

\[ Q[k] = \text{var}(v[k]) \] (2.6)

die \textit{Kovarianzmatrix des Systemrauschens}, und analog

\[ R[k] = \text{var}(w[k]) \] (2.7)

die \textit{Kovarianzmatrix des Messrauschens}.

Im Folgenden wird mit \( \hat{x}[i|j] \) der Schätzwert zur Zeit \( i \), der durch alle Messungen bis einschließlich Zeitpunkt \( j \) bedingt ist, bezeichnet:

\[ \hat{x}[i|j] = E \left[ x[j] | Z^i \right] \]

mit \( Z^i = \{ z[1], z[2], ..., z[j] \} \). Die bedingte Varianz dieser Schätzung bezeichnet man analog mit

\[ P[i|j] = E \left[ (x[i] - \hat{x}[i|j]) \{ x[i] - \hat{x}[i|j] \}^T | Z^i \right] \] (2.8)
Der Ablauf des Algorithmus ist wie folgt:

**Wiederhole:**

1. **Prädiktion:** ausgehend von dem optimalen Schätzwert \( \hat{x}[k|k] \) und seiner Fehlerkovarianz \( P[k|k] \) werden der beste (\textit{a priori}-) Schätzwert \( \hat{x}[k+1|k] \) und seine Fehlerkovarianzmatrix \( P[k+1|k] \) mit Hilfe des Systemmodells extrapoliert. \( P[k+1|k] \) enthält die Systemkovarianz \( Q \) als additiven Anteil. Man transformiert \( P[k+1|k] \) und \( \hat{x}[k+1|k] \) durch das Messmodell und erhält als Vorhersage für den Messwert \( z[k+1|k] \), mit der Varianz \( P_{zz}[k+1|k] \). Die Differenz zwischen vorhergesagtem und tatsächlichem Messwert nennt man Residuum (oder Innovation): \( \nu[k+1] = z[k+1] - \hat{z}[k+1|k] \). Die Varianz dieser Größe ist \( P_{\nu
u}[k+1|k] = P_{zz}[k+1|k] + R[k+1] \).

2. **Korrektur:** mit Hilfe der Messung \( z[k+1] \) wird der extrapolierte Wert \( \hat{x}[k+1|k+1] \) optimal zu \( \hat{x}[k+1|k+1] \) verbessert. Die Fehlerkovarianz \( P[k+1|k+1] \) des korrigierten Schätzwertes wird dabei minimiert. Die Korrekturgleichungen lauten:

\[
\begin{align*}
\hat{x}[k+1|k+1] &= \hat{x}[k+1|k] + K[k+1] \nu[k+1] \\
P[k+1|k+1] &= P[k+1|k] - K[k+1] P_{\nu\nu}[k+1|k] K[k+1]^T
\end{align*}
\]

Hierbei ist \( K[k+1] \) das Kalman-Gain. Es errechnet sich zu

\[
K[k+1] = P_{xz}[k+1|k] P_{\nu\nu}[k+1|k]^{-1}
\]

\( P_{xz}[k+1|k] \) ist die Kreuzkovarianzmatrix

\[
E \left[ \{x[k+1] - \hat{x}[k+1|k]\} \{z[k+1] - \hat{z}[k+1|k]\}^T \right] Z^k
\]

von \( x[k+1] \) und \( z[k+1] \) (Vergleiche 2.8).

Der beschriebene Algorithmus beschreibt das Kalmanfilter-Rahmenwerk in sehr allgemeiner Form. Zu klären bleibt, wie \( \hat{x} \) und seine Kovarianz \( P \) durch die funktionales Modelle propagiert und wie die Kreuzkovarianz \( P_{xz} \) ermittelt wird. Zu ermitteln sind also noch die Größen \( \hat{x}[k+1|k] \), \( P[k+1|k] \), \( \hat{z}[k+1|k] \), \( P_{zz}[k+1|k] \) sowie \( P_{xz}[k+1|k] \). Der Kalmanfilter und seine linearisierenden Varianten verfolgen hierzu jeweils unterschiedliche Ansätze, die in den folgenden Abschnitten erläutert werden.

Die asymptotische Komplexität eines Vorhersage-Korrekturzyklus des Kalmanfilters ist im allgemeinen Fall \( O(2nm^2) + O(m^3) + O(n^3) \) (nach [GGM97]). Hierbei ist \( m \) die Dimensionalität des Zustandsvektors, \( n \) die des Messvektors. Dies gilt auch für die im folgenden beschriebenen Varianten.
2.2.2 Einfacher Kalmanfilter


Die funktionalen Modelle aus (2.4) und (2.5) werden zu linearen Funktionen und man erhält als System- und Messgleichung

\[ x[k+1] = Fx[k] + v[k] \]
\[ z[k] = Hx[k] + w[k] \]

Die Ansteuerung \( u[k] \) wurde hier wie im folgenden aus oben genanntem Grund weggelassen. \( F \) und \( H \) nennt man Zustandsübergangsmatrix und Messmatrix. Prädiktion und Korrektur kann man in diesem linearen Fall durch einfache Vektorgleichungen ausdrücken. Der prädizierte Zustand ergibt sich zu

\[ \tilde{x}[k+1|k] = F\tilde{x}[k|k] \]

Der für die Prädiktion erwartete Fehler beträgt

\[ P[k+1|k] = FP[k|k]F^T + Q[k] \]

Analog wird \( \tilde{x}[k+1|k] \) durch das Messmodell propagiert:

\[ \tilde{z}[k+1|k] = H\tilde{x}[k+1|k] \]
\[ P_{zz}[k+1|k] = HP[k+1|k]H^T \]

Die Kreuzkovarianz ergibt sich zu

\[ P_{xz}[k+1|k] = P[k+1|k]H^T \quad (2.10) \]

Der Kalmanfilter ist im beschriebenen linearen Fall ein optimaler Schätzer in dem Sinne, das er die Wahrscheinlichkeit der Schätzung maximiert.

2.2.3 Erweiterter Kalmanfilter (EKF)

Bei nichtlinearen Modellen wird im einfachsten Fall durch Ableitung am aktuellen Systemzustandsvektor linearisiert (siehe auch Abbildungen 2.5 - 2.6). Für die Prädiktion des Zustandes und seines Fehlers ergeben sich dann

\[ \dot{x}[k+1|k] = f(\tilde{x}[k|k]) \]
\[ P[k+1|k] = J_f P[k|k] J_f^T + Q[k] \]
Abbildung 2.5: Unterwirft man eine normalverteilte Zufallsvariable einer nichtlinearen Transformation $f$, so ist sie im Allgemeinen nicht mehr normalverteilt. Linearisierende Kalmanfilter wie EKF und UKF approximieren die ersten beiden Momente der Verteilung der transformierten Variablen, also Mittelwert und Varianz.

Abbildung 2.6: Beim EKF wird die nichtlineare Transformation $f$ durch Ableitung am Mittelwert approximiert.
\[ J_f \] ist die Jacobi-Matrix von \( f \) an der Stelle \( \hat{x}[k|k] \). Ganz analog wird durch das Messmodell propagiert:

\[
\begin{align*}
\hat{z}[k+1|k] &= h(\hat{x}[k+1|k]) \\
P_{zz}[k+1|k] &= J_h P[k+1|k] J_h^T
\end{align*}
\]

\( J_h \) ist diesmal die Jacobi-Matrix von \( h \), ausgewertet an der Stelle \( \hat{x}[k+1|k] \). Die Kreuzkvarianz ist

\[
P_{xz}[k+1|k] = P[k+1|k] J_h^T
\]

(2.11)

Diese Form des Kalmanfilters heisst \textit{Erweiterter Kalmanfilter (EKF)}. Er wurde erstmals in [Sch70] von Stanley Schmidt vorgeschlagen und heisst nach ihm auch \textit{Kalman-Schmidt-Filter}. Die Nichtlinearitäten der Modelle werden hier bei der Propagierung der Varianzen nur bis zum Grade einer Taylorentwicklung erster Ordnung angenähert, bei großen Abtastzeiten oder stark nichtlinearen Modellen ist der EKF deshalb kein sehr verlässlicher Schätzer. Im nächsten Abschnitt wird eine Variante des Kalmanfilters vorgestellt, die Nichtlinearitäten besser approximiert als der EKF und bei der außerdem die mitunter komplizierter analytische Bildung der Jacobimatrizen nicht erforderlich ist.

### 2.2.4 Unscented Kalman Filter (UKF)

Der von Julier und Uhlmann in [JU97] beschriebene \textit{Unscented Kalmanfilter (UKF)} ist eine nichtlineare Variante des Kalmanfilters, die sich bei der Propagierung der stochastischen Zustandsvariablen durch die Mess- und Systemmodelle vom EKF wesentlich unterscheidet (Abbildung 2.7). Zum Einsatz kommt hierbei die sogenannte \textit{Unscented Transformation} (siehe [JU96]), mit der Varianz und

