Fusion der KinectV2 und LeapMotion zur erweiterten Menüsteuerung in VR

Patrick Höling, Tobias Michel Latta

Zusammenfassung

In dieser Ausarbeitung wird eine Möglichkeit zur Fusion von KinectV2- und Leap Motion-Sensordaten erklärt um eine Menüsteuerung in der Virtuellen Realität (VR) zu implementieren. Motivation dafür liefert die Kombination aus hoher Auflösung der Fingererkennung der Leap Motion sowie dem großen Bewegungsradius der KinectV2. Es wird in Unity auf Basis des KinectV2-VR-Packages von Rumen Filkov eine Menüauswahl in Abhängigkeit des Winkels zwischen linkem Schulter- und Handgelenk aufgebaut. Wenn anschließend die linke Hand geöffnet und dann geschlossen wird, ist ein dem Winkel zugehöriges Event ausgelöst. Handelt es sich dabei um Menü, wird dieses aufgerufen, sprich visualisiert. Es kann sich jedoch auch um ein GameObject mit speziellen Objekteigenschaften handeln. Wird die linke Hand anschließend in den Sichtbereich der am Head-Mounted-Display befestigten Leap Motion gezogen, ist ein Platzieren und Bewegen des Objektes möglich. Eine Erweiterung des Projektes ist möglich und wird aufgrund der verbesserungswürdigen Verwertung der KinectV2-Sensordaten empfohlen. Häufig (mehr als 1 von 10) tritt ein fehlerhaft erkanntes Öffnen/Schließen der Hand auf, wodurch Menüs/Objekte ungewollt (nicht) ausgewählt werden. Eventuell lässt sich eine Optimierung der Gestenerkennung durch eine Tiefpassfilterung der Kinect-Sensordaten bewerkstelligen, damit nur eindeutig und zeitlich über einen gewissen Zeitraum bestehende Gesten erkannt.

# Einleitung

Dieser Aufsatz ist im Zusammenhang mit der Vorlesung *Human Computer Interaction* (HCI), gehalten von Prof. Dr. Birgit Wendholt im Wintersemester 2017/2018 an der Hochschule für Angewandte Wissenschaft Hamburg, entstanden. Während der Vorlesung werden unterschiedliche Implementierungen im Bereich HCI behandelt und in insgesamt drei Laborterminen getestet. Unter anderem handelt es sich dabei um das VR-Headset *HTC Vive*, sowie das Handtracking-System *Leap Motion*. Beides in Verbindung mit der Entwicklungsumgebung Unity. Ebenfalls evaluiert wurde das Ganzkörper-Tracking-System von Microsoft, benannt *Kinect V2*.

Während des Praktikumstermins zur Leap Motion ist einerseits die hohe Genauigkeit des Handtrackings aufgefallen. Es funktioniert, wie folgt: Ein Raster aus Infrarot-Lichtpunkten wird in das FOV gesendet und anschließend wieder aufgenommen. In Abhängigkeit der Entfernung verändert sich die Zeit, welche das Licht benötigt um wieder einfangen zu werden. Mit allen Punkten kann somit das Raster in ein dreidimensionales Bild umgewandelt werden, welches wiederum auf charakteristische Gelenke untersucht wird. Gibt es nun mehrere Übereinstimmungen mit den Gelenken eines beispielhaften Handmodells, wird es auf die detektierten Gelenkpunkte gelegt und eine digitale Repräsentation findet statt. Liegt nun die Hand außerhalb des Infrarot-Rasters, kann dementsprechend keine Detektion stattfinden. Die 27 Freiheitsgrade einer Hand werden nur selten (1 von 100) falsch digitalisiert. Hauptsächlich treten Fehler auf, wenn sich die Hand an den Rand des *Sichtbereichs*/*Field of View* (FOV) begibt oder in einer unvorhergesehenen Position steht, da dann keine plausiblen Übereinstimmungen zu dem Beispiel-Handmodell bestehen.

