HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

swiss science center
TECHNORAMA

Kinect Bodyscanner

Bachelorarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Frühjahrssemester 2011

Autoren:	Felix Egli und Michael Schnyder
Betreuer:	Prof. Dr. Markus Stolze, HSR, Rapperswil
Partner:	Swiss Science Center Technorama, Winterthur
Gegenleser:	Prof. Eduard Glatz, HSR, Rapperswil
Experte:	Markus Flückiger, Zühlke Engineering AG, Schlieren

Impressum

Kontakt
Felix Egli, felix.egli@hsr.ch
Michael Schnyder, michael.schnyder@hsr.ch

Druckdatum, 16.06.2011

Erklärung

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben haben.

Rapperswil, den 16.06.2011

Felix Egli **Michael Schnyder**

Danksagung

Für die Unterstützung während der Bachelorarbeit möchten wir folgenden Personen einen besonderen Dank ausrichten:

Prof. Dr. Markus Stolze für seine Unterstützung und sein wertvolles und konstruktives Feedback.

Markus Hofmann fürs Testen des ersten Paperprototypen und den wertvollen Input.

Thorsten Künnemann und Michel Junge vom Technorama für die Unterstützung bei der Entwicklung des Ausstellungsobjekts.

Daniela Meier und Ramona Rudnicki für das Korrekturlesen der Bachelorarbeit.

Abstract

Im Rahmen dieser Bachelorarbeit sollte ein Ausstellungsstück für die Sonderausstellung „Der vermessen(d)e Mensch“ im Technorama definiert und mit Hilfe geeigneter Technologien entwickelt werden. Dies mit dem Ziel, es anschliessend in den Ausstellungsbetrieb zu integrieren. Aus diesem Grund musste in diesem Projekt besonderes Augenmerk auf die Erfüllung von nicht-funktionalen Qualitätskriterien wie Verständlichkeit und Stabilität gelegt werden.

Die Microsoft Kinect-Technologie, welche ursprünglich für die Gebärdensteuerung der XBox entwickelt wurde, ermöglicht zum ersten Mal die kostengünstige Verarbeitung von Tiefenbildern. Diese Tiefeninformationen können über das inoffizielle OpenNI-Framework aus dem Kinect-Sensor ausgelesen werden. Auf Basis dieser Möglichkeiten wurden unterschiedliche technische Prototypen erstellt, welche jeweils auf Umsetzbarkeit und Resultatgenauigkeit geprüft wurden. Zusammen mit dem Technorama, der Umfang eines ersten Prototyps definiert werden. Ein wichtiger Bestandteil der Arbeit war der Test des erstellten Prototyps im Technorama. Hierbei wurde speziell auf die Verständlichkeit und Stabilität der Applikation geachtet.

Zusätzlich zum Kinect-Sensor wird auch eine Waage für die Personenmessung beigezogen. Diese Hardware wird über das GoIO-Framework angesprochen. Als Entwicklungssprache kam .NET mit C# zum Einsatz. Die grafische Umsetzung wurde mit WPF realisiert. Für die Überwachung der Lösung wurden PerformanceMeter eingesetzt. Diese stellen die Performance-Daten über die in Windows integrierte Schnittstelle auch anderen Applikationen zur Verfügung.

Es wurde ein funktionsfähiger und benutzerfreundlicher Prototyp erstellt, welcher die gesetzten Anforderungen an den Prototypen erfüllt. Durch den Benutzertest im Technorama und Langzeit- und Performancetests konnte dies verifiziert werden. Es sind nur noch wenige, bereits dokumentierte Anpassungen nötig, bis das Ausstellungsobjekt in die Ausstellung integriert werden kann.

Management Summary

Ausgangslage

Das Swiss Science Center Technorama in Winterthur führt regelmässig Sonderausstellungen durch. Für die aktuelle Sonderausstellung „Der vermessen(d)e Mensch“ war das Technorama auf der Suche nach einem ergänzenden Ausstellungsstück und sehr motiviert dieses Ausstellungsstück entwickeln zu lassen.

Im Rahmen dieser Bachelorarbeit sollte dieses Ausstellungsstück definiert und mit Hilfe geeigneter Technologien entwickelt werden, mit dem Ziel dieses in den Ausstellungsbetrieb zu integrieren. Aus diesem Grund musste in diesem Projekt besonderes Augenmerk auf die Erfüllung von nicht-funktionalen Qualitätskriterien wie Verständlichkeit und Stabilität gelegt werden.