Sei $\mathcal{N}$ eine $n$-dimensionale Normalverteilung mit Kovarianzmatrix $P_{xx}$ und Mittelwert $\mu_x$. Man kann eine Stichprobenmenge aus $2n + 1$ Punkten generieren, die die gleiche Kovarianz aufweist. Hierzu wählt man die $2n$ Spalten der positiven und negativen Quadratwurzel der Matrix $(n+\kappa)P_{xx}$, $\kappa$ ist ein Skalierungsfaktor, seine Bedeutung wird weiter unten erläutert. Diese Punkte nennt man $\sigma$-Punkte $\{\sigma_1, \ldots, \sigma_{2n}\}$. War $\mathcal{N}$ nicht mittelwertfrei, so erhält man durch Addition des Mittelwertes $\mu$ zu jedem $\sigma_i$ eine Stichprobenmenge, die in Kovarianz und Mittelwert mit der ursprünglichen Verteilung übereinstimmt. Man nimmt zu dieser Menge noch den Mittelwert $\mu_x$ hinzu, dies ändert Mittelwert und Kovarianz nicht. Zusammenfassend generiert man die verschobenen $\sigma$-Punkte $\mathcal{X}_i$ folgendermaßen:

$$\sigma_i \leftarrow 2n \text{ Spalten oder Zeilen aus } \pm \sqrt{(n+\kappa)P_{xx}}$$

$$\mathcal{X}_0 = \mu$$

$$\mathcal{X}_i = \sigma_i + \mu$$

Um nun $\mathcal{N}$ einer nichtlinearen Funktion $g$ zu unterwerfen, transformieren zunächst die Punkte $\mathcal{X}_i$ einzeln:

$$\mathcal{Y}_i = g(\mathcal{X}_i).$$

Kovarianz $P_{yy}$ und Mittelwert $\mu_y$ der transformierten Verteilung $\mathcal{N}$ sowie die für die Kalmanfilterung erforderliche Kreuzkorrelation $P_{xy}$ erhält man nun durch Mittelung, wobei die $\mathcal{Y}_i$ noch unterschiedlich gewichtet werden:

$$\mu_y = \frac{1}{n+\kappa} \left\{ \kappa \mathcal{Y}_0 + \frac{1}{2} \sum_{i=1}^{2n} \mathcal{Y}_i \right\}$$

$$P_{yy} = \frac{1}{n+\kappa} \left\{ \kappa [\mathcal{Y}_0 - \mu_y] [\mathcal{Y}_0 - \mu_y]^\top + \frac{1}{2} \sum_{i=1}^{2n} [\mathcal{Y}_i - \mu_y] [\mathcal{Y}_i - \mu_y]^\top \right\}$$

$$P_{xy} = \frac{1}{n+\kappa} \left\{ \kappa [\mathcal{X}_0 - \mu_x] [\mathcal{Y}_0 - \mu_y]^\top + \frac{1}{2} \sum_{i=1}^{2n} [\mathcal{X}_i - \mu_x] [\mathcal{Y}_i - \mu_y]^\top \right\}$$

(2.17) und (2.18) kann man sich durch (2.8) beziehungsweise (2.9) plausibel machen.

Die von (2.12)-(2.17) durchgeführte Transformation heißt Unscented Transformation. Sie wird im folgenden mit dem Operator $\mathcal{U}$ bezeichnet:

$$\begin{pmatrix} \mu_y & P_{yy} \end{pmatrix} = \mathcal{U}(\mu_x, P_{xx}, g).$$
Wenn die Berechnung der Kreuzkorrelation mittels (2.18) erforderlich ist, schreiben wir $U_{KK}:

$$
\begin{pmatrix}
\mu_y \\
P_{yy} \\
P_{xy}
\end{pmatrix} = U_{KK}(\mu_x, P_{xx}, g).
$$

Mit der Unscented Transformation kann das Kalmanfilter-Rahmenwerk aus Abschnitt 2.2.1 gefüllt werden. Die Propagierung durch das Systemmodell im Prädiktionsschritt geschieht mit

$$
\begin{pmatrix}
\hat{x}[k+1|k] \\
\hat{P}[k+1|k]
\end{pmatrix} = U(\hat{x}[k|k], P[k|k], f)
$$

und anschließender Addition der Kovarianzmatrix $Q[k]$ des Systemrauschens:

$$
P[k+1|k] = \hat{P}[k+1|k] + Q[k]
$$

Es wird weiter durch das Messmodell propagiert, wobei auch die zur Berechnung des Kalmangains nötige Kreuzkorrelation $P_{xz}$ ermittelt wird:

$$
\begin{pmatrix}
\hat{z}[k+1|k] \\
\hat{P}_{zz}[k+1|k] \\
P_{xz}[k+1|k]
\end{pmatrix} = U_{KK}(\hat{x}[k+1|k], P[k+1|k], h).
$$

Die Kovarianzmatrix $R[k]$ des additiven Messrauschens muss noch addiert werden:

$$
P_{zz}[k+1|k] = \hat{P}_{zz}[k+1|k] + R[k].
$$

Hiermit stehen alle für den Korrekturschritt benötigten Größen zur Verfügung.

Der Skalierungsfaktor $\kappa$ beeinflusst die räumliche Verteilung der $\sigma$-Punkte und ihre Gewichtung. Durch geeignete Wahl von $\kappa$ ist es prinzipiell möglich, die $\sigma$-Punkte beliebig nah an den Mittelwert $\mu$ heranzubringen. Dies kann für bestimmte Funktionen $g$ erstrebenswert sein, da durch eine zu weitläufige Verteilung der $\sigma$-Punkte nicht-lokale Momente der Funktion übergewichtet werden, wie Abbildung 2.8 zeigt. Möchte man die $\sigma$-Punkte sehr nah um den Mittelwert verteilen (dies war für die in Abschnitt 3.2 beschriebenen Modelle erforderlich), muss $\kappa$ negativ gewählt werden. Leider kann dann nicht mehr garantiert werden, dass die transformierte Kovarianz positiv semidefini ist, was für die Kalmanfilterung aber erforderlich ist. Julier und Uhlmann schlagen in [JU02] eine Modifikation der Unscented Transformation vor, bei der sich die Skalierung der $\sigma$-Punkte besser kontrollieren lässt. Bei dieser sogenannten $Scaled$ $Unscented$ $Transformation$ wird der Grad der Skalierung durch einen zusätzlichen Parameter $\alpha$ bestimmt, der beliebig gewählt werden kann, bei garantiert positiver Semidefinitheit der transformierten Kovarianz.

Der UKF ist in den meisten praktischen Situationen ein besserer Schätzer als der EKF, da die Unscented-Transformation Nichtlinearitäten besser einführt als die Ableitung am Systemzustand beim EKF. Der entscheidender Vorteil, der den Ausschlag für den Einsatz des UKF in dieser Arbeit gab, ist die Tatsache, dass sein Entwurf nicht die Ableitung der Modellfunktionen erfordert. Dies vereinfacht den Filterentwurf, insbesondere bei komplizierten Mess- oder Systemmodellen, enorm.
Abbildung 2.8: Auswirkung der Skalierung bei der Unscented Transformation. Eine Gaussverteilung wird durch eine Sinusfunktion transformiert. Im oberen Bild sind die σ-Punkte $X_i$ weit auseinander skaliert, wodurch nicht-lokale Momente der Funktion übergewichtet werden. Die Skalierung im unteren Bild führt zu einer besseren Approximation der transformierten Verteilung. Zu beachten ist, dass bei Skalierung auch die Gewichtung der σ-Punkte verändert werden muss. Im oberen Bild wird $X_0$ stärker gewichtet als im unteren.
2.3 Iterative Closest Point (ICP)


Gegeben sei eine Menge Punkte $P = \{ \vec{p}_0, ..., \vec{p}_n \}$ (*Datenpunkte*). Diese soll rigide transformiert (positioniert) werden, und zwar so, dass sie sich in optimaler Ausrichtung an einer gegebenen Oberfläche $X$ (*Modellform*) befindet. Das Abstandsmaß zwischen einem Einzelpunkt $\vec{p}$ und der Modellform $X$ bezeichnet man mit $d(\vec{p}, X)$. $\vec{y}$ sei der Punkt auf $X$, der diesen kleinsten Abstand zu $\vec{p}$ hat, also $\vec{y} \in X$, so dass $d(\vec{p}, \vec{y}) = d(\vec{p}, X)$. Wir führen den Operator $\mathcal{C}$ ein, der zur Menge $P$ die nächsten Punkte $Y$ auf $X$ berechnet:

$$ Y = \mathcal{C}(P, X). $$

Die rigide Transformation $\theta$ sei die Registrierung der Punktmengen $P$ und $Y$, also die Transformation, die $P$ und $Y$ unter dem Kriterium kleiner Fehlerquadrate in optimale Deckung bringt. Sie lässt sich durch Einzelwertzerlegung (Single Value Decomposition, SVD, siehe [Aru87]) oder Repräsentation im Quaternionenraum ([Hor87]) in geschlossener Form berechnen. Sei $T$ der Operator, mit dem $\theta$ und der resultierende mittlere quadratische Fehler $d$ berechnet wird:

$$ \begin{bmatrix} \theta & d \end{bmatrix} = T(P, Y). $$

Der ICP-Algorithmus kann jetzt formuliert werden:

Wiederhole die folgenden Schritte bis zur Terminierung:

1. Die Iteration wird initialisiert, indem $P = P_0$, $\theta = id$ und $k = 0$ gesetzt werden.

2. Schritte a, b, c und d werden nun bis zur Konvergenz innerhalb der Toleranz $\tau$ iteriert:

(a) Berechne die nächsten Punkte: $Y_k = \mathcal{C}(P_k, X)$

(b) Berechne die Registrierung: $\begin{bmatrix} \theta_k & d_k \end{bmatrix} = T(P_0, Y_k)$

(c) Wende die Registrierung an: $P_{k+1} = \theta_k(P_0)$
(d) Terminiere, wenn die Änderung des kleinsten quadratischen Fehlers unter die gemäß der gewünschten Präzision gewählte Schwelle \( \tau \) fällt:
\[
d_k - d_{k-1} < \tau.
\]

Für diese Arbeit musste insbesondere der in Schritt (b) angewendete Operator \( \mathcal{T} \) angepasst werden, da die in der Literatur im Zusammenhang mit ICP beschriebenen Implementierungen dieses Operators nur für die Registrierung rigider Objekte geeignet sind. Abschnitt 3.2 beschreibt, wie das Unscented Kalmanfilter zum Einsatz kam, um auch die Registrierung artikulärer Objekte zu ermöglichen. Zur effizienten Implementierung des Operators \( \mathcal{C} \) wurde ein kd-Baum verwendet. Diese Datenstruktur wird im folgenden Abschnitt beschrieben.