Der KinectV2-Sensor von Microsoft funktioniert ebenfalls über die Laufzeitmessung von Infrarotlicht. Dabei wird jedoch kein vordefiniertes Lichtraster ausgesendet, sondern großflächig eine Infrarot-Lichtquelle benutzt, welche an dem zu detektierenden Objekt reflektiert wird und anschließend von einer Infrarot-Kamera aufgenommen wird. Dabei ist es ebenfalls möglich, Infrarotlicht aufzunehmen, welches von dem detektierten Objekt selbst ausgesendet wird. Beispielsweise kann so die Temperatur gemessen werden. In der Praxis wird eine Entfernung von 0,5 m bis 2,5 m zur Kinect empfohlen. Außerdem liegt der Fokus auf einer Ganzkörpererkennung, welche bis zu 25 Gelenke des Körpers trackt. Im Bereich der Hand sind nur der Daumen, der Zeigefinger sowie das Handgelenk erfassbar.

Das Ziel dieser Ausarbeitung ist die Kombination von großem Bewegungsraum der Kinect mit dem genauen Handtracking der Leap Motion um eine intuitive Bedienung mit den Händen vor dem Körper hochauflösend abzubilden sowie außerhalb des Leap Motion-FOV grundsätzlich gestengesteuert zu ermöglichen. Durch die Bedienung der VR mit dem eigenen Körper wird die Immersion nicht zuletzt durch die visuelle Repräsentation der eigenen Hände gefördert. Auch muss kein externes Eingabegerät verwendet werden, welches Körperaktionen nur durch Tastendruck aufnehmen kann. Der Nutzer kann intuitiv die Geste aus der Realität vollziehen und sie wird durch die Leap Motion aufgenommen sowie in der VR verarbeitet.

Diese Intuivität ist wichtig um VR als Medium zu etablieren. An mehreren technischen Produkten lässt sich die Relevanz von einfacher Bedienung erkennen. Beispielsweise ist der Erfolg des Apple iPhone der ersten Generation auch darüber zu erklären, dass es eine intuitive Bedienung geschaffen hat, welche es auch weniger technisch-versierten Nutzern ermöglichte komplexe Funktionen abzurufen und zu nutzen. So wird die Hemmschwelle zur Nutzung von Technik herabgesenkt, was wiederum bei den Entwicklern und Unternehmern Interesse weckt, in die gleiche Richtung zu investieren/produzieren. Ohne größere Erklärungen oder Schnittstellen (Werkzeuge) sollen Abläufe mit dem eigenen Körper durchgeführt werden, so wie es auch in der Realität hauptsächlich der Fall ist. Die Fehlerhäufigkeit bei einem Funktionsaufruf soll dabei auf maximal einen Fehlaufruf unter fünf Versuchen beschränkt werden.

Um den Lösungsansatz verständlich zu dokumentieren, wird dieser Aufsatz wie folgt gegliedert: Zuerst findet eine wissenschaftliche Einordnung statt, indem die Inhalte mit bestehenden Arbeiten verglichen werden, welche zum Zeitpunkt des Verfassens aufzufinden waren. Dabei ist hauptsächlich die Arbeit *Multi-Sensor Data Fusion for Hand Tracking using Kinect and Leap Motion* von Benoît Penelle (2014) zu nennen, welche einen sehr wissenschaftlichen Ansatz verfolgt. Darüber hinaus werden verwendete Frameworks und Beispiele zur jeweiligen Implementierung von KinectV2 und LeapMotion näher erläutert. Anschließend findet eine Erläuterung zum derzeitigen Stand der Technik statt, die einen kritischen Blick auf die genutzten Produkte wirft, bevor sie in den Zusammenhang zum Vorlesungsinhalt von *HCI WS17/18* gebracht werden. Diese Einordnung ist sinnvoll um die Sensorik tiefergehend zu erläutern. Der hauptsächliche Teil beschäftigt sich mit der Implementierung des Lösungsansatzes und erklärt ihn durch mehrere Abbildungen aus dem Bereich UML2.

