Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Thorsten Hillebrand & Patrick Tschackert

Evaluierung kamerabasierter Tracking-Ansätze für Motion Capture Applikationen

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Thorsten Hillebrand & Patrick Tschackert

Evaluierung kamerabasierter Tracking-Ansätze für Motion Capture Applikationen

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. rer. nat. Kai von Luck
Zweitgutachter : Prof. Dr. rer. nat. Gunter Klemke

Abgegeben am 23. August 2011

Thorsten Hillebrand & Patrick Tschackert

Thema der Bachelorarbeit

Evaluierung kamerabasierter Tracking-Ansätze für Motion Capture Applikationen

Stichworte

Motion Capture, Training, Sport, ARToolkit, ARToolkit Plus, Kinect, PrimeSense, Infrarot-Tiefenkamera, Passive Marker

Kurzzusammenfassung

Diese Arbeit vergleicht verschiedene kamerabasierte Ansätze, mit denen eventuell ein kostengünstiges Motion-Capture System realisiert werden könnte. Des Weiteren wird ein Konzept für ein solches System vorgestellt.

Thorsten Hillebrand & Patrick Tschackert

Title of the paper

Evaluation of camera-based tracking approaches for motion capture applications

Keywords

Motion Capture, Training, Sports, ARToolkit, ARToolkit Plus, Kinect, Primesense, Infrared Depth Camera, Passive Markers

Abstract

This thesis compares camera-based tracking approaches which may be used to build a motion-capture system. Furthermore, the concept of a (distributed) motion capture system using the described approaches is discussed.

Danksagung

Hochschulsport Hamburg: das Fitnessstudio auf dem Campus Berliner Tor stellte Geräte und Trainer zur Verfügung, um Versuche durchzuführen.

Die Entwickler von libfreenect und OpenNI / NITE haben die Nutzung der Kinect unter freien Betriebssystemen ermöglicht.

Dr. Florian Vogt und Prof. Dr. Kai von Luck für die tatkräftige Unterstützung und viele gute Ratschläge während des Erstellens dieser Arbeit.

Unseren Familien, die uns während des Studiums unterstützt haben,
unseren Freuden für die teilweise willkommene Ablenkung während des Studiums,
und *last but not least* unseren Kommilitonen, die in den ersten Semestern Lerngruppen organisiert haben.

Inhaltsverzeichnis

1. Einführung

1.1. Motivation

Mit der zunehmenden Geschwindigkeit und Leistungsfähigkeit von Computern bzw. Mikrocontrollern aus der Massenproduktion erschließen sich immer wieder neue Aufgaben, die durch Computersysteme gelöst oder zumindest für den Menschen vereinfacht werden können. Viele Arbeiten, die früher nur durch Menschen erledigt werden konnten, werden heute zumeist schneller und effizienter von Maschinen bzw. Computern durchgeführt oder auch unterstützt.

Eine Aktivität, deren Effizienz durch rechnerbasierte Unterstützung gesteigert werden könnte, ist (therapeutisches) Muskeltraining. Die genaue Beurteilung bzw. Bewertung von Bewegungsabläufen erfordert (zumindest Anfangs) immer noch einen Trainer, der bei der jeweiligen Übung anwesend ist und darauf achtet, dass der Trainierende die Bewegungsabläufe korrekt durchführt. Dies beinhaltet u.A. die Bewertung von Körperhaltung und Bewegung der einzelnen Gliedmaßen.

Mit einem System, das eine zuverlässige automatisierte Bewertung von Haltung und Bewegungsabläufen ermöglicht, könnten Kosten für z.B. Sportstudios oder indirekt auch für das Gesundheitswesen reduziert werden. Auch bei Übungen zuhause ist ein Einsatz denkbar, sodass bei selbstständigem Training das Risiko, Übungen auf eine kontraproduktive oder eventuell gesundheitsschädliche Art auszuführen, minimiert wird.

Auch jenseits der Einteilung in wünschenswerte und unerwünschte Bewegungen ist der Nutzen für solch ein System gegeben: Denkbar sind Aufzeichnungen über die allgemeine Körperhaltung, um Fehlhaltungen zu erkennen und ggf. orthopädisch behandeln zu lassen. In solchen Szenarien ist auch eine Speicherung der Daten denkbar, um über einen längeren Zeitraum Entwicklungen in der Körperhaltung zu veranschaulichen.

Denkbar wäre ein Heim Media-PC oder Wohnzimmer PC, der rein durch Gesten gesteuert werden kann. Somit wird weder die Maus noch die Tastatur weiterhin benötigt. Mittels Gesten und Audio Steuerung würde der PC sich komplett lenken lassen.

Im Rahmen dieser Bachelorarbeit sollen verschiedene kamerabasierte Verfahren auf ihre Eignung zur Realisierung eines kostengünstigen Motion Capture Systems verglichen werden. Insbesondere Faktoren wie Genauigkeit, Geschwindigkeit und Beeinflussung durch Umgebungsvariablen sollen untersucht werden.

1.2. Gliederung

Diese Arbeit ist in folgende Bereiche gegliedert:

Im ersten Teil, den **Grundlagen**, wird für das Verständnis dieser Arbeit vorausgesetztes Wissen erläutert sowie Hintergrundinformationen zu Motion Capture genannt.

In der **Analyse** wird die konkrete Ausgangssituation mit den einhergehenden Anforderungen dargestellt, um klare Ziele hinsichtlich eines Motion Capture Systems zu definieren.

Bei der Vorstellung der **Messverfahren** werden deren Funktionsweisen sowie deren jeweilige Schwächen und Stärken anhand von Versuchen erläutert. Anschließend werden in einem Fazit die Unterschiede zwischen den Messverfahren dargestellt mit dem Versuch, das sich am besten eignende zu ermitteln.

Im Kapitel **Entwicklung** wird die Konzeption und das Design eines verteilten Motion Capturing Systems erläutert, welches eines oder mehrere der vorgestellten Messverfahren einsetzt.

2. Grundlagen

2.1. Augmented Reality

Der Begriff “Augmented Reality“ (zu Deutsch “Erweiterte Realität“) wurde erstmals 1990 von dem bei Boeing arbeitendem Ingenieur Tom Caudell verwendet (Brian X. Chen (2009)) und beschreibt ein Verfahren um die menschliche Wahrnehmung der Realität durch zusätzliche Informationen zu erweitern.
So können z.B. einer normalen Videoaufnahme zusätzliche grafische Elemente hinzugefügt werden, die für den Betrachter interessant bzw. informativ sind. Ein Beispiel wäre eine AR-Anwendung, die während eines Spaziergangs in der Stadt herkömmliche Werbeplakate ausblendet und durch personalisierte Werbung o.ä. ersetzt. weitere Anwendungsfälle:

- Spieleprogrammierung
- Skelettmodell
- Bewegungsaufzeichnung
- Objekt-Verfolgung
- Simulation

2.2. Begriffserläuterung

Im Rahmen dieser Arbeit werden folgende Begriffe verwendet:

- **Pose** Eine Körperhaltung ohne Bewegung. Ein Besipiel ist die Kalibrierungspose, die in Abschnitt 4.3.2.4 beschrieben wird. Eine Pose stellt sozusagen eine Momentaufnahme einer Haltung des menschlichen Körpers dar.
- **Geste** Eine Geste ist ein Bewegungsablauf, die einen fest definierten Anfang, Verlauf und ein Ende hat. Beispiele sind Winken, Achselzucken oder Nicken.

- **Bewegung** Eine Bewegung wird in dieser Arbeit als eine fortlaufende Bewegung des menschlichen Körpers ohne klaren Anfang und Ende definiert. Beispiele sind insbesondere sportliche Übungen, die in Abschnitt 3.2 genannt werden.

2.3. Motion Capture

Mit Motion Capture werden im Allgemeinen Verfahren bezeichnet, welche (meist menschliche) Bewegungen aufzeichnen. Die Nutzung von Motion Capture ist etwa in der Film- und Videospielindustrie weit verbreitet, um möglichst realistisch wirkende Bewegungsabläufe in computergenerierten Videosequenzen zu erzeugen.

2.3.1. Skelettmodelle

Der Begriff Skelettmodell beschreibt ein auf das Skelett reduziertes Modell eines menschlichen Körpers. Da jede Bewegung von Gliedmaßen an Gelenken passiert, ist somit eine Reduktion der Datenmenge möglich. Des Weiteren kann durch eine korrekte Modellierung des Skelettmodells “unmöglichen” Bewegungsabläufen vorgebeugt werden, wie zum Beispiel eine Drehung des Kopfes um mehr als 180°.

3. Analyse

3.1. Einleitung

In diesem Kapitel soll die Problemstellung (Motion Capture gestütztes Muskeltraining) analysiert und konkretisiert werden.
Hierbei wird die Ausgangssituation von bisherigen Trainingsabläufen dargestellt, der derzeitige Stand der Technik vorgestellt und die Kriterien für eine passende Realisierung anhand von typischen Szenarien und Anforderungen festgelegt.

3.1.1. Bisheriger Ablauf Muskeltraining

Die Evaluierung der verschiedenen kamerabasierten Motion Capture Ansätze verfolgt das Ziel, Motion Capture effektiv in einen Trainigsablauf für den Menschen einzubinden. Hierzu ist erst eine Betrachtung der Ausgangssituation erforderlich.

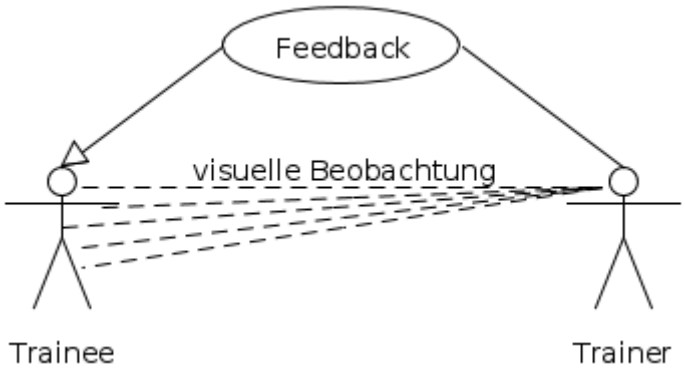

Abbildung 3.1.: Training mit Trainer

Bei Trainingsübungen führt eine Person bestimmte Bewegungen aus, die der Beanspruchung bestimmter Muskelgruppen dient (mit oder ohne Sportgerät). Hierbei ist normalerweise ein menschlicher “Trainer” zugegen, der den “Trainee” (Person, welche die Übung durchführt) überwacht und ggf. bei falscher Durchführung der Übung den Trainee anweist.

Der Trainer ist somit eine Kontrollinstanz, welche den Trainee auf fehlerhafte Bewegungen prüft und Anweisung für Verbesserungen gibt (siehe Abb. 3.1).

3.1.2. Bisheriger Ablauf Motion Capture

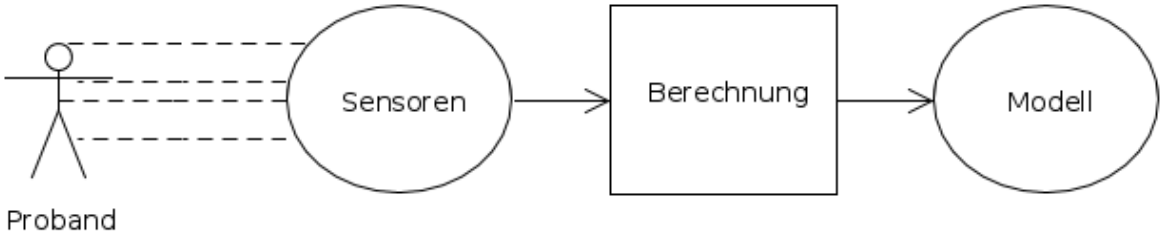

Abbildung 3.2.: Bisheriger Motion Capture Ablauf

Bisherige Motion Capture Einsatzzwecke sind vor allen die Aufzeichnung menschlicher Bewegungen, um sie möglichst realistisch in Filmen und Computerspielen wiederzugeben.
Eine Bewertung bzw. Korrektur der Bewegungen ist bei diesen Szenarien nicht erwünscht oder vorgesehen, da die Bewegungsabläufe immer “korrekt” sind.

Abstrakt gesehen beschränkt sich der Ablauf auf die Erzeugung eines Körpermodells aus Sensorinformationen, wie in Abb. 3.2 beschrieben.

3.1.3. Integration von Motion Capture in Trainigsübungen

Im Rahmen dieser Arbeit wird der bisherige Motion Capture Ansatz um eine Bewertung (und ggf. Korrektur) der Bewegungsabläufe erweitert (siehe Abb. 3.3), die ursprünglich vom Trainer vorgenommen wurde. Bis die Zuverlässigkeit und im akzeptablen Bereich niedrige Fehlerquote eines solchen Systems untersucht ist, empfiehlt es sich dennoch, als letzte Kontrollinstanz einen menschlichen Trainer hinzuzuziehen, der das Motion Capture System auf Fehler überwacht.
Analog dazu könnte man sagen, dass die Rolle des Systems der eines Autopiloten entspricht, da der Autopilot eines Flugzeugs den menschlichen Piloten nicht komplett ersetzt, sondern bei der Wahrnehmung seiner Aufgaben unterstützt.

3.1.4. Bewegungsanalyse vs. Aufzeichnung von Bewegungen

In bereits vorhandenen Motion Capture Systemen geht es, wie oben beschrieben, um die reine Aufzeichnung menschlicher Bewegungen. Um jedoch während einer sportlichen Übung

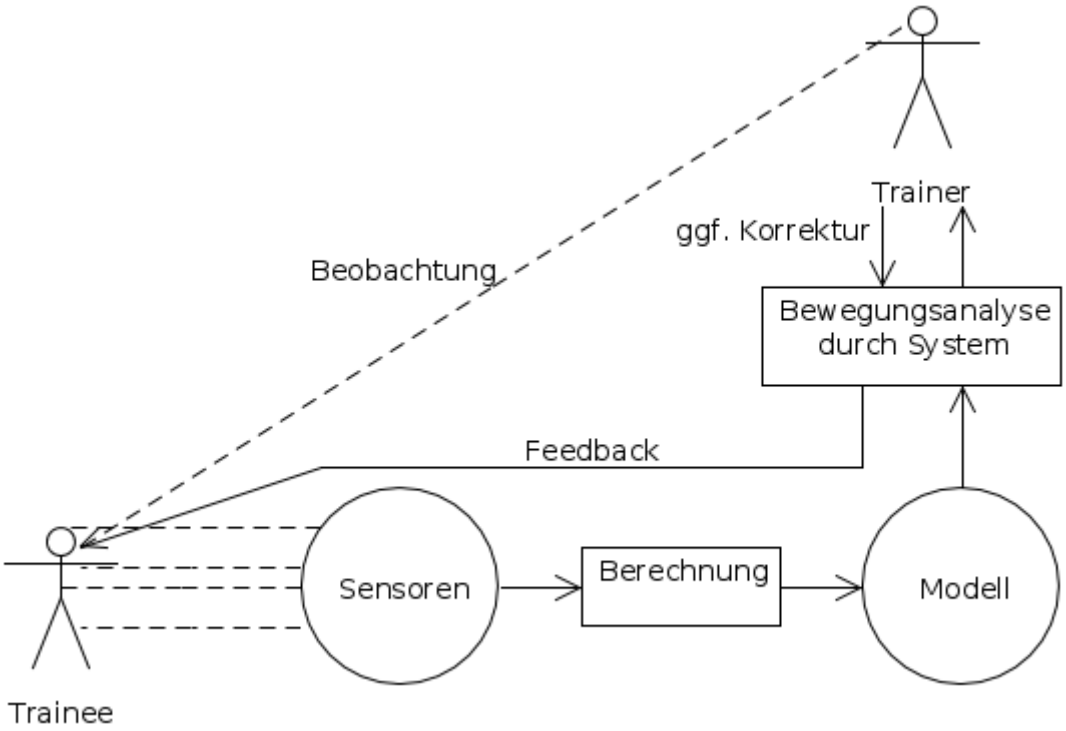

Abbildung 3.3.: Bewertender Motion Capture Ansatz

sinnvolles Feedback geben zu können, muss eine Einteilung von Bewegungsabläufen in sinnvolle (“gute”) und falsche (“schlechte”) stattfinden. Dies ist ein weiterer Grund, das Feedback eines menschlichen Trainers zu berücksichtigen.
Falls tatsächlich einmal auf lange Sicht das Motion Capture System den Trainer ersetzen sollte, muss sich dieses erst durch Lernverhalten die Bewertungen menschlicher Bewegungen zu Eigen machen.

3.2. Anwendungsszenarien

Szenarien dienen dazu, Arbeitsabläufe anhand von Beispielen konkret zu definieren und so besser allgemeine Anforderungen formulieren zu können. Wenn der Ablauf, die beteiligten Akteure und Rahmenbedingungen klar definiert sind, lassen sich Ziele für die Softwareentwicklung leichter festlegen. Die hier beschriebenen Szenarien sollen repräsentative Teilbereiche des weiten Feldes “Muskeltraining” abdecken, ohne jedoch Anspruch auf Vollständigkeit zu erheben.

Die hier beschriebenen Szenarien bauen auf Abschnitt 3.1.3 auf, wo eine mögliche Integration von Motion Capture in eine Trainingssituation erläutert wird.

3.2.1. Szenario 1: Ruderergometer

Abbildung 3.4.: Trainee mit ARToolkit-Markern auf Ruderergometer

Dieses Szenario beschreibt eine Standardsituation in einem Sportstudio, eine Übung an einem Ruderergometer (Rudergerät). Der Trainee sitzt auf einem beweglichen Sitz und zieht an einem Stab, welcher mit einem Seil am Gerät befestigt ist. Die Füße des Trainees sind in einer festen Position fixiert, der Rest des Körpers ist in Bewegung. Je nach Blickwinkel der Aufnahme sind Körperpartien des Probanden vom Gerät partiell verdeckt.
Während die Übung durchgeführt wird, soll die Haltung und die Bewegung des Körpers bzw. der Gliedmaßen bewertet werden, um ein “falsches” Training (d.h. unnütze oder gar schädliche Beanspruchung der Muskeln) zu vermeiden. Der Trainierende führt die Übung durch, seine Bewegungsabläufe werden dem jeweiligen Verfahren entsprechend aufgezeichnet.
Der menschliche Trainer bewertet zunächst die Durchführung der Übungen als “gut” oder “schlecht”. Diese Bewertungen dienen dem Motion Capture System als Maßstab, wie es künftige Bewegungsabläufe zu bewerten hat.
Bewegungen, die vom System als suboptimal erkannt werden, werden in der aufgezeichneten Bewegungssequenz markiert und später dem Trainer vorgeführt, um sicher zu gehen, dass es sich um eine Fehlhaltung handelt. So sollen “false positives”, also fehlerhaftes Anzeigen von vermeintlich falschen Bewegungen vermieden werden.

3.2.2. Szenario 2: Bauchpressen

Dieses Szenario beschreibt die Durchführung sogenannter Bauchpressen (engl. “Crunches”), welche normalerweise auf dem Boden ohne ein Gerät durchgeführt werden. Bauchpressen sind den sogenannten Sit-Ups ähnlich und zielen auf eine Stärkung der Bauchmuskulatur ab. Der Trainee liegt mit dem Rücken auf dem Boden, die Beine rechtwinklig zum

Boden aufgestellt und die Knie bei 90° gebeugt.
Die Durchführung der Übung besteht darin, den Oberkörper vom Boden wiederholt abzuheben. Anders als bei Sit-Ups soll bei dieser Übung das Kreuz nicht vom Boden abheben.
Wie beim ersten Szenarien soll der Trainee von einem Motion-Capture System beobachtet werden, welches eine nicht zweckmäßige Durchführung der Übung anzeigt. Auch hier soll das Feedback bzw. die Rückmeldung des Systems durch einen menschliche Trainer überwacht und ggf. korrigiert werden.

3.2.3. Vergleich der Szenarien

Beide Szenarien beschreiben die Computergestützte Durchführung einer Trainingsübung. Der größte Unterschied zwischen den beiden Szenarien besteht darin, dass bei Szenario 1 eine Sportgerät benutzt wird, welches evtl. Körperpartien des Trainees verdecken und so eine Datenerfassung erschweren könnte.

3.2.3.1. Trainingsumgebung

Die hier beschriebenen Szenarien gehen von einer vordefinierten Umgebung aus, in der das jeweilige Motion Capture System ohne Probleme einsetzbar ist. Dies bedeutet, dass z.B. bei kamerabasierten Ansätzen genug Licht vorhanden ist, bzw. dass keine Infrarotstrahlung das System beeinträchtigt.
Ein normal beleuchteter Innenraum (ohne Objekte, die die Sicht behindern) reicht.

3.3. Anforderungen

Als Anforderungen werden beim Software Engineering Formulierungen verstanden, welche die Aufgabe und das gewünschte Verhalten der Software beschreiben sollen. Hierbei wird zwischen funktionalen und nicht-funktionalen Anforderungen unterschieden, zum Beispiel einerseits zwischen der zu erledigenden Aufgabe (z.B. präzises Motion Capture, funktional) und generellem Verhalten (niedriger Kostenfaktor, nicht funktional).
Des Weiteren ist eine Aufteilung zwischen “harten” und “weichen” Anforderungen möglich, wobei harte Anforderungen die essentielle Funktionalität bzw. Verhalten der Software beschreiben, weiche Anforderungen hingegen sekundäre Ziele, die ggf. auch erst später umgesetzt werden können (“nice-to-have”).
Aus den Szenarien lassen sich folgende Anforderungen an ein Motion-Capture System zur Unterstützung von Muskeltraining formulieren:

3.3.1. Harte Anforderungen

3.3.1.1. Funktional

- **Präzise Aufzeichnung menschlicher Bewegungen**
  Hohe Präzision der Messdaten ist für eine Anwendung im medizinischen Bereich unabdingbar, da eine eventuelle Ungenauigkeit der Daten auch eine Ungenauigkeit der Ergebnisse bedeuten könnte. Erwartet wird, dass zur sinnvollen Auswertung der Daten eine zentimetergenaue Präzision nötig ist, um Fehlhaltungen der Gliedmaßen bei sportlichen Übungen korrekt zu erkennen.
- **Schnelle Aufzeichnung menschlicher Bewegungen**
  Um natürliche menschliche Bewegungen in etwa so präzise wie das menschliche Auge aufzuzeichnen, ist eine Bildfrequenz von ca. 10 Hertz notwendig. Zum Beispiel muss eine Bewegung vollständig aufgezeichnet werden, d.h. einzelne Details der Bewegung dürfen nicht "verloren" gehen. Wäre z.B. der Pfad eines Gelenkpunktes nur lückenhaft aufgezeichnet, könnten Bewegungsnuancen komplett aus der Bewertung herausfallen. Diese Anforderung überschneidet sich teils mit der ersten, da bei der Aufzeichnung von Bewegungen Schnelligkeit auch Präzision bedeutet.
- **Vollständigkeit**
  Um den Anspruch auf Präzision halten zu können, muss das System möglichst vollständige Daten zur Verfügung haben. Optimal wäre eine Betrachtung der Versuchsperson aus mehreren Blickwinkeln, um ein komplettes 360° Modell zu erhalten. Falls eine Datenerfassung in diesem Maße nicht oder nur sehr schwer möglich ist, müssen die fehlenden Daten ggf. extrapoliert, d.h. Näherungswerte mittels Algorithmen berechnet werden.
- **Korrekte Auswertung**
  Eine korrekte Auswertung bzw. Interpretation der Daten ist ebenso wichtig wie die präzise Aufzeichnung derselben. Bei fehlerhafter Bewertung der Bewegungsabläufe durch das System droht eine Festigung von inkorrekten Bewegungsabläufen, die wiederum Haltungsschäden verursachen könnten.

3.3.1.2. Nicht funktional

- **Kostenfaktor**
  Um die Lösung wirtschaftlich oder auch gewinnbringend und flächendeckend in Sportstudios bzw. therapeutischen Einrichtungen einzusetzen, müssen sich die Kosten für die Komponenten in Grenzen halten.

3.3.2. Weiche Anforderungen

3.3.2.1. Funktional

- **Multiuser**
  Das System sollte in der Lage sein, mehrere Benutzer gleichzeitig zu verwalten. Denkbar wäre, den erkannten anonymen Benutzer mit einer Gesichtserkennungssoftware eindeutig zu identifizieren. Somit könnte eine Art Überwachungssoftware erstellt werden, die einen Benutzer permanent im dreidimensionalen Raum verfolgt.
- **eindeutige Erkennung eines Users**
  Die Kombination aus dem Tiefenbild und der Gesichtserkennung ermöglicht eine dauerhafte Erkennung des Anwenders. Die Tiefendaten bieten eine gute Struktur vom Gesicht, die ggf. zur Identifizierung verwendet werden kann. Ebenso könnten die Daten vom Skelettmodell nach der Kalibrierung ausschlaggebende Werte über die Extremitäten der Person enthalten, so dass ein Benutzer daran mit hoher Wahrscheinlichkeit erkannt werden kann.

3.3.2.2. Nicht funktional

- **Performance und Realtime**
  Um Korrekturen an der Bewegung des Trainees vorzunehmen, muss die Rückmeldung während (und nicht nach) der Übung erfolgen. Eine lange Berechnungszeit würde das Benutzen des Systems unnötig erschweren und / oder verkomplizieren. Der Gesamtzeitraum vom Ausführen der Bewegung bis zum Feedback an den Trainee sollte idealerweise nicht mehr als ein paar Sekunden in Anspruch nehmen.
- **Einfacher Auf- und Abbau**
  Ein einfacher Auf- und Abbau des Systems würde eine Anpassung an unterschiedliche Trainingsumgebungen erleichtern und die Kosten niedrig halten, da ein System mehrfach verwendet werden kann.

3.4. Motion Capture Technologie

3.4.1. Geschichtliche Einordnung

Die Aufzeichnung menschlicher Bewegungen zum Zwecke der Interaktion mit Computern ist in der Informatik mindestens seit der Erfindung der Maus ein Thema. Bereits 1964 wurde mit

dem RAND Tablet ein Gerät vorgestellt, das (basierend auf Bewegungen der menschlichen Hand) eine kostengünstige Schnittstelle zwischen Mensch und Computer darstellen sollte (Davis und Ellis (1964)). In jüngerer Vergangenheit ist eine Weiterentwicklung dieser Idee bei Multitouch-Bildschirmen zu beobachten, die simultan Bewegungen mehrerer Punkte auf ihrer Oberfläche verarbeiten können.
Während viele frühe Ansätze wie dieser einen sog. Stylus oder anderes Zubehör voraussetzten und nur einen relativ kleinen Teil möglicher Gesten erkennen konnten (Bewegungen der Hand oder der Finger), haben sich heutzutage die Möglichkeiten verfielfacht. Styli und Mäuse haben mittlerweile ihren festen Platz in der Bedienung von Computern, während Geräte wie z.B. VR-Handschuhe auch erforscht werden (Sturman und Zeltzer (1994)).

3.4.2. Motion Capture im weiteren Sinne

Wenn unter dem Oberbegriff Motion Capture generell die Aufzeichnung menschlicher Bewegungen zu verstehen ist, lassen sich auch folgende Ansätze dort einordnen:

- **Lichtschranken**
  Unter Anderem eingesetzt bei Objektsicherung und Einbruchsschutz. Bei diesem Ansatz löst eine Unterbrechung eines Lichtstrahls ein Signal aus. Vor allem die Anwesenheit von Menschen in einem bestimmten Gebiet soll so durch eine Wahrnehmung ihrer Bewegungen registriert werden.

- **Bewegungsmelder**
  Wie bei Lichtschranken besteht auch hier der Sinn darin, die Anwesenheit von Menschen in einem bestimmten Gebiet zu registrieren. Mittels Ultraschall, elektromagnetischen Wellen oder Infrarot werden Bewegungen aufgezeichnet und lösen ein Signal aus.

Diese Verfahren sind zwar dazu konzipiert, menschliche Bewegungen anzuzeigen, eignen sich jedoch nicht dafür menschliche Bewegungen mit der annähernden Präzision eines menschlichen Trainers aufzuzeichnen. Ein Insekt kann (abhängig von der Umsetzung der Ansätze) theoretisch zu einem “false Positive” führen.

