Multimodale Haussteuerung

Projekt 1 Ausarbeitung

Sobin Ghose

Sommersemester 2015

Diese Ausarbeitung liefert einen Einblick zum Zwischenstand der Projekarbeit in Vorbereitung zur Masterarbeit. Dabei sollen die technischen Grundlagen und die Infrastruktur für multimodale Interfaces zur gestenbasierten Steuerung eines Smart Homes konzeptioniert und entwickelt werden. Zudem wird das Vorgehen während des Projekts und die Weiterentwicklung des Systems zu einem Interaktion-Managment-Systmes beschrieben.

Einführung und Kontext

1 Motivation und Ziele

Mark Weiser hat durch seine in "The computer for the 21st century"[23] beschriebenen Vision das Bild, wie wir mit Computern umgehen, maßgeblich geprägt. Durch die heutige Rechnerallgegenwart (engl. ubiquitous computing) bieten natürliche Schnittstellen eine intuitive Möglichkeit mit unserer Umgebung zu interagieren. So wird aktuell bereits Sprache zur Steuerung des Autonavigationsgeräts oder des Smart Phones genutzt, sowie 2D-Gesten zur Steuerung eines Touchscreens oder 3D-Gesten als Alternative zum Controller bei Spielkonsolen.

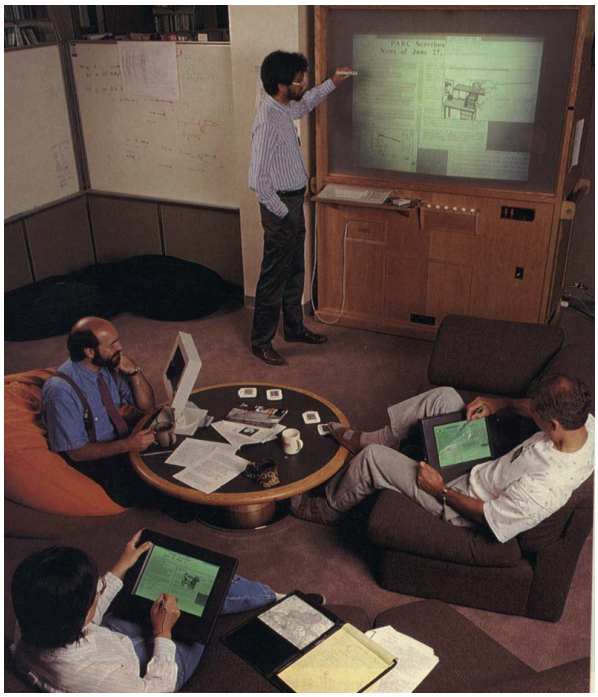


Abbildung 1: Smart Board und Tablets als neue Bedienungskonzepte [23]

Grundsätzlich können einzelne natürliche Schnittstellen lediglich für beschränkte Aufgaben genutzt werden, da umfangreichere Anwendungen weiteres Wissen über den Kontext benötigen. So zeigte bereits 1992 David McNeill [21] auf, dass Rückschlüsse über die Intention einer Geste oder eines Satzes häufig nur unter Betrachtung eines anderen Kommunikationskanals möglich ist. Dies ist insbesondere bei der Interaktion mit Smart Environments und Companion Systemen zu beachten. Auf diesen Zusammenhang wurde bereits

detailliert in der Anwendungen 1 Ausarbeitung[6] eingegangen.
Nachdem in der Anwendungen 2 Ausarbeitung [8] Ansätze für die Sensor Fusion, sowie die Konzeption und den Aufbau eines Interaktion-Managment-Systems vorgestellt wurden, wird in dem Grundprojekt und dem Hauptprojekt die Basis eines Prototypen zur Echtzeiterkennung von Gesten und Sprache entwickelt.

2 Aufbau der Ausarbeitung

Nachdem in der Motivation (1), ein Überblick über die Hintergründe und Ziele des Projekts gegeben wurden, wird im Anschluss die Laborumgebung mit den gegebenen Rahmenbedienungen erläutert (3). Daraufhin wird kurz das Design (3) und die Systemarchitektur des Interaktion-Management-Systems vorgestellt welche eine wohlgeformte Komponententrennung gewährleisten soll. Im Anschluss wird im Kapitel Echtzeiterkennung von Gestenbefehlen (II) auf die verwendeten Hard- und Software Systeme, sowie auf die im Rahmen des Projekts entwickelten Sensor-Agenten eingegangen. Im Anschluss wird auf die Rahmenbedienungen der Sensorfusion (7) eingegangen, sowie auf den entwickelten Fusion-Agenten. Die abschließende Zusammenfassung (III) gibt einen Überblicke der Risiken so wie einen Ausblick über die weitere Entwicklung.

3 Laborumgebung Living Place Hamburg

Dieses Projekt wird im Rahmen der UbiComp Forschungsgruppe des Informatik Departments der HAW Hamburg umgesetzt. Das Living Place Hamburg ist ein Foschungslabor für Smart Home, Ubiquitous Computing sowie Ambient Intelligence sowie für weitere Bereiche.