# Realisierung

## Wissenschaftliche Einordnung

In dem Paper *Multi-sensor data fusion for hand tracking using Kinect and Leap Motion* (Penelle, 2014) wird eine Verbindung zwischen dem hochauflösenden Handtracking der Leap Motion sowie dem großen Bewegungsbereich der Kinect V1 beschrieben. Auf mathematischer Ebene wird beschrieben, wie die Handgelenke bei beiden Sensoren pixel-genau übereinander zu legen sind, wie also eine Kalibrierung und Registrierung beider Komponenten erfolgt. Anwendungsfall ist dabei die Schmerztherapie bei medizinischen Patienten, welche einen Arm verloren haben. Setzt sich der Betroffene ein VR-Headset auf, sollen die Bewegungen des vorhandenen Arms auf das letzte vorhandene Gelenk auf der amputierten Seite gespiegelt werden, um den Eindruck zu erwecken, dass beide Arme vorhanden sind. So können einfache Tätigkeiten ausgeführt werden und die Illusion des rekonstruierten Armes reduziert den Phantomschmerz. Überschneidungen zu diesem Aufsatz finden sich bei der komplementären Nutzung von Leap Motion und einer Kinect-Kamera. Zwar wird hier eine neuere Kinect-Version verwendet, jedoch ist der grundsätzliche Hintergrund einen größeren Bewegungsraum im Vergleich zur LM zu schaffen ebenfalls gegeben. Ein weiterer Unterschied ist die Herangehensweise an die Thematik. Durch die mathematische Beschreibung der Kalibrierungs- und Registrierungs-Algorithmen kann die Methodik auch auf andere Realisierung des Sensortyps angewendet werden. Dabei rückt jedoch die konkrete Implementierung in einer Entwicklungsumgebung in den Hintergrund, was die Ausführbarkeit für den durchschnittlichen Benutzer beschränkt. In diesem Aufsatz ist zwar der mathematische Hintergrund weniger beleuchtet, jedoch findet eine ausführliche Auseinandersetzung mit dem Unity-Projekt statt, bei welcher erstellte und abgeänderte C#-Skripte genau erklärt werden. Mit den mitgelieferten Projektdaten kann somit eine eigene Implementierung vorgenommen werden. Auch ist es dabei von Vorteil, dass direkt eine Visualisierung durch das *RMF\_RadialMenu* implementiert ist. Der Nutzer kann somit mit möglichst wenig Schritten eine Umsetzung für sein eigenes Projekt erstellen.

HIER WEITER

* Arbeit von Penelle2014 🡪 nicht so wissenschaftlicher Ansatz; leichter selbst anzuwenden; Visualisierung der ausgewählten Bereiche; konkreter Menüaufrauf
* LeapMotion wird gemischt aufgefasst -> zukünftig wird direkter Einbau in VR-Headsets angestrebt (Robertson, 2016)
* Kinect wird nicht weiterentwickelt
* Allgemeiner Trend zu Augmented Reality mit HoloLens (eingebaute Gestenerkennung mit möglicher Erweiterung zu Handtracking)

## Verwendete Beispiele und Frameworks

* VR-Package von Rumen Filkov 🡪 Erweiterung durch Leap
* Dynamic UI von LeapMotion 🡪 Erweiterung durch Kinect
* RadialMenuFramework von Brett Gregory 🡪 Visualisierung in VR und Aufruf über Kinect/Leap

## Einordnung *HCI Wintersemester 2017/2018*

* LeapMotion am HeadMountedDisplay als InsideOut-Variante (Nutzer braucht keine Kameras aufstellen, die sich nicht mitbewegen) für Handtracking mit Infrarotlicht-Laufzeitmessung und Modellabbildung (insgesamt 27 DOF in der Hand)
* KinectV2 als Outside-In Bewegungstracker, der unter anderem mit Infrarotlicht-Laufzeitmessung die Position mit 6DOF bestimmt
* Unity als Entwicklungsumgebung für VR-Inhalte (unter anderem zu Nutze machen von *FindObjectByTag*.
* HTC Vive als HeadMounted-Display