Vorgehen/Technologien

Die Microsoft Kinect-Technologie, welche ursprünglich für die Gebärdensteuerung der XBox entwickelt wurde, ermöglicht zum ersten Mal die kostengünstige Verarbeitung von Tiefenbildern. Diese Tiefeninformationen können über das inoffizielle OpenNI-Framework aus dem Kinect-Sensor ausgelesen werden. Auf Basis dieser Möglichkeiten wurden unterschiedliche technische Prototypen erstellt, welche jeweils auf Umsetzbarkeit und Resultatgenauigkeit geprüft wurden. Aufbauend auf dieser Umsetzungsevaluation konnte, zusammen mit dem Technorama, der Umfang eines ersten Prototyps definiert werden. Ein wichtiger Bestandteil der Arbeit war der Test des erstellten Prototyps im Technorama. Hierbei wurde speziell auf die Verständlichkeit und Stabilität der Applikation geachtet.

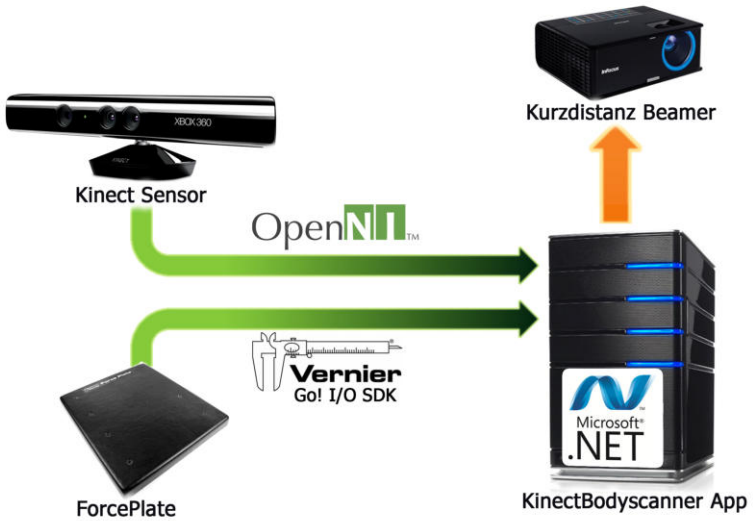


Abbildung (1) Übersicht Architektur

Zusätzlich zum Kinect-Sensor wird auch eine Waage für die Personenmessung beigezogen. Diese Hardware wird über das herstellerspezifische GoIO-Framework angesprochen. Das OpenNI- wie auch das GoIO-Framework, wurde durch einen zusätzlichen Hardware Abstraktionslayer komplett gekapselt, um die Austauschbarkeit jeder dieser Komponenten zu gewährleisten. Als Entwicklungssprache kam .NET mit C# zum Einsatz. Die grafische Umsetzung wurde mit WPF realisiert. Für die Überwachung der Lösung wurden PerformanceMeter eingesetzt. Diese stellen die Performance-Daten über die in Windows integrierte Schnittstelle auch anderen Applikationen zur Verfügung.

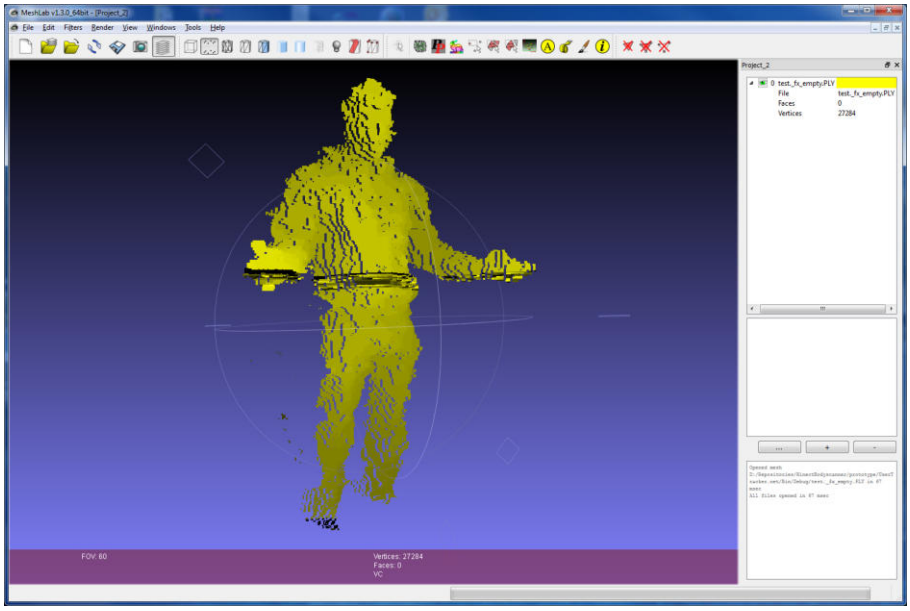
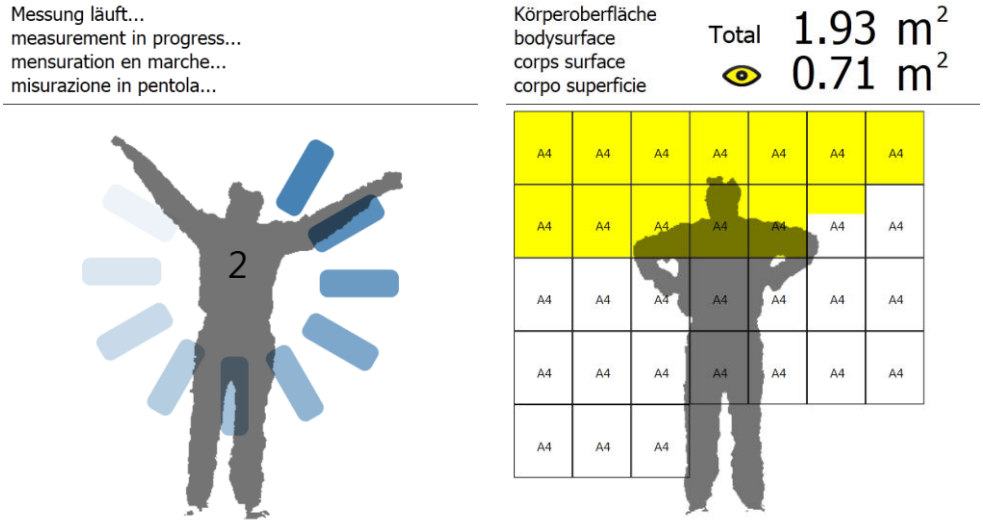


