Interaktionen im dreidimensionalen Raum

Im Kontext von kollaborativen Mixed-Reality-Anwendungen

Christian Blank

HAW Hamburg, Technik und Informatik,
Berliner Tor 7, Hamburg, Germany
christian.blank@hawhamburg.de
http://www.haw-hamburg.de

Zusammenfassung. In der vorliegenden Arbeit werden drei Publikationen vorgestellt, die im Bereich der Gestenerkennung oder der allgemeinen Interaktion im dreidimensionalen Raum angesiedelt sind. Es werden Vor- und Nachteile aufgezeigt und eine Verbindung zur eigenen Arbeit gezogen.

Schlüsselwörter: Mixed-Reality;Gestenerkennung;Interaktion;3D

1 Einleitung

Gesten im Raum können eine alternative Möglichkeit zur Nutzung von Computern sein, sollten aber nicht als Ersatz für die bisherigen Bedienkonzepte (Maus, Tastatur) dienen. Ein zukünftiges Ziel ist es, den Anwender vom Desktop weg in eine ihm vertraute Umgebung zu holen. Dabei muss die Umgebung so geschaffen sein, dass sich Realität und Virtualität verbinden lassen. Durch diese Verbindung und einen Freiraum bei der Interaktion (vgl. multimodale Interaktion) können neue Erlebnisse für den Nutzer erschaffen werden. Die daraus gewonnenen Erkenntnisse können sich auch auf den Forschungsbereich der computergestützten kollaborativen Arbeit (CSCW) auswirken. So wäre es denkbar, eine dreidimensionale Darstellung eines Gebäudes in den korrekten Blickwinkeln der Nutzer anzuzeigen, auf Wunsch zusätzliche Informationen für jeden einzelnen Nutzer bereitzustellen und die gemeinsame Manipulation des Modells mithilfe von Gesten zu ermöglichen.

Wünschenswert ist eine zuverlässige Erkennung von Posen und dynamischen Gesten der Benutzer im dreidimensionalen Raum ohne die Nutzung von am Körper getragener Sensorik. Um diesen Wunsch zu erfüllen, wurden bereits viele mögliche Lösungsansätze entwickelt, die jedoch alle ein gemeinsames Problem haben. Im Falle einer nicht nachvollziehbaren Eingabe durch den Nutzer kommt es zu keiner Reaktion durch das System. Bezogen auf den Einsatz in Mixed Reality kann dieses Fehlen von Feedback zur Verwirrung des Nutzers führen, da sich virtuelle Objekte nicht mehr so verhalten, wie es erwartet wurde. Ein mögliche Abhilfe kann die Einführung von physikbasierten Eingaben schaffen (vgl. [HKI+12], [SYW08], [Pot14]).

Diese Arbeit ist die Weiterführung der Ausarbeitung für Anwendung 1, in der ein Einblick in die Techniken und den Ablauf der Gestenerkennung gegeben wurde ([Bla14]).

1.1 Herausforderung

Im Gegensatz zur zweidimensionalen Gestenerkennung auf berührungsempfindlichen Flächen besitzt die Gestenerkennung von dreidimensionalen Gesten zwei große Herausforderungen, die gelöst werden müssen.

Start-Ende-Problem. Für die Erkennung einer Geste ist es wichtig, sowohl den Beginn als auch das Ende einer Geste zu kennen. Auf Touchgeräten können diese beiden Zeitpunkte mit dem Berühren bzw. dem Loslassen des Touchscreens oder -pads gleichgesetzt werden. Soll eine Geste in der Luft erkannt werden, dann muss anhand der Bewegung der Nutzer abgeschätzt werden, wann eine Geste begonnen und wann geendet hat.

Ungewollte Erkennung. Ein Computer überwacht für die Gestenerkennung alle Bewegungen des Nutzers. So kann es vorkommen, dass auch Bewegungen als Eingaben interpretiert werden, die nicht als solche beabsichtigt waren, etwa weil sie unbewusst oder für einen anderen Kommunikationspartner bestimmt waren. Auf Touchgeräten besteht dieses Problem im Allgemeinen nicht, da der Kontakt mit der Oberfläche hergestellt werden muss.

1.2 Ziele

Ein Ziel dieser Arbeit ist es, ein vertieftes Verständnis zum Thema der dreidimensionalen Gestenerkennung zu erhalten und dieses Verständnis anhand von drei wissenschaftlichen Arbeiten beispielhaft zu demonstrieren. Zudem hilft diese Arbeit bei der genauen Ausrichtung der eigenen Ziele für eine folgende Masterarbeit und zeigt Überschneidungen, aber auch Unterschiede zu aktuellen Forschungsergebnissen.

1.3 Aufbau

Zunächst wird in Abschnitt 1 eine knappe Einführung in das Thema gegeben. In Abschnitt 2 wird die Arbeit [CT07] untersucht. Es folgt [HKI+12] in Abschnitt 3 und [KNQ12] in Abschnitt 4. Anschließend wird in Abschnitt 5 der Schwerpunkt der eigenen Masterarbeit aufgezeigt und mit den vorgestellten Arbeiten verglichen. Zum Abschluss erfolgt eine kurze Zusammenfassung und ein Ausblick auf die noch folgenden Arbeiten.

2 Handgestenerkennung durch SVM-Klassifizierer

In diesem Abschnitt wird die Arbeit [CT07] untersucht. Dabei wird besonders auf zwei Aspekte eingegangen. Zum einen ist das die Erkennung von Gesten durch SVMs[1] und zum anderen die Aushandlung eines endgültigen Ergebnisses bei Verwendung von mehreren unabhängigen SVMs.