## Implementierung

# Reflexion

* Einsetzen des SimpleLeapListeners sowie eigens geschriebenen Methoden aus SimpleHandListener in LeactManager.cs um Funktionalität zu ermöglichen ohne jegliche Änderungen an Leap/Kinect-Komponenten vorzunehmen
* SelectionFollower des RadialMenu fixen
* MultiUser Support mit zweiter Leap
* Menüunterscheidung für Linke/Rechte Hand machen, damit unterschiedliche Menüs mit unterschiedlichen Händen gespawnt werden können 🡪 sogar gleichzeitig jeweils mit links und rechts ein Menü spawnen lassen, welche unabhängig voneinander anwählbar sind (mit jeder Hand autonom Auswahl möglich)
* Geste zum Schließen des Radialmenüs von *Hände in Leap sichtbar* ändern, da es teilweise etwas buggy ist und nicht sofort ausgeblendet wird, wenn LeapHände zu sehen sind🡪 über Kinect könnte Wischgeste mit inaktiver Hand zum Schließen des Menüs führen
* Weiteren Menüpunkt/Geste um Objekte zu Löschen 🡪 wenn Objekt im Anchor ist wird es durch Wischgeste mit anderer Hand gelöscht / oder aber Ziehen des Objektes aus dem Sichtbereich des Leap-Controllers führt zum Löschen
* Hinzufügen von weiteren Menühierarchein wird jetzt bewerkstelligt durch
  + 1. Neues Prefab in MenuFabric
  + 2. Id für Menü vergeben im public RMF\_RadialMenu menu(…)
  + 3. Id im RMF\_RadialMenu-Attribut angeben
  + 4. Menü als Nachfolger eines anderen Menüs deklarieren
  + 5. Ggf. Nachfolger des eingefügten Menüs festlegen
  + Automatisieren dieses Vorgangs durch Add-Methode, mit Parametern zu neuer Id, Menünachfolgern und Menüvorgängern, welche vorherige Schritte selbstständig übernimmt
* Besseres Abfangen von falschen Kinect-Gesten durch Tiefpassfilterung von Sensordaten

Literaturverzeichnis

Im Literaturverzeichnis ordnet ihre die Angaben bitte alphabetisch nach Nachnamen des ersten Autors und dann nach Veröffentlichungsdatum (bei mehreren Titeln von einem Autor).

Es gibt eine Reihe sehr unterschiedlicher Regelungen und Normen für Literaturangaben. Bitte die folgende Syntax verwenden:

* Für Buchzitate: <Autor(en)> <(Jahr)>. <*Titel*>. <Erscheinungsort>: <Verlag>. Die Autoren erscheinen dabei in der Form: <Nachname>, <Abgekürzte Vornamen> – Zwei Autoren werden mit & getrennt. Bei mehr als zwei Autoren werden die letzten beiden ebenfalls mit &, alle weiteren durch Kommata getrennt. Bei Verlagen mit mehreren Verlagsorten wird nur der erste genannt.
* Für Zeitschriftenzitate: <Autor(en)> <(Jahr)>. <Titel>. <*Titel der Zeitschrift*> <*Jahrgangsnummer*(Band-)Nummer>, <Seitenangaben>.
* Für Zitate aus Tagungs- und Sammelbänden: <Einzelbeitrag wie bei Zeitschriften>. In <Herausgeber, analog zu Autor(en)> (Hrsg.): <*Titel*> [eventuell <*Auflage*>]. <Erscheinungsort>: <Verlag>. [S. <Seitenangaben>].

Für das Layout des Literaturverzeichnisses ist das Format <Literatur> zu verwenden. Angaben zur Formatierung der Literaturverweise finden Sie in Abschnitt 4.4.

Nachfolgend sind beispielhaft einige Literaturangaben aufgeführt:

Borghoff, U. M. & Schlichter, J. (1998). *Rechnergestützte Gruppenarbeit – Eine Einführung in Verteilte Anwendungen. 2. Auflage*. Heidelberg: Springer.

DIN EN ISO 9241-110 (2006). *Ergonomics of human-system interaction – Part 110: Dialogue principles*. Beuth, Berlin.