3.4.3. Kommerzielle Motion Capture Systeme

Folgende Ansätze werden derzeit vor Allem in dem in Abb. 3.2 beschriebenen Ablauf verwendet:

- **Optisch**
  Durch viele am menschlichen Körper angebrachte pulsierende LEDs werden die Positionen der einzelnen Marker ermittelt. Die große Anzahl der benötigten Marker und deren aufwändige Kalibrierung stellt einen erheblichen Kostenfaktor dar.

- **Reflexiv**
  Ähnlich dem optischen Ansatz werden die Positionen der einzelnen Marker durch Infrarot ermittelt. Auch hier ist wegen der o.g. Umstände der Kostenfaktor sehr hoch.

Diese Motion Capture Ansätze haben sich in ihren Bereichen als effizient etabliert. Die hohen Kosten bedeuten jedoch, dass ein flächendeckender Einsatz solcher Systeme, zum Beispiel in Trainingsumgebungen, nicht möglich ist.

3.4.4. Kamerabasierte Ansätze

Die rasante Entwicklung und weite Verfügbarkeit von digitalen Kameras bieten neue Ansätze, die eventuell ein kostengünstiges Motion Capture System ermöglichen könnten:

- **Stereo Kamera**
  Zwei Kameras werden leicht versetzt aufgestellt. Der Abstand der beiden Kameras zueinander entspricht in etwa dem menschlichen Augenabstand. Gleichzeitig nehmen die beiden Kameras das Bild auf, wodurch die Berechnung der Tiefendaten ermöglicht wird. Das Verfahren der Stereoskopie (vgl. Devernay und Faugeras (1994)) wird auf die beiden Bilder angewendet um Rückschlüsse auf die Tiefe zu ziehen.

- **Plenoptische Kamera**
  Eine plenoptische Kamera hat eine hohe Tiefenschärfe und das Bild beinhaltet nicht nur die Position und Intensität des Lichtstrahls, sondern auch die Richtung, aus der der Lichtstrahl eintrifft. Rein theoretisch könnte eine plenoptische Kamera 3D Funktionalität zur Verfügung stellen (vgl. Ng u. a. (2005)), ist aber für Motion-Capture Zwecke nicht praktikabel.

- **TOF-Kamera (Time of Flight)**
  Eine TOF-Kamera verwendet Licht innerhalb des infraroten Spektrums. Durch das Verfahren der Laufzeitmessung des Lichts können Rückschlüsse auf die Distanz gezogen werden. Dies geschieht in folgenden Einzelschritten. Durch einen Lichtimpuls wird die Szene beleuchtet. Das reflektierende Licht wird von der Kamera wieder eingesammelt und die Szene wird so rekonstruiert (vgl. Lange und Seitz (2001)). Anhand der Verzögerung, bis das Licht für jeden einzelnen Pixel wieder eingetroffen ist, lässt sich nun die Distanz berechnen. Da sich das Licht mit Lichtgeschwindigkeit ($c = 299.792.458 m/s$) ausbreitet, sind diese Verzögerungen sehr gering.
  Hier ein Beispiel mit bekannter Entfernung zum Objekt:

- t_D: zeitliche Differenz
- D: bekannte Distanz
- c: Lichtgeschwindigkeit = $299.792.458 m/s$ (Konstante)

$$t_D = 2\frac{D}{c} = 2\frac{3m}{299.792.458 m/s^{-1}} = 20ns$$

In diesem Beispiel mit einer bekannten Distanz von 3m zum Objekt ist das reflektierende Licht binnen 20ns wieder beim Ursprung.

Hier ein Beispiel mit bekannter Laufzeitverzögerung:

- D_{max}: maximale Distanz von Kamera zum Objekt
- c: Lichtgeschwindigkeit = $299.792.458 m/s$ (Konstante)
- t_0: bekannte Laufzeitverzögerung

$$D_{max} = \frac{c * t_0}{2} = \frac{299.792.458 m/s^{-1} * 30ns}{2} = 4,5m$$

Mit einer bekannten Laufzeitverzögerung von 30ns kann sich das Objekt nur in maximal 4,5m entfernung befinden.

- **Optische und markerbasierte Verfahren**
  Bei optischen markerbasierten Verfahren werden Marker am Körper befestigt. Die Daten einer normalen digitalen Kamera werden ausgewertet, um hinterher die Position des Markers zu berechnen.
- **Infrarot Tiefen -Sensor/-Kamera**
  Das Sichtfeld einer Infrarotkamera wird mit Punkten ausgeleuchtet, die im Infrarot-Spektrum liegen und somit nicht für das menschliche Auge sichtbar sind. Die Daten der Infrarotkamera werden ausgewertet und somit werden Rückschlüsse auf die Tiefe der angeleuchteten Objekte gezogen. Dies kann durch verschiedene Verfahren realisiert werden. Eine TOF-Kamera zieht Rückschlüsse durch das Laufzeitverfahren, während z. B. bei einer Implementierung des PrimeSense Systems (z. B. Microsoft Kinect) das Verfahren des “Structured Light Coding” (vgl. Albitar u. a. (2007)) zur Anwendung kommt.

Im Rahmen dieser Bachelorarbeit werden konkrete Implementierungen der letzten beiden Verfahren auf ihre Eignung für Motion Capture untersucht, da sie relativ neu und kostengünstig sind.

3.4.4.1. Verschlusszeit und Bildrate von Kameras

Als Verschlusszeit (engl. Shutter Speed) wird die Belichtungszeit von Kameras bezeichnet, d.h. die Zeit, wie lange es dauert ein Bild aufzunehmen. Eine niedrige Verschlusszeit bedeutet, dass Einzelbilder klar und mit möglichst wenig Bewegungsunschärfe aufgenommen werden. Die optische Detektion von passiven Markern (wie bei optisch markerbasierten Verfahren) wird somit erleichtert. Die "Bildrate" bezeichnet die Häufigkeit von Aufnahmen innerhalb einer Sekunde (engl. "Frames per Second"). Eine hohe Bildrate lässt aufgenommene Bewegungen "flüssiger" erscheinen.

3.4.5. Bewertung verschiedener Ansätze

Da im Rahmen dieser Arbeit ein Verfahren ausgewählt werden soll, mit dessen Hilfe möglichst preisgünstig ein Motion-Capture System entwickelt werden kann, sind die Kosten einer der wichtigsten Faktoren, die beachtet werden müssen. Die vorgestellten etablierten Ansätze, z.B. optische und reflexive Verfahren, zählen zum derzeitigen Standard bei Motion-Capture. Die Kosten sind jedoch bei diesen Verfahren so hoch, dass ein flächendeckender Einsatz im mittelständischen oder auch privaten Bereich aus wirtschaftlichen Gründen zur Zeit nicht denkbar ist.
Übrig bleiben die zwei Verfahren, die zumindest die Anforderungen an den Kostenfaktor erfüllen, nämlich optisch markerbasierte und IR-Tiefenkamera Ansätze.

Anforderungen **Verfahren**	Genauigkeit	Geschwindigkeit	Vollständigkeit	Kosten
Optisch	hoch	hoch	ja	> 10 000 €
Reflexiv	hoch	hoch	ja	> 10 000 €
Optisch markerbasiert	variabel[1]	variabel[2]	variabel[1]	je nach Kamera, dazu sonstige Hardware
Tiefenkamera	mittel[3]	30 FPS	ja	ca. 100 € pro Kamera, dazu sonstige Hardware

[1]: je nach Anzahl der eingesetzten Marker
[2]: je nach Bildrate und Verschlusszeit der eingesetzten Kamera
[3]: genaue Tiefendaten, erreicht nicht kommerzielle Systeme

4. Messverfahren

4.1. Einleitung

In diesem Kapitel werden zwei kamerabasierte Ansätze für Motion Capture anhand von vorhandenen Implementierungen vorgestellt und mit Hilfe von Versuchsreihen auf ihre Tauglichkeit überprüft.
Auf Grund der in Abschnitt 3.4.5 vorgenommenen Überlegungen wurden die beiden Ansätze **optisch markerbasiert** und **Infrarot-Tiefenkamera** ausgewählt.
Für den optisch markerbasierten Ansatz wurde das ARToolkit gewählt. Beim ARToolkit steht das Erkennen von passiven Papiermarkern im Vordergrund, die Entwicklung eigener Applikationen ist bewusst einfach gehalten, um einen Einstieg für Entwickler einfach zu gestalten. Dies ermöglicht eine schnelle Entwicklung von Softwarekomponenten, deren einzige Aufgabe die Detektion von Markern ist. Des Weiteren trägt die offene Natur des Projekts (durch GPL lizensiert) dazu bei, den Kostenfaktor minimal zu halten.
Für den Infrarot-Tiefenbild Ansatz wurde die Microsoft Kinect ausgewählt. Die Hauptargumente für die Verwendung dieses Gerätes (und der dazugehörigen Software) liegen einerseits in der Präzision des Gerätes (viel mehr Referenzpunkte zur Feststellung von Koordinaten als optische Marker zur Verfügung stellen könnten) und in den vergleichsweise niedrigen Kosten, da die Kinect als Zubehör zu einer Spielkonsole für den Massenmarkt konzipiert wurde.

4.2. Optisch Markerbasiert: Das ARToolkit

4.2.1. Ursprung und Zweck

ARToolkit ist eine von Hirokazu Kato am Nara Institute of Science and Technology entwickelte Softwarebibliothek, die die Ortung zweidimensionaler passiver Marker in Echtzeit in Videosequenzen ermöglicht (Kato und Billinghurst (1999)).
Der eigentliche Zweck dieser Software ist es, eingehende Videodaten auf Marker zu untersuchen, und virtuelle Objekte an den Markerpositionen zu zeichnen. Das resultierende Videobild wird dann über ein HMD angezeigt, um so Objekte dreidimensional und für den

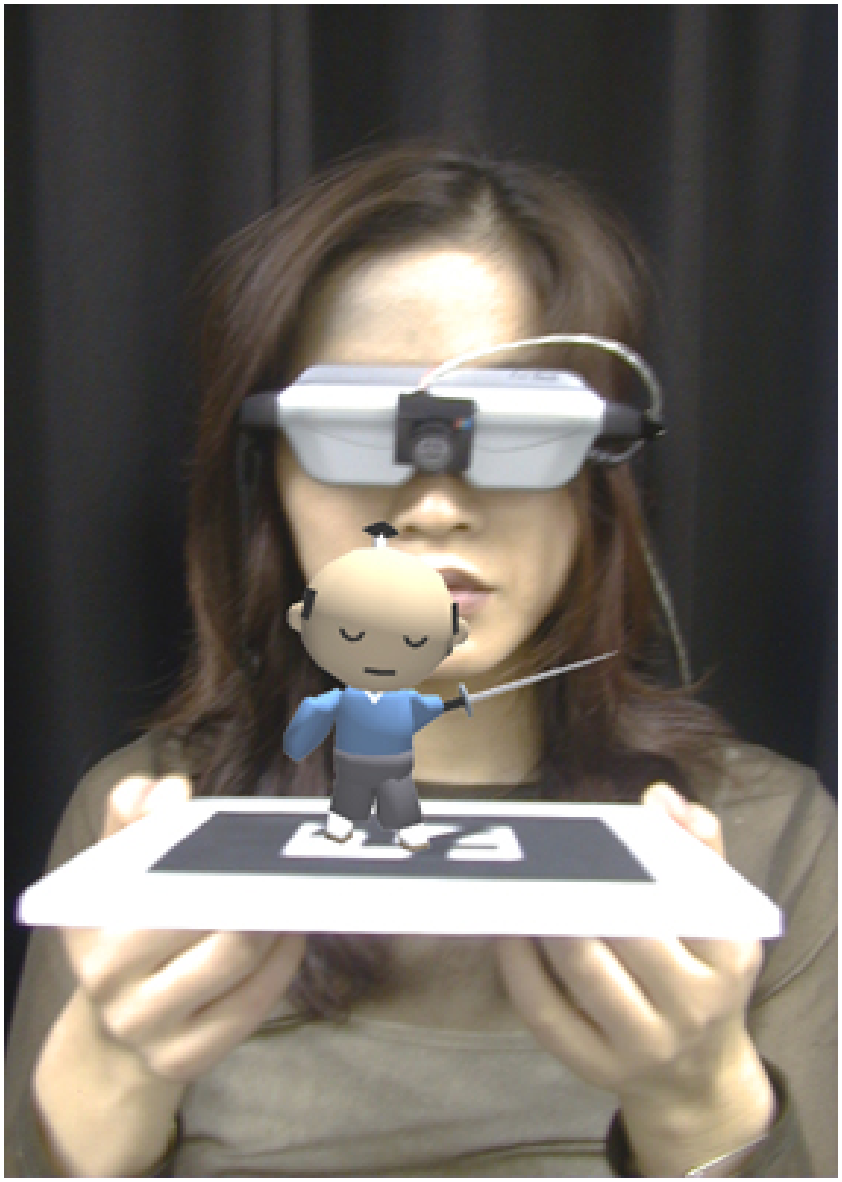

Abbildung 4.1.: ARToolkit-Beispiel
HIT Lab Washington University (2011) Eine einfache AR-Anwendung mit ARToolkit. Die Figur, die auf dem Marker “steht“, wird nachträglich in das Bild eingefügt.

Betrachter perspektivisch korrekt im Raum darstellen zu können. Ursprünglich gedachte Anwendungsfälle sind in Abschnitt 2.1 genannt.

Abbildung 4.2.: Beispielmarker
Auf z.B. Papier ausgedruckt, kann dieser Marker mit ARToolkit verwendet werden, um eine AR-Anwendung zu erstellen.

4.2.1.1. Verbreitung

In den letzten Jahren sind durch das erhöhte Aufkommen sogenannter Smartphones auch mobile Plattformen verfügbar, die eine Echtzeit-Berechnung von Videodaten ermöglichen. So hat die Firma ARToolworks im April 2010 eine erste Betaversion des ARToolkits für Apples iOS herausgegeben (ARToolworks Inc. (2010)). So können auch unterwegs AR-Anwendungen genutzt werden. Des Weiteren ist auch eine ARToolkit Version für Smartphones mit Googles Android-Betriebssystem verfügbar (ARToolworks Inc. (2011)). Bei der Benutzung des ARToolkits auf mobilen Platformen wird normalerweise auf ein HMD verzichtet, stattdessen wird der Bildschirm des mobilen Gerätes zum Anzeigen des Resultates verwendet.
Auch für herkömmliche PC existieren viele alternative Implementierungen bzw. Wrapper für verschiedene Zwecke, z.B. für die Benutzung unter anderen Programmiersprachen als C/C++. Darunter befindet sich auch das ARToolkitPlus, dessen Entwickler größere Präzision bzw. Fehlerkorrektur in Aussicht stellen. Dieses Projekt ist jedoch mittlerweile von den Entwicklern eingestellt und durch den “Studierstube Tracker” ersetzt worden (Graz University of Technology (2011)).

Einsatz zu Motion Capture Zwecken In Sementille u. a. (2004) wurde das ARToolkit bereits als Motion Capture Komponente untersucht und theoretisch eine generelle Eignung für solche Zwecke festgestellt. Ergänzend dazu soll im Rahmen dieser Arbeit die Präzision bzw. Geschwindigkeit des Verfahrens anhand von Versuchen ermittelt werden, um eine Eignung unter reellen Bedingungen zu untersuchen.

4.2.2. Funktionsweise

Ein Programm, welches die ARToolkit Bibliothek benutzt, läuft typischerweise in einer Schleife ab, die folgende Arbeitsschritte beinhaltet:

- Einzelbild von Videoquelle (Kamera) holen
- Marker in dem Bild anhand des schwarzen Randes ermitteln
- Ermittelte Marker anhand des Inhalts identifizieren
- Dreidimensionale Projektionen von computergenerierten Objekten mittels OpenGL an die errechneten Markerpositionen zeichnen
- Ausgabe von generiertem Bild über dem Originalbild

Dieser Ablauf ist in Abb. 4.3 als Flussdiagramm dargestellt.
Die Ausgabe des Programms sind somit Videosequenzen, denen perspektivisch korrekt dreidimensional dargestellte Objekte hinzugefügt wurden.

4.2.2.1. Hardware

Es sind (jenseits von handelsüblicher PC-Peripherie) keine speziellen Hardwareanschaffungen nötig, um eine ARToolkit Anwendung zu realisieren. Als Kamera kann auch eine normale "Webcam" verwendet werden (ein Beispiel ist die im Rahmen einiger Tests verwendete Logitech C500, siehe Abb. 4.4(a)), auch Netzwerkkameras (wie die im Rahmen der Versuchsreihe verwendete Axis211W (siehe Abb. 4.4(b)) können eingesetzt werden.

4.2.3. Bewertung

Das ARToolkit bietet Vorteile, wie auch Nachteile. Zum einen sind Marker erforderlich. Diese müssen eine gewisse Größe haben, damit die Marker auch für das ARToolkit zu unterscheiden sind. Ein 3D-Skelettmodell mit einer Art Marker-Anzug wäre realisierbar, allerdings wird durch die benötigten Marker, wie durch die Kleidung die reelle Position des Markers verfälscht. Es wären viele Marker erforderlich, die dann noch zu einem Modell

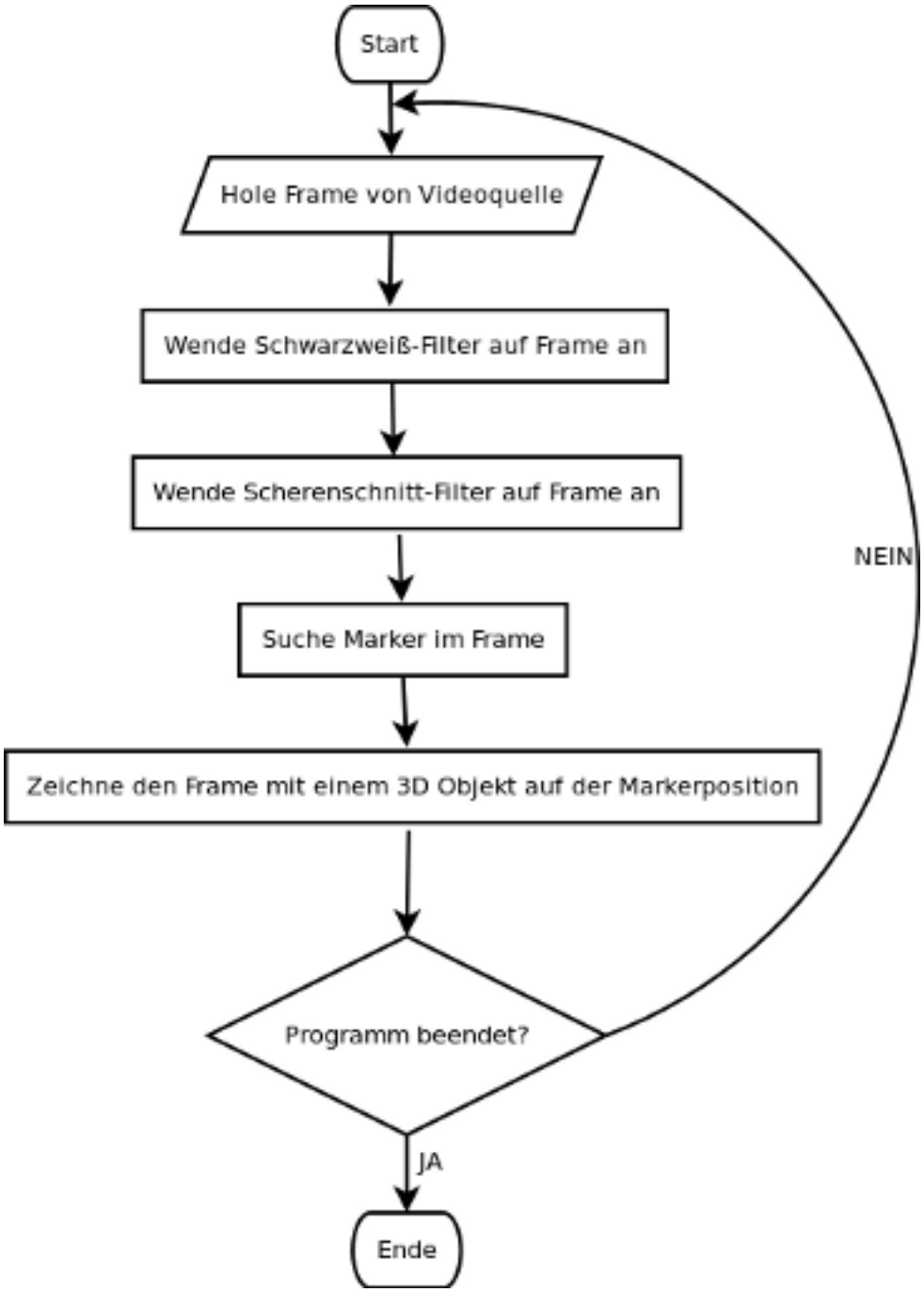

Abbildung 4.3.: Flussdiagramm ARToolkit

(a) Logitech C500 (b) Axis 211W

Abbildung 4.4.: Mit ARToolkit verwendete Kameras (Logitech (2011), Axis (2011))

logisch zusammengefügt werden müssten.
Ein weiterer Nachteil besteht in dem Material, aus dem die Marker sind. Ist das Papier zu dünn, wellt der Marker leicht und wird nicht erkannt. Ist die Pappe zu starr, wird die Position oder die Drehung des Markers verfälscht. Wenn die Oberfläche des Markers spiegelnd ist, wird der Marker nur mit geringer Wahrscheinlichkeit erkannt. Die Ausleuchtung des Raums spielt auch eine wichtige Rolle. Ob direktes oder indirektes Licht verwendet wird, hängt sehr von der Umgebung, wie auch von Material der Marker ab. Wenn die Marker zu klein gewählt werden erhöht sich die Wahrscheinlichkeit, dass die Marker vertauscht werden. Es kann nicht mehr garantiert werden, dass die entsprechende Marker ID auch wirklich vorliegt.

4.2.3.1. Vorteile

- **einfach gestaltete API**
  eine Client-Anwendung, um die Koordinaten von Markern im dreidimensionalen Raum aufzuzeichnen, lässt sich schnell und einfach umsetzen
- **Verwendung mehrerer Kameras möglich**
  durch ein ressourcenschonendes Design der Bibliothek lassen sich mit moderner PC-Hardware mehrere Kameras an einem Rechner betreiben
- **niedrige Kosten**
  Durch kostengünstige Hardware und Verwendung freier Software entstehen minimale Kosten für eine einfache Lösung.

4.2.3.2. Nachteile

- **Markerbeschaffenheit**
  Da die Marker für die Kamera klar einsehbar sein müssen, ist es wichtig, dass die beschrifteten Seiten der Marker immer in Richtung der Kamera orientiert sind und dass die Marker sich nicht verformen. Dies bedeutet, dass ein einfaches “Zeichnen” der Marker auf Textilien negative Auswirkungen auf die Erkennung hätte (z.B. durch Falten in der Kleidung).
- **Markergröße / Behinderung des Trainees**
  Durch die erforderliche Größe und Starrheit der Marker ist eine Behinderung des Trainees bei sportlichen Übungen nicht unwahrscheinlich.
- **hardwareabhängige Genauigkeit**
  Wie in den Vorteilen beschrieben ist eine minimale Lösung mit sehr geringen Kosten verbunden, was jedoch zu Lasten der Genauigkeit des Systems gehen kann. So kann die in Abschnitt 3.4.4.1 beschriebene Verschlusszeit evtl. erst mit einer kostspieligen Kamera erreicht werden.

4.2.4. Versuchsreihe Genauigkeit/Geschwindigkeit

4.2.4.1. Aufbau

Es wurden 3 Marker (IDs 1-3) an einem beweglichen “Arm” (rigides POM-Material) angebracht, mit einer festen Entfernung von 40cm von einem Marker zum Nächsten. Dieser “Arm” ist auf einem Ständer aufgebaut an dem der Marker 0 befestigt ist. Dieser Marker bewegt sich nicht. Ein Bewegungsablauf dauert 6 Sekunden. Die verwendete Kamera ist eine Axis211W Netzwerkkamera, die über Gigabit-LAN angebunden ist. Der “Arm” wurde in 4 verschiedenen Geschwindigkeiten bewegt:

- **Still**: Keine Bewegung
- **Slow**: Eine 190° Bewegung
- **Medium**: Zwei 190° Bewegungen

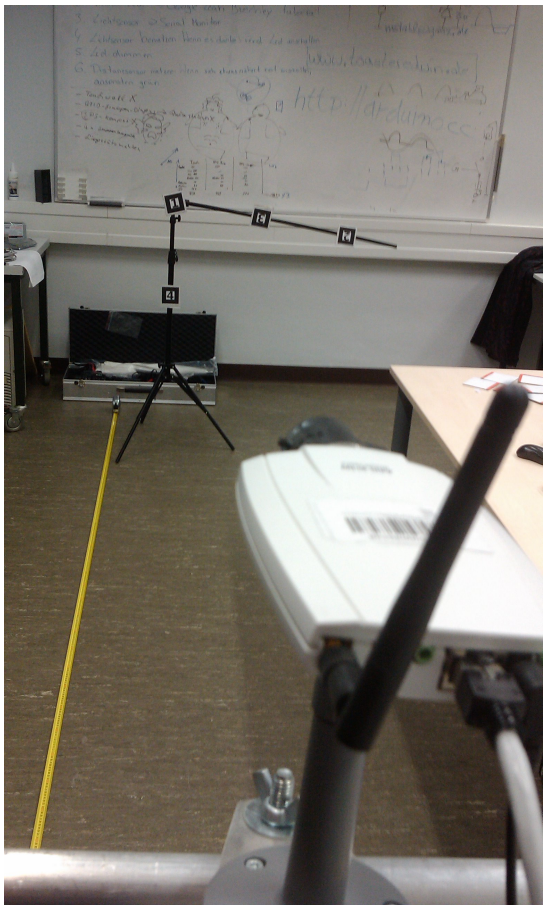

Abbildung 4.5.: Versuchsaufbau

- **Fast**: Drei 190° Bewegungen

Beispiel: Eine “medium” Bewegung wird in 3 Meter Entfernung zur Kamera durchgeführt. Das heißt, dass der Arm innerhalb von 6 Sekunden einmal vor- und einmal zurückbewegt wird.

Diese Versuche werden jeweils für die Entfernungen 2, 2.5, 3, 3.5 und 4 Meter durchgeführt, sodass insgesamt 20 Bewegungsabläufe durchgeführt wurden.

4.2.4.2. Geschwindigkeiten

Ausgehend von der Formel für den Umfang eine Kreises $2\pi r$ und den folgenden Rahmenbedingungen

- Radius für Marker 1: 0.8 Meter
- Radius für Marker 2: 0.4 Meter
- Gradmaß für eine Bewegung: 190°
- Zeitraum für eine Bewegung: 6 Sekunden

lassen sich folgende Geschwindigkeiten näherungsweise berechnen:

- **Slow**:
  Marker 1:
  $$\frac{2 * \pi * 0.8 * (190/360)}{6} = 0.44 m/s$$
  Marker 2:
  $$\frac{2 * \pi * 0.4 * (190/360)}{6} = 0.22 m/s$$
- **Medium**:
  Marker 1:
  $$\frac{2 * \pi * 0.8 * (380/360)}{6} = 0.88 m/s$$
  Marker 2:
  $$\frac{2 * \pi * 0.4 * (380/360)}{6} = 0.44 m/s$$

- **Fast**:
  Marker 1:
  $$\frac{2 * \pi * 0.8 * (470/360)}{6} = 1.09 m/s$$
  Marker 2:
  $$\frac{2 * \pi * 0.4 * (470/360)}{6} = 0.55 m/s$$

Die errechneten Geschwindigkeiten stellen u.a. wegen der irrationalen Zahl π und der Art der Durchführung (Stoppuhr und manuelle Bewegung des “Arms”) nur Näherungswerte dar. Wegen diesen Umständen müssen auch die Versuchsergebnisse nur als Annäherungen verstanden werden.

4.2.4.3. Ergebnisse

Die Versuchsergebnisse variieren je nach Rahmenbedingungen erheblich. Abhängig von Entfernung und Geschwindigkeit des jeweiligen Versuches sind entweder klare Markerpfade (Abb. 4.6) oder nur noch sehr wenige Punkte erkennbar (Abb. 4.8), die dann kaum Rückschlüsse auf die tatsächlichen Pfade zulassen.