2.1 Übersicht und Ziele

Die Autoren Chen und Tseng haben ein System zur Handgestenerkennung entwickelt, dass für Fingerspiele, wie etwa Schere, Stein, Papier verwendet werden kann. Sie sehen dabei eine Geste nur als eine endgültige Haltung der Hand an (vgl. Pose). Zur Erkennung der Gesten werden drei SVMs eingesetzt, die mit unterschiedlichen Ansichten derselben Geste arbeiten.

Die wichtigsten Ziele bei ihrer Arbeit waren eine hohe und stabile Erkennungsrate unter verschiedenen Winkeln und Größen der Handgesten und unterschiedlichen Hautfarben sowie die Fusion der Ergebnisse der Klassifizierer. Neben diesen beiden Punkten wurde auch auf die Performance geachtet.

2.2 Technik

Im folgenden Abschnitt wird das genaue Vorgehen der Autoren beschrieben. Unter anderem werden der Systemaufbau, die Datengrundlage und die Resultate analysiert. Dabei werden auch mögliche Fehler aufgezeigt und es wird auf Lösungsansätze eingegangen, die zu einer Verbesserung führen können.

Drei Webcams werden frontal, links und rechts um die rechte Hand eines Nutzers positioniert. Es wird nur die rechte Hand analysiert. Die Autoren begründen diese Entscheidung mit einer größeren Anzahl an Rechtshändern. Bei einer Aufnahme lösen alle Kameras in etwa zur gleichen Zeit aus. Jeder Kamera wird ein Klassifizierer, bestehend aus einer SVM und einer Pipeline, zur Vorverarbeitung der Bilder zugeordnet.

Es gibt drei Ansichten und drei SVMs. Somit gibt es drei unabhängige Verarbeitungspipelines jeweils bestehend aus Webcam, Vorverarbeitung und SVM.

Trainingsdaten

Die Trainingsdaten für die SVMs stammen von sechs Personen, die für jede Geste 10 Wiederholungen hatten, bei denen sie ihre Hand frei drehen konnten und auch nur einen Teil der Geste zeigen mussten. Da jede der Geste von drei separaten Kameras aufgezeichnet wurde, standen für das Trainingsset 540 Bilder zur Verfügung.

Bevor die Daten für das Training der SVMs verwendet werden können, müssen sie bearbeitet werden. Somit sollen Einflüsse durch Unterschiede, die in der Hautfarbe der Nutzer, in Schatten und Beleuchtung auftreten können, reduziert werden. Alle Bilder werden auf eine Größe von 56 x 42 Pixeln gebracht. Die gewählte Bildgröße entspricht in ihrer Proportion nach Angaben der Autoren in etwa der Proportion einer Hand beim Ausführen einer Geste. Nach der Bearbeitung der Bildgrößen werden die Bilder in Graustufenbilder umgewandelt und die Histogramme werden angeglichen.

[1] Support Vector Machine: binärer Klassifizierer; die heutige Form wurde in [Vap95] vorgestellt

Training der SVMs

Die so erhaltenen Daten werden je nach Kameraposition einem der drei SVMs zugespielt, um die Stützvektoren der Hyperebene zu bestimmen. Die Autoren haben sich für die Verwendung einer radialen Basisfunktion (Gleichung 1) mit der Standardabweichung $\sigma = 8$ als Kernelfunktion entschieden. Ein Problem dabei ist die Verwendung des kompletten Bildes als Feature. Andere Arbeiten entfernen zunächst den Hintergrund und ermitteln dann die Fingerposition, um einen Featurevektor zu erstellen (vgl. [RAHM09]).

$$K(x,x_i) = e^{\frac{-\|x-x_i\|^2}{2\sigma^2}} \tag{1}$$

Da eine SVM nur zwei Klassen voneinander unterscheiden kann, wird eine "One-against-one"-Methode ([CT07]) verwendet, bei der ein Multiklassenproblem auf mehrere binäre Probleme aufgeteilt wird. In dem vorliegenden Fall wäre das dann (Schere, Stein, Papier => (Schere: ja, nein), (Stein: ja, nein), (Papier: ja, nein)). Das Ergebnis ergibt sich aus den Teilergebnissen der einzelnen Probleme.

Test der SVMs

Nachdem die drei SVMs trainiert wurden, können mithilfe von weiteren Daten Tests durchgeführt werden. Dabei haben die Autoren darauf geachtet, dass es keine Schnittmenge zwischen den Trainings- und den Testdaten gibt. Es wurden vier neue Personen eingeladen und von ihnen wurde jeweils die gleiche Anzahl von Aufnahmen gemacht, so dass 360 Bilder für die Testphase zur Verfügung standen. Diese Bilder wurden mit derselben Pipeline aufbereitet, wie auch die Lerndaten. Anschließend wurden die Daten, entsprechend ihres Ursprungs, einer der SVMs zugeführt und ausgewertet. Die Ergebnisse wurden mit den tatsächlichen Werten verglichen. Aus diesen Informationen wurde die Erkennungsgenauigkeit für jeden Klassifizierer bei jeder Geste und die durchschnittliche Erkennungsrate je Klassifizierer ermittelt.

Zusammenfassung der Ergebnisse der Klassifizierer

Für die endgültige Erkennung einer Geste werden die Zwischenergebnisse der einzelnen Klassifizierer zusammengefasst. Dabei wurden drei verschiedene Strategien vorgestellt:

1. Voting: Jeder Klassifizierer besitzt eine Stimme und alle Stimmen haben das gleiche Gewicht.
2. Fusion:
   (a) Die Stimme der Klassifizierer wird mit der Erkennungsrate der ausgewählten Geste aus den Tests multipliziert.
   (b) Die Stimme der Klassifizierer wird mit der durchschnittlichen Erkennungsrate aus den Tests multipliziert.