Im Living Place stehen verschiedenen Sensoren und Aktoren welche über die intrigierte agentenbasierte Middleware verwendet werden können bereit[5]. Das zugrundeliegende Kommunikationskonzept ist dem Publisher-Subscriber Pattern mit der Middleware als Blackboard nachempfunden.
Die bereitgestellte Infrastruktur ermöglicht eine Steuerung der Fenster, Vorhänge, Rollos, so wie des Lichts. Daher bietet das Living Place Hamburg eine hervorragende Umgebung für Wizard-Of-Oz Untersuchungen im HCI Umfeld und ermöglicht eine einfache Umsetzung der gestenbasierten Steuerung der Haustechnik.

Abbildung 2: 3D-Modell des Living Place Hamburg [15]

Für die Einarbeitung in die Infrastruktur der Laborumgebung und der bereit gestellten Middlewaren und Tools wurden sich in folgende Technologien eingearbeitet:

- Git: Versions Contol System für große Projekte
- Scala: Funktionale Programmiersprache welche auf die JVM (JAVA Virtuelle Maschine) aufsetzt

- SBT (Simple Build Tool): Compilierung von Scala sowie Teil des Continuous Integration Prozesses
- Nexus: Als Build Artifact Repository Manager
- Jenkens: Als Continuous Integration Server

Des weiteren wurde sich in das Framework [5], sowie die bereits zu Verfügung stehenden Softwarekomponenten wie, z.B. Leap Motion Agent, Skeleton API, Draw Agent uvw. eingearbeitet und diese weiterentwickelt.

4 Systemarchitektur des Interaktion-Management-Systems

Die Systemarchitektur lehnt sich an die Architektur eines Companion-Systems [12] an, dies soll zukünftige Weiterentwicklungen und das Hinzufügen von weiteren Komponenten zur Kontext-Erfassung und -Interpretation vereinfachen.

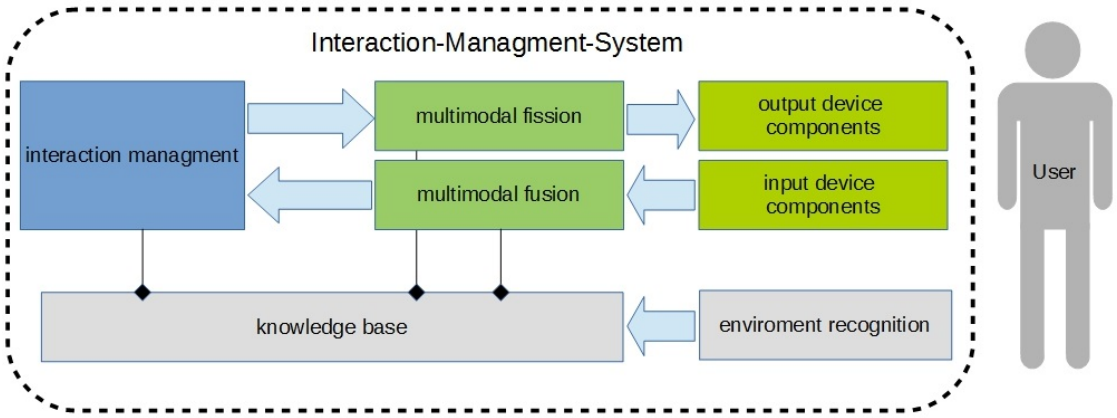


Abbildung 3: Komponenten des Interaktion-Management-Systems [vgl. [12]]

Das Gesamtsystem besteht aus folgenden Komponenten:

Interaktion-Managment Komponente Diese Komponente beinhaltet die Anwendungslogik welche auf Basis der Daten der Multimodale Fusion Komponente und der Wissensdatenbank Entscheidungen über die konkreten Zustände des Systems, der Umwelt und vor allem des Nutzer, trifft. Auf dieser Basis kann eine Interpretation der Nutzeraktionen durchgeführt werden. Zudem werden die Steuerungsbefehle an die Multimodal-Fission Komponente weiter geleitet.

Multimodale-Fusion Komponente Diese Komponente interpretiert die Daten der Gesten- und Sprachsensoren. Diese Daten werden im Vorfeld fusioniert um präzisere und zuverlässigere Aussagen über den Nutzer treffen zu können. Die Ergebnisse werden in Aggregierter Modalität-unabhängiger Form an die Interaktion-Management Komponente weitergeleitet.

Multimodal-Fission/User-Feedback Komponente Auf Basis der Modalität-unabhängigen Information der Interaktion-Managment Komponente kann die Multimodal-Fisson Komponente durch Nutzung der Kontext-Informationen der Wissensdatenbank entscheiden, über welches Interface, der Situation entsprechend, mit dem Nutzer interagiert werden soll. Dies kann besonders wichtig sein sollte ein Befehl des Nutzers nicht eindeutig erkannt worden sein.

Wissensdatenbank Die Wissensdatenbank hält Informationen über die verschiedenen Aktoren des Smart Homes so wie weitere, für den Kontext relevante, Informationen bereit.