### 2.4 kd-Bäume

Der kd-Baum (kd steht für \( k \)-dimensional) ist eine Baumstruktur zur Optimierung mehrdimensionaler Suchanfragen, die zuerst in [Ben75] beschrieben wurde. Im Rahmen dieser Arbeit wird er verwendet, um die Suche nächster Nachbarn im ICP-ähnlichen Teil des Algorithmus zu optimieren, außerdem bei der Implementierung der in Abschnitt 3.1.2 beschriebene Filterstufe. Dieser Abschnitt erklärt kurz den Aufbau von kd-Bäumen und den Ablauf der nächst-Nachbar-Anfrage.

Beim kd-Baum handelt sich um einen verallgemeinerten binären Suchbaum, der \( k \)-dimensionale Schlüssel verwendet. Ohne Beschränkung der Allgemeinheit wird angenommen, dass als Daten \( k \)-dimensionale Vektoren im Baum abgelegt werden sollen und gleichzeitig als Schlüssel verwendet werden. Das Einfügen in den Baum geschieht genauso wie beim einfachen binären Baum, man wählt aber zum Indizieren auf Ebene \( i \) des Baumes das \( i \) mod \( k \)-te Element des Schlüssels:

\[ p \]

sei der einzufügende Schlüssel. Setze initial als nächsten zu besuchenden Knoten \( n \) die Wurzel des Baumes. Iteriere nun bis zur Terminierung folgende Schritte:

- Setze den Diskriminator \( d = i \) mod \( k \). \( i \) ist die Tiefe von \( n \).

  - \( \tilde{q} \) sei der bei \( n \) gespeicherte Schlüssel. Vergleiche die \( d \)-ten Elemente \( q_d \) und \( p_d \) von \( \tilde{q} \) und \( \tilde{p} \). Gilt \( q_d < p_d \), setze als den nächsten zu besuchenden Knoten \( n \) den linken Nachfolger von \( n \), ansonsten den rechten. Gibt es hier keinen Nachfolger, so füge \( \tilde{p} \) als neuen Knoten hier ein und terminiere.

Anschaulich repräsentiert jeder Knoten \( n \) eine \((k - 1)\)-dimensionale Hyperebene, die auf der Hyperebene des Vaterknotens senkrecht steht. Neu eingefügte Punkte werden dem linken oder rechten Teilbaum von \( n \) zugeordnet, je nachdem, auf welcher Seite der Hyperebene sie sich befinden. Die Hyperebene teilt die zum Punkt gehörige Region, die für \( k = 2 \) rechteckig ist, im allgemeinen Fall ein Hyperquader. Abbildung 2.9 veranschaulicht dies für \( k = 2 \).
Abbildung 2.9: kd-Baum für k = 2. Die Punkte sind in der Reihenfolge ihres Einfügens numeriert.
Abbildung 2.10: Nächster-Nachbar-Suche. Es soll der nächstengelegene Punkt zu \( q \) gefunden werden. Man steigt zunächst im Baum bis zu \( \overline{p} \) ab. Durch Backtracking gelangt man zunächst zum Vater \( \overline{p}_2 \). Der Bounds-Overlap-Ball-Test (BOB) mit dem Abstand des bisher besten Kandidaten \( \overline{p}_1 \) ist für die zu \( \overline{p}_2 \) gehörige Region positiv, deshalb wird dieser Punkt überprüft. \( \overline{p}_3 \) liegt näher an \( q \) als \( \overline{p}_1 \), der Kreisradius für BOB wird entsprechend angepasst (rechts). \( \overline{p}_3 \) ist ein Kind von \( \overline{p}_2 \), das noch überprüft werden muss, da sich seine Region noch mit dem Kreis überlappt. Der Algorithmus terminiert nach Backtracking zum Wurzelknoten \( \overline{p}_4 \), da alle seine Kinder entweder bereits überprüft wurden, oder BOB fehlschlägt.

Der kd-Baum ist eine geeignete Datenstruktur, um das Problem nächster Nachbarn zu lösen (Operator \( C \) aus Abschnitt 2.3), also zu einem Punkt \( q \) den nächstengelegenen zu finden. Hierzu geht man zunächst wie beim Einfügen vor und findet so einen Blattknoten des Baumes. Der dort gespeicherte Punkt liegt in der Regel \( q \) schon recht nah, der wahre nächste Nachbar könnte aber auch in angrenzenden Regionen liegen. Durch Backtracking werden deshalb noch alternative Äste des Baumes durchsucht. Es müssen alle Äste untersucht werden, deren Regionen sich mit einer Kugel um \( q \), deren Radius der Abstand zum nächsten bisher gefundenen Punkt ist, überlappen (Bounds-Overlap-Ball-Test). Abbildung 2.10 zeigt den Ablauf des Algorithmus.

Will man die \( k \) nächstengelegenen Punkte finden, so merit man sich statt des besten bisher gefundenen Kandidaten eine Liste der \( k \) besten, der Ablauf ist ansonsten gleich. David Mounts Softwarebibliothek ANN (siehe [Mou98]), die für diese Arbeit zum Einsatz kam, enthält effiziente Implementierungen der beschriebenen Suchverfahren.
2.5 Kinematische Modellierung artikulärer Objekte

Das im Rahmen dieser Arbeit für die Kalmanfilterung gewählte Messmodell enthält als wesentlichen Teil die Vorwärtskinematik des menschlichen Oberkörpers. In diesem Abschnitt werden die entsprechenden mathematischen Modelle eingeführt.

Matrixmultiplikation zu beschreiben:

\[
\begin{pmatrix}
  x_{\text{transformiert}} \\
  y_{\text{transformiert}} \\
  z_{\text{transformiert}} \\
\end{pmatrix}
= \begin{pmatrix}
  r_{11} & r_{12} & r_{13} & t_x \\
  r_{21} & r_{22} & r_{23} & t_y \\
  r_{31} & r_{32} & r_{33} & t_z \\
  0 & 0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
  x \\
  y \\
  z \\
  1 \\
\end{pmatrix}
\]

\((t_x, t_y, t_z)^T\) ist die translatorische Komponente, \(r_{ij}\) sind die Elemente der Rotationsmatrix. Wie man sieht, muss der ursprüngliche Koordinatenvektor um eine Komponente erweitert werden, die immer gleich eins ist.

Die Rotationsmatrizen sind mit Eulerwinkeln parametrisiert:

\[R(\alpha, \beta, \gamma) = \begin{pmatrix}
  \cos\gamma & \sin\gamma & 0 \\
  -\sin\gamma & \cos\gamma & 0 \\
  0 & 0 & 1 \\
\end{pmatrix} \begin{pmatrix}
  \cos\beta & 0 & \sin\beta \\
  0 & 1 & 0 \\
  -\sin\beta & 0 & \cos\beta \\
\end{pmatrix} \begin{pmatrix}
  1 & 0 & 0 \\
  0 & \cos\alpha & \sin\alpha \\
  0 & -\sin\alpha & \cos\alpha \\
\end{pmatrix}
\]

Die Winkel \(\alpha, \beta, \gamma\) entsprechen der Rotation um die Achsen \(xyz\) (Pitch-Yaw-Roll), in dieser Reihenfolge. \(\gamma\) ist bei allen Gelenken konstant, lediglich die Rotationsmatrix, die die Gesamtlage des Körpers beschreibt, erlaubt Drehung um alle Achsen.

Zur Transformation eines Vektors von einem Koordinatensystem in ein anderes muss man nun alle Transformationsmatrizen entlang der kinematischen Kette, die beide Koordinatensysteme verbindet, mit einander multiplizieren. Man erhält eine homogene Transformationsmatrix, die die Gesamttransformation repräsentiert.
Kapitel 3

Trackingalgorithmus


3.1 Segmentierung und weitere Vorverarbeitung

Aus einem Stereobildpaar lässt sich mittels Stereobildverarbeitung (siehe Abschnitt 2.1) ein Disparitätenbild generieren, aus dem eine Punktwolke rekonstruiert werden kann, die einer 3D-Ansicht der beobachteten Szene entspricht. Dank der räumlichen Kalibrierung aller Kameran gegen einander (Abschnitt 2.1.5) können die einzelnen Punktfolgen aus deren lokalen Koordinatensystemen in ein gemeinsames Weltkoordinatensystem transformiert werden. Der Kern des Trackingalgorithmus (siehe Abschnitt 3.2) setzt eine hinreichend gute Freistellung (Segmentierung) der beobachteten Person vom Hintergrund voraus. Dieser wichtige Vorverarbeitungsschritt wird in diesem Abschnitt beschrieben. Abbildung 3.1 präsentiert einen Überblick der einzelnen Stufen der Vorverarbeitung.

3.1.1 Vordergrundsegmentierung auf Disparitätenbildern

Um eine erste, grobe Vordergrundsegmentierung zu erhalten, wurden die aus der Stereobildverarbeitung stammenden Disparitätenbilder (Abbildung 3.2 (a)) herangezogen. Dieses Verfahren unterscheidet sich von der konventionellen Vordergrundsegmentierung auf Intensitäts- oder RGB-Bildern entscheidend, ist aber prinzipiell einfacher. Die Vordergrundsegmentierung auf Disparitätenbildern ist im Vergleich zur konventionellen Segmentierung im RGB-Farbraum robuster gegenüber wechselnden Beleuchtungsverhältnissen und Schattennwürfen. Das hier
Abbildung 3.1: Die einzelnen Schritte der Vorverarbeitung.
3.1. SEGMENTIERUNG UND WEITERE VORVERARBEITUNG

Abbildung 3.2: Vordergrundsegmentierung auf Disparitätenbildern. (a) Ein Einzelbild der Stereoabfolge. (b) Disparitätenbild. Schwarze Bereiche enthalten keine gültigen Tiefeninformationen. (c) Ergebnis der Vordergrundsegmentierung. Die Vordergrundpixel sind mit ihrer Disparität dargestellt, Hintergrund ist schwarz.

beschriebene Verfahren ist ähnlich zu [EKB98].