Abschließende Tests und Auswertung

In einem weiteren Testlauf mit 540 Bildern und der gleichen Vorverarbeitung wie bei den Lerndaten wurden die verschiedenen Strategien gegeneinander getestet und das Ergebnis zeigt, dass die Erkennungsrate von Strategie 2 b am besten abschneidet. Ob Personen als Tester verwendet wurden, die bereits zuvor mit dem System in Berührung gekommen sind, geht nicht aus der Arbeit hervor. Mit $93,33\%$ liegt sie deutlich über der durchschnittlichen Erkennungsrate der Frontansicht ($73,3\%$) und ist auch ein wenig besser als die Erkennungsrate der rechten Ansicht ($92,5\%$). Daraus kann man schließen, dass eine Zusammenführung von Ergebnissen eine Verbesserung der Gesamtleistung bei der Erkennung gegenüber der Erkennung von einzelnen Positionen bringt.

Die schlechten Werte der Frontansicht können durch den großen Anteil an Hintergrundinformationen im Verhältnis zur Gesteninformation in den Aufnahmen der Frontkamera begründet sein. Die Autoren schlagen eine Veränderung der Kameraposition an eine höher gelegene Stelle vor ([CT07]). Ob diese Änderung eine Verbesserung mit sich bringt, wurde nicht weiter untersucht. Ebenfalls kann das durchgeführte Resizing Fehler verursachen, da es die Hand deformiert.

2.3 Zusammenfassung

Das System ist in der Lage, Handgesten in verschiedenen Winkeln zu erkennen, wobei die Nutzer nicht in der Ausrichtung ihrer Hand eingeschränkt sind. Auch die Größe der Hand und die Hautfarbe der Person haben keinen Einfluss auf die Ergebnisse.

Es muss jedoch berücksichtigt werden, dass sich nur auf drei Gesten konzentriert wurde. Für die Bedienung eines Computersystems müssten jedoch wesentlich mehr Gesten zur Verfügung stehen. Zudem werden die drei Gesten nur durch eine Pose und nicht durch einen Bewegungsablauf definiert, was eine weitere Schwäche des vorgestellten Systems aufzeigt. Die Erkennung der Handgesten erfolgt nur auf der Basis von Einzelbildern und umgeht somit Probleme der Synchronisation der Kameraaufnahmen und der Erkennung von Start- und Endpunkten von Gesten[2].

In den vorherigen Abschnitten wurde das von Chen und Tseng entwickelte System zur Handgestenerkennung analysiert und bewertet. In den Abschnitten 2.2 und 2.3 wurde auf Vor- und Nachteile eingegangen. Die Verbindung zur eigenen Arbeit wird in Abschnitt 5 erläutert.

3 Physikbasierte Interaktion unter Verwendung von Partikeltracking

Die Arbeit [HKI+12] von Hilliges et al. besitzt eine Sonderstellung in der Reihe der Publikationen, die in dieser Ausarbeitung untersucht werden, da sie sich nicht mit Gestenerkennung, sondern mit der Interaktion im 3D-Raum befasst. Die Arbeit ist dabei in zwei Abschnitte unterteilt. Im ersten Teil werden das Setup und die Algorithmen beschrieben und im zweiten Teil werden umfangreiche Tests und Nutzerstudien erstellt und ausgewertet. Für diese Ausarbeitung hat der erste Teil eine wesentlich höhere Relevanz, sodass auf die Ergebnisse des zweiten Teils nur kurz eingegangen wird.

Dieser Ansatz ist deshalb interessant, da er eine erweiterte Möglichkeit der Interaktion bietet und mit Gestenerkennung kombiniert werden könnte (Abschnitt 5.2).

3.1 Zielstellung

Die von Microsoft Research in Kooperation mit der RWTH Aachen entstandene Arbeit versucht mithilfe der Modellierung von physikalischen Eigenschaften von virtuellen Objekten eine Interaktion zwischen realen und virtuellen Objekten zu erzielen. Dabei können sowohl starre Objekte, wie Bücher oder Schalen, als auch verformbare Objekte, wie etwa Hände oder Blätter, zur Interaktion verwendet werden.

Die Autoren haben einen Tisch konzipiert und gebaut, bei dem der Nutzer durch ein transparentes Display einen Interaktionsbereich einsehen kann. Der Nutzer ist dabei in der Lage, mit seinen Händen in diesen Bereich zu greifen und mit virtuellen Objekten, die über das Display angezeigt werden, zu interagieren.

3.2 Umsetzung

Der Holodesk, den die Autoren entwickelt haben, zeichnet sich gegenüber anderen Tabletop-Systemen dahingehend aus, dass ein Nutzer keine weiteren Sensoren oder Brillen tragen muss ([NMKT07], [PB11]). Der Wunsch nach “Come as you are” ([DBG13]) wird somit stark in den Fokus gerückt.

In diesem Abschnitt wird zunächst der Aufbau des Holodesks beschrieben und es wird kurz erklärt, wie das Gefühl der Tiefe für den Nutzer erzeugt wird. Anschließend wird ein Einblick in das Partikeltracking gegeben, dass zur Erkennung von Objekten im Interaktionsbereich verwendet wird. Im letzten Teil werden die Ergebnisse der Arbeit erläutert.