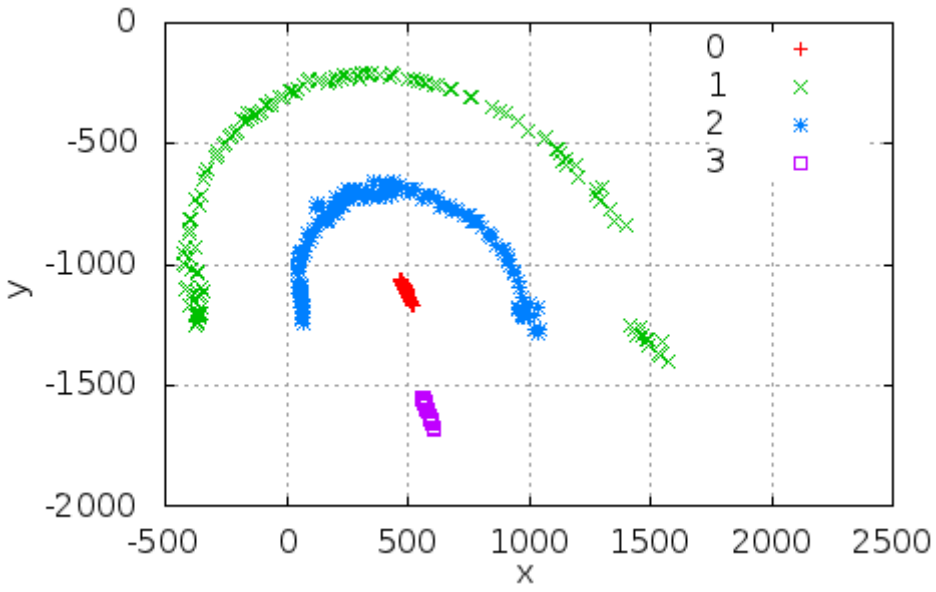

Abbildung 4.6.: Versuch “slow” bei 2 Meter Entfernung zur Kamera

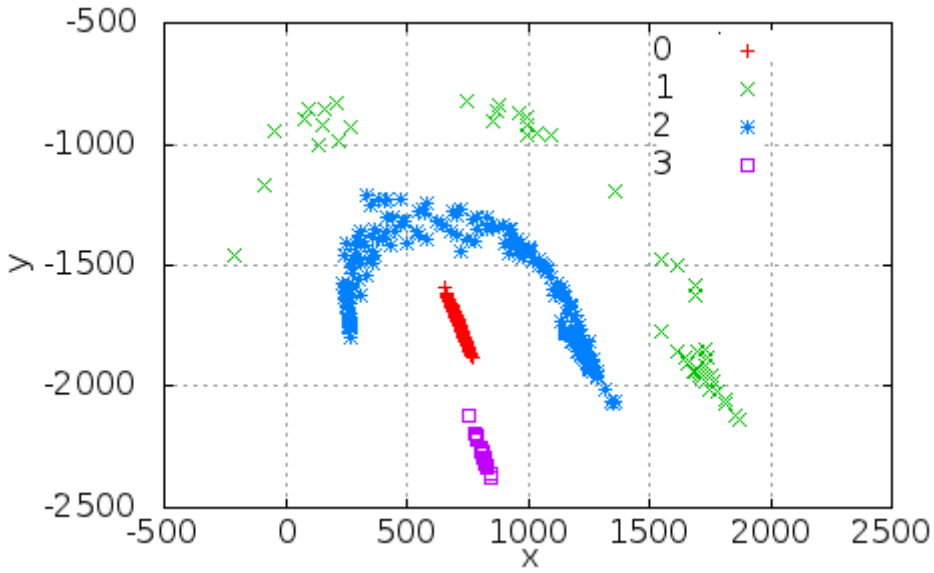

Abbildung 4.7.: Versuch “medium” bei 3 Meter Entfernung zur Kamera

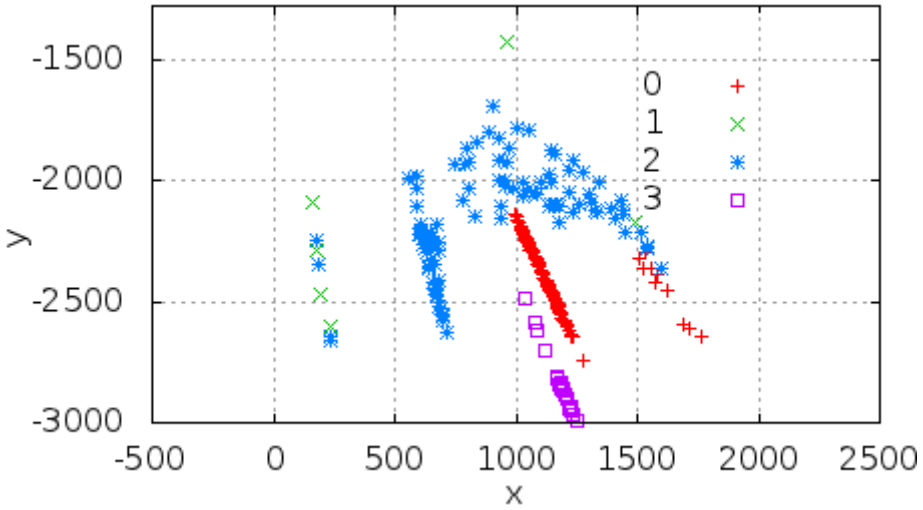

Abbildung 4.8.: Versuch “fast” bei 4 Meter Entfernung zur Kamera

4.2.4.4. Auswertung der Ergebnisse

- Abbildung 4.9: Hier werden die durchschnittlichen Confidence Values (siehe Abschnitt 5.1.2.1) der Marker unter den verschiedenen Versuchsbedingungen verglichen.
- Abbildung 4.10: Dieser Graph zeigt die durchschnittlich erkannten Markerdaten pro Pfad. Pfad bedeutet in diesem Zusammenhang die Markerdaten pro ID, also z.B. alle

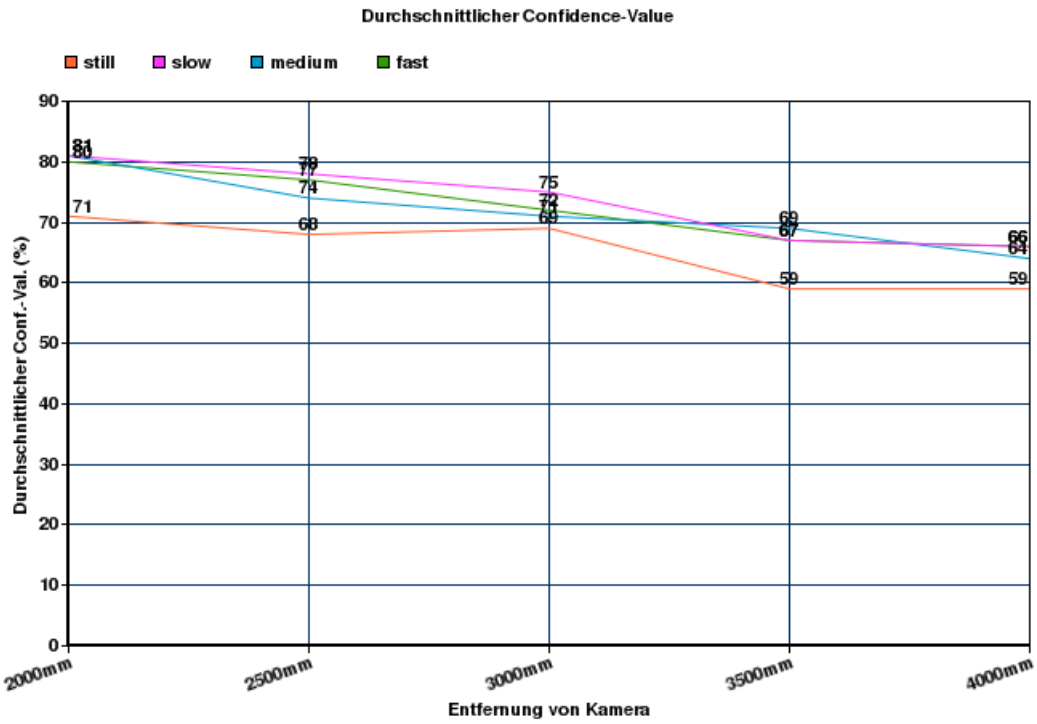

Abbildung 4.9.: Durchschnittliche Confidence Values als Graphen

empfangenen Markerdaten für die Marker-ID 1. Alle empfangenen Markerdaten mit der ID 1 bilden den Pfad des Markers 1.

- Abbildung 4.11: Dieser Graph zeigt erkannte Marker pro Bewegungsablauf, z.B. wurden bei dem Bewegungsablauf “fast” in 3m Entfernung 1133 Marker erkannt.
- Abbildung 4.12: Dieser Graph zeigt (prozentual) die durchschnittliche korrekte Zuordnung der Markerdaten. Da die verwendeten Marker-IDs (0-3) bekannt sind, können andere IDs nicht vorkommen. Werden jedoch andere Marker erkannt, gilt dies als Fehl-Erkennung. So gab es z.B. beim Versuchsaufbau ’still’ in 3m Entfernung 33.2% Fehl-Erkennungen.

4.2.4.5. Fazit

Die visuelle Aufbereitung der Ergebnisse stellt deutlich dar, dass Pfade von Markern unter ungünstigen Rahmenbedingungen (erhöhte Geschwindigkeit des Markers, zunehmende Entfernung zur Kamera) bei weitem nicht komplett sind (Abb. 4.8). Das Ziel, natürliche Bewegungen des menschlichen Körpers komplett bzw. mit einer hohen Präzision aufzuzeichnen,

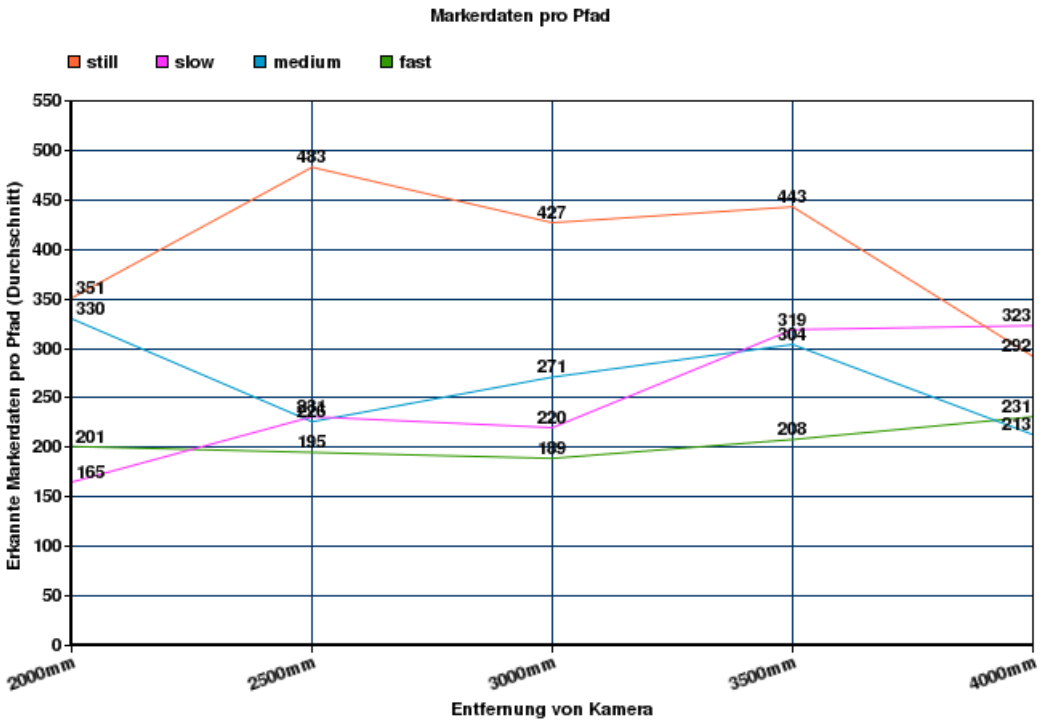

Abbildung 4.10.: Erkannte Markerdaten pro Pfad als Graph

ist mit nur einer ARToolkit-Kamera nicht möglich. Es müssen also entweder mehrere Kameras eingesetzt werden, um eine komplette Aufzeichnung der Bewegung zu erhalten, oder ARToolkit muss mit anderen Motion Capture Verfahren kombiniert werden.

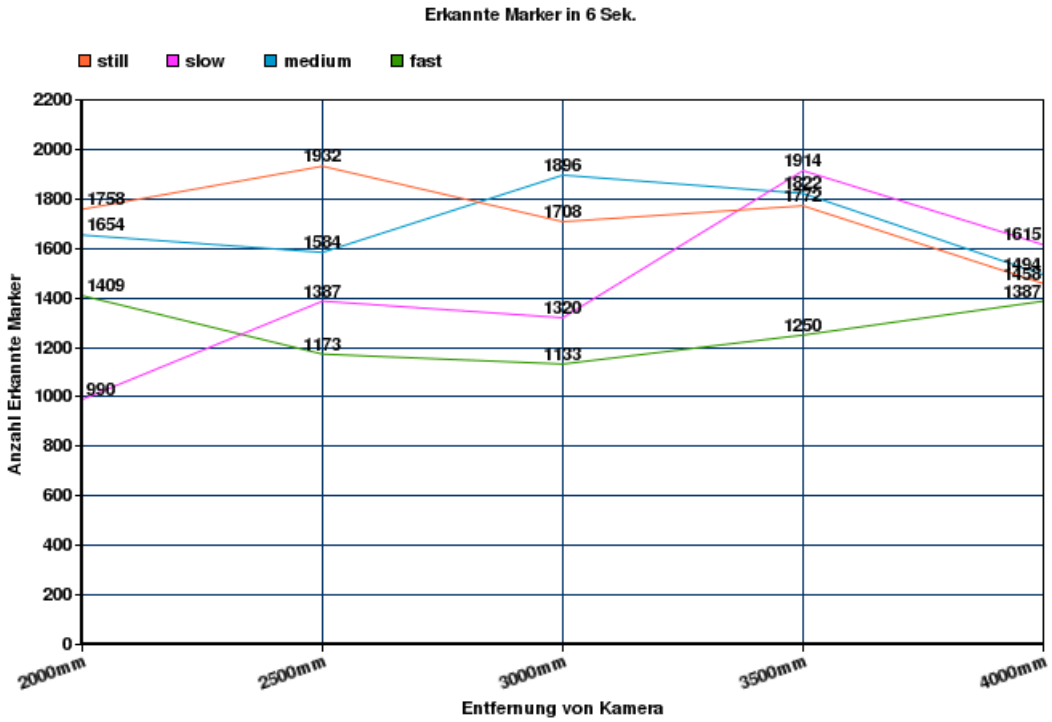

Abbildung 4.11.: Anzahl der Erkannten Marker pro Versuch als Graph

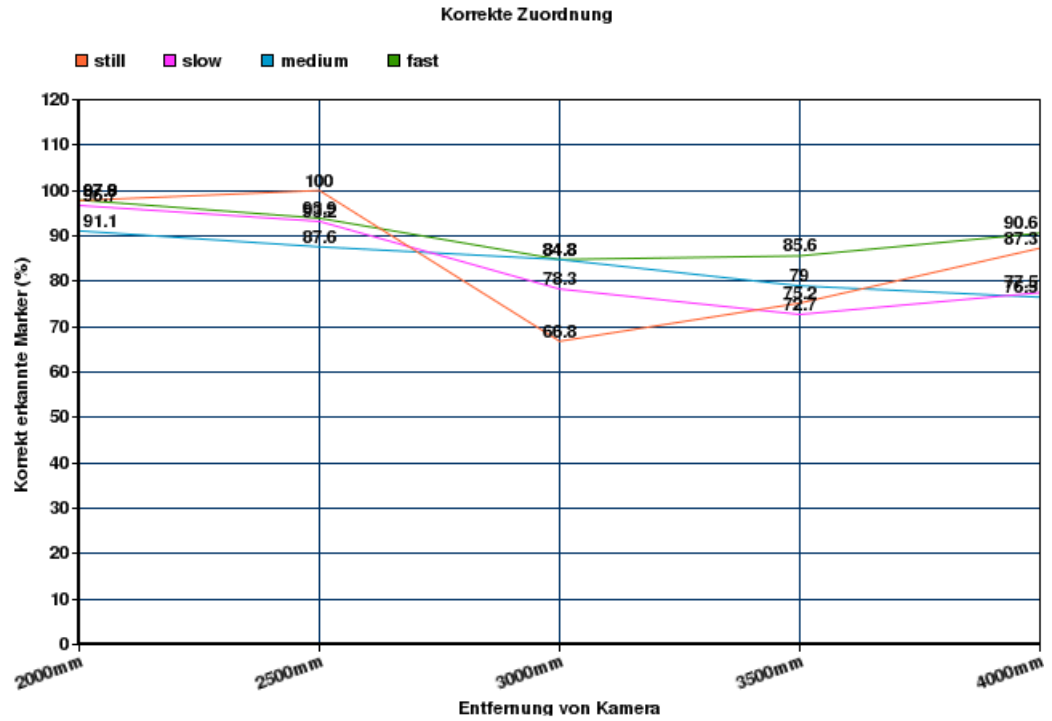

Abbildung 4.12.: Prozentuale Darstellung korrekt erkannter Markerdaten

4.3. Infrarot-Tiefenkamera: Die Microsoft Kinect

Abbildung 4.13.: Foto eines Kinect-Gerätes
Microsoft (2011)

4.3.1. Ursprung und Zweck

Die Kinect ist ein von Microsoft und dem Israelischen Unternehmen PrimeSense entwickeltes Peripheriegerät für die XBox 360 (Microsoft (2010)). Der Zweck des Gerätes ist die Bedienung der XBox 360 ohne Spielcontroller durch Gesten und Sprachbefehle. Durch den Einsatz eines solchen Gerätes soll der Spielspaß gesteigert sowie das im Spiel Erlebte realer wahrgenommen werden.

4.3.2. Funktionsweise

4.3.2.1. Tiefenkamera

Die Kinect arbeitet selbst nicht mit einer TOF-Kamera (PrimeSense Ltd. (2011)). Das Verfahren, welches bei der Kinect zum Einsatz kommt nennt sich **Structured Light Coding** (Albitar u. a. (2007)). Bei diesem Verfahren wird ein Muster aus einer Vielzahl von Infrarotpunkten ausgestrahlt. Der integrierte CMOS-Sensor kennt das verwendete Muster und sammelt die Daten der Infrarotpunkte der Szene. Nun liegen zwei verschiedene Muster vor. Anhand der Verzerrung des Musters im Vergleich zum Original können nun die Tiefeninformationen berechnet werden. Hierzu wird die Technik der Stereoskopie (Devernay u. a. (2002)) verwendet. Als Ergebnis liegt ein Tiefenbild der Szene im Format einer PointCloud vor. Bei dieser Technik wird Licht aus dem infraroten Spektrum verwendet. Der gravierendste Vorteil besteht darin, dass das Umgebungslicht keinen Einfluss auf die Ergebnisse hat. Ebenso ist diese Technik unabhängig von der Ausleuchtung des Raums und funktioniert somit auch in einer komplett unbeleuchteten Szene.

Abbildung 4.14.: Tiefendaten der Kinect
Kyle McDonald (2011)

4.3.2.2. Weitere Hardware

Die Kinect verfügt zudem über eine traditionelle RGB-Kamera, wie sie auch in diversen Webcams oder Mobiltelefonen zu finden ist. Diese hat eine Auflösung von 640*480 Pixel bei 30Hz.

Für die Audioverarbeitung ist ein Array aus 4 Mikrofonen eingebaut. Durch die Verwendung eines Mikrofon-Arrays lässt sich mittels Beamforming (Fischer und Simmer (1996)) die Quelle des Signals lokalisieren. Für eine höhere Zuverlässigkeit oder auch Reinheit des Signals wurden Filter verwendet, welche das Echo entfernen und Rauschen unterdrücken.

4.3.2.3. Software

Als Library für die Kinect wurde OpenNI/Nite verwendet. Bei dieser Library ist das Skeleton-Tracking bereits integriert, zusätzlich können die Tiefendaten als PointCloud und das RGB-Bild abgegriffen werden.

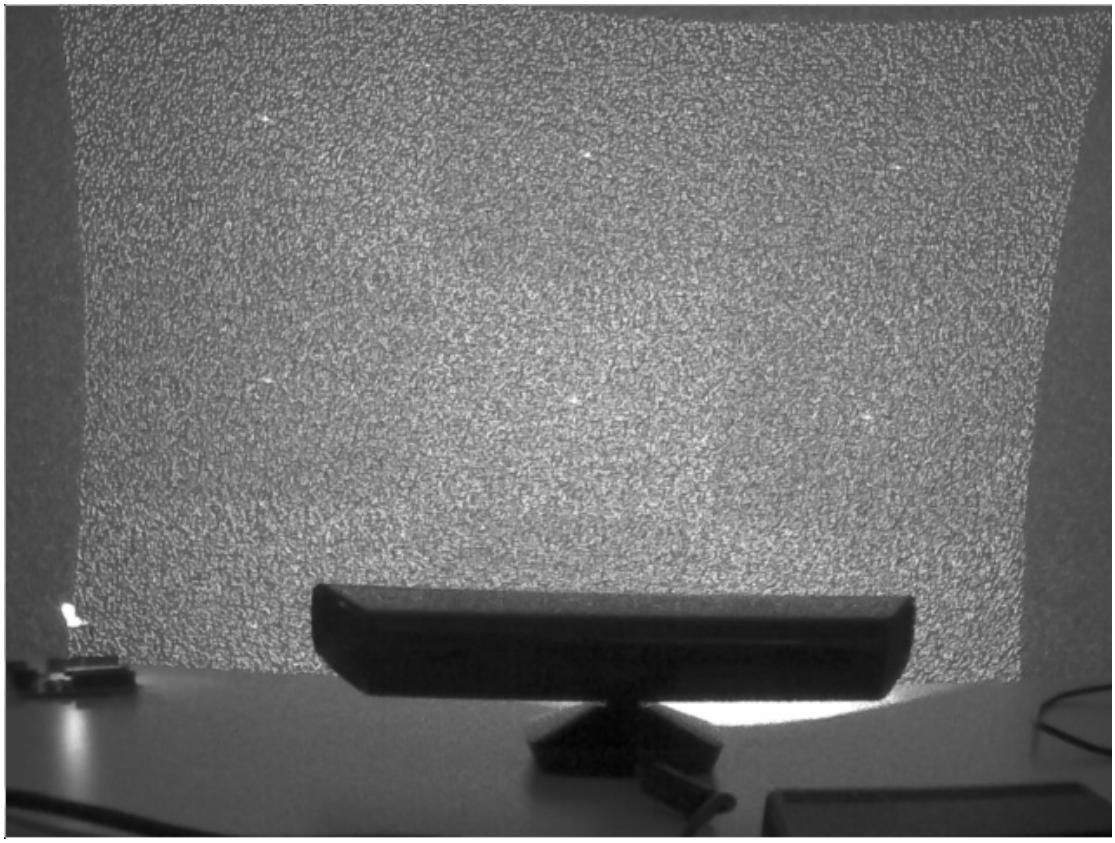

Abbildung 4.15.: Darstellung des Structured Light
Arne Bernin (2010)

- **libfreenect**
  Von Freiwilligen u.a. durch Reverse-Engineering entwickelter Treiber zur Verwendung der Kinect mit PC-Hardware. Greift die Tiefen- und Videodaten des Gerätes ab und gibt sie weiter.
- **Point Cloud Library (PCL)**
  Mit OpenNI ist eine Anbindung der Kinect an das ROS (Robot Operating System) und die enthaltene Point Cloud Library möglich. Die PCL stellt eine einfache Möglichkeit dar, Tiefendaten als PointCloud (“Wolke” aus 3D-Koordinaten von Punkten im Raum) zu komprimieren, zu speichern und weiterzuverarbeiten.
- **OpenNI**
  Ein u.a. von PrimeSense unterstütztes Framework zur Verwendung von Kinects und Kinect-ähnlichen Geräten. Ziel ist es, die kombinierten Daten von Eingabegeräten zur Verfügung zu stellen und so eine Steuerung von Computern zu ermöglichen.
- **NITE**

Middleware zur Verwendung mit OpenNI, vor allem auf die Erkennung von Gesten und zur Errechnung von Skelettmodellen konzipiert.

- **Kinect for Windows SDK**
  Im Juni 2011 veröffentlichte Microsoft ein offizielles Software Development Kit (SDK) für die Kinect, um die Nutzung unter Windows zu ermöglichen. Dieser Ansatz wird in dieser Arbeit nicht untersucht bzw. bewertet, da dazu die Veröffentlichung zu spät erfolgte.

In der Versuchsreihe zur Kinect wurde OpenNI mit NITE benutzt, um die relevanten Punkte des Skelettmodells zu ermitteln.

4.3.2.4. Funktionsweise von NITE

Algorithmen Die NITE-Bibliothek verarbeitet mittels Computer Vision Algorithmen den Stream der unbearbeiteten Tiefendaten.

- **Scene Segmentation**
  In diesem Schritt werden die einzelnen User vom Hintergrund getrennt und bekommen eine ID zugewiesen.
- **Hand point detection and tracking**
  Die Hand des Users wird erkannt und verfolgt.
- **Full Body Tracking**
  Basierend auf dem Ergebnis der Scene Segmentation werden die einzelnen Punkte des Skelettmodells berechnet.

Session Management

- **Not in Session**
  In diesem Zustand existiert keine aktive Session. Das System überprüft die Szene und sucht nach einer Geste zum fokussieren. Sobald eine Geste erkannt und akzeptiert wird, befindet sich das System im Zustand “In Session”.
- **In Session**
  In diesem Zustand wurde eine Geste erfolgreich erkannt und eine Session wurde gestartet, zum Beispiel wird der Punkt einer Hand verfolgt oder das Skelettmodell eines Users wird berechnet.

- **Quick Refocus**
  Dieser Zustand ist optional. Wenn die Verfolgung des Handpunktes nicht mehr funktioniert, muss die laufende Session nicht gleich beendet werden. Innerhalb einer gewissen Zeitperiode kann der Punkt wieder erkannt werden oder die Session kann über eine neue Geste wiedergewonnen werden.

User Segmentation Ziel der User Segmentierung ist es, den User vom Rest der Daten zu trennen und zu verfolgen. Jedem User wird eine eindeutige ID zugewiesen. Als Datenstruktur wird eine Label Map verwendet, bei der jeder Pixel eine ID bekommt, welche der User-ID entspricht. Der Skelettmodellalgorithmus verwendet diese Label Map und generiert daraus das Skelettmodell. Ungenauigkeiten, die bei der Segmentierung entstanden sind, werden zu Ungenauigkeiten im Skelettmodell führen. Folgende Szenarios können zu Problemen bei der Segmentierung führen:

- Wenn der Benutzer sich zu nah an einem Objekt oder der Wand befindet, kann es zu Problemen bei der Segmentierung kommen.
- Wenn zwei Benutzer sich berühren, kann es ebenfalls zu Problemen bei der Segmentierung kommen.
- Durch die Verdeckung eines Benutzer durch 2 andere Benutzer kann es dazu kommen, dass die Benutzer-ID vertauscht wird.
- Sensor bewegen während Segmentierung

Skeleton Tracking Sobald der Benutzer das Kamerablickfeld betritt, wird er als aktiver Benutzer erkannt. Die Kinect wartet nun auf die Kalibrierungspose (Abb. 4.16). Die Kalibrierung

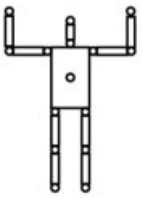

Abbildung 4.16.: Skelettmodell: Kalibrierungspose

dauert ca. 3 Sekunden, währenddessen sollten zumindest die Arme und der Kopf komplett sichtbar sein, die Knie und der Torso sollten gerade sein und nicht verbogen. (Ebenso sollte nicht zu weite Kleidung getragen werden, dies wirkt sich auf die berechneten Extremitäten aus.) Die optimale Distanz beträgt 2,5m.
Das Skelettmodell besteht aus 15 einzelnen Punkten (Abb. 4.17). Jeder Punkt beschreibt eine Position und umfasst die X, Y und Z Koordinaten. Zudem ist ein Wahrscheinlichkeitswert

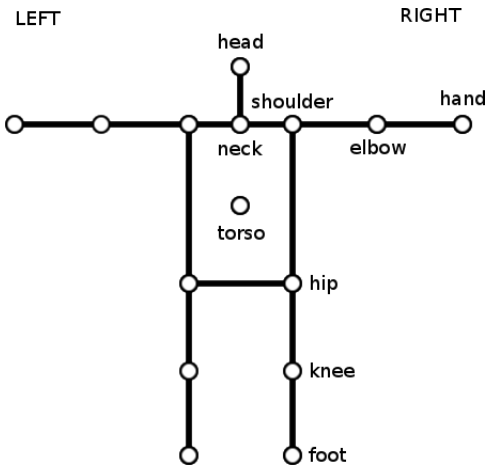

Abbildung 4.17.: Skelettmodell: Verfügbare Punkte der NITE Softwarebibliothek für das Skeleton Tracking

in der Datenstruktur vorhanden. Dieser sagt aus, ob der Punkt erfolgreich erkannt wurde oder ob der Punkt nicht eindeutig bei der Segmentierung identifiziert werden konnte. Bei erfolgreicher Erkennung ist dieser Wert auf 1 gesetzt, ansonsten auf 0.
Zu jedem Punkt ist zusätzlich eine 3x3 Rotationsmatrix (orthonormal) enthalten.
Die linke Hand des Users repräsentiert die rechte Hand des Modells. Das Bild ist somit gespiegelt. Damit das Skelettmodell der realen Welt entspricht, muss das Bild erneut gespiegelt werden.
Es wird empfohlen die Punkte aus der API zu verwenden, da diese durch den Skelettmodellalgorithmus bereits optimiert wurden. Ein weiterer Ansatz zur Berechnung des Skelettmodells kann verfolgt werden durch die Position und Orientierung des Torso in Kombination mit den Orientierungsmatrizen der Schulter (shoulder) und des Ellenbogen (elbow).