Die Ungenaugkeit der errechneten Disparität $d$ ist unabhängig von der Entfernung des beobachteten Punktes. Die Entfernung $z$ hängt mit der Disparität über einen Zusammenhang der Form $z \sim \frac{1}{d}$ (Vergleiche 2.1) zusammen, die resultierende Ungenaugigkeit von $z$ ist also nicht unabhängig vom Betrag von $d$ beziehungsweise $z$. Aus diesem Grund ist es ratsam, die Disparitätenbilder statt der echten Tiefenbilder zu verarbeiten.

Existiert ein Modell $D_{Modell}$ des statisch angenommenen Hintergrundes in Form eines Disparitätenbildes, so ist das Konzept der Vordergrundsegmentierung denkbar einfach: Um ein aufgenommenes Disparitätenbild $D$ zu segmentieren, wird die Disparität $d$ jedes Pixels aus $D$ mit der des entsprechenden Pixels in $D_{Modell}$, $d_{Modell}$, verglichen. Gilt $d > d_{Modell}$, so wird das Pixel dem Vordergrund zugeordnet, ansonsten dem Hintergrund. Es wird für die Unterscheidung der Operator $\mathcal{V}(d, d_{Modell})$ eingeführt und schrittweise verfeinert. Im idealisierten Fall kann man $\mathcal{V}$ wie beschrieben wählen:

\[
\mathcal{V}(d, d_{Modell}) = \begin{cases} 
  d > d_{Modell} & : \text{Vordergrund} \\
  \text{sonst} & : \text{Hintergrund}
\end{cases}
\] (3.1)

In der Praxis müssen einige weitere Faktoren beachtet werden: Zunächst enthalten die Disparitätenbilder durch die in Abschnitt 2.1.4 beschriebene Nachverarbeitung Pixel ohne gültige Tiefeninformation. Finden man gültige Tiefeninformation an Stellen, an denen das Modell als ungültig markiert ist, so zum Beispiel, wenn eine Person mit gemustertem Hemd vor einem sehr schwach texturierten Teil des Hintergrundes steht, so können die entsprechenden Pixel dem Vordergrund zugeordnet werden:

\[
\mathcal{V}_1(d, d_{Modell}) = \begin{cases} 
  \text{gültig}(d) \land \text{ungültig}(d_{Modell}) & : \text{Vordergrund} \\
  d > d_{Modell} & : \text{Vordergrund} \\
  \text{sonst} & : \text{Hintergrund}
\end{cases}
\] (3.2)
Prinziell gilt auch der komplementäre Fall, steht also eine Person mit schwach texturiertem Hemd vor einer gemusterten Tapete, so könnte man auch hier Vordergrundpixel ausmachen. Dieser Fall wird im Folgenden ignoriert, da Vordergrundpixeln ohne gültige Tiefeninformation für die Weiterverarbeitung wertlos sind.

Die Disparitätenbilder sind verrauscht, hieraus würden bei einer strengen Diskriminierung wie in (3.1) und (3.2) Artefakte resultieren. Deshalb wird ein Schwellwert $s$ in die Ungleichung eingeführt:

$$
V_2(d, d_{Modell}) = \begin{cases} 
\text{gültig}(d) \land \text{ungültig}(d_{Modell}) & : \text{Vordergrund} \\
\text{sonst} & : \text{Hintergrund}
\end{cases}
$$


Man kann ein Hintergrundmodell erzeugen, indem man die aufgenommenen Disparitätenbilder laufend mit exponentieller Lernrate mittelt. Das neue Hintergrundmodell $d_{k\text{Modell}}^k$ ($k$ ist der diskrete Zeitindex der Bildfolge) errechnet sich für jedes neue Disparitätenbild $d^k$ zu

$$
d_{k\text{Modell}}^k = (1 - \alpha)d_{k\text{Modell}}^{k-1} + \alpha d^k
$$

mit der Lernrate $\alpha$, die zwischen null und eins liegen muss. Ähnlich wird ein Array $G_{\text{model}}$ gelernt, das angibt, wo im Hintergrund mit gültigen Disparitäten zu rechnen ist:

$$
g_k^{\text{Modell}} = (1 - \alpha_g)g_{k\text{Modell}}^{k-1} + \alpha_g g^k
$$

$g^k$ enthält bei einer gültigen Disparität eins, ansonsten null. Entsprechend zeigt $G_{\text{Modell}}$ für jedes Pixel an, mit welcher Häufigkeit dort in den letzten Bildern der Folge gültige Tiefeninformation vorhanden war. Das Array $G_{\text{Modell}}$ wird verwendet, um die Prädikate gültig und ungültig aus (3.3) auszuwerten. Hierzu wird der Wert von $G_{\text{Modell}}$ an der jeweiligen Stelle mit einem Schwellwert verglichen. Abbildung 3.2 (c) zeigt die Wirkung des Verfahrens.

### 3.1.2 Filterung der 3D-Daten

Transformiert man die segmentierten Disparitätenbilder zu Punktwolken (siehe 2.1) und fusioniert die Messungen der Einzelkameras (2.1.5), erhält man eine Punktwolke, in der zwar die meisten zum Hintergrund gehörigen Punkte entfernt sind, einige aber noch als Artefakte verblieben sind. Sie resultieren aus fehlerhafter berechneten Korrespondenzen bei der Stereobildverarbeitung sowie mangelhafter Segmentierung. Diese Artefakte werden in einer nachgeschalteten Filterung entfernt.
Abb Bildung 3.3: Filterung der vorsegmentierten Punktwolke. Exemplarisll wird \( \vec{p}_1 \), \( \vec{p}_2 \) und \( \vec{p}_3 \) für \( k = 4 \) überprüft. \( \vec{p}_1 \) wird entfernt, da der eingezeichnete Abstand zu groß ist. Schrittweise würde im weiteren Verlauf das vollständige Artefakt um \( \vec{p}_1 \) entfernt werden. \( \vec{p}_2 \) und \( \vec{p}_3 \) werden nicht entfernt.

Die auftretenden Artefakte sind ihrem Charakter nach Punktbündel kleiner Ausdehnung und Mächtigkeit. Um sie zu entfernen, kommt der \( k \)-nächste-Nachbarn-Operator \( C_k \) zum Einsatz. \( C_k(\vec{p}, P) \) ermittelt zu einem Punkt \( \vec{p} \) die \( k \) nächstgelegenen Punkte aus \( P \). In Abschnitt 2.4 wird eine effiziente Implementierung beschrieben.

Sei \( P = \{ \vec{p}_0, ..., \vec{p}_n \} \) die ungefilterte, vorsegmentierte Punktwolke. Für alle \( \vec{p}_i \) ermittelt man sukzessive die \( k \) nächsten Nachbarn in \( P \): \( C_k(\vec{p}_i, P) \). Man entfernt \( \vec{p}_i \), wenn der Abstand eines der Elemente aus \( C_k(\vec{p}_i, P) \) zu \( \vec{p}_i \) einen Maximalwert \( d_{\text{max}} \) überschreitet (siehe Abbildung 3.3). Abbildung 3.4 zeigt die Wirkung des Verfahrens.

Die amortisierten Kosten für die Nächste-Nachbar-Suche in \( n \) Punkten ist bei der Implementierung mit kd-Bäumen (Abschnitt 2.4) \( \mathcal{O}(\log n) \), die Anwendung des Verfahrens auf alle \( n \) Punkte hat dementsprechend die Komplexität \( \mathcal{O}(n \log n) \).

### 3.2 Tracking artikulärer Objekte in Punktwolken

Die in Abschnitt 3.1 durchgeführte Vorverarbeitung liefert das vom Hintergrund freigestellte Abbild der zu verfolgenden Person als 3D-Punktwolke \( P_m \). Das im Rahmen dieser Arbeit entwickelte System verwendet ein Registrierungsverfahren, um ein artikulares Modell des menschlichen Oberkörpers (siehe Abschnitt 2.5)
Abbildung 3.4: Filterung der 3D-Daten. Die schwarzen Punkte wurden durch das Artefaktfilter entfernt.

dieser Punktwolke nachzuführen. Hierzu wurde der bekannte ICP-Algorithmus modifiziert, um die Registrierung nicht-rigider Körper zu ermöglichen. Das Problem der Registrierung zweier Punktwolken wurde dazu als lineares Schätzproblem formuliert, das mit einem Kalmanfilter gelöst werden kann. Da das beteiligte funktionale Messmodell nichtlinear ist, kam eine linearisierende Variante des Kalmanfilters (Unscented Kalmanfilter, Abschnitt 2.2.4) zum Einsatz, um es zu linearisieren.

Abschnitt 3.2.1 beschreibt zunächst das verwendete Systemmodell, Abschnitt 3.2.2 das Messmodell sowie die Generierung eines geeigneten Messvektors aus der von der Vorverarbeitung gelieferten Punktwolke. Abschnitt 3.2.3 demonstriert dann detailliert den Ablauf der Schätzung.

3.2.1 Systemmodell

Der Kalmanfilter schätzt einen numerischen Zustandsvektor x. Dieser beschreibt bei uns die Konfiguration des menschlichen Oberkörpers und enthält die Parameter der in Abschnitt 2.5 beschriebenen Gelenke beziehungsweise ihrer Rotationsmatrizen. In die Zustandsschätzung fließt wie in Abschnitt 2.2.1 beschrieben Wissen über die Dynamik des Zustandes x ein, das in Form des funktionalen Systemmodells f aus Gleichung (2.4) sowie der Kovarianzmatrix Q, die die Verlässlichkeit von f beschreibt, eingebracht wird. Da die Bildrate der verwendeten
3.2. TRACKING ARTIKULARER OBJEKTE IN PUNKTWOLKEN

Videohardware nicht besonders hoch ist (~10 Hz), die Bewegung einer Person aber vergleichsweise hochfrequent und schwer vorhersehbar, wurde als Dynamik des Systems Konstanz \( f(x) = x \) angenommen, allerdings bei entsprechend hoher Kovarianz \( Q \). Der Systemprozess ist dann schon ein mehrdimensionaler, symmetrischer Random Walk mit Kovarianz \( Q \). Wir nehmen an, dass die Bewegungen der einzelnen Körperteile beziehungsweise Gelenkwinkel stochastisch unabhängig voneinander sind, \( Q \) hat also Diagonalform.