Aufbau von Holodesk

Die Ausgabe des Holodesks besteht aus einem oberhalb des Kopfes montierten LCD und einem transparenten Spiegel, der das Bild des LCD reflektiert und gleichzeitig die Sicht in den Interaktionsbereich zulässt. Als Sensoren kommen eine Kinect-Kamera und eine Webcam zum Einsatz. Die Kinect-Kamera ist so ausgerichtet, dass sie den gesamten Interaktionsbereich abdeckt. Die Webcam wird auf den Kopf des Nutzers ausgerichtet und für das Headtracking verwendet. Auf die verwendete Methode wird nicht weiter eingegangen und es wird nur der Hinweis gegeben, dass sie in OpenCV implementiert ist.

[2] Das Problem der Bestimmung von Start- und Endpunkt von Nutzereingaben wird in der Literatur oft auch als König-Midas-Problem bezeichnet. In einer Sage erhält der gierige König Midas die Gabe alles was er berührt in Gold zu verwandeln. Diese Gabe wird ihm zum Verhängnis, da er somit keine Nahrung mehr zu sich nehmen kann.

Für die Kalibrierung der Kameras und des LCD werden Schachbrettmuster verwendet und die extrinsischen Faktoren miteinander verrechnet, um die absoluten Positionen der einzelnen Geräte zu erhalten. Durch das Wissen über die Positionen der Kameras, des LCD und des Kopfes kann der Viewport in Abhängigkeit von der Kopfposition und -ausrichtung korrekt berechnet werden. Die dadurch erreichte Bewegungsparallaxe in Kombination mit der berechneten Überdeckung von realen und virtuellen Objekten reichen aus, um einen sehr guten Eindruck von räumlicher Tiefe zu erhalten.

Die Meshes, die für die Ermittlung von Überdeckungen benötigt werden, werden in mehreren Schritten auf der Grafikkarte unter Verwendung von CUDA und HLSL[3] berechnet. Zu Beginn wird ein Tiefenbild als Referenzhintergrund erstellt. Dafür wird der Durchschnitt aus mehreren Tiefenbildern der Kinect-Kamera ohne Objekte im Interaktionsraum ermittelt.

1. Separierung des Vordergrundes unter Verwendung des Referenzhintergrundes
2. Anwendung eines bilateralen Filters zur Reduzierung von Rauschen
3. Laden eines flachen Meshes in GPU-Speicher
4. Parallele Projektion eines Pixels des Tiefenbildes als 3D-Punkt und Speicherung des Punktes in Textur
5. Z-Werte des Meshes werden auf Basis der Textur verändert
6. Berechnung der Normalen anhand des Kreuzproduktes der Vektoren zu Nachbarpunkten
7. Filterung von falschen Tiefenwerten

> Dieser Ansatz erlaubt das Erstellen von Meshes von physikalischen Objekten in Echtzeit unter Verwendung eines kompletten Tiefenbildes. [HKI+12]

Durch das Mesh können Überdeckungen von virtuellen Objekten, die durch reale Objekte entstehen, berechnet werden. Ebenfalls ist ein realistischer Schattenwurf in die virtuelle Szene möglich.

Partikeltracking

Neben dem Erzeugen von Meshes ist das Partikeltracking ein weiterer wichtiger Berechnungsschritt, der ebenfalls rechenintensiv ist. Es wird für die Bewegungsabschätzung und die physikalische Repräsentation von realen Objekten zur Interaktion mit virtuellen Objekten verwendet.

Für die Aktualisierung der Partikelpositionen wird das Verfahren **Depth-Aware Optical Flow** verwendet. Dabei wird der Versatz von Pixeln nicht direkt auf den Tiefenbildern berechnet, sondern es wird das RGB-Bild der Kinect verwendet, da es besser texturiert ist und somit robustere Ergebnisse liefert.

1. Rektifizierung von RGB- und Tiefenbild in Frame i und $i+1$
2. Vordergrundsegmentierung der RGB-Bilder
3. Berechnung des optischen Flusses unter Verwendung der Energiefunktion unter Gleichung 2
4. Berechnung des Versatzes im Tiefenbild
5. Aktualisierung der Positionen der Partikel

Partikel werden gelöscht, wenn sie mit Hintergrundpixeln übereinstimmen oder eine Lebenszeit von 150 Frames überschritten haben. Die Partikel als Gesamtheit geben eine gute Approximation der realen Objekte und können sowohl Festkörper als auch verformbare Objekte widerspiegeln. Durch sie können Kräfte in der Physiksimulation modelliert werden, wie etwa seitlich wirkende Kräfte oder Reibungskräfte.

$$arg\,min \int_u (I_i(u) - I_{i+1}(u + p_i(u)))^2 + \alpha * \nabla p_1(u)^2 \tag{2}$$

Ergebnisse

In ihren Nutzerstudien gehen die Autoren auf verschiedene Problemstellungen ein und untersuchen das Verhalten der Nutzer bei freier Bedienung und bei vorgegebenen Aufgaben. Dabei werden unterschiedliche Setups, wie etwa indirekte Eingabe und direkte Eingabe oder monoskopische und stereoskopische Wiedergabe verwendet.

[3] High Level Shader Language

Eine indirekte Eingabe hat wesentlich länger gedauert als die direkte Eingabe (bis zu 370ms langsamer). Die Eingabe unter Verwendung von stereoskopischer Wiedergabe ist schneller gewesen, als die bei monoskopischer Wiedergabe, hat aber bei einigen Nutzern nach kurzer Zeit zu Unwohlsein geführt. Ebenso hat sie dem Ziel, dass ein Nutzer keine Geräte tragen muss, widersprochen, da bei Verwendung von steroskopischer Wiedergabe eine entsprechende Brille getragen werden muss. Ein wichtiger Faktor für die Bearbeitungszeit einer Aufgabenstellung war die Positionierung der Elemente. So hat es einen Unterschied gemacht, ob ein Element vor, auf oder hinter der Bildschirmebene lag.