4.3.3. Bewertung

4.3.3.1. Vorteile

Die Besonderheit der Kinect liegt in der Tiefenkamera. Diese basiert auf Infrarot-Technologie und funktioniert so auch in absoluter Dunkelheit. Die Funktionsweise ist in Abschnitt 4.3.2.1 näher beschrieben. Mit der Verwendung von OpenNI/NITE als Softwarebibliothek für die Kinect werden dem Programmierer die Daten des 2D RGB-Bildes und des 3D Tiefenbildes zur Verfügung gestellt. Zusätzlich bietet diese Kombination die Grundlagen für die Gestenerkennung und für das Skeleton Tracking. Die abgegriffenen Daten können nun mit weiteren Softwarebibliotheken bearbeitet und gefiltert werden.

4.3.3.2. Bekannte Probleme

Bei der Kalibrierung, wie auch bei der aktiven Verwendung kann es zu etwaigen Fehler kommen (vgl. Primesense Inc. (2010)):

- Wenn ein Arm sich zu nah am Körper befindet, könnte dieser nicht mehr von Körper getrennt werden oder Ungenauigkeiten aufweisen. Weiterhin könnten die Arme vertauscht werden, wenn diese sich zu nah am Körper befinden.
- Die Beine werden ungenau erkannt. Die Genauigkeit erhöht sich, je weiter die Beine voneinander entfernt sind. Schnelle Bewegungen, Tritte oder Hocken können zu Fehlinterpretationen führen.
- Die Erkennung wird beeinträchtigt, wenn der Kopf nicht sichtbar ist.
- Sehr schnelle Bewegungen führen zu Genauigkeitsfehlern.

4.3.4. Versuchsreihe Genauigkeit/Geschwindigkeit

4.3.4.1. Aufbau

Analog zur ARToolkit Versuchsreihe wurde ein Proband mit der Kinect-Tiefenkamera aufgezeichnet. Dies war nötig, da der “Arm” aus dem ARToolkit-Versuch nicht von der Software, die bei der Kinect das Skelettmodell errechnet, als solcher erkannt werden würde.
Die Koordinaten bestimmter Gelenkpunkte am Arm des Probanden (Handgelenk, Ellenbogen und Schulter) wurden aufgezeichnet. Der sonstige Aufbau bzw. Ablauf des Versuchs entspricht genau den Rahmenbedingungen aus dem ARToolkit-Versuchs, es wurden die gleichen Geschwindigkeiten und Abstände zur Kamera eingehalten.

4.3.4.2. Geschwindigkeiten

Ausgehend von der Formel für den Umfang eine Kreises $2\pi r$ und den folgenden Rahmenbedingungen

- Radius für Handgelenk: ca. 0.68 Meter
- Radius für Ellenbogen: ca. 0.38 Meter
- Gradmaß für eine Bewegung: 190°
- Zeitraum für eine Bewegung: 6 Sekunden

lassen sich folgende Geschwindigkeiten näherungsweise berechnen:

- **Slow**:
  Handgelenk:
  $$\frac{2 * \pi * 0.68 * (190/360)}{6} = 0.38 m/s$$
  Ellenbogen:
  $$\frac{2 * \pi * 0.38 * (190/360)}{6} = 0.21 m/s$$
- **Medium**:
  Handgelenk:
  $$\frac{2 * \pi * 0.68 * (380/360)}{6} = 0.75 m/s$$
  Ellenbogen:
  $$\frac{2 * \pi * 0.38 * (380/360)}{6} = 0.42 m/s$$
- **Fast**:
  Handgelenk:
  $$\frac{2 * \pi * 0.68 * (470/360)}{6} = 0.93 m/s$$
  Ellenbogen:
  $$\frac{2 * \pi * 0.38 * (470/360)}{6} = 0.52 m/s$$

Die errechneten Geschwindigkeiten stellen u.a. wegen der irrationalen Zahl π und der Art der Durchführung (Stoppuhr und manuelle Bewegung des Arms) nur Näherungswerte dar. Wegen diesen Umständen müssen auch die Versuchsergebnisse nur als Annäherungen verstanden werden.

4.3.4.3. Ergebnisse

Wie bei dem äquivalenten Versuch zum ARToolkit variieren die Ergebnisse teils erheblich voneinander. Auch hier gibt es bei günstigen Versuchsbedingungen (z.B. niedrige Geschwindigkeit) klar erkennbare Markerpfade (Abb. 4.18), oder aber ungenaue Messungen (Abb. 4.19).

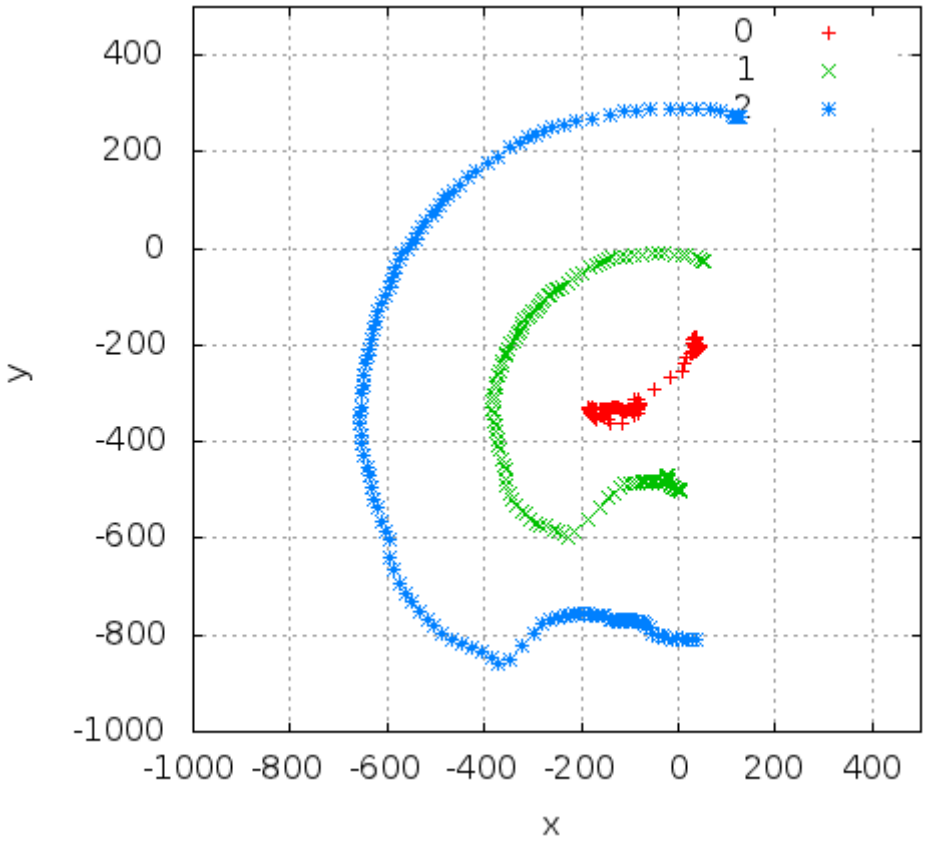

Abbildung 4.18.: Versuch “slow” bei 2 Meter Entfernung zur Kamera

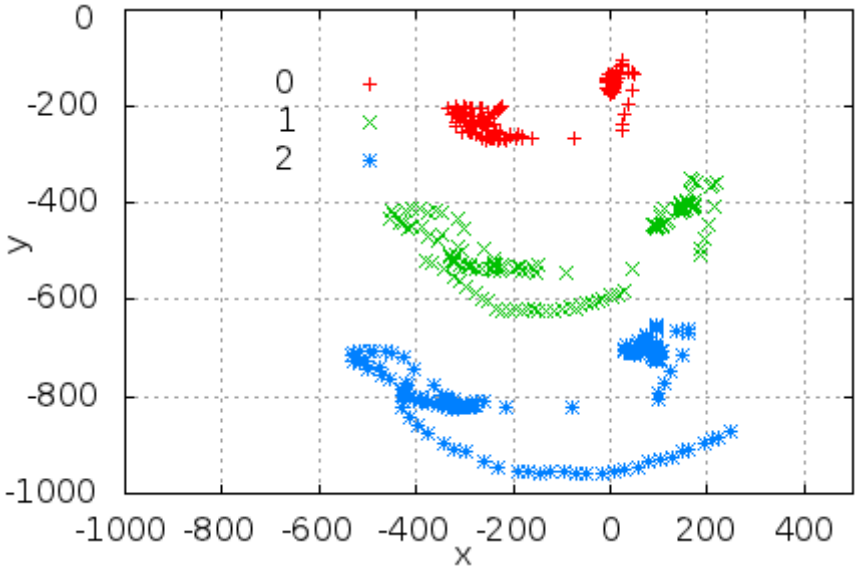

Abbildung 4.19.: Versuch “medium” bei 3 Meter Entfernung zur Kamera

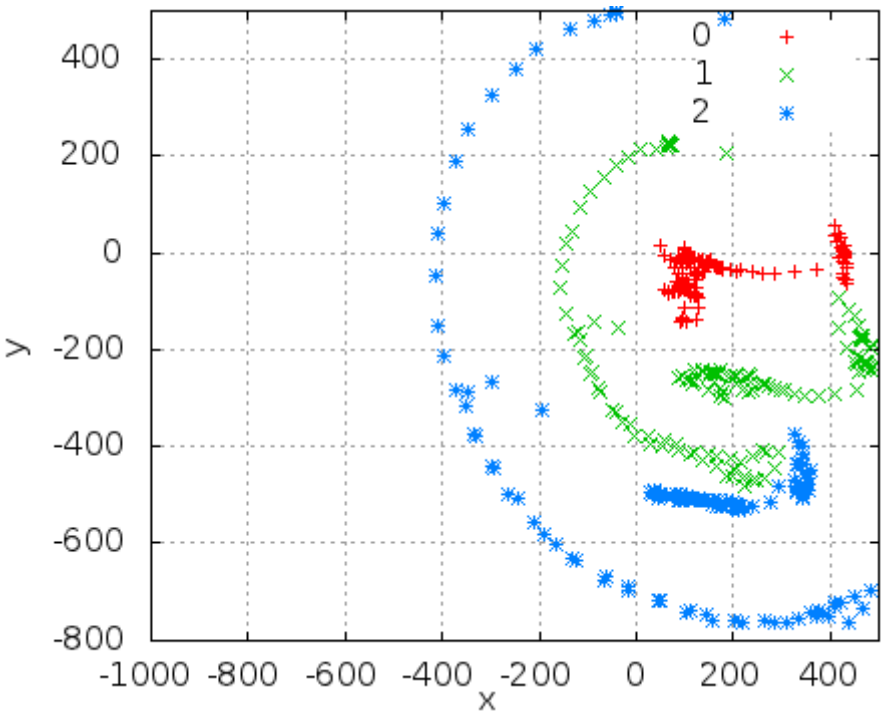

Abbildung 4.20.: Versuch “fast” bei 4 Meter Entfernung zur Kamera

4.3.4.4. Auswertung der Ergebnisse

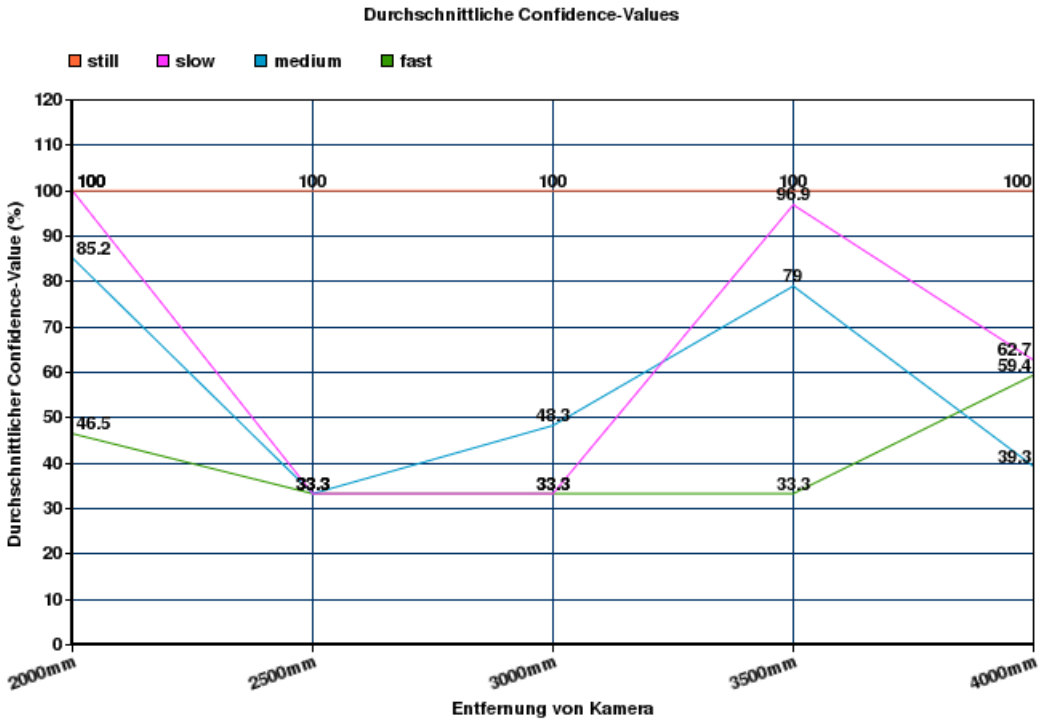

Abbildung 4.21.: Durchschnittliche Confidence Values als Graphen

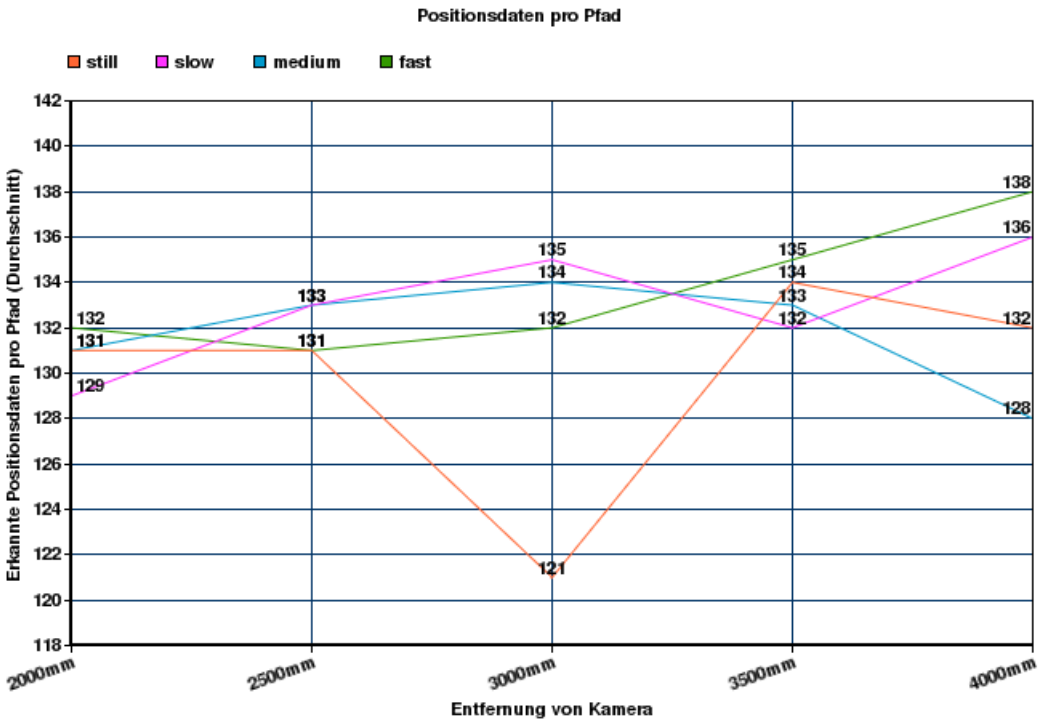

Abbildung 4.22.: Erkannte Markerdaten pro Pfad als Graph

4.4. Vergleich und Bewertung der Ansätze

Mit Hilfe der durchgeführten Versuche lassen sich zur Präzision und Nutzbarkeit der zwei Verfahren bereits einige Schlüsse ziehen.

4.4.1. Präzision

Während das ARToolkit bei der Verwendung als Motion-Capture Komponente einigermaßen “zweckentfremdet” wird, ist der ursprüngliche Verwendungszweck der Kinect eine Aufgabe, die dem Motion-Capture nicht unähnlich ist. Einige Unterschiede bei der grundsätzlichen Implementierung der Systeme sorgen bereits für eine ungleiche Ausgangssituation.

Bei dem ARToolkit werden beispielsweise einzelne Marker erkannt, aus deren Position Rückschlüsse auf die Körperhaltung des Trainees gezogen werden. Die Kinect verfährt in genau entgegengesetzter Weise: durch die Auswertung der gesamten Tiefendaten des Raumes wird ein menschlicher Körper gesucht (Kalibrierung). Aus den nachfolgenden Daten werden dann Rückschlüsse auf die Positionsdaten der neuralgischen Punkte gezogen, die das Skelettmodell bilden sollen. So kann beispielsweise ein Objekt innerhalb der Versuchsumgebung, welches aussieht wie ein Marker, für einen Fehler des ARToolkits sorgen (diese

Abbildung 4.23.: Fehlerkennung einer Markers bei dem ARToolkit

Situation trat bei ersten Tests des Öfteren auf, siehe Abb. 4.23, dort wird ein eingerahmter Zettel auf dem Flur fälschlicherweise als Marker erkannt).

4.4.2. Geschwindigkeit

Die Erkennungsrate von Markern beim ARToolkit ist nicht konstant. Das bedeutet, dass zwischen der Bildrate, der angeschlossenen Kamera und der Erkennungsrate der Softwarebibliothek ARToolkit meist unterschiede bestehen. So werden z.B. beim ARToolkit Bilder komplett verarbeitet, in denen jedoch aufgrund von Bewegungsunschärfe keine Marker erkannt werden. So kann eine Kamera mit 30 Bildern pro Sekunde Daten liefern, die Anzahl der erkannten Marker kann jedoch erheblich abweichen.

4.4.3. Kosten

4.4.3.1. ARToolkit

Bei ARToolkit werden lediglich ein handelsüblicher PC, ein paar günstig herzustellende Marker, Kameras und die nötige Software benötigt. Je nach Anspruch an die Präzision des Systems können diese Kosten jedoch variieren: Mit Betrachtung des Trainees aus mehreren Blickwinkeln durch mehrere Kameras steigen die Kosten. Auch ist eine Nutzung von hochwertigen (und somit teureren) Kameras mit einer niedrigen Belichtungszeit und einer hohen Bildrate und Auflösung empfehlenswert.

4.4.3.2. Kinect

Im ersten Augenblick erscheint der Ansatz mit der Kinect-Kamera teurer, da ein Sensor allein bereits ca. 100 € kostet. Wenn man bedenkt, dass bei dem optischen Ansatz mit dem ARToolkit wahrscheinlich mehrere hochwertige Kameras benötigt werden, verkehrt sich dieser Eindruck in das Gegenteil, da eine Hochgeschwindigkeitskamera meist erheblich mehr als 100 € kostet.

4.5. Fazit

Die oben genannten Designunterschiede der beiden Ansätze wirken sich zuungunsten des ARToolkits aus, wobei fairerweise gesagt werden muss, dass die Software nie zu Motion-Capture Zwecken gedacht war.
Für den Ansatz mit Infrarot-Tiefendaten spricht vor Allem das Preis-Leistungs-Verhältnis mit Blick auf Motion-Capture. Dazu muss gesagt werden, dass allein durch die Nutzung als Videospiel-Zubehör und der damit verbundenen Massenherstellung eine Infrarot-Tiefenkamera zu einem relativ niedrigen Preis verfügbar ist.
Der Infrarot-Tiefenbild Ansatz wird somit als erfolgversprechender bewertet, wenn eine komplette, möglichst präzise Aufzeichnung menschlicher Bewegungen erreicht werden soll.

4.5.1. Simultane Nutzung beider Ansätze

Denkbar wäre es auch, beide Ansätze ergänzend zueinander zu benutzen. Mit dem Einsatz von hochauflösenden Kameras mit dem ARToolkit könnte eine zusätzliche Stufe der Fehlerkorrektur hinzukommen, falls beim Infrarot-Ansatz Fehl-Erkennungen auftreten sollten.
So könnten einzelne Gliedmaßen des Trainees mit passiven Markern ausgestattet werden,

um für zusätzliche Genauigkeit zu sorgen. Ein anderes Szenario, in welchem beide Technologien ergänzend zueinander eingesetzt werden könnten, wird in Abschnitt 6.3.1 beschrieben.

5. Entwicklung eines verteilten Motion Capture Systems

5.1. Komponenten

5.1.1. Übersicht

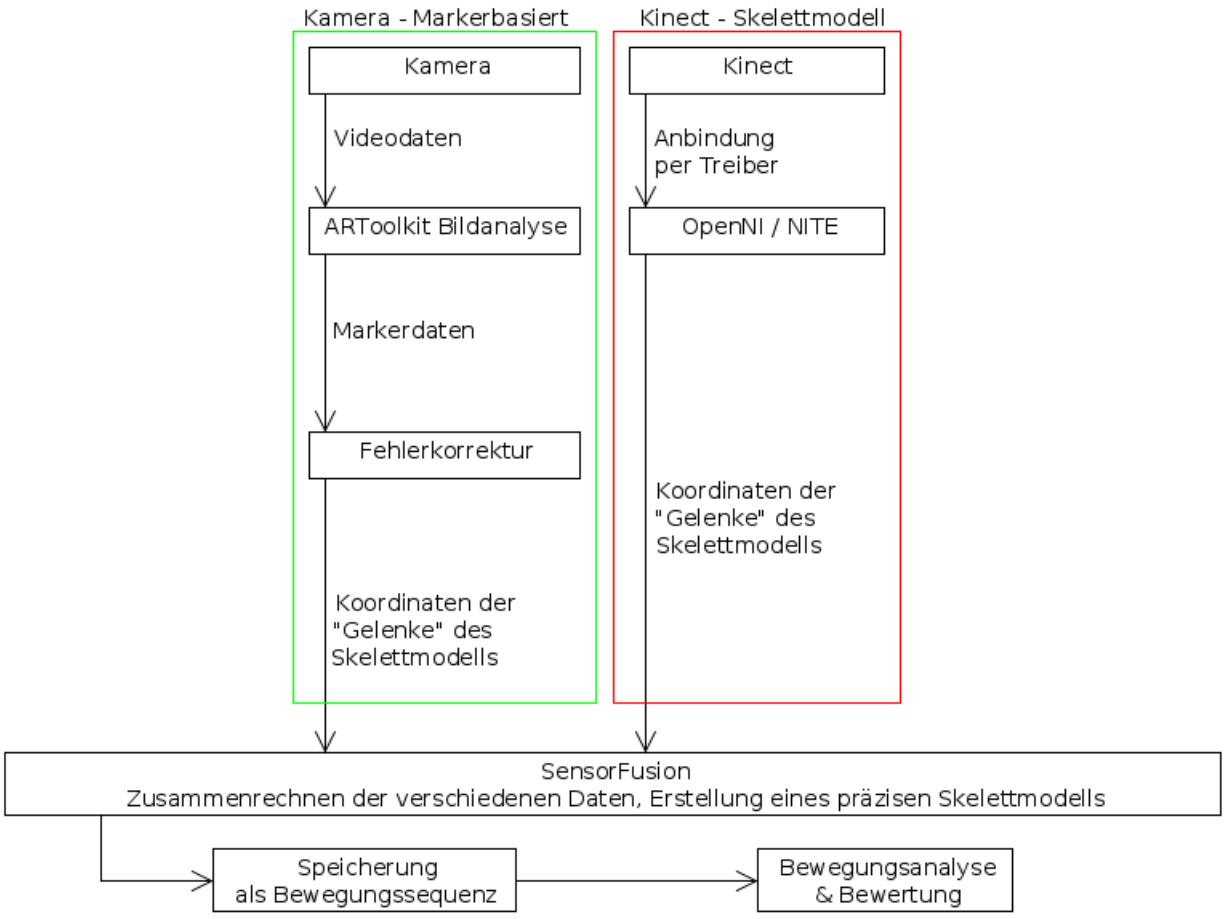

Abbildung 5.1.: Abstrakte Übersicht: Schrittweise Verarbeitung der Daten

Die Abbildung 5.1 beschreibt auf abstrakte Weise die Schrittweise Verarbeitung der Sensordaten. Die im Diagramm vertikal gruppierten Verarbeitungsschritte beschreiben jeweils eine Art der Datengewinnung. So ist es denkbar, nur eines der Verfahren mehrfach einzusetzen, z.B. die Gewinnung von Daten mittels der Kinect (Skelettmodell).
Auf jeden Fall ist das Mischverhältnis der Sensoren frei bestimmbar, um das System den Randbedingungen und Anforderungen anzupassen. In einer schlecht beleuchteten Umgebung wäre z.B. die Kinect der konventionellen Kamera vorzuziehen, da die Gewinnung von Infrarot-Daten unabhängig von der Lichtverhältnissen ist. Durch ein solches “Fine Tuning” lässt sich die Präzision der Messergebnisse erhöhen.