4.1 Design Zusammenfassung

Das Design soll dem Nutzer - je nach Situation - ermöglichen über verschiedene Geräte und Kanäle mit dem System in Interaktion zu treten. Dies soll gewährleisten, dass die Vorlieben, Möglichkeiten des Nutzers berücksichtigt werden.
Durch die Trennung der verschiedenen Bereiche sowie der Austausch von modalitätsunabhängigen Informationen soll eine losen Koppelung der Systemkomponenten gewährleisten. Dies ermöglicht eine semantische Vermengung von verschiedenen Kommunikationskanälen. Zudem kann eine Austausch der Sensorik oder der Aktoren Transparent für das restliche System geschehen. Die Multimodle-Fission Komponente soll dem Nutzer, durch die Wahl eines geeigneten Kommunikationskanals für das User-Feedback, bei der Steuerung unterstützen. Dies ist besonders wichtig, da in Studien wie z.B. [1] gezeigt wurde, dass Nutzer eine direkte Reaktion des Systems erwarte. In der Studie wie auch in [2] wurde zudem gezeigt, dass die Semantische Vermischung verschiedener Steuerungskanäle zur natürlichen Interaktion mit Smart Objects dazu gehören. Die Multimodal-Fusion Komponente soll diese Semantische Vermischung durch eine Transformation zu aggregierten modalitäts-unabhängigen Nachrichten ermöglichen. Die Interaktion-Managment Komponente hält intern ein Modell des Systems, der Umwelt sowie des Nutzers bereit um eine Interpretation der Nutzereingaben vorzunehmen.

Echtzeiterkennung von Gestenbefehlen

5 Sensorik und Framworks

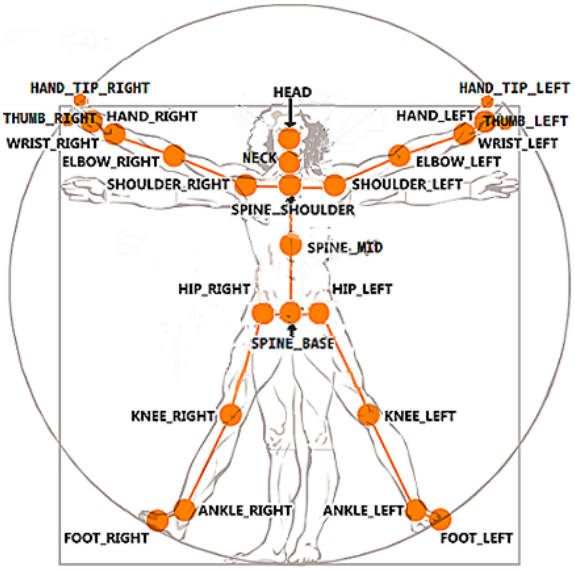
Die Umsetzung der Gestenerkennung soll mit der Kinect 2 und dem Leap Motion Controller durchgeführt werden. Diese sollen im Folgenden kurz vorgestellt werden.

5.1 Kinect 2

Die zweite Version der Kinect für Windows bietet durch das Microsoft SDK bereits eine Erkennung von drei Standardgesten (Hand auf, Hand geschlossen, Pistole). Das SDK liefert einen Stream mit Skelett Modellen von bis zu sechs Nutzern, welche mit jeweils 25 Gelenken gleichzeitig erfasst werden (siehe Abbildung 4b). Zudem liefert die Kinect noch weitere Daten, die bezüglich der Kontexterfassung sinnvoll sind. Durch den Audio Stream, den Video Stream, ein Time-of-Light Stream, Infrarot Stream, sowie das Face Tracking ist die Kinect ein vielseitig nutzbarer Sensor, der ebenfalls umfangreiche Kontextinformationen liefern kann.
Die 3D Tiefeinwahrnehmung der Kinect deckte einen Bereich von 0,5m bis 4,5 Meter ab. Die “sweet area” wird jedoch mit 1,0m - 4,0m angegeben. Erste Alpha-Tests zeigten das die Daten außerhalb der “sweet area” ungenau und unzuverlässig sind, daher muss der Mindestabstand von 1,0m unbedingt eingehalten werden um brauchbare Ergebnisse zu erhalten.

J4KSDK Microsoft liefert für die Kinect lediglich eine SDK für C# und C++, da die Agenten der Middleware jedoch in einer JVM (Java Virtuel Maschine) ausgeführt werden müssen, musste hierfür eine z.B. auf JNI (Java Native Interface) basierte Zugriff umgesetzt werden. Eine solche Umsetzung wurde bereits vom Digital Worlds Institute der Universität von Florida entwickelt [3]. Auf Basis des J4KSDKs konnte eine Überführung zu der im Living Place Projekt genutzten Skeleton-API, durch die Entwicklung des Kinect2-Agent, umgesetzt werden.

(a) Kinect 2 von Microsoft



(b) Joint Modell der Kinect 2

SkeletonModell Die Kinect liefert für 25 Joints die Position und die Richtung des Joints. Der Arm, sowie die Hände werden dabei durch vier Joints repräsentiert. Auf Basis dieser vier Joints kann später eine Sensor Fusion mit der Leap Motion umgesetzt werden. Für die Umsetzung der Überführung zur aktuellen Skeleton-API mussten zusätzliche Joints berechnet werden, da diese nicht direkt von der Kinect bzw. J4KSDK geliefert werden.

5.2 Leap Motion

Der Leap Motion Controler liefert zu einer hohen Bildrate von bis zu 200 Fps ein detailliertes Handmodell (vgl. Abbildung 5b). In jedem Frame werden alle gefundenen Hände und weitere “Pointable Object” (z.B. ein Stift) bereitgestellt. Zu jeder Hand lassen sich Daten wie Position und Neigung zu den einzelnen Fingern auslesen [vgl. [6], [7]]. Durch das SDK können fünf Gesten erkannt werden: Kreisbewegungen, Greifen, Touch nach vorne, Swipe Gesten sowie Key Tip Gesten [vgl. [18]].