3.3 Zusammenfassung

Ein großer Vorteil der vorgestellten Arbeit zeigt sich in der Vielfalt an realistischen Interaktionsmöglichkeiten zwischen verschiedenen realen und virtuellen Objekten. So ist es beispielsweise möglich, einen virtuellen Ball auf einem realen Papier herunter rollen zu lassen, um ihn anschließend in der Hand aufzufangen und ihn in die Luft zu werfen. Die Autoren haben zudem eine sehr ausführliche Nutzerstudie unter Berücksichtigung verschiedener Szenarien durchgeführt.

Da die Szene von der Kinect-Kamera von oben betrachtet wird, kann es leicht zu Überdeckungen kommen, wenn ein Nutzer versucht, ein Objekt von oben zu greifen. Da das System keine Annahmen über die Objekte im Interaktionsbereich macht, können fehlende Informationen nicht interpoliert werden. Eine weitere Möglichkeit zur Lösung wäre die Verwendung von mehreren Kinect-Kameras, was jedoch zu Problemen bei der Berechnung der Meshes in Echtzeit führen würde. Die Verwendung von einer rein physikbasierter Interaktion ohne Interpretation hat den Nachteil, dass das System kein Wissen über die Interaktion besitzt.

4 Kontinuirliche Gestenerkennung auf Basis von Gestentemplates

Kristensson et al. stellen in ihrer Arbeit eine Methode vor, mit der Gesten, die mit den Händen und Armen ausgeführt werden, erkannt werden können. Dabei verwenden sie einen auf Wahrscheinlichkeitsverteilung basierenden Algorithmus, der in jedem Schritt eine Prognose erstellt. Der Algorithmus vergleicht dabei die Bewegung des Nutzers mit einer Sammlung von Gestentemplates und ermittelt die Wahrscheinlichkeit für jede Geste ([KNQ12]).

4.1 Zielstellung

Die Autoren haben einen Algorithmus entwickelt, der auf Basis von Wahrscheinlichkeitsverteilung eine von einem Nutzer gemachte Geste jeder Geste, eines zuvor definierten Gestenalphabets, eine Wahrscheinlichkeit zuordnet und diese Wahrscheinlichkeit mit dem Fortschreiten der Geste weiter aktualisiert.

Ein Ziel der Autoren war es, die Gesten in einem kontinuierlichen Strom von Eingaben zu erkennen. Um dieses Ziel zu erreichen, arbeiten sie mit einer Input-Zone, in dessen Bereich Eingaben akzeptiert werden. Ein Großteil der Arbeit basiert auf einer bereits veröffentlichten Arbeit der Autoren ([KD11]), in der sie einen Algorithmus zur Erkennung von Gesten bei Stift- und Toucheingabe vorgestellt hatten. In der darauf aufbauenden Arbeit wurde der Algorithmus so angepasst, dass er dreidimensionale Eingaben von zwei Händen analysieren konnte.

Ebenfalls sollte geklärt werden, ob Gesten mit einer oder mit zwei Händen besser erkannt werden können und ob die Wahl der Eingabemethode und der Distanzfunktion Auswirkungen auf die Erkennung von unvollständigen Gesten hat. Die Autoren wollten dem Nutzer eine Möglichkeit bieten, eine Fehlerkennung noch während der Eingabe zu korrigieren.

4.2 Umsetzung

In diesem Abschnitt werden zunächst der Algorithmus aus [KD11] und die gemachten Anpassungen vorgestellt und es wird auf die verwendeten Distanzmaße eingegangen. Anschließend wird das Konzept der Input-Zone erklärt und zum Schluss werden die Ergebnisse der Untersuchungen vorgestellt.

Algorithmus

Zunächst der ursprüngliche Algorithmus:

1. Berechnung der Liklihood $P(I_i|\omega_j)$ für jedes Template ω_j im Alphabet ω anhand von Nutzer-Input I, wobei I n Punkte besitzt und I_i ein Teilinput mit $0 < i \leq n$.
   (a) Berechnung $P_l(I_i|\omega_j) = arg \max_{S_k \in S_j \in \omega_j} D(I,S)$. $D()$ ist eine Distanzfunktion.
   (b) Berechnung der Wahrscheinlichkeit des Endzustandes $E(I_i|\omega_j)$ (Gleichung 4). S_{last} ist das letzte Segment in w_j.
2. Anwendung des Satzes von Bayes für jedes Template ω_j (Gleichung 5). Die a-priori-Wahrscheinlichkeit für ein Template ist $P(\omega_j)$. k ist der Index für die Iteration über das Gestenalphabet.

Der Algorithmus wird für jeden Punkt der Eingabe wiederholt und berechnet die Wahrscheinlichkeit für jedes Template des Alphabets. Für eine bessere Aussage kann der Mittelwert der Wahrscheinlichkeiten für eine einzelne Gesten über mehrere Zeitpunkte berechnet werden.

$$P(I_i|\omega_j) = P_l(I_i|\omega_j)E(I_i|\omega_j) \tag{3}$$

$$E(I_i|\omega_j) = 1 + \kappa exp\left(-(1 - D(I_i, S_{last}))^2\right) \tag{4}$$

$$P(\omega_j|I_i) = \frac{P(\omega_j)P(I_i|\omega_j)}{\sum_k P(\omega_k)P(I_i|\omega_k)} \tag{5}$$