5.1.2. Kamera - Markerbasiert

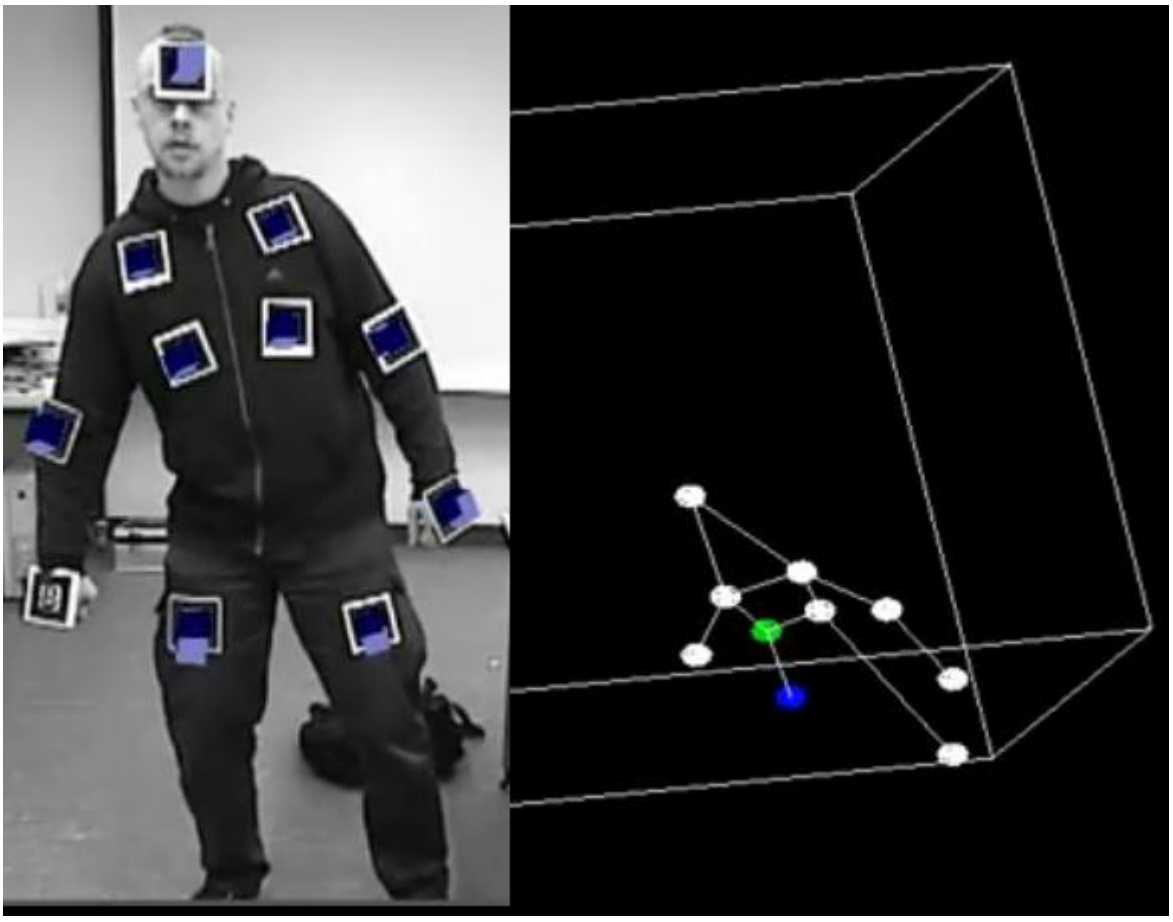

Abbildung 5.2.: Primitives Skelettmodell realisiert mit ARToolkit

Mit Hilfe der ARToolkit Bibliothek werden aus dem Videobild die Positionsdaten der Marker im dreidimensionalen Raum berechnet.

Die Bildrate richtet sich nach der maximal verfügbaren Bildrate der eingesetzten Kamera. Pro aufgenommenem Bild wird der Algorithmus von ARToolkit verwendet, um Marker in dem Bild zu ermitteln und zuzuordnen.
Die Aufzeichnung der Markerdaten vom ARToolkit in einer Datenbank oder in einem Dateisystem stellt kein Problem wie bei Abschnitt 5.3.2.2 dar, da durch die Ermittlung der Markerpositionen eine Auswahl zu einem Skelettmodell ähnlich zu Abschnitt 5.1.3 bereits getroffen ist.

5.1.2.1. Datentypen

Das ARToolkit ermittelt pro Einzelbild die Markerdaten. Für jeden erkannten Marker wird eine Datenstruktur mit folgenden Eigenschaften erstellt und weitergegeben:

- **Confidence Value**: Der Confidence Value ist ein Wert zwischen 0 und 100, der die prozentuale Wahrscheinlichkeit, dass ein erkannter Marker tatsächlich ein Marker ist, darstellt. Das heißt bei einem Confidence Value von 50 ist die Wahrscheinlichkeit, dass sich an der angegebenen Position tatsächlich der gesuchte Marker befindet, 50%. Wenn mehrere Marker mit der gleichen ID ermittelt werden, obwohl nur ein Marker solchen Typs genutzt wird, kann anhand des Confidence Value bestimmt werden, welche Datenstruktur mit der höchsten Wahrscheinlichkeit den tatsächlich vorhandenen Marker beschreibt.
- **ID**: Beim Start des ARToolkit-Clients werden sog. Marker-Patterns geladen, die das Aussehen des Markers beschreiben. Jedem dieser Patterns wird eine Identifikationsnummer zugeordnet. Dieser Wert beinhaltet die Identifikationsnummer, so dass die Markertypen voneinander unterschieden werden können.
- **Position**: Die Position des Markers im dreidimensionalem Raum. X, Y und Z Koordinaten als Gleitkommazahlen (double). Die Grenzwerte dieser Zahlen hängen von der verwendeten Kamera ab.
- **Direction**: Der Rotationswert des Markers. Vier mögliche Werte (0,1,2,3), die beschreiben, ob der Marker z. B. “auf dem Kopf steht”.

5.1.3. Kinect - Skelettmodell

Diverse Bibliotheken (OpenNI und NITE) unterstützen die direkte Extraktion der Skelettmodelle aus der Kinect. Konkret werden die 3D Koordinaten bestimmter neuralgischer Punkte (insbesondere Gelenke) weitergereicht und mit Linien verbunden. Hiermit wird eine drastische Reduktion der Datenmenge erreicht, die in Abschnitt 5.3.2.2 beschrieben wird. Anstatt

Abbildung 5.3.: Skelettmodell der Kinect

die dreidimensionalen Koordinaten von 307200 Punkten zu übermitteln, werden einige wenige Punkte (momentan 15) gespeichert, aus denen sich Rückschlüsse auf die Körperhaltung ziehen lassen:

- Kopf
- Brust
- Schulter (links / rechts)
- Ellenbogen (links / rechts)
- Hand (links / rechts)
- Bauch
- Hüfte (links / rechts)
- Knie (links / rechts)
- Fuß/ Unterschenkel (links / rechts)

5.1.3.1. Datentypen

Die Aufzeichnung bzw. Weitergabe dieser Informationen findet ebenfalls als Auflistung von Punkten im dreidimensionalen Raum statt. Jeder Punkt beschreibt eine Position und um-

fasst die X, Y und Z Koordinaten. Zu jedem Punkt ist zusätzlich eine 3x3 Rotationsmatrix (orthonormal) enthalten.

5.1.3.2. Verlässlichkeit

Bei der Verwendung des Skelettmodells kann es auch zu Problemen kommen. Probleme entstehen hauptsächlich bei der Segmentierung. Mit der Segmentierung werden verschiedene Bildbereiche voneinander getrennt, so dass diese möglichst eindeutig als Objekt oder Benutzer erkannt werden können. In der Dokumentation zu der NITE-Bibliothek sind einige bekannte Probleme aufgelistet (vgl. Primesense Inc. (2010)).

- Wenn der Benutzer sich zu nah an einem Objekt oder der Wand befindet, kann es zu Problemen bei der Segmentierung kommen.
- Wenn zwei Benutzer sich berühren, kann es ebenfalls zu Problemen bei der Segmentierung kommen.
- Durch die Verdeckung eines Benutzer durch 2 andere Benutzer kann es dazu kommen, dass die Benutzer-ID vertauscht wird.
- Sensor bewegen während Segmentierung

5.1.4. SensorFusion

In der Komponente SensorFusion sollen die Daten der (evtl. verschiedenen) Messverfahren zusammengerechnet und ggf. durch Interpolation korrigiert werden. Abgesehen von der Skalierung evtl. unterschiedlicher Größenordnungen je nach Messverfahren ist auch eine Fehlerkorrektur nötig, um ein möglichst präzises Ergebnis zu erhalten.
Ein Beispiel für so ein Verfahren ist der Kalman-Filter.

5.1.4.1. Kalman-Filter

Der Kalman-Filter ist ein 1960 von Rudolf E. Kalman beschriebenes rekursives Verfahren zur Berechnung von Zuständen in einem System, von dem immer nur ungenaue bzw. verrauschte Messwerte bekannt sind (Kalman (1960)). Das Verfahren geht von der Annahme aus, dass gewonnene Messwerte eine inhärente Ungenauigkeit besitzen, die korrigiert werden muss. Folgende Variablen werden als gegeben vorausgesetzt bzw. müssen im ersten Schritt “händisch” geschätzt werden:

- M: Messwert

- V_G: Varianz des zu messenden Wertes (als Anfangswert), absoluter Wert
- V_M: Varianz des Messergebnisses, absoluter Wert
- G: geschätzter Wert (als Anfangswert)

Folgende Variablen werden durch das Verfahren errechnet:

- W: Gewichtung (zwischen 0 und 1)
- E: errechnete Schätzung
- V_E: Varianz der errechneten Schätzung

Die Ergebnisse werden im ersten Schritt der Rekursion auf folgende Weise errechnet:

$$W = \frac{V_G}{V_G + V_M}$$

$$E = G + W * (M - G)$$

$$V_E = \frac{V_G * V_M}{V_G + V_M}$$

Nachdem der erste Schritt durchgeführt wurde, wird mit den folgenden rekursiven Formeln die Berechnung fortgesetzt:

$$E_i = E_{i-1} + W * (M - E_{i-1})$$

$$V_{Ei} = \frac{V_{Ei-1} * V_M}{V_{Ei-1} + V_M}$$

wobei E_{i-1} die errechnete Schätzung des vorherigen Rekursionsdurchlaufs ist.

5.1.5. Speicherung als Bewegungssequenz

Abgeschlossene Bewegungsabläufe können leicht rekonstruiert und analysiert werden. Die einzelnen Gelenkkoordinaten werden für jeden einzelnen Frame während eines Testlaufs in eine Datenbank geschrieben (Abb. 5.4). Auf diese Weise kann jeder Bewegungsablauf zur weiteren Analyse detailliert betrachtet werden. Die Tabelle “user” repräsentiert den Benutzer, optional zur Bernutzer-ID kann ein Name hinterlegt werden. Zu Testzwecken wurde ein Standardbenutzer mit dem Namen “Manfred Mustermann” in die Datenbank eingetragen. Die Bezeichnung zu jedem Marker oder Gelenkknoten mit einer eindeutigen ID ist in der Tabelle “joints” hinterlegt. In der Tabelle “run” wird ein einzelner Testlauf festgehalten. Ein Testlauf ist mit einem Benutzer verknüpft und kann zusätzlich durch ein Label benannt werden. Die Startzeit zu jedem Testlauf wird als Unix-Timestamp ebenso festgehalten, wie der Zeitpunkt

beim Beenden.
Die relevanten Gelenkkoordinaten werden in die Tabelle “points” geschrieben:

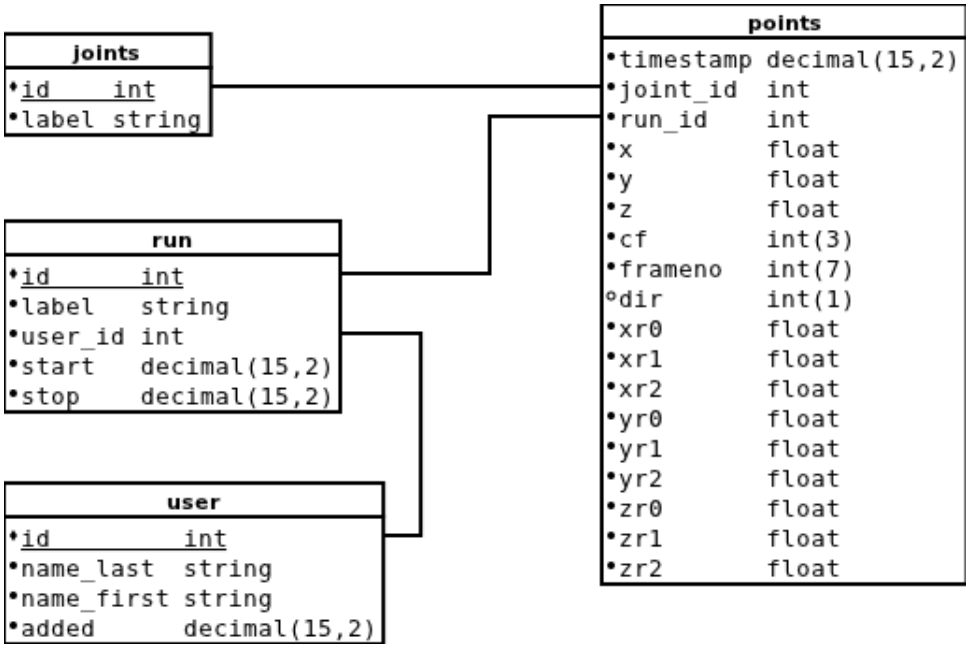

Abbildung 5.4.: Realisierung der Datenbank

- **Timestamp** Der Timestamp wurde verwendet, um die zeitliche Zuordnung exakt zu bestimmen. Mit einer Genauigkeit von $\frac{1}{100} ms$ kann die Abarbeitungsreihenfolge genau eingehalten werden. Vorausschauend wird der Timestamp auch festgehalten, damit bei einer Weiterentwicklung dieses Systems mit mehreren Kameras die Zeitliche Differenz bestimmt werden kann, sodass die richtigen Frames zusammengefügt werden. Wenn die Aufnahmen an 2 verschiedenen Clients gemacht werden, muss darauf geachtet werden, dass die Zeiten möglichst synchronisiert sind, damit keine Ungenauigkeiten durch leichte zeitliche Differenzen entstehen.
- **Joint ID** Die Joint ID bestimmt den erkannten Marker bzw. Gelenkpunkt.
- **Koordinaten** Für X, Y und Z ist je eine Spalte in der Datenbank vorhanden.
- **Confidence Value** Der Confidence Value entspricht der prozentualen Wahrscheinlichkeit, dass es sich wirklich um den erkannten Marker handelt. Der Wert liegt zwischen 0 und 100.
- **Framenummer** Die fortlaufende Framenummer wird zur weiteren Identifizierung der Reihenfolge verwendet.

- **Direction** Die Ausrichtung gibt bei markerbasierten Systemen die Ausrichtung des Markers an.
- **Rotationsmatrix** Zusätzlich zu jedem Gelenkpunkt wird die Orientierungsmatrix gespeichert. Diese liegt in einer 3x3 Matrix vor und wird in der Datenbank in insgesamt 9 Feldern zu jedem Datensatz gespeichert.

5.1.6. Bewegungsanalyse & Bewertung

Mit einer Physik-Engine wie Bullet Physics (vgl. Erwin Coumans (2011)) lassen sich Simulationen realisieren. Das Skelettmodell der Kinect kann zu der 3D-Engine portiert werden. Nun können physikalische Eigenschaften für die losen Punkte des Modells bestimmt werden. Nachdem einzelne Extremitäten, wie Arme und Beine eindeutig zusammengesetzt sind, ist hier eher die Relation von Arm zu Torso und Bein zu Torso relevant. Hier wird nun bestimmt, wie groß der Winkel zueinander maximal sein kann. Hier muss nun entschieden werden, ob die vorliegenden Winkel innerhalb realistischer physikalischer Grenzen liegen.

5.2. Konzeption als verteiltes System

Um das Gesamtsystem so skalierbar wie möglich zu halten, wurde der Prozess Bewegungsabläufe aufzuzeichnen und auszuwerten in kleinere Aufgaben geteilt. Skalierbarkeit wird somit ermöglicht, da jeder Teilprozess auf einem eigenen System ausgeführt werden kann. Da die Kommunikation der Komponenten untereinander per TCP/IP geschieht, sind unter dem Gesichtspunkt der Verteilung mehrere Szenarien denkbar, um sich den Bedürfnissen der Situation anzupassen. Zu Testzwecken oder bei Szenarien, in denen wenige Messgeräte (Kinects oder Webcams) genutzt werden, ist ein Aufbau wie in Abb. 5.5 denkbar. Der Gegensatz hierzu ist in Abb. 5.6 dargestellt. In diesem Szenario ist das System auf mehrere Computer verteilt, womit die Belastung für die einzelne CPU herabgesetzt wird. Auch die Minimierung von Latenzzeiten ist somit in gewissem Maße möglich.

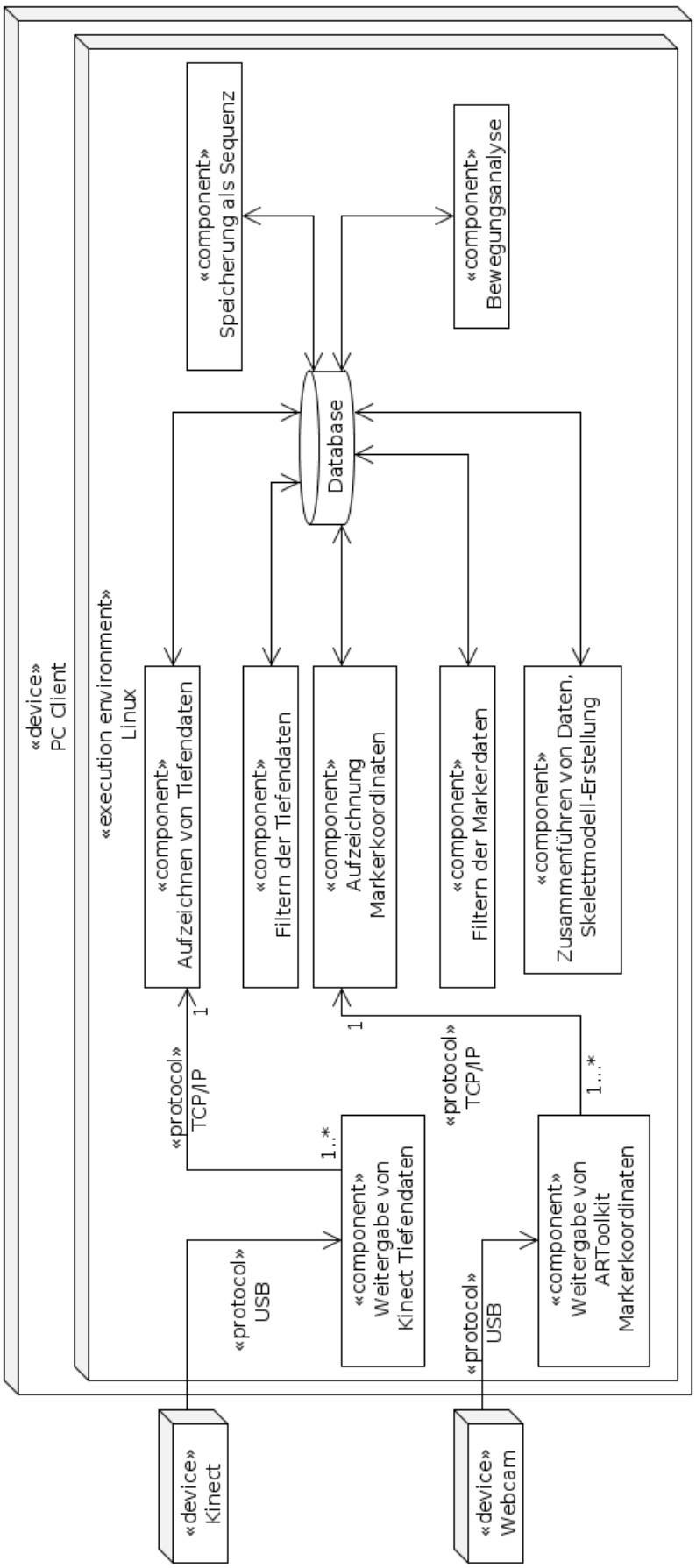

Abbildung 5.5.: UML Verteilungsdiagramm: Komplettsystem auf einem Rechner

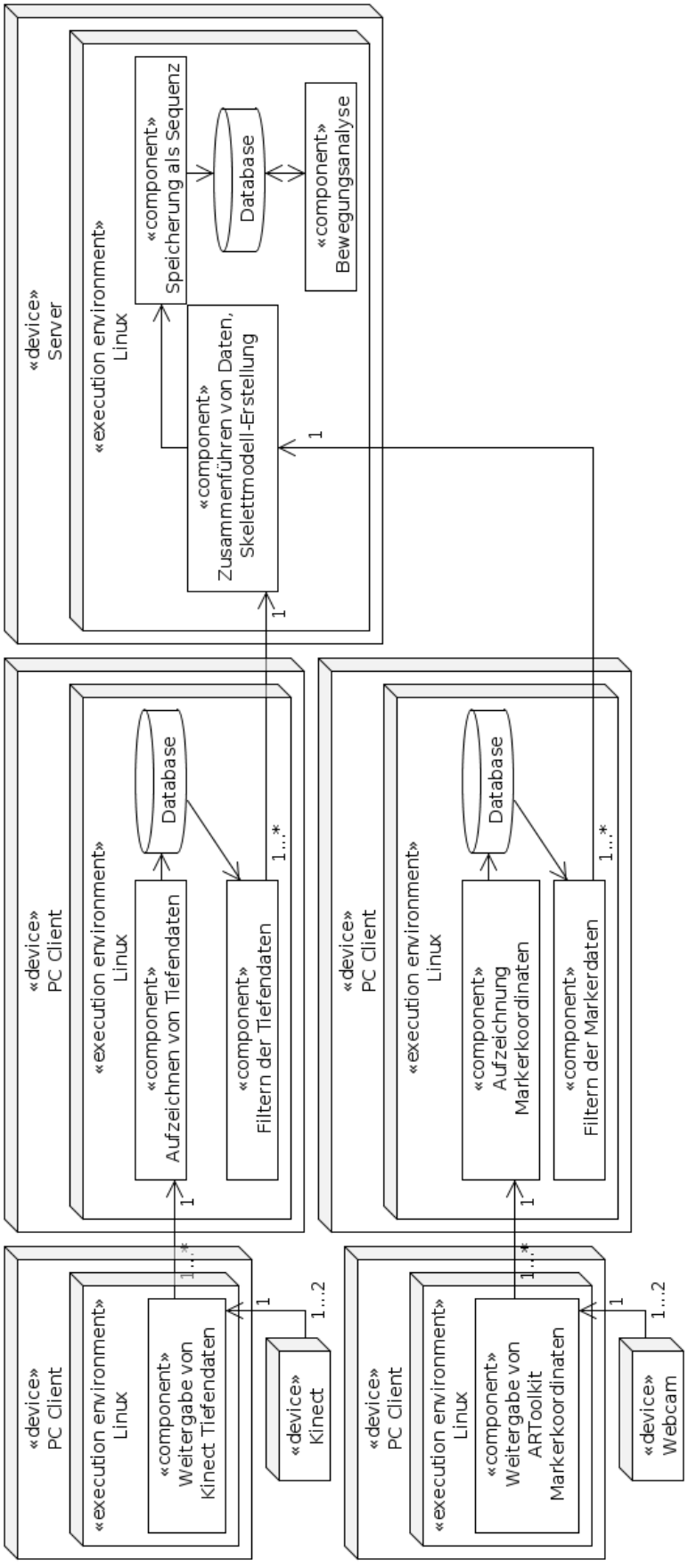

Abbildung 5.6.: UML Verteilungsdiagramm: Realisierung als Verteiltes System auf mehrerern Rechnern

5.3. Ausblick: Weitere Aufzeichnungsverfahren

In diesem Abschnitt wird beschrieben, wie weitere Variationen der untersuchten Ansätze zukünftig in das vorgestellte System eingegliedert werden können.

5.3.1. Erweitertes Design

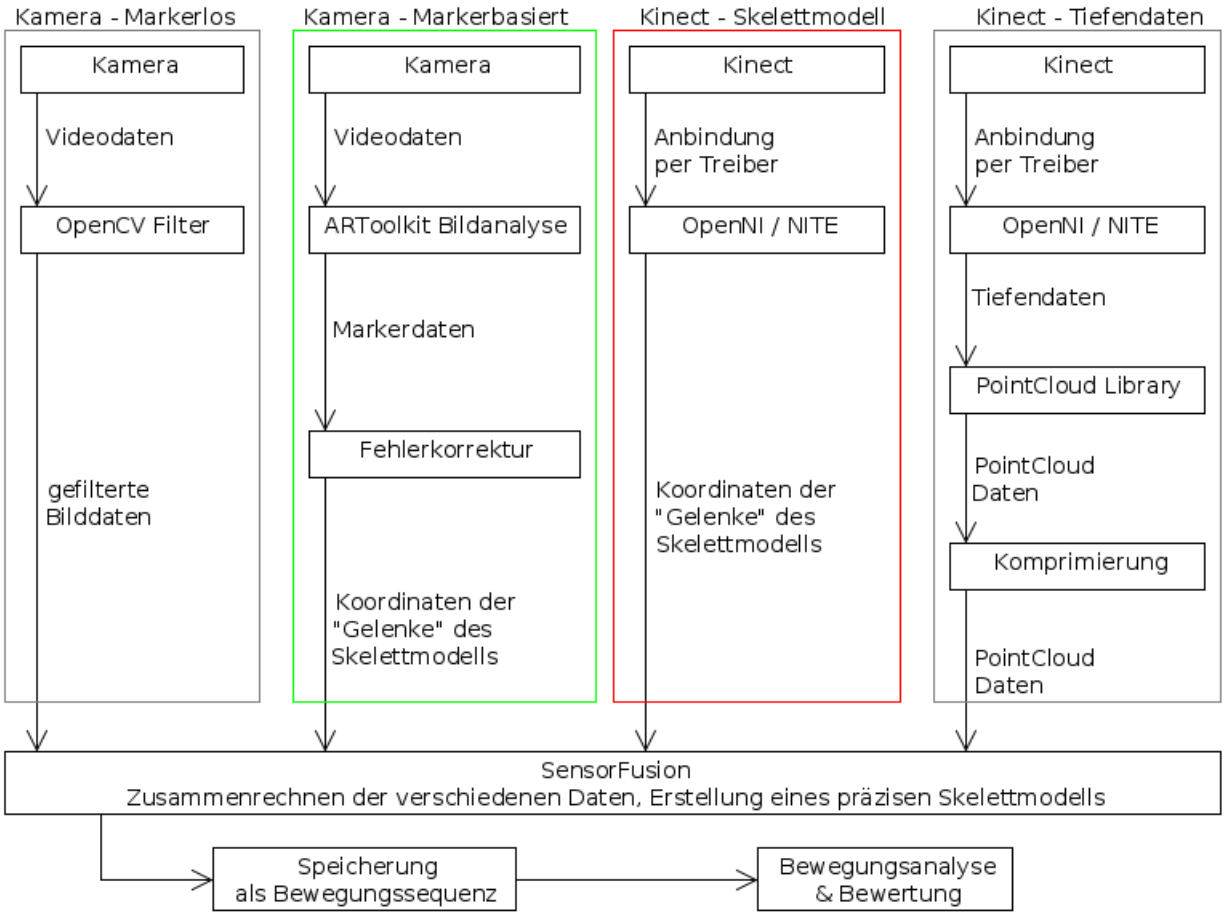

Abbildung 5.7.: Abstrakte Übersicht: Schrittweise Verarbeitung der Daten (erweitert)

5.3.2. Kinect - Tiefendaten

5.3.2.1. Datenstruktur

Die Weitergabe der Tiefendaten findet über ein mehrdimensionales Array von Integer (ganzzahligen) Werten statt. Das Array hat dieselbe Größe wie das Bild der Kinect RGB-Kamera, also 640 * 480 Punkte. Pro Punkt wird der Tiefenwert gespeichert. Aus diesem Array lassen sich X, Y und Z Koordinaten errechnen, die die Position der mit Infrarot angeleuchteten Punkte im dreidimensionalen Raum beschreiben. Bei Bedarf lassen sich parallel die Bilder der RGB-Kamera auslesen und als Textur über die Tiefendaten legen.

5.3.2.2. Problem der Datenrate

Bei diesem Schritt ist die Datenrate schwer in den Griff zu bekommen, wenn die gesamte Tiefenkarte gespeichert werden soll. Da der Infrarot-Sensor der Kinect bei 30 Bildern pro Sekunde arbeitet, kommt somit folgende Datenrate zusammen:

$$SpeicherbedarfPunkt * AnzahlPunkte * Bildrate = x\frac{bytes}{sec}$$

Die Gleichung lässt sich wie folgt aufschlüsseln:

- **Größe eines Punktes im Speicher**
  In der benutzten Umgebung (C in einem 32-bit Unix) ist ein Integer Wert typischerweise 2 Bytes groß. Da jeder Punkt 3 Werte hat (x, y, z), die seine Koordinaten im dreidimensionalen Raum darstellen, ist ein Punkt somit 3 * 2 Bytes, also 6 Bytes groß.
- **Gesamtanzahl der Punkte in der Tiefenkarte**
  Die vom Infrarotsensor der Kinect zurückgegebene Tiefenkarte enthält 640*480 Punkte (307200).
- **Bildrate**
  Der Infrarotsensor der Kinect arbeitet bei 30 Bildern pro Sekunde.

$$6 * 640 * 480 * 30 = 55296000\frac{bytes}{sec}$$

55296000 Byte, also 54000 Kilobyte oder 52.73438 Megabyte pro Sekunde. Berücksichtigt werden muss, dass bei dieser Berechnung diverse Overhead-Daten (die durch das Übertragungsprotokoll usw. entstehen), noch nicht aufgeführt sind. Die tatsächliche Datenrate dürfte das Ergebnis dieser Rechnung noch übertreffen. Somit folgt, dass eine Übermittlung der gesamten Tiefenkarte technisch nur schwer umzusetzen ist.