(a) Leap Motion Controller

(b) Leap Motion Controller

6 Gestenerkennung

Neben den durch die SDKs erkannten Gesten müssen eigene Verfahren zur Gestenerkennung entwickelt werden. Die ist notwendig da die Gestenerkennung auf Basis des fusionierten Skelett Modell arbeiten soll. Im folgenden werden einige theoretische Überlegungen bzgl. der Verfahren angeführt (vgl. [8]):

6.1 Lernende Verfahren

Für die Erkennung und Analyse von Gesten gibt es grundsätzlich zwei verschiedene Herangehensweisen: Maschinenlernen und Heuristik basierende Verfahren. Eine gute Übersicht der Gestenerkennung liefert die Abhandlung [17]. Die beiden Herangehensweisen sollen im Folgenden kurz vorgestellt werden.

Üblicherweise werden Maschinen-Lernen-Verfahren zur Klassifizierung von 3D-Gesten genutzt. Dafür müssen wichtige Charakteristiken (Features) der aufgenommen Geste extrahiert werden und an den Klassifizierungsalgorithmus zur Einordnung übergeben werden[vgl. [17]]. Wie bei den meisten lernenden Verfahren, muss das System mit einer Vielzahl von hochwertigen Beispieldaten trainiert werden, um zuverlässige Klassifizierungen durchführen zu können. Die Auswahl aussagekräftiger Features ist eine der zentralen Herausforderungen der 3D-Gestenerkennung durch Maschinenlernen.

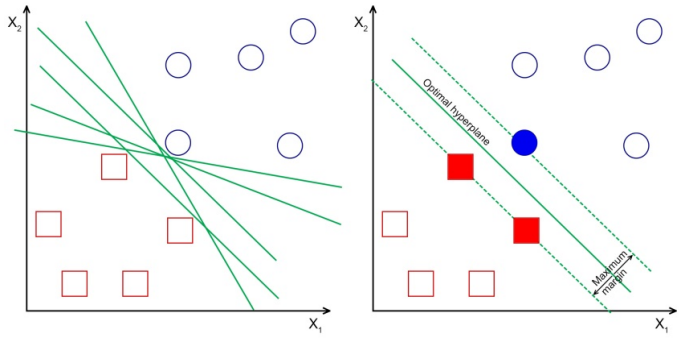


Abbildung 6: SVM: Beispiel einer optimalen Hyperebene in einem zweidimensionalen Vektorraum [17]

Folgende konkrete Verfahren des Maschinenlernens werden häufig genutzt:

- Support Vector Machines (SVM): SVM gehören zu den Überwachtes-Lernen Verfahren. Jede Geste aus den Testdaten wird durch ein Objekt in einem mehrdimensionalen Vektorraum dargestellt. Dabei wird grundsätzlich jedes Feature in einer Dimension des Vektorraums beschrieben. Aufgabe der SVM ist es mehrere Hyperebenen in dem Raum zu definieren, welche auf Basis der Trainingsdaten, den maximalen Abstand zwischen den verschiedenen Gesten beschreibt. Diese Grenzen der Hyperebenen werden für die Klassifizierung von neuen Daten genutzt. (Siehe Figur 6)

- Hidden Markov Models (HMMs): HMMs sind stochastische Modelle, welche auf einer Markov-Kette mit endlichen Zuständen(für welche die Markow-Annahme besteht) aufgebaut sind und einer Menge zufälliger Übergangsfunktionen. HMMs können auf verschiedene Art und Weise für die Klassifizierung eingesetzt werden. Einige Ansätze dazu werden in [17] erläutert.

- Conditional Random Fields (CRFs): CRFs sind wie HMMs stochastische Modelle. Jedoch sind bei CRFs die vorangegangenen Zustände gegenüber von HMMs nicht versteckt. Zudem haben sie nicht die potentielle Gefahr des “labeling bias” Problems, welches bei HHMs durch eine geringe Entropie der Übergangswahrscheinlichkeit der Zustände auftritt[vgl. [16]]. CRFs können ebenso wie HMMs auf verschiedene Weisen für die Gestenerkennung eingesetzt werden.

- Decision Trees (DTs): Entscheidungsbäume sind geordnete und gerichtete Graphen, bei denen Knoten, mit Ausnahme der Blätter, Entscheidungsregeln beschreiben. Konkret bedeutet dies, dass jeder Knoten eine Entscheidung über ein Feature trifft. Durch die Aneinanderreihung von Entscheidungen, von der Wurzel bis zum Blatt, wird die Klassifizierung durchgeführt. Die Entscheidungsbäume werden in der Gestenerkennung durch rekursive Verfahren wie das “top-down induction of decision trees” (TDIDT) erlernt bzw. aufgebaut.

- Decision Forests (DFs): DFs sind eine Erweiterung von DTs. DFs sind eine Menge von zufällig trainierten Entscheidungsbäumen. Die Nutzung von DFs gegenüber zu DTs führt meist zu einer hören Robustheit und zu einer besseren Generalisierung der Ergebnisse. Genauere Informationen zu Decision Forests findet man u.a. in [4].