Der vorgestellte Algorithmus wurde so angepasst, dass er nun Vektoren für die linke und die rechte Hand verarbeiten kann, wie in Gleichung 6 zu sehen ist. Die Vektoren bestehen dabei jeweils aus einem Tupel (x,y,z,t). Wenn sich die Hand in der Input-Zone befindet, dann wird eine Projektion in den 2D-Raum vorgenommen und der Z-Anteil der Koordinate sowie der Zeitstempel verworfen ($I_i = [(x_1,y_1),(x_2,y_2),...,(x_i,y_i)]$ als Beispiel für die linke Hand). Mit den erhaltenen Daten können anschließend Vergleiche mit dem Gestenalphabet gemacht werden. Im Vergleich zu [KD11] sind in [KNQ12] 16 Gesten zum Gestenalphabet hinzugekommen, die mit zwei Händen simultan eingegeben werden.

$$P^{(lr)}(\omega_k^{(lr)}|I_i^{(l)}|I_j^{(r)}) = \frac{P(\omega_k^{(lr)})P(I_i^{(l)}|\omega_k^{(l)})P(I_j^{(r)}|\omega_k^{(r)})}{\sum_n P(\omega_n^{(lr)})P(I_i^{(l)}|\omega_n^{(l)})P(I_j^{(r)}|\omega_n^{(r)})} \tag{6}$$

Als Distanzmaß (Gleichung 7) wird eine gewichtete Kombination aus den Mittelwerten von euklidischer Distanz (Gleichung 8) und Drehwinkel (Gleichung 9) verwendet. Die Gewichtung kann über λ beeinflusst werden, σ_e und σ_t sind geschätzte Varianzen für die beiden Maßen. Die Parameter I und S sind zwei Vektoren, die verglichen werden sollen. I repräsentiert die Eingabe des Nutzers und S ist ein Template. Sie bestehen aus den Punkten $a_1,a_2,...,a_n$ respektive $b_1,b_2,...,b_n$. d_t ist der Winkel zwischen zwei Linien in Radiant ([KD11]).

$$D(I,S) = exp\left(-\left[\lambda\left(\frac{x_e^2}{\sigma_e^2}\right) + (1-\lambda)\left(\frac{x_t^2}{\sigma_t^2}\right)\right]\right) \tag{7}$$

$$x_e = \frac{1}{n}\sum_{i=1}^{n} \|a_i - b_i\| \tag{8}$$

$$x_t = \frac{1}{n-1}\sum_{i=2}^{n} d_t(a_i, a_{i-1}, b_i, b_{i-1}) \tag{9}$$

Input-Zone

Die Verwendung einer Input-Zone ist keine neue Idee (vgl. [PBWI96]), doch sind bisher eher fest definierte Bereiche als Zone eingesetzt worden. Diese Zonen sind sehr robust, können aber bei der Eingabe hindern, da sie die Erwartungen der Nutzer meist nicht erfüllen. Die Autoren haben sich jedoch für die Verwendung eines relativen Maßes entschieden, dass die Geschwindigkeit der Hand und des Körpers sowie die Entfernung der Hand zum Körper berücksichtigt (Gleichung 10). v_b ist die Geschwindigkeit des Körpers, v_h die Geschwindigkeit der Hand und d_h ist die Distanz zwischen Hand und Körper des Nutzers. γ_{v_b}, γ_{v_h} und γ_{d_h} sind empirisch ermittelte Grenzwerte ([KNQ12]). Für die Ermittlung wurden mehrere Nutzer mit einem Controller ausgestattet, den sie betätigen mussten, wenn sie vor und nach dem Ausführen einer Geste der Meinung waren, die Input-Zone zu betreten bzw. zu verlassen.

$$z(v_b, v_h, d_h) = \begin{cases} 1, & \text{falls } v_b < \gamma_{v_b} \wedge v_h < \gamma_{v_h} \wedge d_h > \gamma_{d_h} \\ 0, & \text{sonst} \end{cases} \tag{10}$$

Ergebnisse

Auf Basis von 1700 Gesten von 18 unterschiedlichen Nutzern wurden Vergleiche in Bezug auf die Eingabeverfahren und die gewählten Distanzfunktionen mit vollständigen und unvollständigen Gesten aufgestellt. Es konnte festgestellt werden, dass Zweihandgesten besser als Einhandgesten erkannt werden. Es gibt 16 Zweihandgesten und im Vergleich dazu 43 Einhandgesten in dem verwendeten Gestenset und beide Erkennungsraten fallen relativ hoch aus (92,3 % bei der Erkennung von Einhandgesten und 96,2 % bei Zweihandgesten).

Bei der Verwendung der euklidischen Distanz ohne Drehwinkel sinkt die Erkennungsrate um bis zu 10 %. Eine Kombination aus beiden Distanzfunktionen führt zu minimal besseren Ergebnissen als die Verwendung von Drehwinkeln allein.

Durch die kontinuierliche Analyse ist das System in der Lage, schon nach 20 % der ausgeführten Geste mit einer Wahrscheinlichkeit von etwa 51 % die korrekte Geste zu bestimmen. Im Gegensatz dazu wird bei einem System, das eine Eingabe mit dem vollständigen Template vergleicht, die richtige Geste mit einer Wahrscheinlichkeit von 12 % erkannt ([KNQ12]).

4.3 Zusammenfassung

In der Arbeit von Kristensson et al. werden viele interessante Ansätze vorgestellt. Die meisten der Ideen stammen aus zuvor veröffentlichten Arbeiten, wurden in der Form aber noch nicht zusammen gezeigt. So bietet die Verwendung von gewichteten Distanzmaßen eine flexiblere Lösung, die auch zu Testzwecken gut verwendet werden kann. Die Definition einer relativen Input-Zone und die kontinuierliche Analyse auf Basis von Gestentemplates ist ein interessanter Ansatz für die Lösung des Start-Ende-Problems. Das verwendete Gestenset enthält bereits eine Vielzahl an Gesten und ist darüber hinaus leicht erweiterbar.