Tiefenfilter Ein Weg, die Datenmenge zu reduzieren, wäre die Punkte der Tiefe nach zu filtern, d.h. nur Punkte in einem bestimmten Abstand zum Sensor zu speichern. Dies hätte den Vorteil, dass keine Daten (abgesehen von dem Hintergrund) weggelassen werden. Selbst wenn z.B. $\frac{2}{3}$ der Punktdaten auf diese Weise eliminiert wären, würde die Datenrate jedoch immer noch ca. 17 Megabyte pro Sekunde betragen. Diese Lösung allein ist also nicht praktikabel.

Extrahierung und Komprimierung der Tiefendaten mittels PCL Mit Hilfe der Point Cloud Library ist über den OpenNI / NITE Treiber eine direkte Extrahierung der Tiefendaten möglich. Zwar besteht bei dieser Lösung dasselbe Problem wie bei 5.3.2.2, jedoch stellt die PCL Algorithmen zur Komprimierung der Tiefendaten zur Verfügung. Mit Hilfe dieser Komprimierungs-Algorithmen ist eine bis zu 20-fache Reduzierung der Daten möglich (Julius Kammerl (2011)).

5.3.3. Kamera - Markerlos

Bei markerlosen Systemen / Lösungen geht es darum, dem einzelnen Bild möglichst viele Informationen zu entnehmen. Je nach Anwendungsgebiet und Umgebung können verschiedene Filter kombiniert werden, damit die für den Einsatz relevanten Daten aus dem RAW-Frame gefiltert werden können. Somit entsteht weniger Overhead, wenn eine weitere Verarbeitung über das Netzwerk oder über die Festplatte eingeplant ist. Durch die richtige Kombination an Filtern erhält man somit nur die interessanten Bildbereiche.

OpenCV ist eine Softwarebibliothek, welche Algorithmen für Bildbearbeitung und maschinelles Sehen zur Verfügung stellt. Entworfen wurde OpenCV für C und C++, wobei auch Wrapper für andere Sprachen existieren. Ursprünglich wurde OpenCV von Intel entworfen, heute wird die Bibliothek hauptsächliche von Willow Garage gepflegt. Unter anderem sind in der Bibliothek Algorithmen für Gesichtsdetektion, 3D-Funktionalität, Haar-Klassifikatoren, wie auch verschiedene Filter (Sobel, Canny, Gauß).
Zusätzlich sind Funktionen für die Kamerakalibrierung enthalten.

- **Gaussian Blur Weichzeichner**: Durch das Weichzeichnen eines Bildes wird der Kontrast verringert. Anschließend lassen sich bestimmte Bildbereiche, getrennt durch Farben oder Helligkeitsbereiche, wie auch der Kombination aus beidem, leichter trennen. Die Details des Bildes wurden verwischt, somit erhöht sich die Wahrnehmung erwünschter Bilddetails.
- **Background Substraction**: Durch die Eliminierung nicht beweglicher Pixel oder Bereiche in einem Video-Stream durch den Vergleich des aktuellen Bildes mit dem Vorgängerbild, lässt sich der bewegliche Teil des Bildes herausfiltern. Background Sub-

straction kann entweder auf einem gewöhnlichen RGB-Bild erfolgen, ebenso kann das Verfahren aber auch auf eine Point Cloud angewendet werden. Somit wird der Daten-Overhead reduziert, zugleich fallen uninteressante Daten aus dem Bild oder aus der Datenstruktur heraus.

6. Zusammenfassung und Fazit

6.1. Zusammenfassung

Im Rahmen dieser Bachelorarbeit wurden erst im Kapitel 3: Analyse verschiedene etablierte und unkonventionelle Motion-Capture Ansätze vorgestellt und theoretisch nach mehreren Faktoren bewertet. Hierbei spielten Genauigkeit, Geschwindigkeit und vor Allem die Kosten eine Rolle.
Auf Grund des letztgenannten Faktors wurden die beiden günstigsten Verfahren zur weiteren Betrachtung ausgewählt.

In Kapitel 4 wurden zwei der vorgestellten Ansätze, optisch markerbasiert und Infrarot-Tiefenbild, genauer vorgestellt sowie ihre Funktionsweise erläutert. Mit Hilfe von Versuchsreihen wurde die Genauigkeit und der Einfluss verschiedener Variablen wie Geschwindigkeit und Entfernung untersucht. Dabei stellte sich heraus, dass der Infrarot-Tiefenbild Ansatz eine höhere Toleranz gegenüber schnellen Bewegungen hat und genauere Ergebnisse produziert.

In Kapitel 5 wurde das Konzept eines verteilten Motion-Capture Systems vorgestellt, welches zur Aufzeichnung menschlicher Bewegungen einen oder beide der vorgestellten Ansätze verwendet. Es wurde ebenfalls ein Ausblick auf Erweiterungen des Systems gegeben, welche einen Zuwachs von Präzision bei der Bewegungsaufzeichnung bedeuten könnten.

6.2. Fazit

Die im Rahmen dieser Arbeit durchgeführten Versuche und Überlegungen zu kamerabasierten Motion-Capture Ansätzen haben gezeigt, dass bei dem aktuellen Stand der Technik Motion-Capture Systeme zu einem Bruchteil ihrer vorherigen Kosten umgesetzt werden können. Natürlich ist die Präzision der vorgestellten kamerabasierten Verfahren nicht auf dem gleichen Niveau wie bei den traditionellen Motion-Capture Verfahren, eine brauchbare Genauigkeit kann sich jedoch mit mehreren Sensoren und sinnvoller Auswertung und Interpretation der Daten erreichen lassen.

Insbesondere der Infrarot-Tiefenbild-Ansatz bietet eine Präzision, die für den Massenmarkt vor ein paar Jahren aufgrund der hohen Kosten noch unerreichbar gewesen wäre.

6.3. Ausblick

6.3.1. Andere Verwendungszwecke vorgestellter Technologien

Zusätzlich zur strikten Interpretation von Motion-Capture sind weitere Szenarien denkbar, unter denen beide im Kapitel Messverfahren vorgestellten Technologien zusammen eingesetzt werden können. So wäre z.B. ein Szenario denkbar, bei dem sich ein Proband durch einen "kahlen" Raum bewegt, an dessen Wänden (und anderen Oberflächen) ARToolkit-Marker angebracht sind. Mit einem HMD und einer daran befestigten Kamera könnte eine komplette Einrichtung des Raumes virtuell für den Betrachter dargestellt werden. Wenn der Raum ebenfalls mit Infrarot-Tiefenkameras ausgestattet wäre, könnte der Raum auf den Probanden reagieren bzw. sich verändern. Ein Beispiel für so eine Anwendung wäre eine virtuelle Kunstgalerie oder eine virtuelle Besichtigung von historischen Bauwerken.

6.3.2. Überlegungen zum Datenschutz

Der eigentliche Anfangspunkt für diese Arbeit war die Idee, eine Wohnung oder einen anderen Raum mit Sensoren auszukleiden, die im Raum vorhandene Personen beobachten und so eine Steuerung der Haushalts- bzw. Unterhaltungselektronik ermöglichen sollte. Diese Art der Nutzung der vorgestellten Verfahren würde eine sehr komfortable Nutzung bzw. Steuerung von Geräten im Haushalt darstellen, jedoch auch potenzielle Risiken für den Datenschutz birgt. Angenommen, so ein System existiere in einer kommerziellen Form auf dem Markt, könnte der Hersteller mit dem System gewonnene Daten auswerten, um z.B. personalisierte Werbung beim Anwender anzuwenden. Würde ein auf eine Privatwohnung ausgerichtetes Motion-Capture System Aufzeichnungen des Verhaltens der Bewohner speichern, könnte dies bei unabsichtlicher Weitergabe der Daten eine empfindliche Beeinträchtigung der Privatsphäre nach sich ziehen.
Ein Beispiel für kontroversen Umgang mit persönlichen Daten bietet aktuell die Social-Networking Seite Facebook, bei der seit Ihrer Entstehung personenbezogene Werbung und Gesichtserkennung (vgl. Jens Ihlenfeld (2011)) als Features hinzugekommen sind. Gerade die Hinzunahme von biometrischen Daten wie Gesichtsdaten muss genau betrachtet werden, um eventuell unerwünschte Verwendungszwecke auszuschließen.
Es liegt an den Entwicklern solcher Systeme, die Sicherheit von persönlichen Daten zu gewährleisten, an den Herstellern, mit den unter Umständen so gewonnenen Daten sorgsam

und nicht missbräuchlich umzugehen, und letztendlich auch beim Verbraucher, sich ins Bewusstsein zu rufen, dass der Einsatz von komplexer Informationstechnologie im Privatleben immer überlegt sein muss.

Literaturverzeichnis

[Albitar u. a. 2007] ALBITAR, I.C. ; GRAEBLING, P. ; DOIGNON, C.: Robust Structured Light Coding for 3D Reconstruction. In: *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, oct. 2007, S. 1 –6. – ISSN 1550-5499

[Arne Bernin 2010] ARNE BERNIN: *Kinect chess board meta pattern*. Website. 2010. – Online verfuegbar unter http://livingplace.informatik.haw-hamburg.de/blog; letzter Zugriff 2011-07-22.

[ARToolworks Inc. 2010] ARTOOLWORKS INC.: *ARToolworks announces ARToolKit for iPhone beta 1*. Website. April 2010. – Online verfuegbar unter http://www.artoolworks.com/community/forum/viewtopic.php?f=4&t=1042/; letzter Zugriff 2011-04-23.

[ARToolworks Inc. 2011] ARTOOLWORKS INC.: *Products - ARToolworks*. Website. 2011. – Online verfuegbar unter http://www.artoolworks.com/products/; letzter Zugriff 2011-04-23.

[Axis 2011] AXIS: *Axis 211W Netzwerk Kamera*. Website. 2011. – Online verfuegbar unter http://www.axis.com/de/products/video/camera/productguide.htm; letzter Zugriff 2011-07-13.

[Brian X. Chen 2009] BRIAN X. CHEN: *If You're Not Seeing Data, You're Not Seeing*. Website. August 2009. – Online verfuegbar unter http://www.wired.com/gadgetlab/2009/08/augmented-reality/; letzter Zugriff 2011-04-23.

[Davis und Ellis 1964] DAVIS, M. R. ; ELLIS, T. O.: The RAND tablet: a man-machine graphical communication device. In: *Proceedings of the October 27-29, 1964, fall joint computer conference, part I*. New York, NY, USA : ACM, 1964 (AFIPS '64 (Fall, part I)), S. 325–331. – URL http://doi.acm.org/10.1145/1464052.1464080

[Devernay und Faugeras 1994] DEVERNAY, F. ; FAUGERAS, O.D.: Computing differential properties of 3-D shapes from stereoscopic images without 3-D models. In: *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, jun 1994, S. 208 –213. – ISSN 1063-6919

[Devernay u. a. 2002] DEVERNAY, Frédéric ; BANTICHE, Olivier ; COSTE-MANIÈRE, Ève: Structured light on dynamic scenes using standard stereoscopy algorithms / INRIA. URL http://hal.inria.fr/inria-00072111/en/, Juni 2002 (RR-4477). – Forschungsbericht

[Erwin Coumans 2011] ERWIN COUMANS: *bullet - Bullet is a professional free 3D Game Multiphysics Library*. Website. 2011. – Online verfuegbar unter http://code.google.com/p/bullet/; letzter Zugriff 2011-08-15.

[Fischer und Simmer 1996] FISCHER, Sven ; SIMMER, Klaus U.: Beamforming microphone arrays for speech acquisition in noisy environments. In: *Speech Communication* 20 (1996), Nr. 3-4, S. 215 – 227. – URL http://www.sciencedirect.com/science/article/pii/S0167639396000544. – Acoustic Echo Control and Speech Enhancement Techniques. – ISSN 0167-6393

[Graz University of Technology 2011] GRAZ UNIVERSITY OF TECHNOLOGY: *ARToolkitPlus*. Website. 2011. – Online verfuegbar unter http://handheldar.icg.tugraz.at/artoolkitplus.php; letzter Zugriff 2011-08-18.

[HIT Lab Washington University 2011] HIT LAB WASHINGTON UNIVERSITY: *ARToolkit Home Page*. Website. 2011. – Online verfuegbar unter http://www.hitl.washington.edu/artoolkit/; letzter Zugriff 2011-04-19.

[Jens Ihlenfeld 2011] JENS IHLENFELD: *Datenschutz: Facebook aktiviert automatische Gesichtserkennung*. Website. 2011. – Online verfuegbar unter http://www.golem.de/1106/84038.html; letzter Zugriff 2011-06-24.

[Julius Kammerl 2011] JULIUS KAMMERL: *Compressing Point Clouds*. Website. 2011. – Online verfuegbar unter http://www.pointclouds.org/news/compressing-point-clouds.html; letzter Zugriff 2011-06-14.

[Kalman 1960] KALMAN, R. E.: A New Approach to Linear Filtering and Prediction Problems. In: *Transactions of the ASME - Journal of Basic Engineering* (1960), Nr. 82 (Series D), S. 35–45. – URL http://www.cs.unc.edu/~welch/kalman/media/pdf/Kalman1960.pdf

[Kato und Billinghurst 1999] KATO, Hirokazu ; BILLINGHURST, Mark: Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System. In: *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality*. Washington, DC, USA : IEEE Computer Society, 1999, S. 85–. – URL http://portal.acm.org/citation.cfm?id=857202.858134. – ISBN 0-7695-0359-4

[Kyle McDonald 2011] KYLE MCDONALD: *Photos of Kyle McDonald*. Website. 2011. – Online verfuegbar unter http://www.flickr.com/photos/kylemcdonald/5174106004/in/photosof-kylemcdonald/; letzter Zugriff 2011-06-07.

[Lange und Seitz 2001] LANGE, R. ; SEITZ, P.: Solid-state time-of-flight range camera. In: *Quantum Electronics, IEEE Journal of* 37 (2001), mar, Nr. 3, S. 390 –397. – ISSN 0018-9197

[Logitech 2011] LOGITECH: *Logitech C500*. Website. 2011. – Online verfuegbar unter http://www.logitech.com/de-de/435/5866; letzter Zugriff 2011-07-13.

[Microsoft 2010] MICROSOFT: *PrimeSense Supplies 3-D-Sensing Technology to 'Project Natal' for Xbox 360*. Website. Maerz 2010. – Online verfuegbar unter http://www.microsoft.com/Presspass/press/2010/mar10/03-31PrimeSensePR.mspx; letzter Zugriff 2011-04-23.

[Microsoft 2011] MICROSOFT: *Kinect for Xbox 360*. Website. 2011. – Online verfuegbar unter http://www.xbox.com/en-US/Xbox360/Accessories/Kinect/kinectforxbox360; letzter Zugriff 2011-04-23.

[Ng u. a. 2005] NG, Ren ; LEVOY, Marc ; BRÉDIF, Mathieu ; DUVAL, Gene ; HOROWITZ, Mark ; HANRAHAN, Pat: Light Field Photography with a Hand-Held Plenoptic Camera. URL http://graphics.stanford.edu/papers/lfcamera/, April 2005. – Forschungsbericht

[Primesense Inc. 2010] PRIMESENSE INC.: *Prime Sensor NITE 1.3 Algorithms notes*. Website. 2010. – Online verfuegbar unter http://www.primesense.com/?p=515; letzter Zugriff 2011-08-12.

[PrimeSense Ltd. 2011] PRIMESENSE LTD.: *Primesense FAQ*. Website. 2011. – Online verfuegbar unter http://www.primesense.com/?p=535; letzter Zugriff 2011-07-22.

[Sementille u. a. 2004] SEMENTILLE, Antonio C. ; LOURENÇO, Luís E. ; BREGA, José Remo F. ; RODELLO, Ildeberto: A motion capture system using passive markers. In: *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*. New York, NY, USA : ACM, 2004 (VRCAI '04), S. 440–447. – URL http://doi.acm.org/10.1145/1044588.1044684. – ISBN 1-58113-884-9

[Sturman und Zeltzer 1994] STURMAN, D.J. ; ZELTZER, D.: A survey of glove-based input. In: *Computer Graphics and Applications, IEEE* 14 (1994), jan, Nr. 1, S. 30 –39. – ISSN 0272-1716

Anhang

A. Erläuterung zu Graphen

Die in den nachfolgenden zwei Abschnitten gezeigten Graphen sind im Rahmen der Auswertung der beiden Versuchsreihen aus Kapitel 4 entstanden. Bei den Versuchen aufgezeichnete Daten wurden mittels GNUPlot als Graphen dargestellt.