- Weitere lernende Verfahren: Es können auch weitere Verfahren wie Template Learning, Maschienenlernen durch endliche Automaten oder Neuronale Netze eingesetzt werden. Weitere Informationen dazu liefert [17].

6.2 Heuristik basierende Verfahren

Die auf Heuristik basierenden Verfahren werden seltener verwendet, obwohl sie in Ansätzen wie [24] sehr gute Ergebnisse liefern. Heuristische Verfahren eignen sich besonders, wenn es lediglich eine kleine Anzahl von einfachen Gesten gibt. Heuristisch basierte Verfahren haben gegenüber den lernenden Verfahren den Vorteil, dass sie keine Trainingsdaten benötigen und durch einfache Regeln umgesetzt werden können [vgl. [17]].

Ein Beispiel dazu liefert [24], wo z.B auf Basis des Skelettmodells der Kinect von Microsoft durch eine simple Regel ein Springen des Nutzers erkannt werden konnte.

$$J = H_y - \bar{H}_y < C$$

Dabei steht H_y für die aktuelle Höhe des Kopfes (Y-Achse der Kopfposition), $\bar{H}_y$ für die kalibrierte normale Höhe des Kopfes der Person. C ist eine konstante Größe, die den Schwellwert zwischen Stehen und Springen beschreibt. Das Ergebnis J ist $wahr$ oder $falsch$, je nachdem, ob die Differenz zwischen H_y und $\bar{H}_y$ C übersteigt.

Abbildung 7: RealEdge Prototype auf Basis heuristischer Verfahren [24]

6.3 Verfahrensauswahl für die Gestenerkennung

Für intuitive Gesten scheinen heuristische Verfahren zur Gestenerkennung besser geeignet zu sein. Zudem können die von aktuellen Sensoren bereitgestellten Skelett- und Handmodelle unmittelbar verarbeitetet werden, ohne zusätzliche Feature Bildung. Da heuristische Verfahren keine umfangreichen Trainingsdaten benötigen, lassen sich somit auch schneller weitere Gesten dem System hinzufügen. Andererseits besteht die Gefahr bei heuristischen Verfahren, dass durch das Hinzufügen einer neuen Geste eine manuelle Anpassung der anderen Gesten nötig wird. Dies kann auftreten, wenn Effekte der neuen Geste zuvor nicht in Betrachtung gezogen worden sind.

Zumal die in dieser Ausarbeitung vorgestellten komplizierteren metaphorischen Gesten keine Veränderungen auf der Z-Achse berücksichtigen, muss im Rahmen des Projekts geprüft werden, ob bereits bekannte Frameworks, wie z.B. OpenCV zur Gestenerkennung auf Basis von 2D-Bilder, ausreichen. So können z.B. die erkannten Punkte bei Imaginery Interfaces [10] einfach durch eine SVM klassifiziert werden.

7 Sensor Fusion

Die erste Version eines Fusion-Agents für die Sensor Fusion wurde auf Basis des Kalman Filters [13] und einer Bachelorarbeit aus der UbiComp Arbeitsgruppe [11] umgesetzt.

Dies ermöglicht eine gewichtete Fusion von den Sensor Daten der Kinect und der Leap Motion. Dieser erste Ansatz ist jedoch für die Anwendung im Smart Home nur bedingt einsetzbar, da die in [11] eingesetzte Kalibrierung nur funktioniert wenn beide Sensoren parallel zueinander auf der X-Achse stehen und der Winkel der Kameras um genau 90° um die X-Achse gedreht ist (vgl.[11]). Zudem ist die Implementierung lediglich für zwei Sensoren ausgelegt.

Kalman Filter für zwei Sensoren

$$x = x_1 * \frac{\sigma_2^2}{(\sigma_1^2+\sigma_2^2)} + x_2 * \frac{\sigma_1^2}{(\sigma_1^2+\sigma_2^2)}$$

- x Fusionierter Wert von beiden Sensoren
- x_1 Sensor Wert 1
- x_2 Sensor Wert 2
- σ Standartabweichung
- σ^2 Varianz

Standardabweichung Kinect Vom Hersteller selber wird keine Standardabweichung für die Tiefen-Informationen angegeben. Daher müssen verschiedene wissenschaftliche Studien hinzugezogen werden. In [20] wird eine Standartabweichung von 0,75mm bei einer Entfernung von 1,2m und 5,6mm bei einer Entfernung von 3,5m. Der Durchschnitt aller Messungen von 1m-5m hatte eine Standardabweichung von 8,1mm. In [9] wird die Standardabweichung als Formel angegeben:

$$0.5417 * Z^2 - 0,885 * Z + 2,7083$$

Dabei steht Z für die Entfernung zum Sensor. Bei einer Entfernung von 1,2m ergibt sich somit eine Standardabweichung von 2,4mm und für 3,5m eine Standardabweichung von 6,2mm.
Daher wird für erste Alphatests eine Standardabweichung von 5mm konfiguriert. Im weiteren Verlauf des Projekts soll der Agent auch Funktionen für die Standard Abweichung entgegen nehmen können.

Standardabweichung Leap Motion Der Herstelle gibt eine Standardabweichung von 0,001mm an [19]. In einen unabhängigen Versuch konnten Werte von 0.05 mm bei stillstehenden Objekten und 1.0 mm bei Objekten in Bewegung als Standardabweichung ermittelt werden [22].