3.2.2 Messmodell und Generierung des Messvektors

Das für die Filterung verwendete Messmodell \( h(x) \) aus Gleichung 2.5 stützt sich auf die in Abschnitt 2.11 beschriebene Vorwärtskinematik des Oberkörpers und den Closest Point-Operator aus Abschnitt 2.4. Das Messmodell wird in diesem Abschnitt schrittweise aufgebaut.

Zunächst tastet man eine Menge Punkte \( P_z \) von der Oberfläche des Oberkörpermödells ab. \( P_z \) soll eine Hypothese für die Punktwoche sein, die man im Systemzustand \( x \) beobachten würde. Das Oberkörpermödell wird aus affin transformierten Kegelstämmen zusammengesetzt (Abbildung 3.5 (a)). Abbildung 3.5 (b) zeigt, wie ein im Koordinatenursprung liegender und an der \( z \)-Achse ausgerichteter Kegelstumpf abgetastet wird. Es gilt

\[
p(h, \alpha) = (r_1 + h \frac{r_2 - r_1}{H}) \begin{pmatrix} \cos \alpha \\ \sin \alpha \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ h \end{pmatrix}.
\]

Einige Körperteile bestehen aus mehreren derartigen Stümpfen, die gegebenenfalls gegen den Koordinatenursprung verschoben und gedreht werden müssen. Dies ist der Fall beim Torso, der aus einem großen Stumpf und zwei kleineren, die die Schultern modellieren, besteht. Der Hauptstumpf des Torsos wird in \( y \)-Richtung abgeflacht. Der Kopf wird aus zwei auseinandergesetzten Kegelstämmen modelliert, ebenso die Unterarme, die zusammen mit den Händen als ein in sich starres Körperteil modelliert werden.

Die so erhaltenen Punkte werden nun aus den lokalen Koordinatensystemen der einzelnen Körperteile mittel Vorwärtskinematik (siehe Abschnitt 2.5) in das Weltkoordinatensystem transformiert. Die Abbildungen 3.5 (c)-(d) zeigen zwei verschiedene Möglichkeiten, die Modelloberfläche abzutasten. Abschließend werden Punkte, die im Inneren des Modells liegen oder die durch Verdeckung auf keiner Kamera sichtbar sind, entfernt (Abbildung 3.5 (e)).

Wir führen den Operator \( \mathcal{G} \) ein, der die Punktmenge \( P_z = \{ p_{0z}, p_{1z}, \ldots, p_{nz} \} \) zu einem gegebenen Zustand \( x \) wie beschreiben generiert:

\[
P_z = \mathcal{G}(x).
\]  

Man berechnet nun mit dem im Abschnitt 2.11 eingeführten Operator \( C \) die zu \( P_z \) korrespondierenden Punkte \( P_z = \{ p_{0z}, p_{1z}, \ldots, p_{nz} \} \) aus der von der Vorver-
Abbildung 3.5: Generierung der Modellpunkte \( P_h \). Bild (a): Das Oberkörpermodell ist aus 11 affin transformierten Kegelstümpfen zusammengesetzt. Bild (b) zeigt das Ablasen eines Kegelstumpfes. Man beschreibt einen Punkt \( p \) des Stumpfmantels zweckmäßigerverweise wie im Bild über die Höhe \( h \) und den Winkel \( \alpha \). Der Stumpf selber wird über die beiden Radien \( r_1 \) und \( r_2 \) der Stirnflächen sowie seine Höhe \( H \) beschrieben. Die Modellpunkte werden von den Kegelstümpfen (c) deterministisch oder (d) zufällig abgetastet. Schließlich werden Punkte, die in keiner Kamera sichtbar wären, entfernt. (e) zeigt dies für den Fall, das nur eine Kamera verwendet wird, die von links auf das Modell gerichtet ist.
Abbildung 3.6: Messmodell. Die schwarzen Punkte $p_i^z$ entsprechen der prädizierten Messung, die weißen Punkte, $p_i^z$, der Messung selbst. Das rechte Bild zeigt, wie aus jeweils einer Korrespondenz $(p_i^z, p_i^z)$ eine Komponente von $\tilde{z}$ und $z$ gebildet wird. Wie man sieht, sind die Komponenten des Residuums $\nu_i = \tilde{z}_i - z_i$ die Abstände der korrespondierenden Punkte.

arbeitung gelieferten Punktfolge $P_M$. Dieser Schritt ist identisch zum Vorgehen im gewöhnlichen ICP-Algorithmus aus Abschnitt 2.3:

$$P_2 = C(P_2, P_M).$$

Im nächsten Schritt muss $P_2$ mit $P_2$ registriert werden. Beim ICP-Algorithmus zur Registrierung starrer Körper kommen hier unterschiedliche Verfahren zum Einsatz (Singularwertzerlegung, Quaternionenrepräsentation), die aber im artikelaren Fall nicht anwendbar sind. Wir betrachten deshalb das Problem der Registrierung als lineares Schätzproblem, das mit einem Kalmanfilter gelöst werden kann.

Die Menge $P_2$ repräsentiert hier in der Notation aus Abschnitt 2.2.1 die Messung $z[k + 1]$, $P_2$ die vorhergesagte Messung $\hat{z}[k + 1|k]$. Der diskrete Zeitindex $k$ wird in diesem Abschnitt weggelassen und nur $z$ und $\hat{z}$ geschrieben. Die Vektoren $z$ und $\hat{z}$ müssen aus $P_2$ und $P_2$ generiert werden.

Der direkteste Ansatz, aus den Punktmengen Vektoren zu bilden, wäre, die Einzelpunkte direkt zu Vektoren zusammenzuführen, aus $P = \{p_0, p_1, ..., p_m\}$ würde also der Vektor

$$\left( p_0^z \ p_0^\nu \ p_1^z \ p_1^\nu \ p_2^z \ p_2^\nu \ \cdots \ p_{m-1}^z \ p_{m-1}^\nu \ p_{m-1}^z \ p_{m-1}^\nu \right)^T,$$

mit $p_i = (p_i^z \ p_i^\nu \ p_i^z)^T$. Analog würden aus $P_2$ und $P_2$ die $3m$-dimensionalen Vektoren $z$ und $\hat{z}$ generiert. Durch einen Trick ist es möglich, die Dimensionalität
von $z$ und $\hat{z}$ auf $m$ zu reduzieren. Dies ist besonders lohnenswert, da die Komplexität des Korrekturumschrittes des Kalmanfilters kubisch mit der Dimensionalität des Messvektors zusammenhängt (siehe Abschnitt 2.2.1). Die Idee hierbei ist, das bei der Bewegung eines Punktes aus $p_i^* \in P_z$ hauptsächlich der Anteil der Bewegung, der zu seinem korrespondierenden Punkt $p_i^* \in P_z$ hin (oder davon weg) verläuft, für eine gute Passung ausschlaggebend ist. Deshalb werden $p_i^*$ und $p_i^*$ auf ihre Verbindungsgerade projiziert. Abbildung 3.6 illustriert das Verfahren. Aus $p_i^*$ und $p_i^*$ werden also die skalaren Werte

\[
\hat{z}_i = \frac{p_i^* - p_i^*}{\|p_i^* - p_i^*\|} \cdot p_i^* \\
z_i = \frac{p_i^* - p_i^*}{\|p_i^* - p_i^*\|} \cdot p_i^*. \tag{3.5}
\]

berechnet. Die Vektoren $\hat{z}$ und $z$ werden nun durch Zusammenführen aller $\hat{z}_i$ und $z_i$ gebildet:

\[
\hat{z} = \begin{pmatrix} \hat{z}_0 & \hat{z}_1 & \cdots & \hat{z}_{m-1} \end{pmatrix}^T \tag{3.7}
\]

\[
z = \begin{pmatrix} z_0 & z_1 & \cdots & z_{m-1} \end{pmatrix}^T \tag{3.8}
\]

Die Generierung der prädizierten Messung aus den korrespondierenden Punktmengen $P_z$ und $\hat{P}_z$ mittels (3.5) und (3.7) wird zum Operator $\hat{Z}$ zusammengefasst:

\[
\hat{z} = \hat{Z}(P_z, \hat{P}_z). \tag{3.9}
\]

Die Messung selbst wird mit dem Operator $Z$ generiert ((3.6) und (3.8)):

\[
z = Z(P_z, P_z). \tag{3.10}
\]

Wie man gesehen hat, muss zur Auswertung des Messmodells der Closest Point Operator $C$ angewendet werden. Aus diesem Grund müsste $C$ im Korrekturumschritt des UKF $2n+1$ mal ($n$: Dimensionalität des Zustandsraumes) ausgewertet werden (für jeden $\sigma$-Punkt einmal). Da man durch Skalierung der Unscented Transformation (siehe Abschnitt 2.2.4) die $\sigma$-Punkte beliebig nah zueinander bringen kann, kann man sich damit begnügen, die Punktkorrespondenzen im Korrekturumschritt nur einmal herzustellen, und zwar basierend auf dem prädizierten Zustand $\hat{x}$. Das entsprechende Messmodell wird hier $h_\delta(x)$ genannt. Es lässt sich zusammenfassend formulieren als

\[
h_\delta(x) = \hat{Z}(P_z, \hat{P}_z), \tag{3.11}
\]

mit $\hat{P}_z = \mathcal{G}(\hat{x})$ und $P_z = \mathcal{C}(\mathcal{G}(\hat{x}), P_m)$. $P_m$ ist die segmentierte und vorverarbeitete Punktwolke aus der Stereobildverarbeitung. Der zugehörige Messvektor $z$ wird mit (3.10) erzeugt.
3.2. TRACKING ARTIKULÄRER OBJEKTE IN PUNKTWOLKEN

Abbildung 3.7: Einfaches artikulares Modell mit den drei Freiheitsgraden α, β und γ. Das Modell ist bei r fest.