B. Graphen: Versuchsreihe ARToolkit

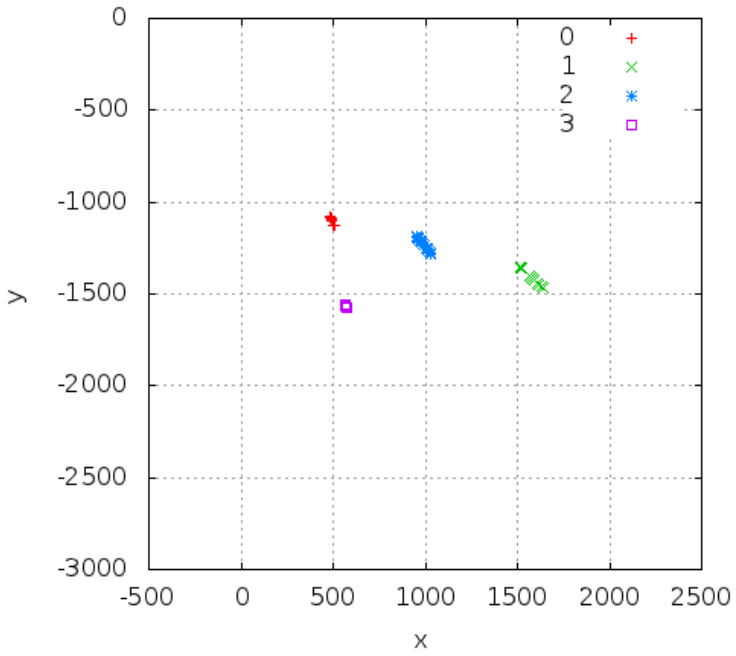

Abbildung B.1.: Versuch “still” bei 2 Meter Entfernung zur Kamera

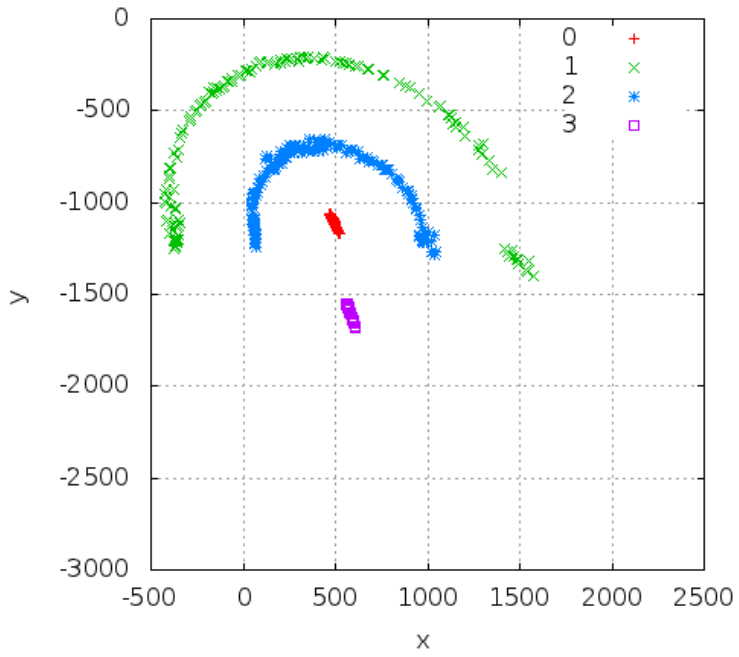

Abbildung B.2.: Versuch “slow” bei 2 Meter Entfernung zur Kamera

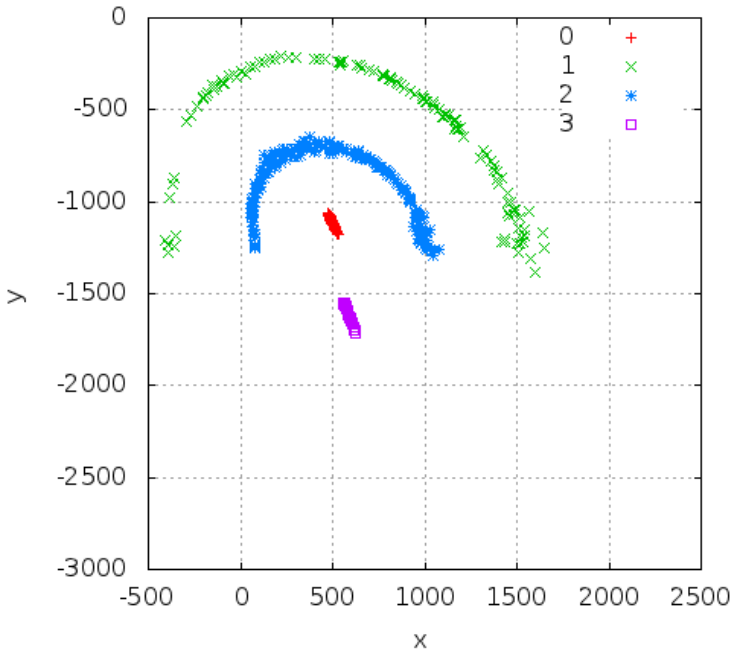

Abbildung B.3.: Versuch “medium” bei 2 Meter Entfernung zur Kamera

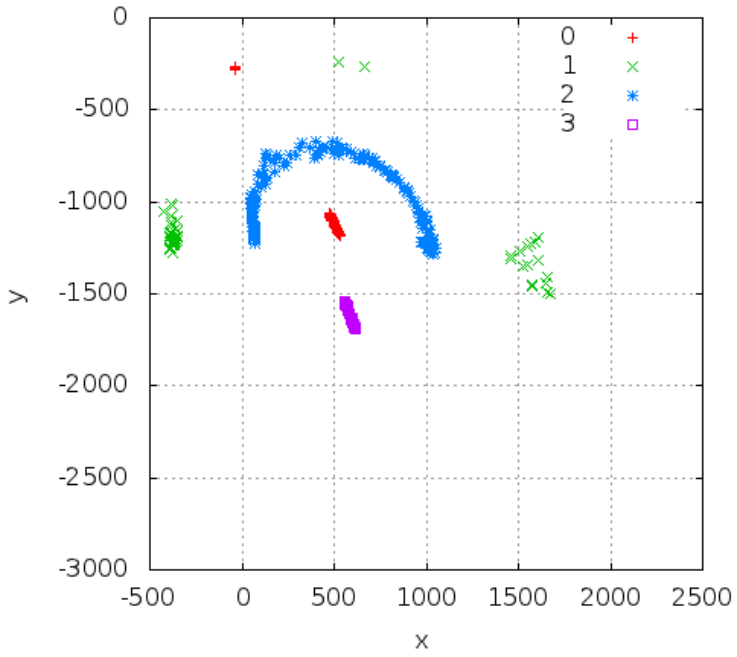

Abbildung B.4.: Versuch “fast” bei 2 Meter Entfernung zur Kamera

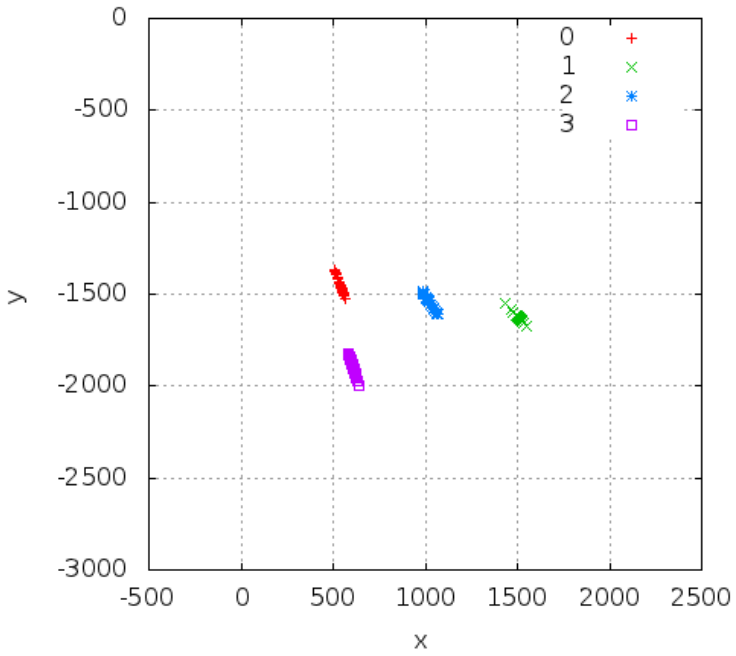

Abbildung B.5.: Versuch “still” bei 2,5 Meter Entfernung zur Kamera

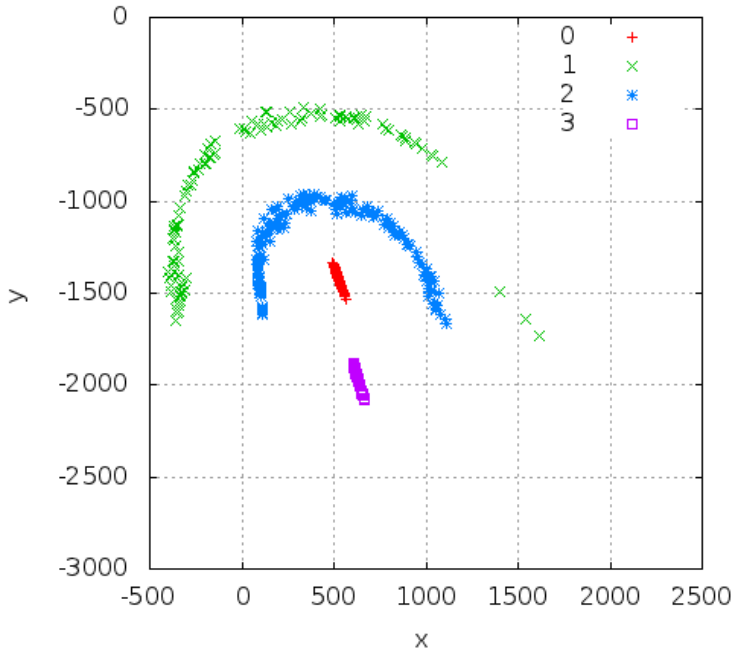

Abbildung B.6.: Versuch “slow” bei 2,5 Meter Entfernung zur Kamera

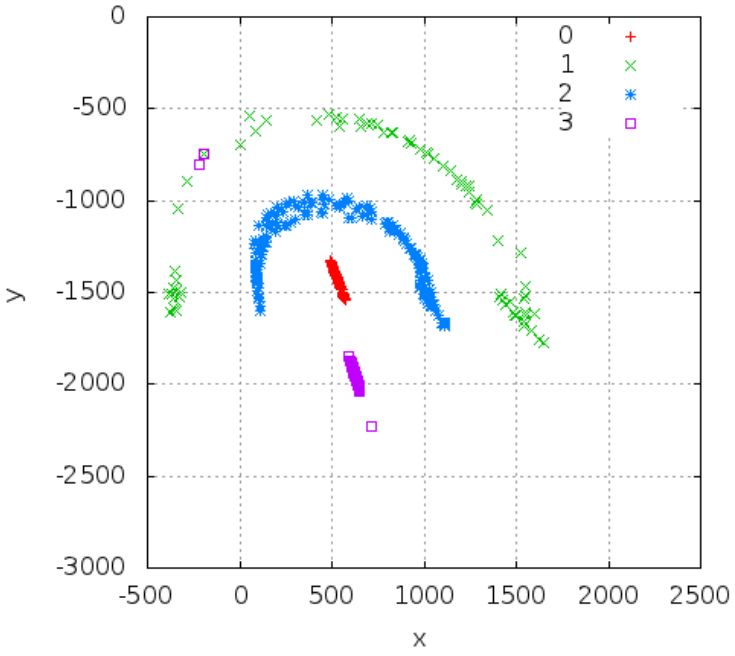

Abbildung B.7.: Versuch “medium” bei 2,5 Meter Entfernung zur Kamera

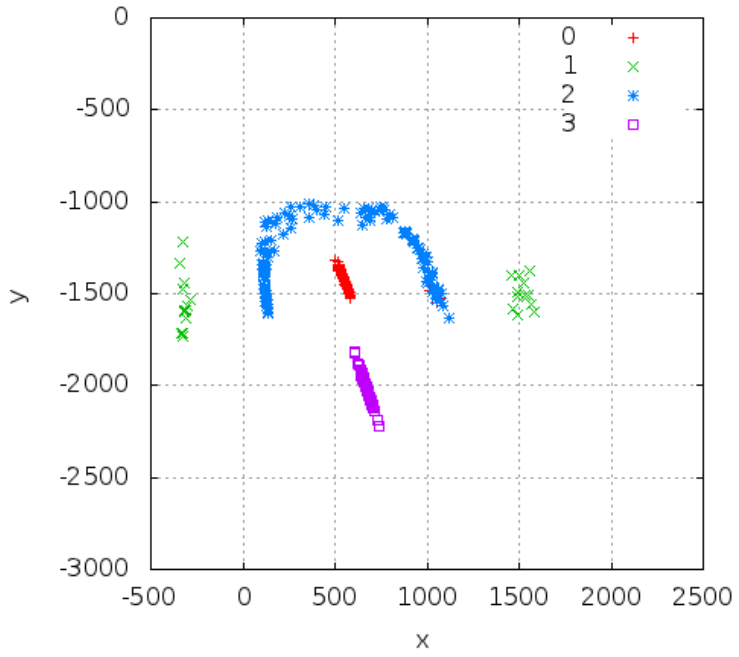

Abbildung B.8.: Versuch “fast” bei 2,5 Meter Entfernung zur Kamera

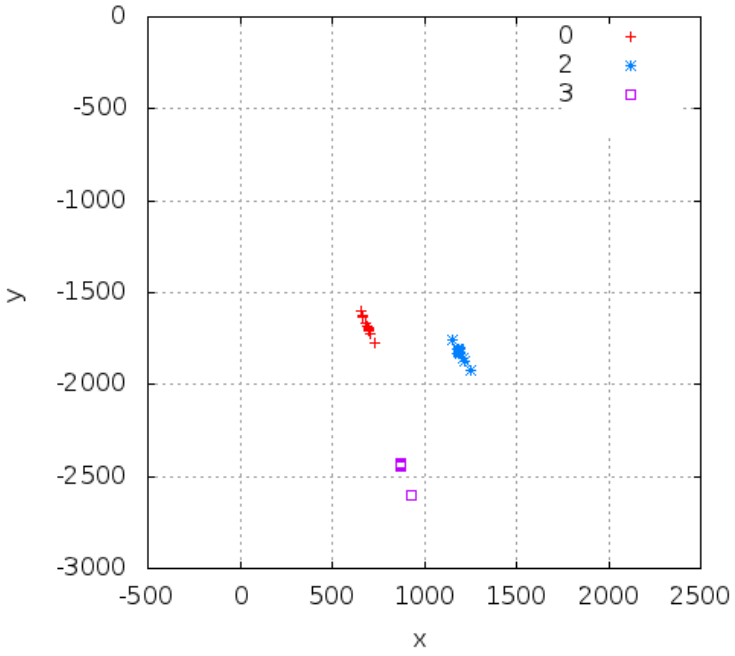

Abbildung B.9.: Versuch “still” bei 3 Meter Entfernung zur Kamera

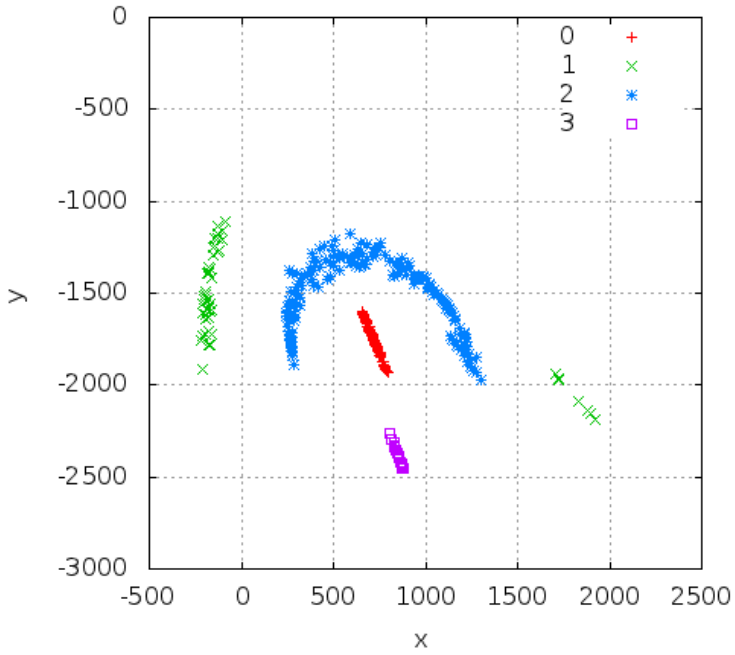

Abbildung B.10.: Versuch “slow” bei 3 Meter Entfernung zur Kamera

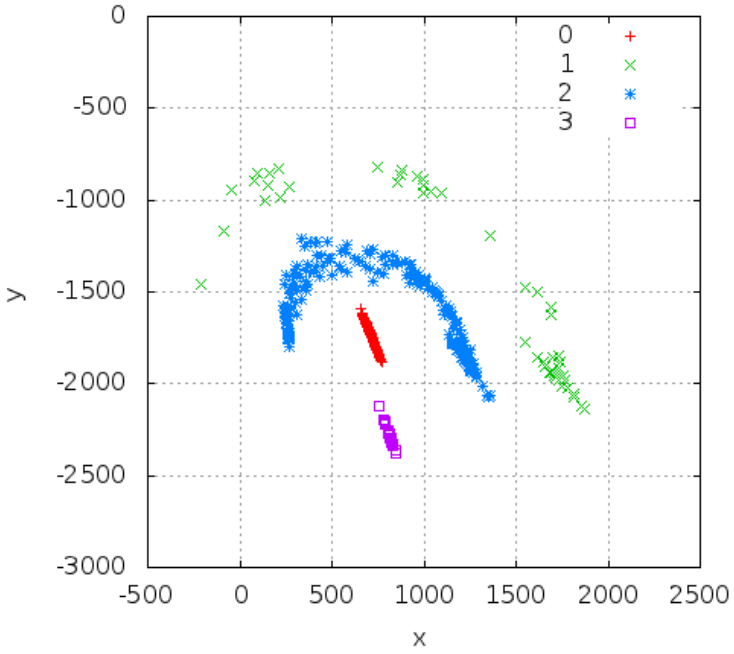

Abbildung B.11.: Versuch “medium” bei 3 Meter Entfernung zur Kamera

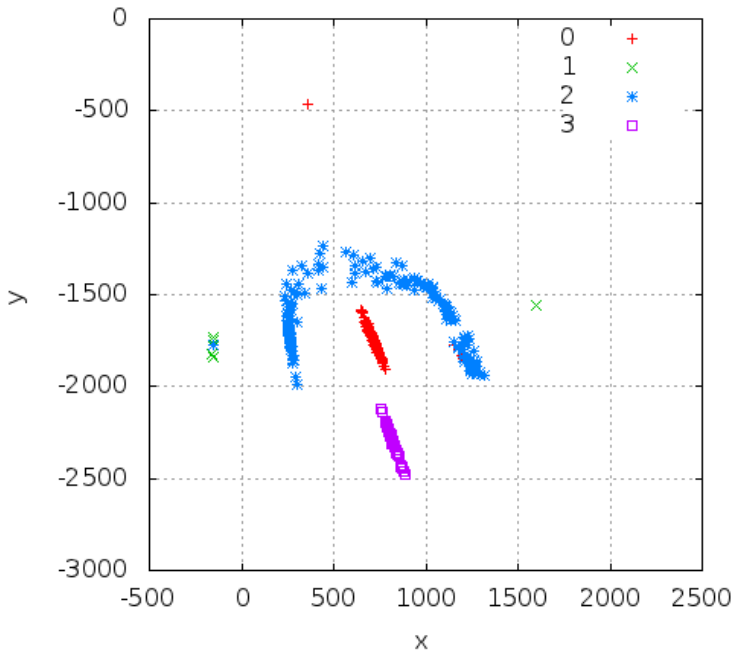

Abbildung B.12.: Versuch “fast” bei 3 Meter Entfernung zur Kamera

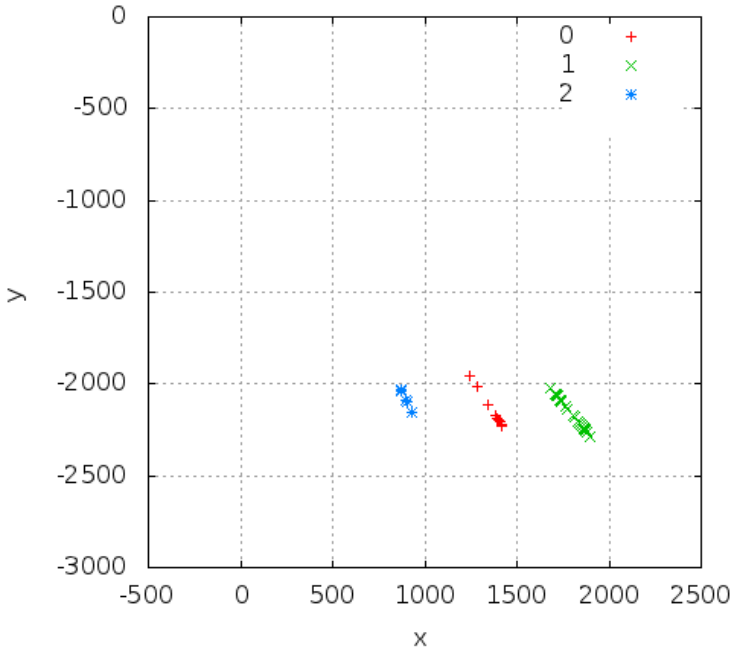

Abbildung B.13.: Versuch “still” bei 3,5 Meter Entfernung zur Kamera

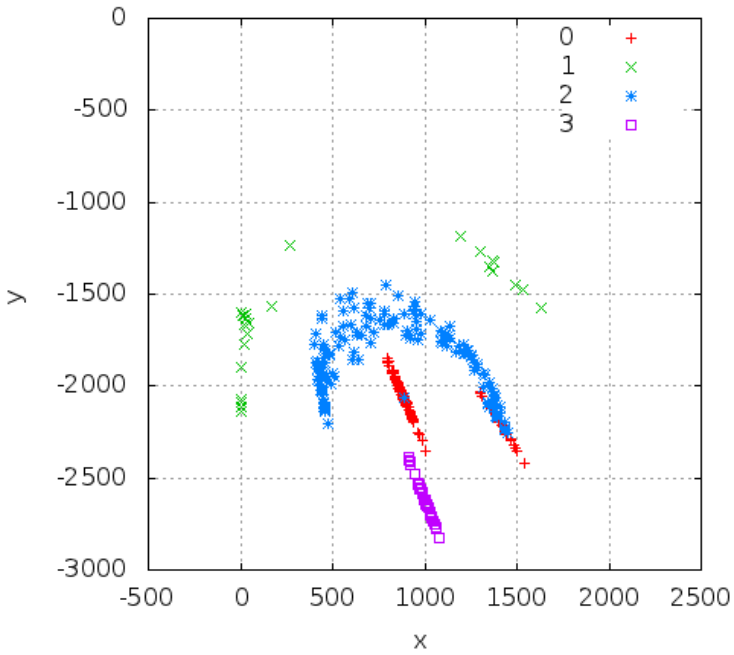

Abbildung B.14.: Versuch “slow” bei 3,5 Meter Entfernung zur Kamera

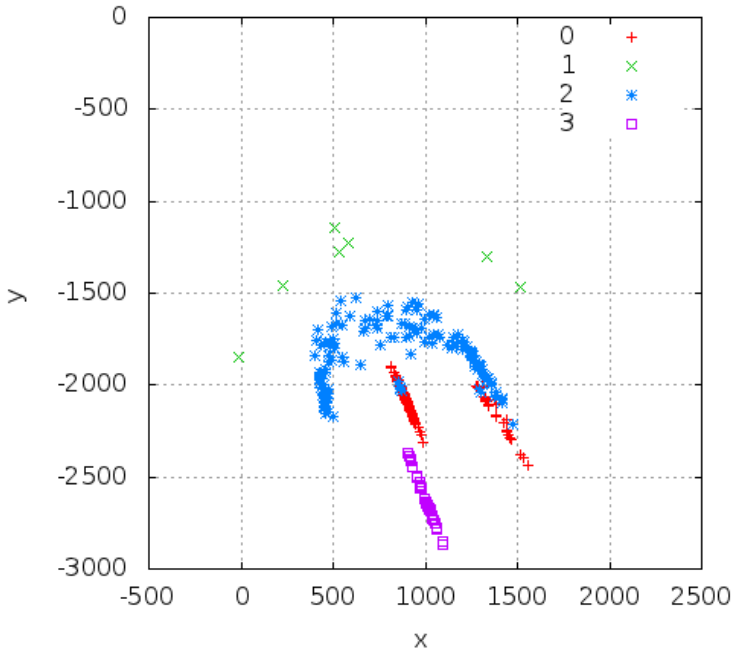

Abbildung B.15.: Versuch “medium” bei 3,5 Meter Entfernung zur Kamera

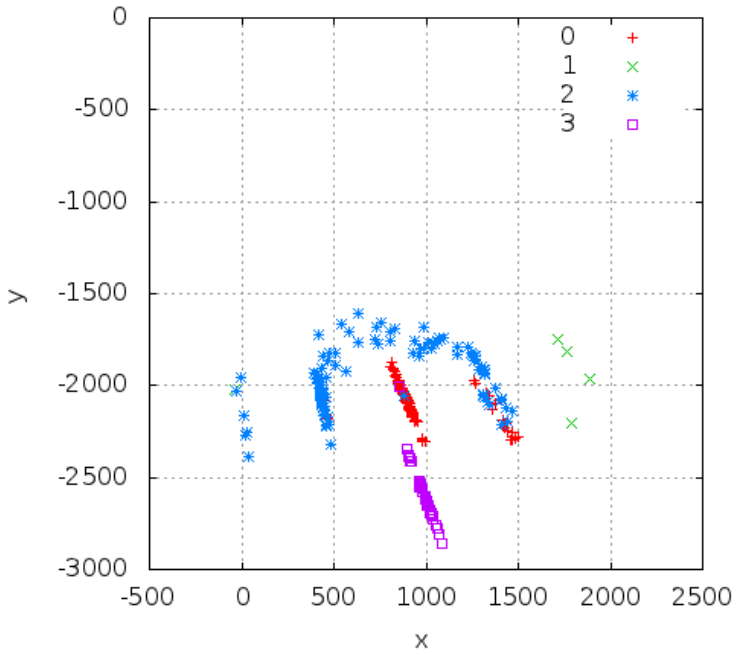

Abbildung B.16.: Versuch “fast” bei 3,5 Meter Entfernung zur Kamera

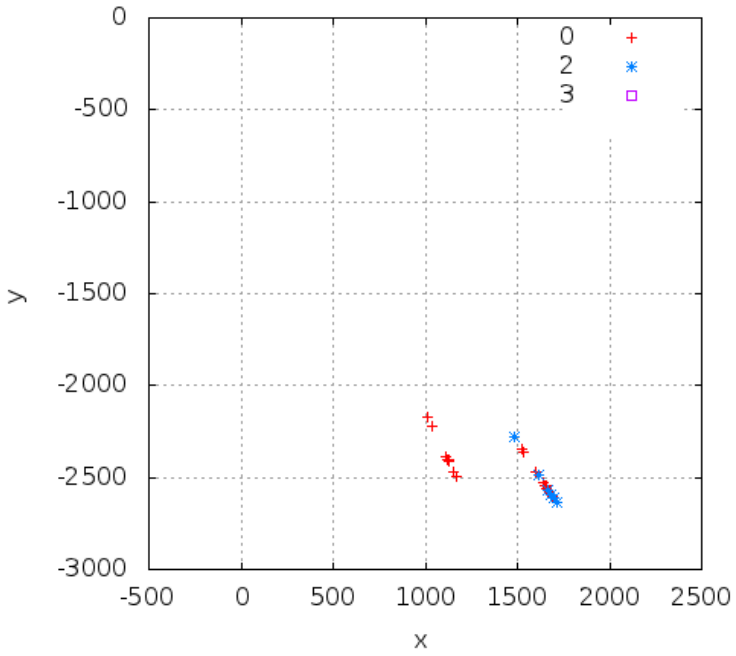

Abbildung B.17.: Versuch “still” bei 4 Meter Entfernung zur Kamera

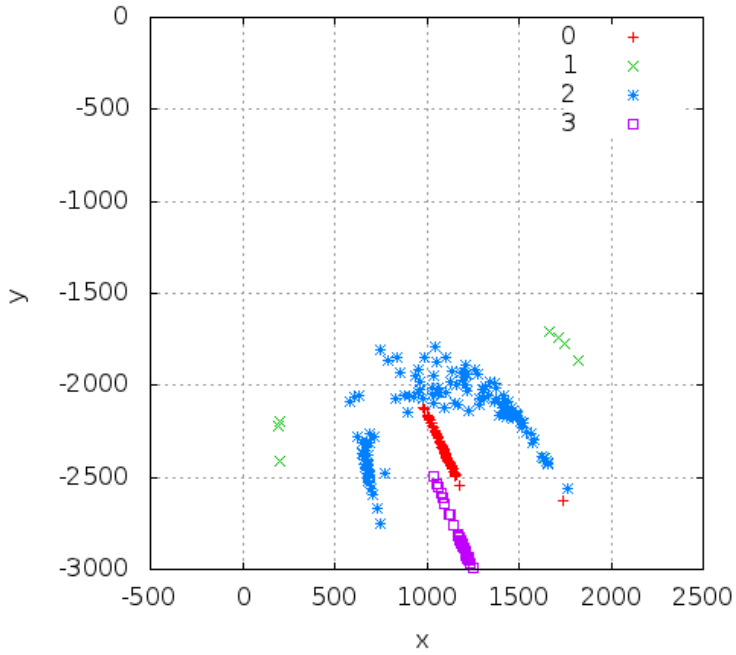

Abbildung B.18.: Versuch “slow” bei 4 Meter Entfernung zur Kamera

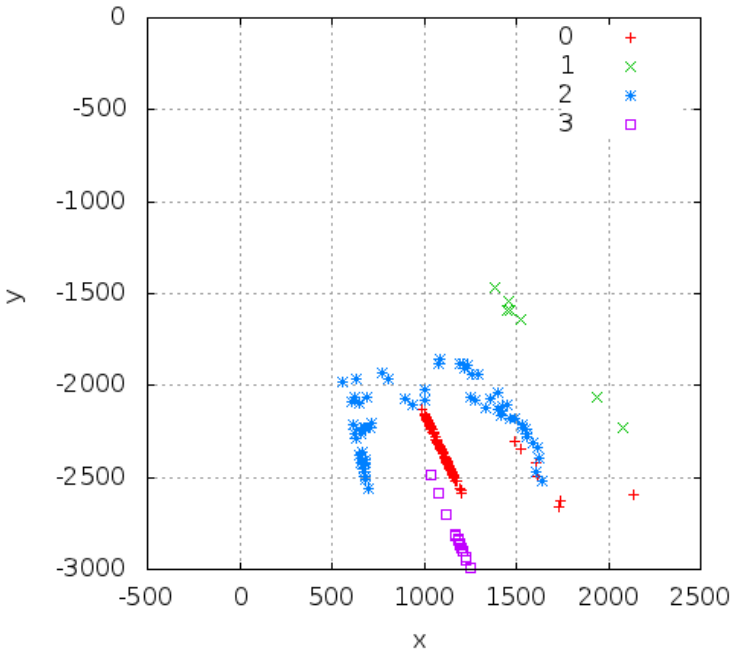

Abbildung B.19.: Versuch “medium” bei 4 Meter Entfernung zur Kamera

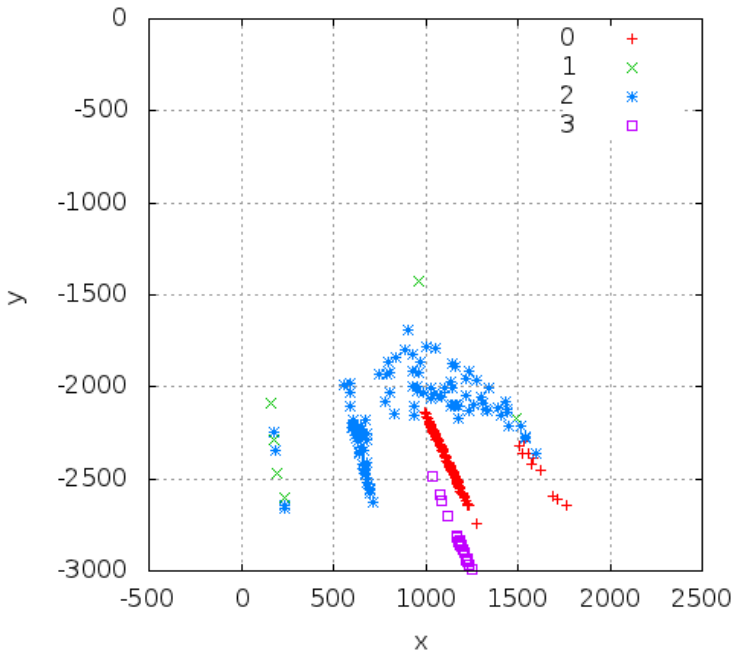

Abbildung B.20.: Versuch “fast” bei 4 Meter Entfernung zur Kamera

C. Graphen: Versuchsreihe Kinect

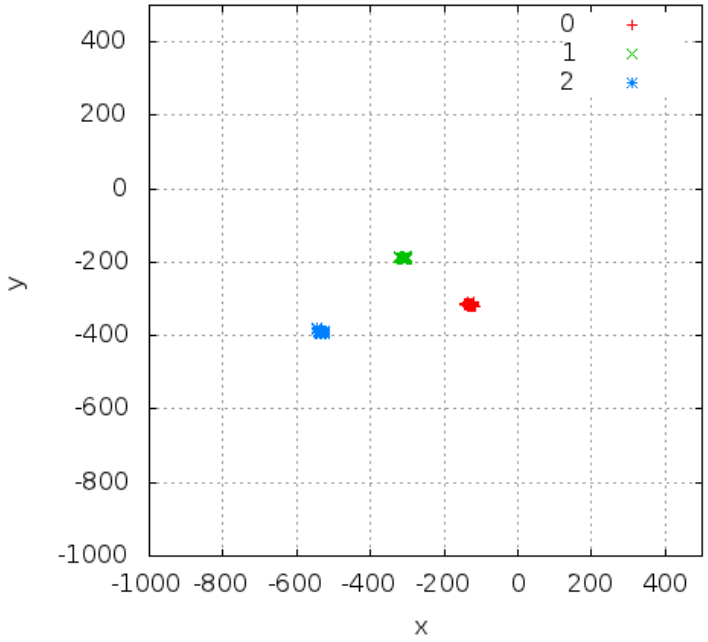

Abbildung C.21.: Versuch “still” bei 2 Meter Entfernung zur Kamera

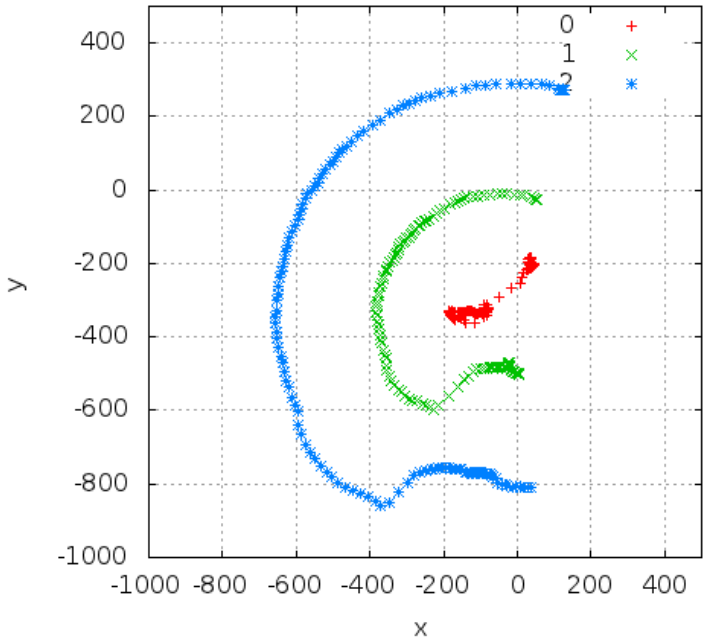

Abbildung C.22.: Versuch “slow” bei 2 Meter Entfernung zur Kamera

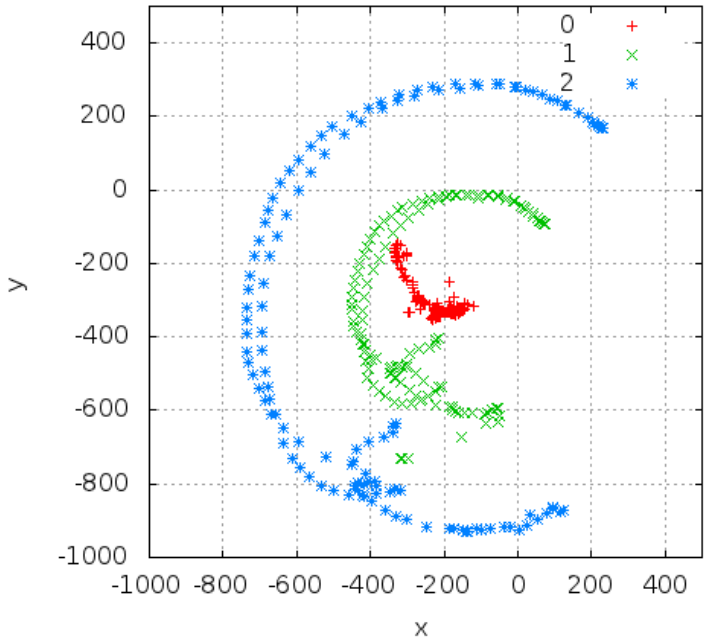

Abbildung C.23.: Versuch “medium” bei 2 Meter Entfernung zur Kamera

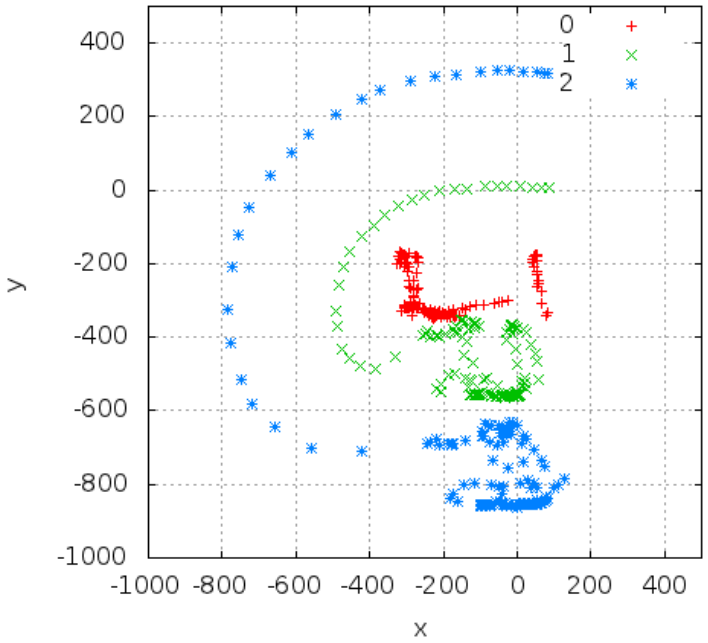

Abbildung C.24.: Versuch “fast” bei 2 Meter Entfernung zur Kamera

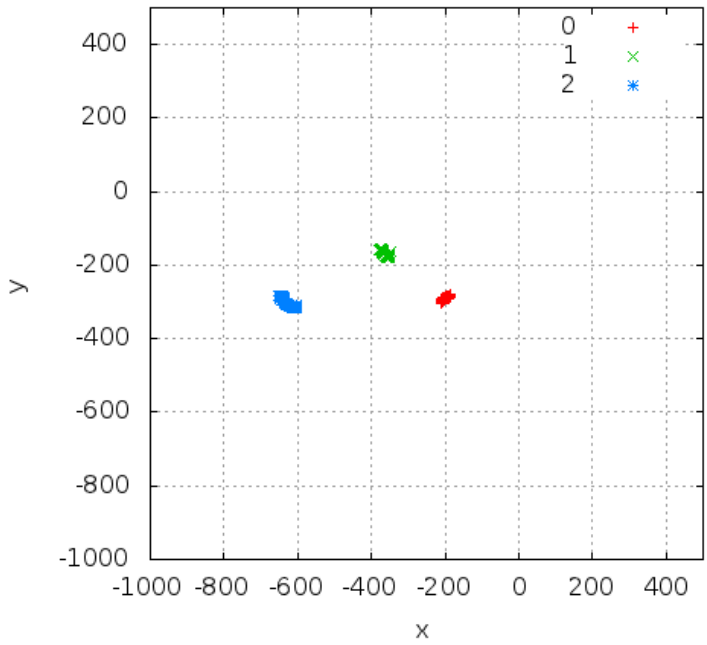

Abbildung C.25.: Versuch “still” bei 2,5 Meter Entfernung zur Kamera

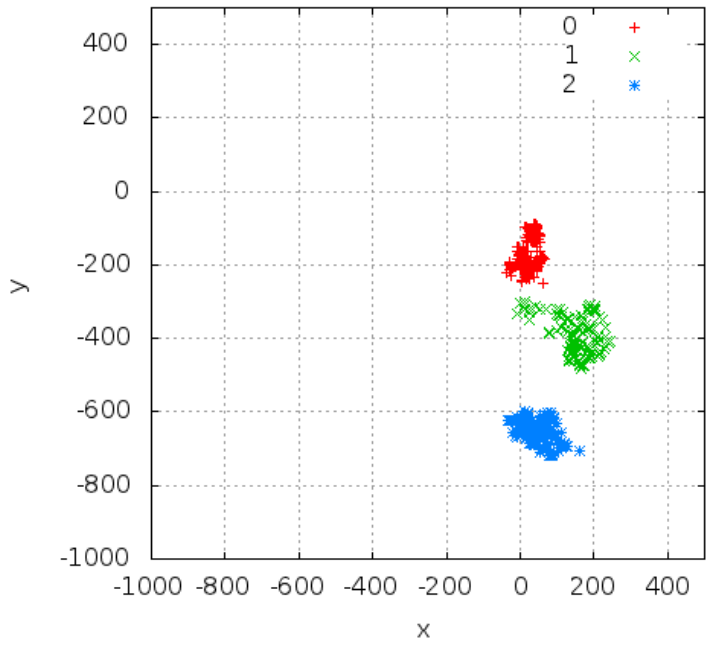

Abbildung C.26.: Versuch “slow” bei 2,5 Meter Entfernung zur Kamera

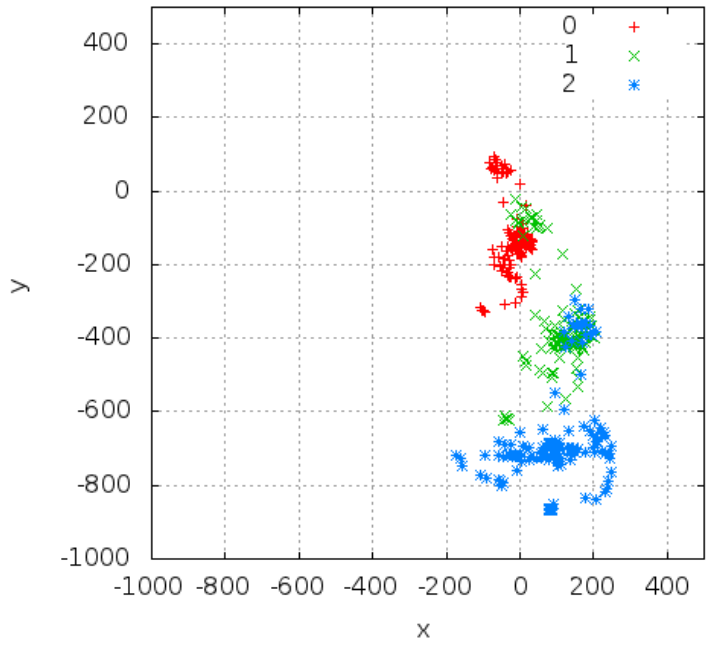

Abbildung C.27.: Versuch “medium” bei 2,5 Meter Entfernung zur Kamera

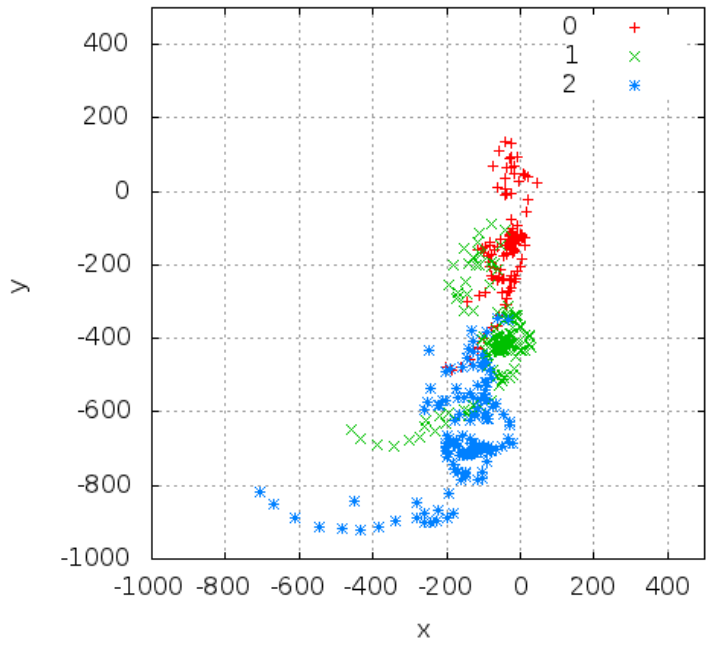

Abbildung C.28.: Versuch “fast” bei 2,5 Meter Entfernung zur Kamera

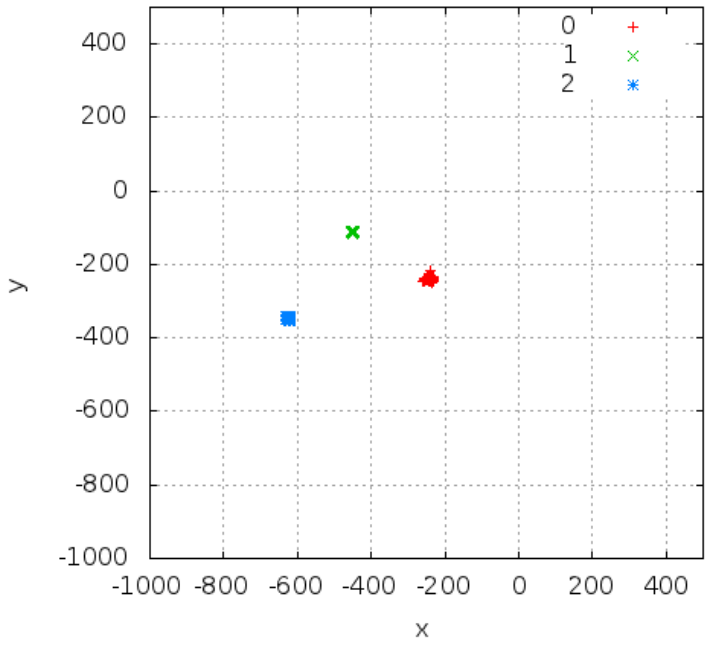

Abbildung C.29.: Versuch “still” bei 3 Meter Entfernung zur Kamera

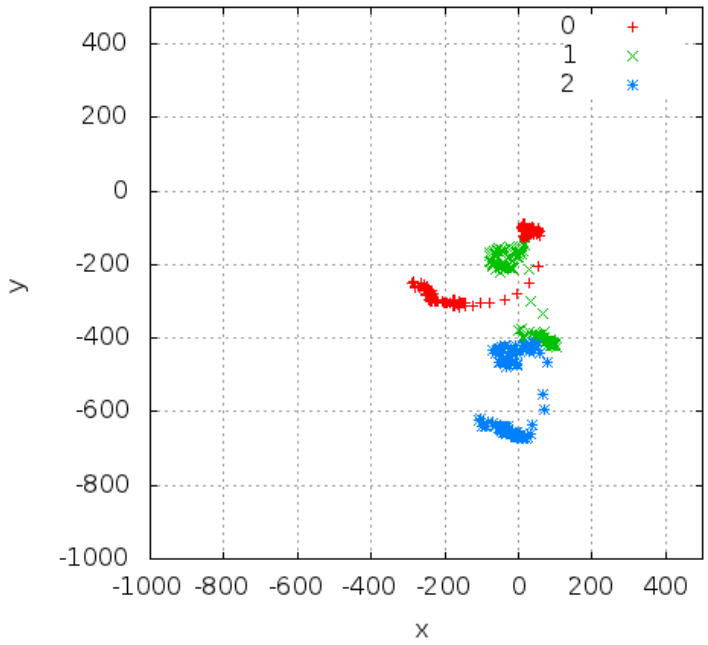

Abbildung C.30.: Versuch “slow” bei 3 Meter Entfernung zur Kamera

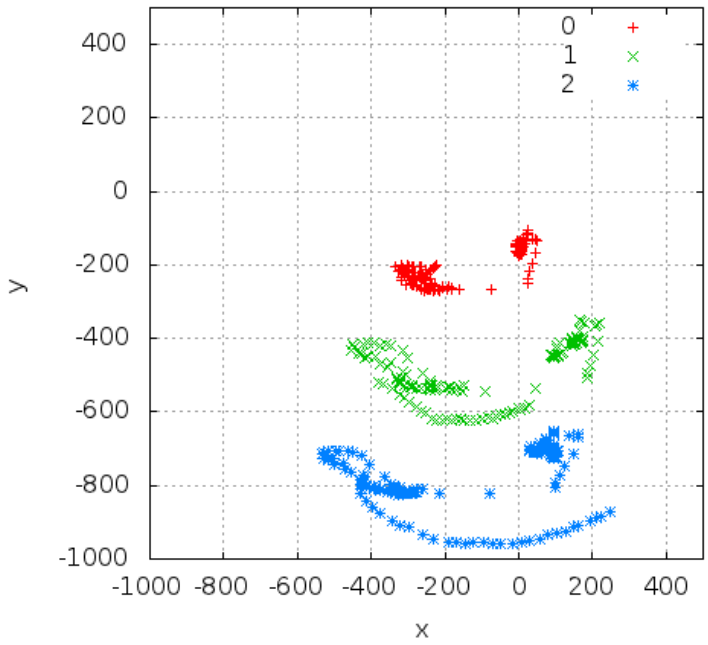

Abbildung C.31.: Versuch “medium” bei 3 Meter Entfernung zur Kamera

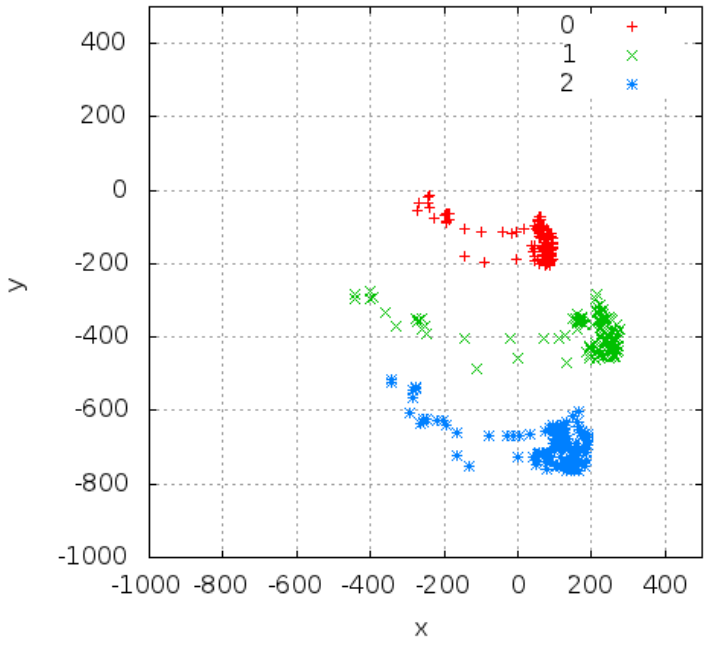

Abbildung C.32.: Versuch “fast” bei 3 Meter Entfernung zur Kamera