7.1 Schnittstellen

In Vorbereitung des Fusion Agenten wurde ein Kinect v2 Agent entwickelt welcher die Daten der Kinect an die Middelware übergibt. Der Fusion Agent abonniert diese Daten, so wie die Daten des bereits im Projekt vorhandenen Leap Motion Agentens. Die Daten werden in ein generischen Skelett der Skeleton API umgewandelt und anderen Agenten über die Middelware bereitgestellt. So konnte z.B. der Draw Agent diese Daten direkt abonnieren und die fusionierten Skelette zeichen.

Zusammenfassung

8 Ausblick

Im folgenden soll das weitere Vorgehen im Hauptprojekt skizziert werden. Dabei wird der Fokus auf die Implementation einen Prototypen zur Gestenerkennung gelegt.

8.1 Erkennung von Zeigegesten/3D Modell Analyse

Zur Analyse und Erkennung von Zeigegesten soll ein 3D-Model des Labors genutzt werden. In dieses soll eine 3D-Echtzeitdarstellung des Skeletts und eine Visualisierung der erkannten Zeigerichtung umgesetzt werden.

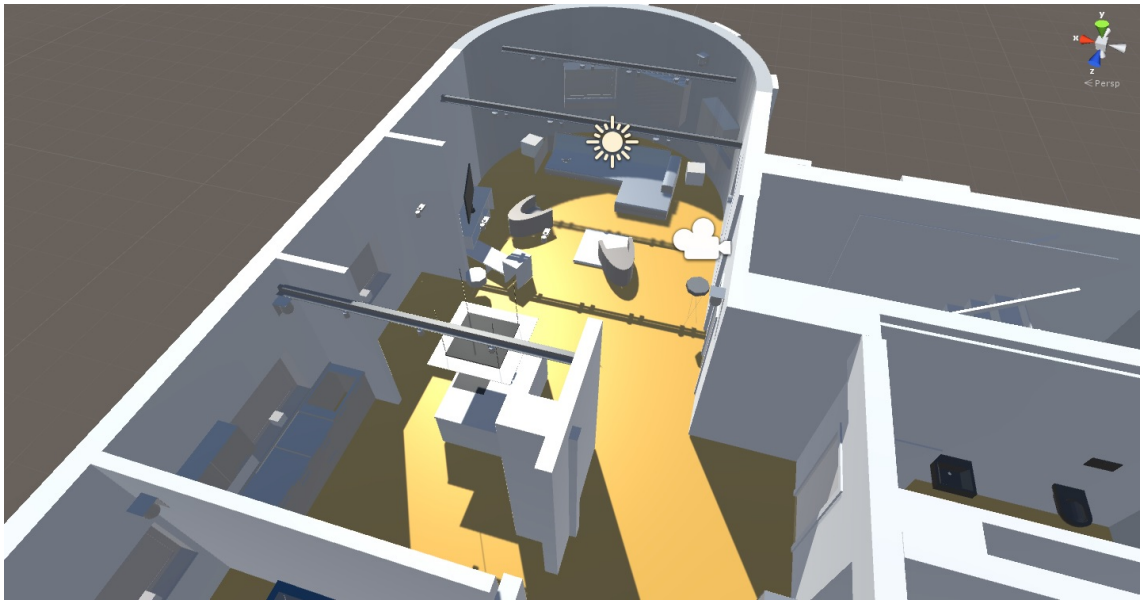


Abbildung 8: 3D-Model des Living Places

Dies kann ebenfalls das Debuggen bei der Entwicklung der Zeigegeste erleichtern. Das Modell soll als webbasiert 3D-Modellierung zur Verfügung gestellt werden um die Plattform Unabhängigkeit zu gewährleisten. Dabei wird auf das 3D-Modell von [14] aus der Ubicomp Arbeitsgruppe zurückgegriffen.

8.2 J4KSDK erweiterung

Das J4KSDK ([3]) wurde noch nicht vollständig auf die Kinect v2 umgestellt. Daten für Skelett-, Video-, und Tiefenbild-Stream, werden wie bei der Kinect 1 bereitgestellt, jedoch werden keine weiteren Kontextinformationen über die Ebene (Boden), das Gesicht des Nutzers (Emotion, Brille, Mund auf/zu, etc.) und weitere Features der Kinect v2 für

Windows bereitgestellt. Daher muss in Zusammenarbeit mit dem Entwickler der Middelware ein weiterer Konnektor zum System auf Basis von C# einwickelt werden. Alternativ könnte auch das J4KSDK durch zusätzliche JNI Funktionen erweitert werden, sollte die Universität von Florida den Quellcode offen legen.

8.3 Sensor Fusion

Bei ersten Tests stellten sich weiter Anforderungen für die Sensor Fusion heraus, welche bei der Entwicklung einen Prototypen zu berücksichtigen sind.

- Kalibrierung verbessern: Die hier auf Basis von [11] verwendete Sensor Fusion lässt sich in dieser Form lediglich für 2 Sensoren verwenden. Zudem müssen die Sensoren parallel zu einander stehen, welches das Erkennen des Nutzers von verschiedenen Seiten stark einschränkt.
- Funktion als Standardabweichung: Das System sollte in der Lage sein Funktionen aus der Konfiguration auszulesen und diese zur Berechnung der Standardabweichung verwenden. Dafür muss das Framework erweitert werden.
- Sensor Fehler: Auf Basis der letzten Messungen sollte ein Sensor-Smoothing so wie weitere Constrains(z.B. zusammenhängendes Skelett) genutzt werden um fehlerhafte Daten zu erkennen.