3.2.3 Ablauf der Filterung

In diesem Abschnitt wird ein vollständiger Vorhersage-Korrektur-Zyklus des Filters demonstriert. Es wird aber aus Gründen der Anschaulichkeit ein stark vereinfachtes Zustandsmodell mit nur 3 Winkeln verwendet, dessen Kinematik in Abbildung 3.7 dargestellt ist. Der Zustandsvektor hat die Form \( \mathbf{x} = (\alpha, \beta, \gamma)^T \).

Man geht von der Zustandsschätzung zum letzten diskreten Zeitpunkt \( k \) aus. Die Schätzung besteht aus dem Mittelwert \( \hat{\mathbf{x}}[k|k] \) und der Kovarianzmatrix \( \mathbf{P}[k|k] \), die die Ungenauigkeit der Schätzung beschreibt. Zur Erinnerung: Mit \( \hat{\mathbf{x}}[i|j] \) wird die Schätzung zum Zeitpunkt \( i \), unter Einbeziehung aller Messungen bis zum Zeitpunkt \( j \), bezeichnet (vergleiche Abschnitt 2.2.1). Die Extrapolation von \( \hat{\mathbf{x}}[k+1|k] \) und \( \mathbf{P}[k+1|k] \) (Prädiktion) ist im hier betrachteten Fall trivial, da als Systemmodell Konstanz angenommen wird (siehe Abschnitt 3.2.1). Es vergrößert sich also lediglich die Kovarianz der Schätzung um das Systemrauschen \( \mathbf{Q} \):

\[
\hat{\mathbf{x}}[k+1|k] = \hat{\mathbf{x}}[k|k] \\
\mathbf{P}[k+1|k] = \mathbf{P}[k|k] + \mathbf{Q}.
\]

Abbildung 3.8 veranschaulicht dies. Die folgenden Ausführungen befassen sich meistens mit den prädizierten Größen \( \hat{\mathbf{x}}[k+1|k] \) und \( \mathbf{P}[k+1|k] \), die Zeitindizes werden dann weggelassen.

Abbildung 3.9 zeigt die aus \( \hat{\mathbf{x}} \) und \( \mathbf{P} \) mittels (2.12)-(2.14) generierten \( \sigma \)-Punkte \( \{x_0, ..., x_6\} \), die weiter unten gebraucht werden. Sie sind eine vollständige Repräsentation von Mittelwert und Kovarianz der Zustandsschätzung, \( \hat{\mathbf{x}} \) und \( \mathbf{P} \) liessen sich aus ihnen durch gewichtete Mittelung wie in (2.16) und (2.17) reproduzieren.

Nun wird der Korrekturschritt vorbereitet, indem die Punktkorrespondenzen hergestellt werden, auf denen das Messmodell \( h_\hat{\mathbf{x}} \) basiert. Zunächst wird eine wohldefinierte Menge Punkte vom Modell, das sich im prädizierten Zustand \( \hat{\mathbf{x}} \) befindet, abgetastet. Es wird angenommen, dass für das hier betrachtete Modell ein Operator \( \mathcal{G} \) existiert, der analog zu (3.4) funktioniert, \( \mathcal{G}(\hat{\mathbf{x}}) \) liefert, wie in Abbildung 3.10 ersichtlich, die Punktmenge \( P_\hat{\mathbf{x}} = \{p_0, ..., p_6\} \). Durch Anwendung des Closest Point-Operators \( \mathcal{C} \) erhält man die Menge korrespondierender Punkte
Abbildung 3.8: Prädiktion von $\hat{x}[k+1|k]$ und $P[k+1|k]$. Die Ungenauigkeit $P[k|k]$ der letzten Zustandsschätzung $\hat{x}[k|k]$ ist durch die gestrichelten Linien angedeutet. Sie vergrößert sich (gepunktete Linien) bei der Prädiktion um das Systemrauschen $Q$.

Abbildung 3.9: Unscented-Repräsentation des prädictierten Zustandes und seiner Ungenauigkeit. Jeder der 7 σ-Punkte \( \{x_0, ..., x_6\} \) repräsentiert eine vollständige Konfiguration des Modelles, hier schwarz dargestellt. Für die Darstellung wurde eine diagonale Kovarianzmatrix gewählt (keine Korrelation der Winkel untereinander), deshalb überlappen sich die Modelle teilweise, dies ist im Allgemeinen nicht der Fall.

Abbildung 3.10: Korrespondenzen. Es wurde die Punktmenge $G(\hat{x}) = P_2 = \{p_0, ..., p_5\}$ vom Modell abgetastet (schwarz). Sie soll mit den korrespondierenden weissen Punkten $P_2 = \{p_0', ..., p_5'\}$ registriert werden.
3.2. TRACKING ARTIKULÄRER OBJEKTE IN PUNKTWOLKEN

\[ (\tilde{x}, P) \quad \rightarrow \quad \{X_0, \ldots, X_n\} \quad \xrightarrow{h_\tilde{x}} \quad \{Y_0, \ldots, Y_{2n}\} \quad \xrightarrow{\text{gewichtete Mittelung}} \quad (\tilde{z}, P_{zz}) \]

\[ \{X_0, \ldots, X_n\} \quad \xrightarrow{\text{gewichtete Mittelung}} \quad P_{xz} \]

\[ \{Y_0, \ldots, Y_{2n}\} \]

\[ \text{Produkte } (\tilde{X}_i - \tilde{x})(\tilde{Y}_i - \tilde{z})^T \]

\[ P_z = \{P_{z0}, \ldots, P_{zn}\}; P_z \text{ soll nun durch geeignete Korrektur von } \alpha, \beta \text{ und } \gamma \text{ mit } P_z \text{ registriert werden.} \]

Das Kalmanfilter ermittelt die optimale Korrektur des Zustandsvektors, indem er einen linearen Zusammenhang zwischen dem Zustandsvektor und dem Messvektor herstellt. Das UKF bedient sich hierzu der Unscented Transformation (Abbildung 3.12), die hier folgendermaßen abläuft: Die \( \sigma \)-Punkte \( \{X_0, \ldots, X_n\} \) werden zunächst durch das Messmodell transformiert. Hierzu werden in einem ersten Schritt für alle \( X_i \) die Punkte \( G(X_i) \) abgetastet (Abbildung 3.11). Dann erfolgt die Projektion auf die Verbindungsgerade des jeweiligen korrespondierenden Punktpares. Durch zusammenführen der Projektionen aller \( \sigma \)-Punkte des Modells erhält für jeden \( \sigma \)-Punkt \( X_i \) einen sechsdimensionalen Vektor, \( Y_i \).

Durch Mittelung der \( Y_i \) erhält man den prädizierten Messvektor \( \tilde{z} \) und seine Varianz \( P_{zz} \). Die Projektion und anschließende Mittelung ist für eine Komponente von \( \tilde{z} \) in Abbildung 3.13 illustriert. Die dargestellte Varianz von \( \tilde{z}_0 \) ist das 0-te Diagonalelement der Matrix \( P_{zz} \). Die Unscented-Transformation ermittelt alle Elemente dieser Matrix, gegebenenfalls auch Korrelation der Komponenten.
Abbildung 3.13: Transformation von \(\{X_0, ..., X_6\}\) durch das Messmodell \(h_z\), zweiter Schritt. Man betrachte die Umgebung von \(p_{z2}^0\) (Kasten aus Abbildung 3.11), die hier so rotiert wurde, dass \(p_{z2}^0 - p_{z2}^0\) entlang der Abszisse liegt. Die Projektion der zu \(p_{z2}^0\) gehörenden Punkte ist angedeutet (es sind tatsächlich sieben Punkte, sie liegen aber teilweise übereinander). Die Komponente \(z_0\) und ihre Varianz erhält man nun durch gewichtete Mittelung dieser Projektionen. Die Varianz von \(z_0\) wird durch das Messrauschen \(R\) vorgegeben und entspricht dem erwarteten Fehler der Messung. Wie man sieht, enthält jede skalare Komponente \(v_i = \hat{z}_i - z_i\) des Residuums die Abstände der korrespondierenden Punkte.

Abbildung 3.14: Kreuzkоварианц. Qualitative Darstellung der Kreuzkоварианцmatrix \(P_{xz}\) für das in diesem Abschnitt beschriebene Beispiel. Schwarz bedeutet starke positive Korrelation, weiß starke negative Korrelation. Man kann beispielsweise ablesen, dass eine Vergrößerung des Winkels \(\gamma\) den Abstand \(p_{z2}^5p_{z2}^5\) verkleinern würde. Die Abstände \(v_0\) bis \(v_3\) dagegen sind mit dem Winkel \(\gamma\) unkorreliert.
3.2. TRACKING ARTIKULÄRER OBJEKTE IN PUNKTWOLKEN


von \( \tilde{z} \) untereinander. Beispielsweise ist eine positive Korrelation von \( \tilde{z}_0 \) und \( \tilde{z}_1 \) zu erwarten, da die entsprechenden Punkte \( p^0_0 \) und \( p^1_1 \) nahe beieinander auf dem gleichen Schenkel des Modells liegen. Das Residuum \( \nu = (\nu_0 \nu_1 \nu_2 \nu_3 \nu_4 \nu_5)^T \) ist die Differenz \( \tilde{z} - z \) von prädiziertem und aquiriertem Messvektor. Wie man sieht, enthalten es hier die Abstände der korrespondierenden Punkte. Seine Varianz ergibt sich aus \( P_{zz} \) durch Addition des Messrauschens: \( P_{\nu} = P_{zz} + R \).