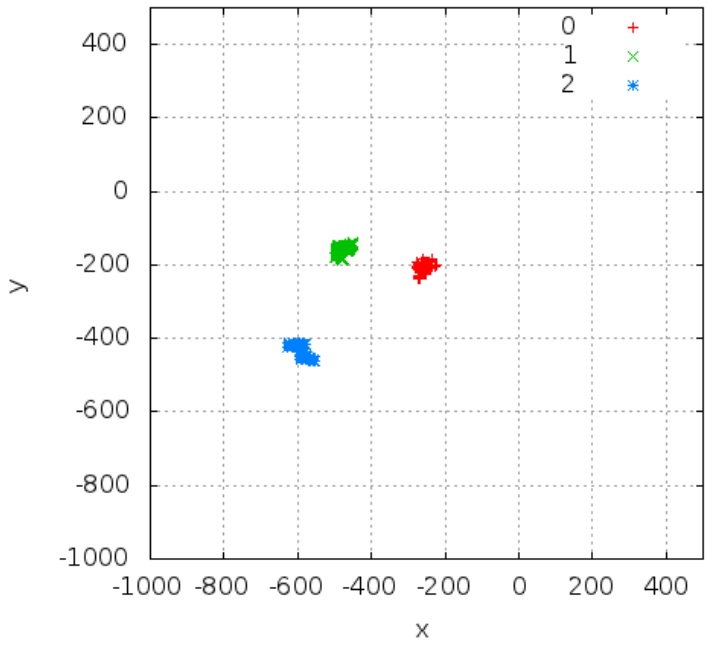

Abbildung C.33.: Versuch “still” bei 3,5 Meter Entfernung zur Kamera

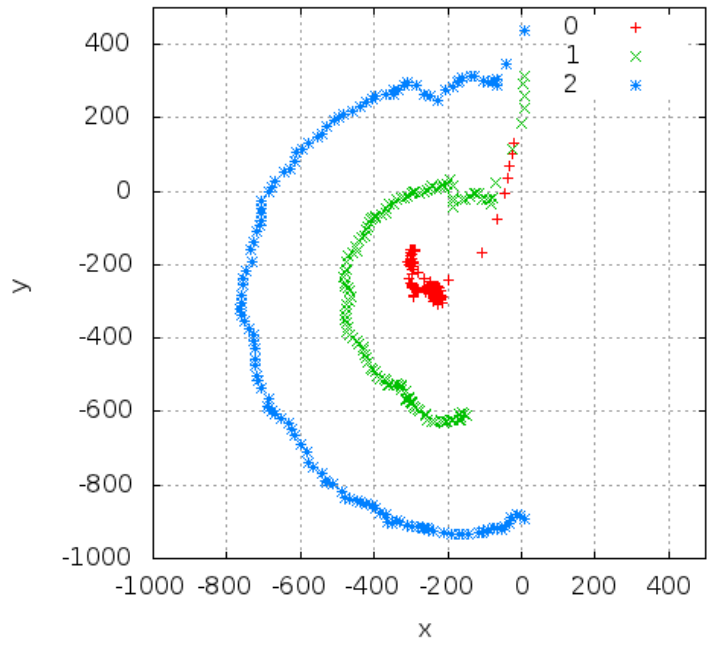

Abbildung C.34.: Versuch “slow” bei 3,5 Meter Entfernung zur Kamera

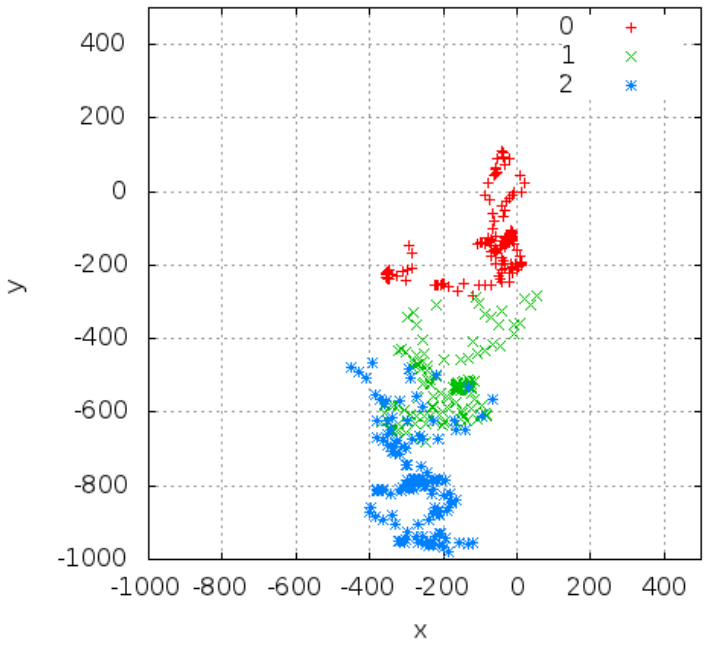

Abbildung C.35.: Versuch “medium” bei 3,5 Meter Entfernung zur Kamera

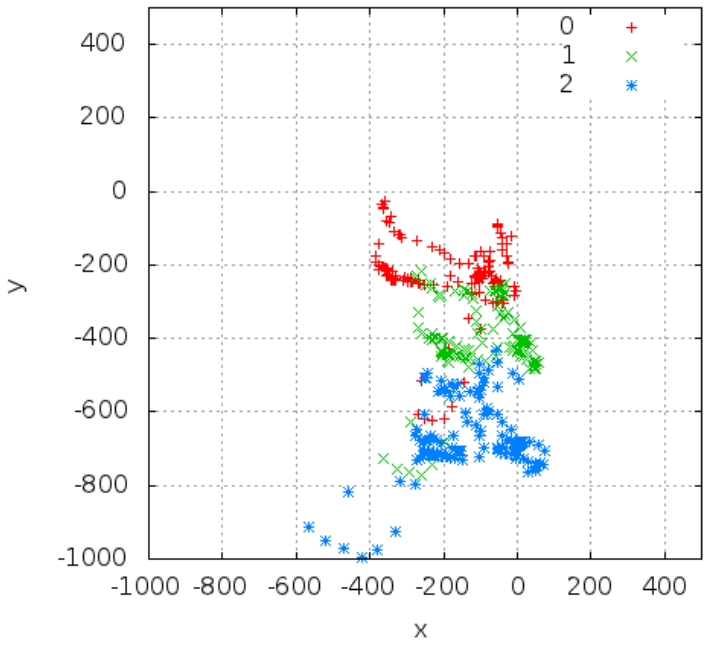

Abbildung C.36.: Versuch “fast” bei 3,5 Meter Entfernung zur Kamera

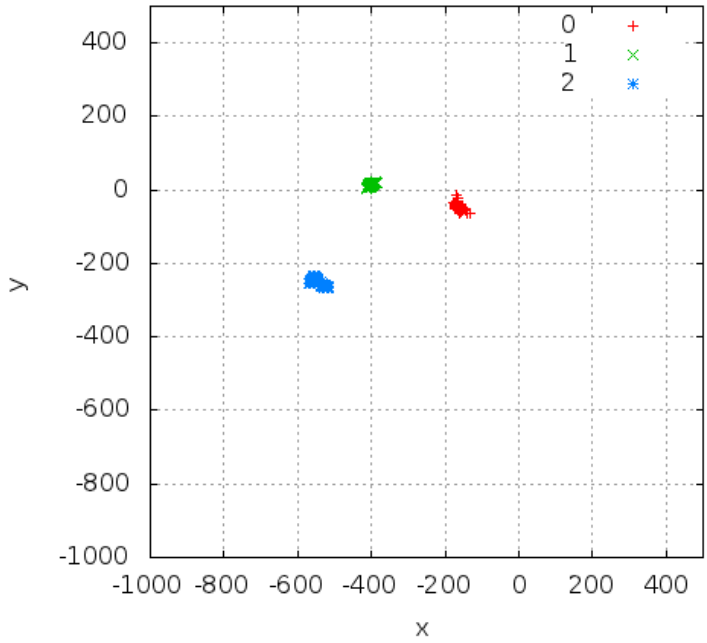

Abbildung C.37.: Versuch “still” bei 4 Meter Entfernung zur Kamera

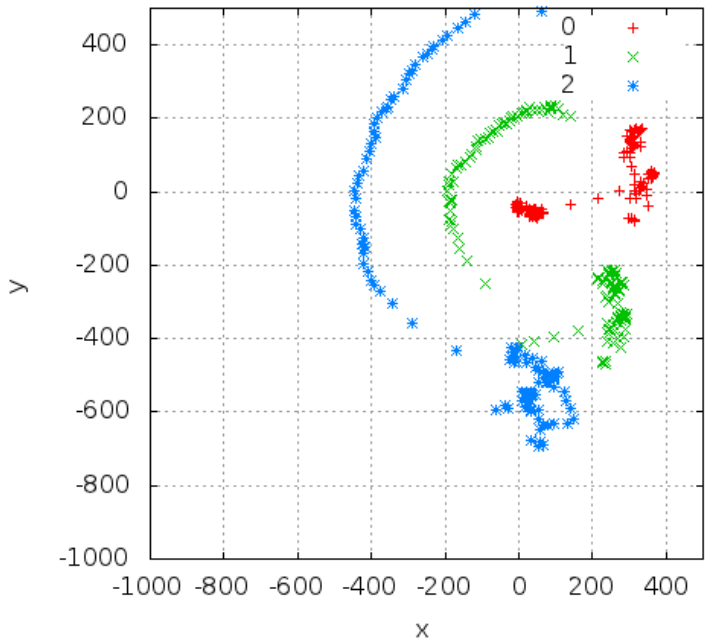

Abbildung C.38.: Versuch “slow” bei 4 Meter Entfernung zur Kamera

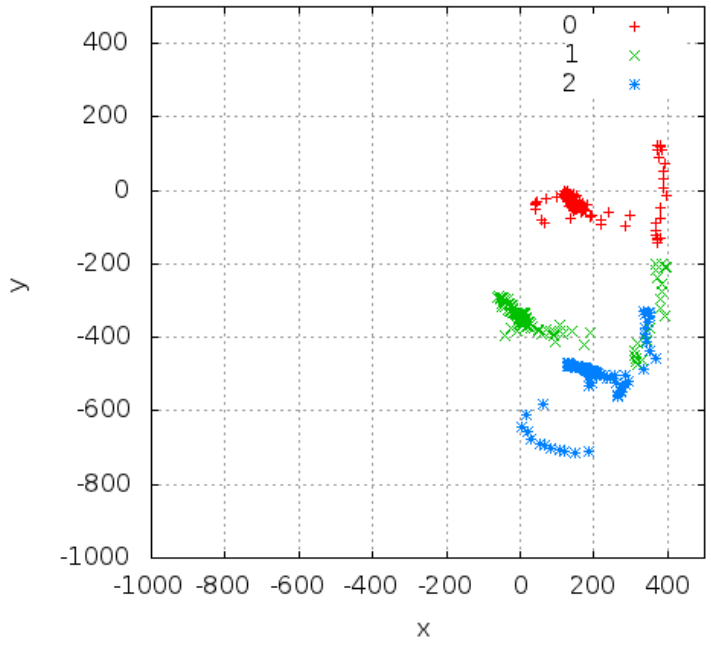

Abbildung C.39.: Versuch “medium” bei 4 Meter Entfernung zur Kamera

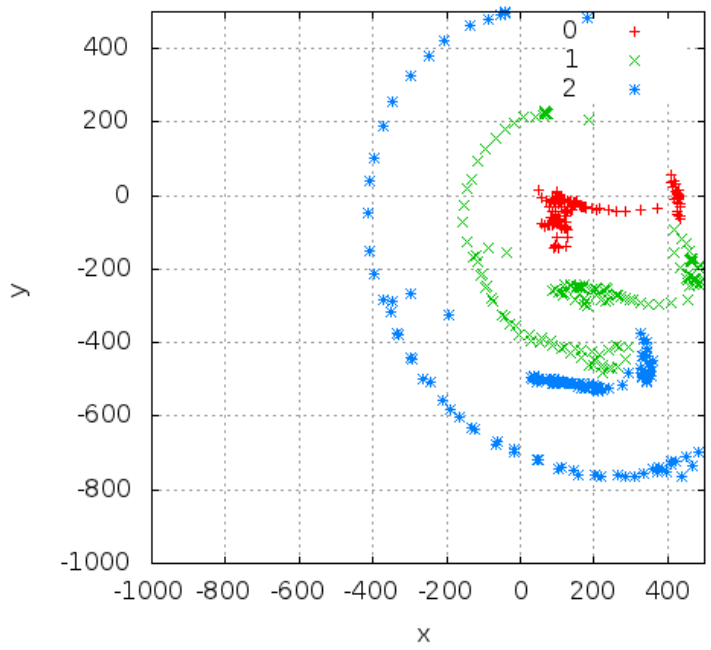

Abbildung C.40.: Versuch “fast” bei 4 Meter Entfernung zur Kamera

Abbildungsverzeichnis

Glossar

API Application Programming Interface - Programmierschnittstelle

AR Augmented Reality - ”Erweiterte Realität”

ARToolkit Augmented Reality Toolkit - Softwarebibliothek für AR

Beamforming Verfahren zur Lokalisierung einer Audioquelle mittels eines Mikrofon-Arrays

Bullet Physics Softwarebibliothek für physikalische Simulationen

Confidence Value Gibt die Wahrscheinlichkeit für die Korrektheit eines Datensatzes an

CPU Central Processing Unit - Hauptprozessor eines Rechners

GPL GNU Public License eine Lizenz für “freie” Software

Hertz Anzahl von regelmäßigen Wiederholungen pro Sekunde

HMD Head Mounted Display - Am Kopf befestigte Anzeige

IR Infrarot

Marker Im Falle von ARToolkit: quadratische kleine Fläche mit speziellem Muster, welches eine Erkennung ermöglicht

Middleware Softwareschicht zur Verbindung zweier Komponenten

OpenCV Open ComputerVision

OpenGL Open Graphics Library - Freie Softwarebibliothek zur Darstellung dreidimensionaler Szenen

PCL Point Cloud Library - Bibliothek zur Verarbeitung von Point-Cloid Daten

Point Cloud engl. “Punktwolke”, Datenstruktur zur Speicherung von vielen Punkten in einem dreidimensionalen Raum

PrimeSense Referenzimplementierung der gleichnamigen Firma einer Infrarot-Tiefenkamera

RAW Daten im RAW Format sind nicht bearbeitet. Es handelt sich um Rohdaten, z. B. von einem Sensor

RGB Rot Grün Blau

Smartphone Mobiltelefon mit Computerfunktionatiät

Software Engineering Fachgebiet, welches Verfahren für Design, Implementierung und Modifizierung von Software beinhaltet

Spielcontroller kleines Gerät zur Bedienung / Steuerung einer Spielkonsole

Stream (Daten-)Strom

Structured Light Coding Verfahren zur Berechnung der Tiefendaten

Stylus Eingabestift für elektronische Geräte, wie z.B. Smartphones oder Tablet-PC’s

Tablet-PC Rechner im “Tablett”-Format, alle Komponenten sind schmal hinter einem Bildschirm verbaut

VR-Handschuh Virtual-Reality Handschuh, der auf Grund von Lage- und Krümmungssensoren Bewegungen der menschlichen Hand aufzeichnet

Wrapper Softwarekomponente, die eine Datenstruktur in einer eigentlich fremden Umgebung nutzbar macht

XBox 360 Spielkonsole von Microsoft

Stichwortverzeichnis

Versicherung über Selbstständigkeit

Hiermit versichern wir, die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt zu haben.

Thorsten Hillebrand verfasste Kapitel 1 und Abschnitte 3.1, 3.3, 4.3, 4.4, 5.1.4, 5.1.5, 5.1.6, 5.3.2, 5.3.3, 6.3.
Patrick Tschackert verfasste Kapitel 2 und Abschnitte 3.2, 3.4, 4.1, 4.2, 4.5, 5.1.1, 5.1.2, 5.1.3, 5.2, 5.3.1, 6.1, 6.2.

Hamburg, 23. August 2011	
Ort, Datum	Unterschrift

Hamburg, 23. August 2011	
Ort, Datum	Unterschrift