8.4 Spracherkennung

Die Möglichkeit der semantischen Vermischung von Gesten- und Sprachsteuerung sollte im Hauptprojekt weiter verfolgt werden. So konnte u.a. in [1] gezeigt werden das viele Nutzer intuitiv auf solch eine semantische Vermischung zurückgreifen. So könnte z.B ein Nutzer über eine Geste ein Objekt auswählen und über einen Sprachbefehl die Art der Manipulation wählen.

9 Problemstellungen und Risiken

9.1 Reaktionszeit und User Feedback

Die Wizard-of-Oz Studien zeigten auf, dass die Probanden eine schnelle Reaktion des Systems erwarteten. So wird in verschiedenen Untersuchungen dazu angegeben, dass eine Antwortzeit von weniger als 100 ms dem Nutzer fließend vorkommen. Längere Reaktionszeiten führen bei vielen Nutzern zu Verwirrung da sie sich nicht sicher sind ob das System ihrer Geste erkannt hat. Die schnelle Reaktionszeit muss durch eine gute Architektur und Implementierung des Systems sichergestellt werden. Eine zusätzliche Möglichkeit dies zu gewährleisten kann durch eine Täuschung des Nutzers erreicht werden. So wird dem Nutzer beispielsweise mitgeteilt, dass das System eine 360° Kreisbewegung mit der Hand erkennt. Dabei erkennt das System bereits bei 270° die Geste und verhält sich dementsprechend.
Darüber hinaus sollte dem Nutzer durch ein Feedback mitgeteilt werden, dass die Geste erkannt worden ist und die Anfrage bearbeitet wird. Dies ist besonders wichtig, sollte der Nutzer Objekte ansteuern, welche er nicht sehen kann oder bei denen keine direkt bzw. unverzüglich wahrnehmbare Veränderung stattfindet (z.B. Verändern der Temperatur der Bodenheizung). Dieses Feedback kann durch Licht-, Soundsignale oder haptisches Feedback umgesetzt werden. Daher wurde im Rahmen des Projekts ein Vibration-Agenten und eine benutzerfreundliche API zur Ansteuerung des kabellosen Vibrations-Armbands umgesetzt. Konzeption und Entwicklung eines Prototypen eines kabellosen Vibrations-Armbands für ein zeitnahes haptisches Nutzer-Feedback.

9.2 Risiken

Ein grundsätzliches Problem bei einer Gegenüberstellung verschiedener Gesten besteht darin, dass andersartige Ansätze aus unterschiedlichen Abhandlungen sich nur schwer vergleichen lassen. So wird z.B. in einigen Abhandlungen für die Gestenerkennung kein kontinuierlicher Datenstrom genutzt und daher das Start-End-Problem nicht beachtet. Zudem werden häufig auch grundverschiedene Sensoren zum Motion Tracking verwendet. Dies erschwert die Beurteilung und die Gegenüberstellung von verschiedenen konzeptionellen Ansätzen und Verfahren zur Gestenerkennung beträchtlich.
Daher sollten im Master Seminar erste Nutzerstudien durchgeführt werden, um die Entscheidung auf Basis eigener Messungen treffen zu können.
Ein weiteres Risiko bildet die Möglichkeit der Datenerfassung der Sensorik. Grundsätzlich kann eine kamerabasierte Gestenerkennung nicht alle Bereiche einer Wohnung abdecken. Diese schränkt die Usability ein und kann schnell zur Ablehnung bei den Nutzern führen. Daher soll das System vorerst nur in einem dem Nutzer bekannten Bereich der Wohnung zur Verfügung gestellt werden. Im Living Place Hamburg würde sich z.B. das Wohnzimmer anbieten. Dieser Bereich könnte von mehreren Kinect und Leap Motion Sensoren abgedeckt werden.
Durch die Ergebnisse des geplanten Wizard-of-Oz Experiments müssen evtl. weitere Gesten dem System hinzugefügt werden. Sollten dabei komplizierte Bewegungsabläufe genutzt werden, besteht die Gefahr, dass diese Gesten sich nur schwer über ein Heuristikbasierendes Erkennungsverfahren identifizieren lassen. Ebenso muss überprüft werden, welchen konkreten Technologien für die Selektion eines Objekts bei kontextsensitiven Befehlen zum Einsatz kommen können.
Ferner muss beachtet werden, dass keine der hier vorgestellten Studien eine längere Nutzung des Systems von den Probanden erforderte. Daher können aktuell keine Aussagen über die physikalische Ergonomie oder die Langzeit-Akzeptanz der Gesten getroffen werden. Dementsprechend sind eigene Vorstudien mit trainierten und untrainierten Probanden durchzuführen. Dabei könnte man ebenfalls einen Vergleich zwischen intuitiven und effektiven Gesten erstellen.