Aus dem Residuum wird eine Korrektur des Zustandes \( \tilde{x} \) berechnet. Es wird hierzu mit dem Kalmagain \( K \) gewichtet: \( \tilde{x}[k+1|k+1] = \tilde{x} + K \nu \). In das Kalmagain gehen die Varianz \( P_{\nu} \) des Residuums und die mittels (2.18) berechnete Kreuzkovaarianzmatrix \( P_{x\nu} \), die ein Maß für den linearen Zusammenhang zwischen einer Änderung des Zustandsvektors und der resultierenden Änderung am Residuum darstellt, ein. Die Kovaarianz ist für das hier betrachtete Beispiel in Abbildung 3.14 qualitativ dargestellt. Das Kalmagain berechnet sich zu \( K = P_{xz} P_{\nu}^{-1} \). Die Korrektur der Zustandsschätzung kann nun durchgeführt werden:

\[
\begin{align*}
\tilde{x}[k+1|k+1] &= \tilde{x} + K \nu \\
P[k+1|k+1] &= P - KP_{\nu}K^T.
\end{align*}
\]

Man iteriert den in diesem Abschnitt beschriebenen Korrekturschritt mehrmals, da analog zum gewöhnlichen ICP-Algorithmus zu erwarten ist, das sich die Korrespondenzen mit jedem Mal verbessern und die Registrierung nach einigen Iterationen konvergiert.

3.2.4 Kollisionsvermeidung

Situationen, in denen Körperteile sich einander sehr nahe kommen, sind für das System schwer handhabbar, da es hier häufig zur Bildung falscher Korrespon-

Um dies zu vermeiden, werden Posen, bei denen sich Körperteile sehr nahe kommen oder überschneiden, von vornherein vermieden. Wird nach der Registrierung des Modells mit der Punktwolke eines Satzes Einzelbilder eine solche Pose festgestellt, wird das Modell in einen Zustand gesetzt, bei dem die Körperteile bestimmte Mindestabstände einhalten. Mindestwerten werden hierbei erzwungen für

- den Abstand des Ellbogens vom Torso
- den Abstand der Hand vom Torso
- den Winkel, den Ober- und Unterarm bilden.

Kapitel 4

Experimente und Ergebnisse

Das Trackingsystem wurde auf realen Videodaten getestet. Ziel war außer der Evaluierung auch die Ermittlung optimaler Werte für zahlreiche Parameter des Systems (Tuning, siehe Abschnitt 4.2.1 für eine Übersicht der Parameter). Abschnitt 4.1 beschreibt zunächst das verwendete Szenario und erläutert, wie eine Referenztrajektorie erstellt wurde, mit der die Ausgaben des Systems für verschiedene Parametersätze verglichen wurden. Abschnitt 4.2.2 präsentiert die Ergebnisse dieser Vergleiche, und Abschnitt 4.3 beschreibt, wie sich das System bei ausgewählten, unterschiedlich schwierig zu trackenden Bewegungsabläufen verhält.

4.1 Datensammlung und Erstellung der Referenztrajektorien


Die aufgenommenen Bildfolgen wurden wie in Abbildung 4.3 manuell annotiert. Aus den gewonnen Daten wurde eine Grundwahrheit ermittelt, indem das


4.2 Parametersatz und Tuning

4.2.1 Parametersatz


Messrauschen und Systemrauschen (R und Q aus Abschnitt 2.2.1) können in recht weiten Grenzen variiert werden ohne signifikante Änderung der Trackinggenauigkeit. Für den Standardparametersatz wurde die Standardabweichung des Messrauschens auf zwei Zentimeter festgesetzt. Dies bedeutet, das angenommen wird, das die Position eines zu einem abgetasteten Modellpunkt durch nearest-neighbour zugeordneten Messpunktes im Mittel um zwei Zentimeter vom tatsäch-
lich korrespondierenden Punkt abweicht. Die Standardabweichung des gewählten Systemrauschen beträgt für die Körperposition $\sigma_{\text{pos}} = 100$ mm, es wird also angenommen, das sich der Proband von Einzelbild zu Einzelbild im Mittel 10 Zentimeter bewegt. Entsprechend wurde für die Körperdrehung (drei Achsen) eine Genauigkeit von $\sigma_{\text{rot}} = 20^\circ$ angenommen, für Ellbogen- und Schultergelenke (jeweils zwei mal zwei Achsen) $\sigma_{\text{gelenk}} = 40^\circ$.

Wie in Abbildung 3.5 gesehen gibt es grundsätzlich zwei Möglichkeiten, Punkte vom Körpermodell abzutasten. Im Allgemeinen wurde für die Experimente eine zufällige Abtastung gewählt, da sich zeigte, dass, insbesondere wenn wenig Punkte abgetastet wurden, bei deterministischer Abtastung die Neigung besteht, das der Tracker in lokalen Minima hängenbleibt. Der Standardwert für die Anzahl abgetasteter Punkte ist $m = 64$.


Wie gesehen läuft das Tracking nach Art einer iterativen Registrierung ab. Für jedes Einzelbild werden standardmäßig $i = 4$ Iterationsschritte verwendet.

In Abschnitt 3.2.4 wurde erläutert, wie ein Mindestabstand der Hand beziehungsweise des Ellbogens vom Torso erzwungen wird. Als Mindestabstand wurde $d_{\text{min}} = 150$ mm gewählt.

Im Allgemeinen wurden alle vier Kameras zum Tracking herangezogen (N, O, S und W).

### 4.2.2 Einfluss einzelner Parameter auf die Trackinggenauigkeit


Tabelle 4.1 führt zunächst die Ergebnisse für den Referenzparametersatz auf. Die Ergebnisse für P3 und P4 sind etwas schlechter, dies ist einerseits auf deren besonders kontrastarme Kleidung zurückzuführen (siehe Abbildung 4.2), andererseits darauf, das in den Sequenzen besonders schwierig zu verfolgende Aktivitäten durchgeführt wurden (siehe auch Abschnitt 4.3).

Tabelle 4.2 zeigt, wie sich die Ergebnisse verändern, wenn die Anzahl der Modellpunkte variiert wird. Der Gewinn an Genauigkeit fällt geringer als erwartet aus, von $m = 24$ abgesehen. Zu erwähnen ist hier, dass die Rechenzeit des Algorithmus vornehmlich von $m$ (Komplexität: $\mathcal{O}(m^3)$) abhängt. Abbildung 4.4
### 4.2. Parametersatz und Tuning

<table>
<thead>
<tr>
<th>Torsoposition [mm]</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>Zusammen</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>65</td>
<td>53</td>
<td>85</td>
<td>67</td>
<td>67</td>
</tr>
<tr>
<td>Torsolängsachse [°]</td>
<td>9,1</td>
<td>8,2</td>
<td>11,2</td>
<td>9,2</td>
<td>9,4</td>
</tr>
<tr>
<td>Torsquerachse [°]</td>
<td>9,5</td>
<td>13,6</td>
<td>16,4</td>
<td>16,1</td>
<td>13,9</td>
</tr>
<tr>
<td>Oberarme [°]</td>
<td>22,0</td>
<td>33,5</td>
<td>40,3</td>
<td>32,8</td>
<td>32,1</td>
</tr>
<tr>
<td>Unterarme [°]</td>
<td>27,6</td>
<td>25,6</td>
<td>27,8</td>
<td>42,3</td>
<td>30,8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\sigma_{pos}$</th>
<th>$\sigma_{rot}$</th>
<th>$\sigma_{gelenk}$</th>
<th>Abtastung</th>
<th>$m$</th>
<th>Proportionen</th>
<th>$i$</th>
<th>$d_{min}$</th>
<th>Kameranamen</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 mm</td>
<td>20°</td>
<td>40°</td>
<td>zufällig</td>
<td>64</td>
<td>individuell</td>
<td>4</td>
<td>150 mm</td>
<td>NOSW</td>
</tr>
</tbody>
</table>

**Tabelle 4.1:** Ergebnisse für den Referenzparametersatz.

<table>
<thead>
<tr>
<th>m = 24</th>
<th>m = 40</th>
<th>m = 64</th>
<th>m = 120</th>
</tr>
</thead>
<tbody>
<tr>
<td>Torsoposition [mm]</td>
<td>89</td>
<td>71</td>
<td>67</td>
</tr>
<tr>
<td>Torsolängsachse [°]</td>
<td>10,8</td>
<td>9,1</td>
<td>9,4</td>
</tr>
<tr>
<td>Torsquerachse [°]</td>
<td>18,5</td>
<td>13,7</td>
<td>13,9</td>
</tr>
<tr>
<td>Oberarme [°]</td>
<td>41,4</td>
<td>31,8</td>
<td>32,1</td>
</tr>
<tr>
<td>Unterarme [°]</td>
<td>41,9</td>
<td>31,1</td>
<td>30,8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\sigma_{pos}$</th>
<th>$\sigma_{rot}$</th>
<th>$\sigma_{gelenk}$</th>
<th>Abtastung</th>
<th>$m$</th>
<th>Proportionen</th>
<th>$i$</th>
<th>$d_{min}$</th>
<th>Kameranamen</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 mm</td>
<td>20°</td>
<td>40°</td>
<td>zufällig</td>
<td>s. Tabelle</td>
<td>individuell</td>
<td>4</td>
<td>150 mm</td>
<td>NOSW</td>
</tr>
</tbody>
</table>

**Tabelle 4.2:** Auswirkung der Anzahl abgetasteter Punkte $m$ auf den mittleren Truckingfehler. Die Ergebnisse des Referenzparametersatzes sind hervorgehoben.

<table>
<thead>
<tr>
<th>Torsoposition [mm]</th>
<th>Kameran NS</th>
<th>Kameran NOSW</th>
</tr>
</thead>
<tbody>
<tr>
<td>Torsolängsachse °</td>
<td>64</td>
<td>62</td>
</tr>
<tr>
<td>Torsquerachse °</td>
<td>7,6</td>
<td>8,8</td>
</tr>
<tr>
<td>Oberarme °</td>
<td>12,1</td>
<td>13,1</td>
</tr>
<tr>
<td>Unterarme °</td>
<td>34,1</td>
<td>29,4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>σ_{pos}</th>
<th>σ_{rot}</th>
<th>σ_{gelenk}</th>
<th>Abtastung</th>
<th>m</th>
<th>Proportionen</th>
<th>i</th>
<th>d_{min}</th>
<th>Kameran</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 mm</td>
<td>20°</td>
<td>40°</td>
<td>zufällig</td>
<td>64</td>
<td>individuell</td>
<td>4</td>
<td>150 mm</td>
<td>s. Tabelle</td>
</tr>
</tbody>
</table>


zeigt die experimentell ermittelte Abhängigkeit zwischen m und der für ein Einzelbild benötigten Verarbeitungszeit. Hierzu kommt noch die Rechenzeit für die Stereobildverarbeitung (0,05 Sekunden pro Kamera) und Segmentierung (0,13 Sekunden für die fusionierte Punktwolke).