Das Gestenset für Zweihandgesten ist kleiner als das für Einhandgesten, sodass Vergleiche zwischen den Ergebnissen der Erkennung nicht unbedingt aussagekräftig sind. Als Eingabe wird das komplette Skelett der Kinect-Kamera verwendet, aber für die Erkennung der Geste werden nur die Bewegungen der Handflächen analysiert. Eine Ausrichtung des Nutzers oder die Körperhaltung wird ebenso nicht berücksichtigt wie die Bewegung der Finger. Ein weiterer negativer Punkt ist die einfache Projektion $R^4 \rightarrow R^2$ durch $(x,y,z,t) \rightarrow (x,y)$. Nutzereingaben in der Tiefe werden somit komplett verworfen.

5 Bezug zur eigenen Arbeit

In den drei voran gegangenen Abschnitten wurden die Arbeiten von Chen und Tseng, Hilliges et al. und Kristensson et al. vorgestellt und es wurden Vor- und Nachteile aufgezeigt. Im folgenden Abschnitt werden die drei Arbeiten im Hinblick auf die eigene Masterarbeit bewertet.

5.1 Chen und Tseng

Der vorgestellte Algorithmus zur Erkennung von Gesten ist nicht für die Analyse von Bewegungen geeignet. Um neue Gesten hinzuzufügen, müssen Test- und Trainingsdaten erstellt und von allen drei SVMs verarbeitet werden. Die Strategien der Zusammenführung sind sehr einfach und wenig flexibel. Der derzeitige Plan für die eigene Arbeit sieht keine Verwendung von mehreren Recognizern im fertigen System vor. Aus der jetztigen Sicht werden also keine Ideen von dieser Arbeit übernommen. Es lässt sich jedoch die Frage ableiten, wie man einen Deskriptor bzw. Merkmalsvektor für eine Geste modellieren kann, wenn die Geste über mehr als nur ein Frame andauert und der dynamische Anteil der Geste mit in die Auswertung einfließen soll.

5.2 Hilliges et al.

Die Arbeit von Hilliges et al. beinhaltet den Ansatz der Interaktion durch Simulation von physikalischen Eigenschaften. Allein durch diesen Ansatz lassen sich eine große Anzahl von Interaktionsmöglichkeiten durchführen. Eine direkte Kommunikation mit dem Computer kann durch diese Art der Bewegungserkennung aber nicht entstehen, da es keine Interpretation von Bewegungen gibt.

In der eigenen Arbeit soll die Erkennung von Gesten im Mittelpunkt stehen, doch auch die Unterstützung von physikbasierter Verarbeitung der Eingabe ist angedacht. Durch die Kombination der beiden Ansätze können möglicherweise Schwächen, die die beiden Ansätze jeweils besitzen, ausgeglichen werden. Zum Beispiel kann eine physikbasierte Interpretation erfolgen, wenn kein passende Geste gefunden wurde. Ob dieser Ansatz interaktive Reaktionszeiten ermöglicht, bleibt bisher offen und muss in den folgenden Arbeiten untersucht werden.

5.3 Kristensson et al.

Es wurden viele interessante Ideen in der Arbeit vorgestellt, so etwa die relative Input-Zone, die Verwendung von Gestentemplates und die Nutzung eines gewichteten Distanzmaßes. In der eigenen Arbeit sollen viele dieser Ideen verwendet werden, jedoch soll sich die eigene Arbeit in einem entscheidenden Punkt von der Arbeit von Kristensson et al. abheben. Ein großer Kritikpunkt und eine Vereinfachung der Probleme, die im Bereich der dreidimensionalen Gestenerkennung auftreten, ist die Projektion der Gestenkoordinaten auf eine kleinere Dimension. Diese Projektion soll in der eigenen Arbeit nicht ausgeführt werden. Es ist also wichtig, eine mögliche Repräsentation für Gesten zu finden, die als Templates verwendet werden können. Diese müssen dann zum Teil invariant gegen Rotation und Skalierung sein.

Ebenfalls wird sich die eigene Arbeit darin unterscheiden, dass eine Nicht-Erkennung von Gesten zu einer Weiterverarbeitung führen kann, wie bereits in Abschnitt 5.2 beschrieben.

5.4 Weitere Einflüsse

Kontextwahl. Eine Kontextwahl kann von Vorteil sein, wenn eine Eingabe durch eine Geste im allgemeinen Fall zu ungenau ist. Es stehen verschiedene Ansätze (vgl. [GBB10], [KD11]) zur Auswahl und die Entscheidung für oder gegen einen Ansatz kann voraussichtlich nur durch die Gegenüberstellung beider Lösungen in einer Evaluation erfolgen.

Abstraktionsschicht. Bereits in [EGG+03] wurde auf die Verwendung einer Abstraktionsschicht eingegangen, um bei der Erkennung von Gesten unabhängig von den eigentlichen Sensoren zu sein. Diese Idee soll auch in der eigenen Arbeit aufgegriffen werden. Als Abstraktionsschicht kommt Trame[4] zum Einsatz. Dadurch kann die Analyse der Gestenerkennung immer auf Skeletten arbeiten und ist unabhängig von Änderungen der Sensoren oder der Schnittstellen.

[4] `https://intergitlab.informatik.haw-hamburg.de/christian/trame`

6 Fazit

In dieser Arbeit wurden drei Konzepte für Interaktion im dreidimensionalen Raum vorgestellt. Das erste und das dritte Konzept befassen sich mit der Erkennung von Gesten, das zweite behandelt die physikbasierte Interaktion.