Friedrich, J., Herrmann, T., Peschke, M. & Rolf, A. (Hrsg.) (1995). *Informatik und Gesellschaft*. Heidelberg: Spektrum Akademischer Verlag.

Nake, F. (1993). *Die erträgliche Leichtigkeit der Zeichen*. Baden-Baden: Agis.

Oberquelle, H. (1991). MCI – Quo Vadis? Perspektiven für die Gestaltung und Entwicklung der Mensch-Computer-Interaktion. In Ackermann, D. & Ulich, E. (Hrsg.): *Software-Ergonomie '91*. Stuttgart: Teubner, S. 9-24.

Raskin, J. (1994). Intuitive equals Familiar. *Communications of the ACM, 37*(9), 17-18.

Robertson, A (2016). *Leap Motion's revamped hand tracking is getting built straight into VR headset*. <https://www.theverge.com/2016/2/17/11021214/leap-motion-hand-tracker-virtual-reality-orion-mobile-vr>

Penelle, B. (2014). *Multi-Sensor Data Fusion for Hand Tracking using Kinect and Leap Motion*.Université Libre Brussels. Brüssel. <https://dl.acm.org/citation.cfm?id=2620710>

* (optional => VR als bedeutendes Medium in der dritten Entwicklungsphase)
* Während des Praktikums sind Grenzen der Handerkennung und großer Bewegungsradius von Kinect aufgefallen sowie der geringe Bewegungsradius der LeapMotion, welche dafür aber eine gute Handerkennung implementiert.
* Vorstellung positive Eigenschaften beider Sensoren zu kombinieren um eine gute Handerkennung im nahen Sichtbereich (120° FOV, max. 1 m Entfernung) des HMD-Trägers mit Gestenerkennung im Bereich um den HMD-Träger
* Beitrag zu Informatik: liefern Basis um die Eingabemethoden in VR durch bestehenden Sensoren zu verbessern, um das Interagieren intuitiver zu machen da keine Controller sondern nur der eigene Körper verwendet wird
* Es soll erreicht werden, dass man in einer Umgebung ohne große Erklärung Menüpunkte auswählen kann
* Ziel: hinreichend gute Bedienung in der VR mit minimaler Interaktion realer Gegenstände
* Anforderung: Fehlerhäufigkeit beschränken auf maximal „sehr häufig“ (1 von 5 Versuchen)
* Aufbau der Arbeit:
  + Vergleich:
    - Arbeit von Penelle2014 🡪 nicht so wissenschaftlicher Ansatz; leichter selbst anzuwenden; Visualisierung der ausgewählten Bereiche; konkreter Menüaufrauf
      * VR-Package von Rumen Filkov 🡪 Erweiterung durch Leap
    - Dynamic UI von LeapMotion 🡪 Erweiterung durch Kinect
    - RadialMenuFramework von Brett Gregory 🡪 Visualisierung in VR und Aufruf über Kinect/Leap
  + Stand der Technik:
    - LeapMotion wird gemischt aufgefasst -> zukünftig wird direkter Einbau in VR-Headsets angestrebt (Robertson, 2016)
    - Kinect wird nicht weiterentwickelt
    - Allgemeiner Trend zu Augmented Reality mit HoloLens (eingebaute Gestenerkennung mit möglicher Erweiterung zu Handtracking)
  + Einordnung in Vorlesungsinhalt:
    - LeapMotion am HeadMountedDisplay als InsideOut-Variante (Nutzer braucht keine Kameras aufstellen, die sich nicht mitbewegen) für Handtracking mit Infrarotlicht-Laufzeitmessung und Modellabbildung (insgesamt 27 DOF in der Hand)
    - KinectV2 als Outside-In Bewegungstracker, der unter anderem mit Infrarotlicht-Laufzeitmessung die Position mit 6DOF bestimmt
    - Unity als Entwicklungsumgebung für VR-Inhalte (unter anderem zu Nutze machen von *FindObjectByTag*.
    - HTC Vive als HeadMounted-Display
    - Umfang / Aufbau und Inhalt