Im Allgemeinen wurde das kinematische Modell den ungefähren Proportionen der Probanden individuell angepasst. Tabelle 4.4 zeigt die Trackingergebnisse mit
4.3 Analyse kurzer Bewegungsabläufe im Detail


Schwierigkeiten ergeben sich meistens dann, wenn, wie in Abschnitt 3.2.4 beschrieben, Gliedmaßen sich stark einander annähern oder berühren. Die Zuordnungen zwischen Modell- und Messpunkten basiert auf räumlicher Nähe, deshalb kommt es in diesen Fällen häufig zu falschen Korrespondenzen. Die in 3.2.4 beschriebene Kollisionsvermeidung kann in vielen Fällen Divergenz verhindern, allerdings dauert es mitunter einige Zeit, bis die richtige Zuordnung wiedergefunden wird. Gelegentlich bleibt eine Gliedmaße auch für lange Zeit falsch eingepasst, beispielsweise bleibt dann eine Hand am Körper oder am Kopf „hängen“.


Ähnlich problemlos wird die Bewegung von P2, die in Abbildung 4.7 (Seite 58) dargestellt ist, verfolgt. Die Fehler sind in Abbildung 4.8 (Seite 59) aufgetragen. In Abbildung 4.9 (Seite 60) ist ein Ausschnitt der P3-Sequenz wiedergegeben.
P3 beugt sich vor und geht in die Hocke, um sich die Schuhe zu binden. Abbildung 4.10 (Seite 61) zeigt, wie sich die Fehler in diesem Ausschnitt entwickeln. Es gibt hier teilweise beträchtliche Abweichungen von bis zu $170^\circ$ gegenüber der Referenz. Trotzdem wird die Szene augenscheinlich recht gut getrackt, insbesondere werden die Fehler nach dem Aufrichten der Person (ungefähr beim Einzelbild 320) rasch wieder geringer.

4.3. ANALYSE KURZER BEWEGUNGSABLAÜFE IM DETAIL
Abbildung 4.5: Bewegungsablauf: „Peitschenhieb“. Einzelbildindices (von oben): 1479, 1507, 1577, 1651, 1753
Abbildung 4.6: Fehlerentwicklung während des Bewegungsablaufes „Peitschenhieb“
Abbildung 4.7: Bewegungsablauf „Travolta“. Einzelbildindices (von oben): 412, 421, 436, 443, 450
Abbildung 4.8: Fehlerentwicklung während des Bewegungsablaufes „Truvolta“.
Abbildung 4.10: Fehlerentwicklung während des Bewegungsablaufs „Schuhe binden“.
Abbildung 4.11: Bewegungsablauf: „Arme kreuzen“. Einzelbildindices (von oben): 390, 455, 488, 503, 520
Kapitel 5
Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit wurde ein System zur Verfolgung der artikularen Bewegung des menschlichen Oberkörpers entwickelt. Der Algorithmus führt ein dreidimensionales kinematisches Modell des Oberkörpers einer 3D-Punktwolke nach, die durch Stereobildverarbeitung und verschiedene Segmentierungsschritte aus den Bildfolgen mehrerer Stereokameras gewonnen wurde. Das Trackingproblem wurde hierzu als die Registrierung zweier Punktwolken aufgefasst. Die Registrierung wurde als ein lineares Schätzproblem formuliert, das mit einem Unscented Kalmanfilter gelöst wurde.


Das System wurde ursprünglich mit den Daten von vier Kameras betrieben. Der Betrieb mit weniger Kameras ist auch möglich, die Beschränkung auf zwei Kameras brachte geringe Einbußen bei der Trackinggenauigkeit.

Das Tracking basiert ausschließlich auf der Verwendung von durch Stereobildverarbeitung gewonnenen Tiefeninformationen. Diese sind im wesentlichen invariant gegenüber den Beleuchtungsverhältnissen, Schattenwürfen sowie unterschiedlicher Farbe und Musterung der Bekleidung. Das System zeichnet sich des-
halb durch eine hohe Robustheit gegenüber einer Veränderung dieser Faktoren aus.


Anhang A

Implementierungsdetails


A.1 Klassenstruktur

Abbildung A.1 zeigt die am Tracking beteiligten Klassen. Um die Übersichtlichkeit zu wahren, wurden die Objektfunktionen nicht vollständig aufgeführt. Funktionen, die nicht direkt am Tracking beteiligt sind, wurden ausgelassen, im wesentlichen sind dies jene, die der Initialisierung des Systems oder der Visualisierung der Trackingergebnisse dienen. In Einzelfällen betrifft dies auch die Parameterlisten der Funktionen.


Die Klassen ArticulatedTrackerObserveModel, ArticulatedTrackerPredictModel und Scaled_unscented_filter leiten von Klassen der BAYES++-Bibliothek ab (siehe [Ste03]).

Scaled_unscented_filter erweitert die Funktionalität der von BAYES++ für den Unscented Kalmanfilter bereitgestellten Filterklasse um das Skalierungsverfahren aus [JU02]. Die Klasse enthält als öffentliche Attribute die momentane Zustandsschätzung, bestehend aus dem Vektor x und seiner Kovarianzmatrix P.

ArticulatedTrackerPredictModel enthält das für die Kalmanfilterung verwendete dynamische Modell, derzeit nur die einfache Implementierung aus Abschnitt 3.2.1. In Form der Kovarianzmatrix Q könnte hier zusätzliches Wissen über die Dynamik des menschlichen Körpers eingebracht werden.
Objektbezeichner | Typ
---|---
tracker | ArticulatedTracker
predict_model | ArticulatedTrackerPredictModel
observe_model | ArticulatedTrackerObserveModel
filter | Scaled_unscented_filter
disparityfilters[n] | DispFilter[n]
pointsynth | PointCloudPersonModelWithOcclusion
depthsynths[n] | GIDepthSynthWidget[n]
models[n] | GIPersonModel[n]

Tabelle A.1: Objekte des Trackingsystems. disparityfilters[n], depthsynths[n], und models[n] sind jeweils ein Arrays aus n Objekten, n ist die Anzahl verwendeter Sterokameras.

ArticulatedTrackerObserveModel enthält die Implementierung des Messmodells aus Abschnitt 3.2.2 und dient hierzu an die Klasse PointCloudPersonModelWithOcclusion. Die Klasse GIDepthSynthWidget dient zur Generierung eines synthetischen Tiefenbildes, das bei der Entfernung verdeckter Punkte verwendet wird und verwendet hierzu Routinen der OpenGL-Bibliothek [WNDS99]. Die Anbindung an OPENGL geschah über das QT-GUI-Toolkit [BS04].

Die Klasse DisparityFilter hat im Wesentlichen zwei Funktionen: mit processPair(...) wird aus einem Stereobildpaar ein Disparitätenbild erzeugt. dispToCoord(...) führt die Konvertierung eines Pixels des Disparitätenbildes in Weltkoordinaten durch.

### A.2 Ablauf bei der Verarbeitung eines Einzelbildsatzes


tracker.updateFrame(...) (0) wird nacheinander für die mit der Klasse FGSegmenter voresegmentierten Disparitätenbilder aller Kameras aufgerufen. Alle Aufrufe bis auf den Letzten verhalten sich wie in Abbildung A.2 und führen ausschließlich dazu, das im Array measured_points die aus den Disparitätenbildern mittels des entsprechenden disparityfilters ermittelten 3D-Punkte akkumuliert werden (1 - 1.1).

Erst der letzte Aufruf stößt zusätzlich die Verarbeitung (2 - 4.2) an. Zu-
nächst wird der Prädiktionsschritt des Kalmanfilters durchgeführt (2). Anschließend wird der Messvektor $z$ berechnet, der aus der gemessenen Punktwolke in `measured_points` und der momentanen Zustandschätzung `filter.x` abgeleitet wird (3). Die Berechnung des Messvektors ist in Abbildung A.4 im Detail wiedergegeben. Der Messvektor wird schließlich in die Zustandschätzung eingebracht (Kalmanfilter-Korrekturschritt), hierzu wird er über die Funktion `observe_scaled` zusammen mit dem zu verwendenden Messmodell an das Filterobjekt übergeben (4). Im Rahmen des Korrekturschrittes werden die $σ$-Punkte berechnet (4.1) und durch das Messmodell transformiert (4.2). Die Auswertung des Messmodells ist in Abbildung A.5 dargestellt.
A.2. ABLAUF BEI DER VERARBEITUNG EINES EINZELBILDSATZES 71

Abbildung A.4: Kollaborationsdiagramm zur Berechnung des Messvektors z (oben). 

\[ n \] ist die Anzahl verwendeter Stereoskamerae. Die Entfernung verdeckter Punkte 
gechieht mit Hilfe eines synthetisierten Tiefenbildes. Dieses wird erzeugt, indem ei-
ne polygonale Repräsentation (links unten) des Oberkörperrmodells mittels 
\texttt{GLPersonModel.draw_body()} (3.1.1.1) in einen OPENGL-Kontext gezeichnet wird. Die 
OPENGL-Library kann gleichzeitig ein Tiefenbild der gezeichneten Szenerie erstellen, 
den sogennannten z-Buffer (rechts unten). Dieser wird von \texttt{GLDepthSynthWidget} ausge-
lesen (3.2).

Abbildung A.5: Kollaborationsdiagramm zur Auswertung des Messmodells.
Literaturverzeichnis


[BS04] BLANCHETTE, Jasmin ; SUMMERFIELD, Mark: C++ GUI Programming with Qt 3. Prentice Hall PTR, 2004