10 Fazit

In Projekt 1 konnte sich erfolgreich in eine Vielzahl von Tools und APIs so wie das bereitgestellte Framework eingearbeitet werden. Zudem konnte ein Fusion Agent sowie ein Kinect 2 Agent für das System entwickelt und getestet werden. Infolgedessen konnten weitere Problemstellungen identifiziert werde. Durch die umfangreiche Einarbeitung in die Sensorik, so wie deren Programmierschnittellen konnte die Infrastruktur für einen Prototypen für die Multi-modale Haussteuerung entwickelt werden. Zudem wurden umfangreiche Recherche bzgl. der Architektur von multimodalen Systemen vorgenommen. Dies ermöglicht ein Design und die Architektur Planung des Prototypen. Ferner konnten aus den Erfahrungen der praktischen Umsetzung des Fusion-Agenten wichtige Einsichten zur Problemstellung und Weiterentwicklung gewonnen werden.

Literatur

[1] Dimitra Anastasiou, Cui Jian, and Christoph Stahl. A german-chinese speech-gesture behavioural corpus of device control in a smart home. In *Proceedings of the 6th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '13, pages 62:1–62:6, New York, NY, USA, 2013. ACM.

[2] Dimitra Anastasiou, Cui Jian, and Desislava Zhekova. Speech and gesture interaction in an ambient assisted living lab. In *Proceedings of the 1st Workshop on Speech and Multimodal Interaction in Assistive Environments*, SMIAE '12, pages 18–27, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[3] A. Barmpoutis. Tensor body: Real-time reconstruction of the human body and avatar synthesis from rgb-d. *Cybernetics, IEEE Transactions on*, 43(5):1347–1356, Oct 2013.

[4] Antonio Criminisi, Jamie Shotton, and Ender Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Found. Trends. Comput. Graph. Vis.*, 7(2–3):81–227, February 2012.

[5] Tobias Eichler. Agentenbasierte middleware zur entwicklerunterstützung in einem smart-home-labor. 2014. Erreichbar online unter `http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/eichler.pdf`.

[6] Sobin Ghose. Anwendungen 1 ausarbeitung - kontextabhaengige interpretation von 3d-gesten. 2014.

[7] Sobin Ghose. Konzeption und evaluation eines interaktiven badezimmerspiegels. 2014. Erreichbar online unter `http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/bachelor/ghose.pdf`; besucht am 06.6.2014.

[8] Sobin Ghose. Anwendungen 2 ausarbeitung - multimodale haussteuerung. 2015.

[9] H. Gonzalez-Jorge, P. Rodríguez-Gonzálvez, J. Martínez-Sánchez, D. González-Aguilera, P. Arias, M. Gesto, and L. Díaz-Vilariño. Metrological comparison between kinect i and kinect {II} sensors. *Measurement*, 70(0):21 – 26, 2015.

[10] Sean Gustafson, Daniel Bierwirth, and Patrick Baudisch. 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*, UIST '10, pages 3–12, New York, NY, USA, 2010. ACM.

[11] Christian Gutjahr. Modellbasierte sensorfusion von kamerabasierter bewegungsverfolgung. 2015. Erreichbar online unter `http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/bachelor/gutjahr.pdf`.

[12] Frank Honold, Felix Schüssel, Florian Nothdurft, and Peter Kurzok. Companion technology for multimodal interaction. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, ICMI '12, pages 67–68, New York, NY, USA, 2012. ACM.

[13] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[14] Bastian Karstaedt. Kontextinterpretation in smart homes auf basis semantischer 3d gebäudemodell. 2012. Erreichbar online unter `http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/karstaedt.pdf`; besucht am 12.6.2015.

[15] Bastian Karstaedt. Kontextinterpretation in smart homes auf basis semantischer 3d gebäudemodelle. 2012. Erreichbar online unter `http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/master/karstaedt.pdf`.

[16] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[17] Joseph J. LaViola, Jr. An introduction to 3d gestural interfaces. In *ACM SIGGRAPH 2014 Courses*, SIGGRAPH '14, pages 25:1–25:42, New York, NY, USA, 2014. ACM.

[18] Inc. Leap Motion. Leap overview - gestures. 2014. Erreichbar online unter `https://developer.leapmotion.com/documentation/java/devguide/Leap_Overview.html#gestures`; besucht am 10.6.2015.

[19] Inc. Leap Motion. This is precisely about accuracy. 2014. Erreichbar online unter `https://www.leapmotion.com/product`; besucht am 10.6.2015.

[20] Mark A. Livingston, Jay Sebastian, Zhuming Ai, and Jonathan W. Decker. Performance measurements for the microsoft kinect skeleton. 2014. Erreichbar online unter `http://www.nrl.navy.mil/itd/imda/sites/www.nrl.navy.mil.itd.imda/files/pdfs/2012_VRposter_kinectSkelPerf.pdf`.

[21] David McNeill. *Hand and Mind: What Gestures Reveal about Thought*. University of Chicago Press, 1992.

[22] Frank Weichert, Daniel Bachmann, Bartholomäus Rudak, and Denis Fisseler. Analysis of the accuracy and robustness of the leap motion controller. *Sensors*, 13(5):6380, 2013.

[23] Mark Weiser. The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.*, 3(3):3–11, July 1999.

[24] Wingrave C. Laviola J. Roberts T. Williamson, B. and P. Garrity. Natural full body interaction for navigation in dismounted soldier training. in interservice/industry training, simulation, and education conference (i/itsec 2011), 2103–2110., 2011.