In diesem Abschnitt wird der aktuelle Stand kurz umrissen und es wird ein kurzer Ausblick auf die noch folgenden Arbeiten gegeben.

6.1 Aktueller Stand

Im Hinblick auf die Masterarbeit wurde in Projekt 1 ein Konzept für die Gestenerkennung erstellt und es wurde Trame implementiert und getestet. Für einen ersten Prototypen wurden vereinfachte Regeln für ein heuristisches Verfahren aufgestellt und implementiert. Die derzeitige Version der Gestenerkennung erkennt einfache Körpergesten, wie etwa gehobene Hände und Arme.

Einen umfassenden Überblick über den bisherigen Stand erhält man im dem Bericht zu Projekt 1[5].

6.2 Ausblick

Für Projekt 2 ist geplant, die Implementierung voranzubringen und dabei auf einen zu [KNQ12] vergleichbaren Ansatz umzusteigen. Verschiedene Fragen wurden in der vorliegenden Arbeit gestellt und sollen am Ende von Projekt 2 beantwortet werden können. Neben der reinen Implementierung soll auch eine Evaluierung erfolgen. Der Rahmen dafür muss noch abgesteckt werden.

Ebenfalls ist die Integration der Gestensteuerung als weitere Komponente in den Mixed-Reality-Tisch von I^2E angedacht.

Literatur

[Bla14] BLANK, Christian: *Gesten im dreidimensionalen Raum*. `http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2013-aw1/blank/bericht.pdf`. Version: März 2014

[CT07] CHEN, Yen-Ting ; TSENG, Kuo-Tsung: Multiple-angle Hand Gesture Recognition by Fusing SVM Classifiers. In: *Automation Science and Engineering, 2007. CASE 2007. IEEE International Conference on*, 2007, S. 527–530

[DBG13] DIONISIO, John David N. ; BURNS, William G. ; GILBERT, Richard: 3D Virtual Worlds and the Metaverse: Current Status and Future Possibilities. In: *ACM Comput. Surv.* 45 (2013), Juli, Nr. 3, 34:1–34:38. `http://dx.doi.org/10.1145/2480741.2480751`. – DOI 10.1145/2480741.2480751. – ISSN 0360–0300

[EGG+03] EISENSTEIN, Jacob ; GHANDEHARIZADEH, Shahram ; GOLUBCHIK, Leana ; SHAHABI, Cyrus ; YAN, Donghui ; ZIMMERMANN, Roger: Device independence and extensibility in gesture recognition. In: *Virtual Reality, 2003. Proceedings. IEEE* IEEE, 2003, S. 207–214

[GBB10] GUSTAFSON, Sean ; BIERWIRTH, Daniel ; BAUDISCH, Patrick: Imaginary Interfaces: Spatial Interaction with Empty Hands and Without Visual Feedback. In: *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*. New York, NY, USA : ACM, 2010 (UIST '10). – ISBN 978–1–4503–0271–5, 3–12

[HKI+12] HILLIGES, Otmar ; KIM, David ; IZADI, Shahram ; WEISS, Malte ; WILSON, Andrew: HoloDesk: direct 3d interactions with a situated see-through display. In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems* ACM, 2012, S. 2421–2430

[KD11] KRISTENSSON, Per O. ; DENBY, Leif C.: Continuous recognition and visualization of pen strokes and touch-screen gestures. In: *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling* ACM, 2011, S. 95–102

[KNQ12] KRISTENSSON, Per O. ; NICHOLSON, Thomas ; QUIGLEY, Aaron: Continuous Recognition of One-handed and Two-handed Gestures Using 3D Full-body Motion Tracking Sensors. In: *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*. New York, NY, USA : ACM, 2012 (IUI '12). – ISBN 978–1–4503–1048–2, 89–92

[NMKT07] NAKASHIMA, Kousuke ; MACHIDA, Takashi ; KIYOKAWA, Kiyoshi ; TAKEMURA, Haruo: A 2D–3D integrated tabletop environment for multi-user collaboration. In: *Computer Animation and Virtual Worlds* 18 (2007), Nr. 1, S. 39–56

[PB11] PRACHYABRUED, Mores ; BORST, Christoph W.: Dropping the ball: Releasing a virtual grasp. In: *3D User Interfaces (3DUI), 2011 IEEE Symposium on* IEEE, 2011, S. 59–66

[5] `http://i2e.informatik.haw-hamburg.de/assets/docs/p1/p1_blank_2014.pdf`

[PBWI96] POUPYREV, Ivan ; BILLINGHURST, Mark ; WEGHORST, Suzanne ; ICHIKAWA, Tadao: The Go-go Interaction Technique: Non-linear Mapping for Direct Manipulation in VR. In: *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology*. New York, NY, USA : ACM, 1996 (UIST '96). – ISBN 0–89791–798–7, 79–80

[Pot14] POTRATZ, Olaf: *Ein Framework für physikbasierte 3D Interaktion mit großen Displays*, HAW Hamburg, Diplomarbeit, 2014

[RAHM09] RASHID, O. ; AL-HAMADI, A. ; MICHAELIS, B.: A framework for the integration of gesture and posture recognition using HMM and SVM. In: *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on* Bd. 4, 2009, S. 572–577

[SYW08] SONG, Peng ; YU, Hang ; WINKLER, Stefan: Vision-based 3D finger interactions for mixed reality games with physics simulation. In: *Proceedings of The 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry* ACM, 2008, S. 7

[Vap95] VAPNIK, Vladimir N.: *The nature of statistical learning theory*. New York : Springer-Verlag, 1995