Abbildung (2) Darstellung der verarbeiteten PointCloud

Ergebnis

Es wurde ein funktionsfähiger und benutzerfreundlicher Prototyp erstellt, welcher die gesetzten Anforderungen an den Prototypen erfüllt. Durch den Benutzertest im Technorama und Langzeit- und Performancetests konnte dies verifiziert werden. Es sind nur noch wenige, bereits dokumentierte Anpassungen nötig, bis das Ausstellungsobjekt in die Ausstellung integriert werden kann. Die Ausstellung läuft bis Ende 2012.



(a) Messung einer Person

(b) Oberflächenanzeige mit DIN-A4 Blättern

Abbildung (3) Resultatanzeige

Inhaltsverzeichnis

Teil I

Technischer Bericht

1 Einführung

1.1 Problemstellung, Vision

Das Technorama startet im Frühjahr 2011 die Ausstellung „Der vermessene Mensch“. Als Teil dieser Ausstellung wollte das Technorama ursprünglich einen kommerziellen 3D Scanner nutzen, um die Körpermasse von Besuchern zu erheben und Informationen wie Körpervolumen, Oberfläche und Grösse anzuzeigen. Aufgrund der hohen Kosten kommerzieller Laser-Scanner wurde dieser Plan vom Technorama verworfen. Aus dieser Situation ergab sich die Gelegenheit für die HSR eine Bachelorarbeit auszuschreiben in der die Verwendbarkeit des Microsoft Kinect Sensors für die 3D Analyse von Besuchern getestet wird.
Der Kinect Sensor wurde von Microsoft für die Gebärden-Steuerung von XBox Spielen entwickelt. Seit Anfang 2011 sind für diesen Sensor Open Source Libraries verfügbar, welche es erlauben die 3D Informationen an einem angeschlossenen PC auszulesen.

1.2 Aufgabenstellung, Ziele

Für das Technorama soll prototypisch ein Ausstellungsobjekt entwickelt werden, welches dem Besuchern Informationen zu seinem Körper anzeigt.
Die hierfür notwendigen Informationen sollen mittels einem Microsoft Kinect Sensor (und ggf. weitere technische Mittel wie Waage, Distanzmesser) gesammelt werden.
Für das Ausstellungsobjekt sind gute Verständlichkeit der dargestellten Informationen, einfache Bedienbarkeit und hohe technische Robustheit wichtige Eigenschaften. Das Ausstellungsobjekt soll so ausgelegt sein, dass im Regelfall Technorama Besucher im Alter von 7 bis 70 Jahren in der Lage sein sollen ohne Lesen einer Anleitung die Funktion des Ausstellungsobjektes zu erfassen und innerhalb von 4 Minuten eine aus ihrer Sicht bereichernde Interaktion mit dem Ausstellungsobjekt abschliessen können. Besucher sollen, soweit sinnvoll, zur korrekten Bedienung des Systems angeleitet werden und auf die vom System getroffene Annahmen hingewiesen werden. Weiterhin kann auf Möglichkeiten zur Verbesserung der Resultate hingewiesen werden (Manipulationssicherheit). Das Ausstellungsobjekt soll zudem für den Dauerbetrieb ausgelegt sein. Dies bedeutet, zum Beispiel, dass 8 Stunden Betrieb ohne Absturz überstanden werden, und das Objekt durch Schalten der zentralen Stromzufuhr im Technorama an- und abgeschaltet werden kann (niedriger Wartungsaufwand im Betrieb).
Es ist wahrscheinlich, dass das System welches im Rahmen der BA erstellt wird nicht alle diese Eigenschaft vollständig erreichen kann. Es sind aber trotzdem im Projektablauf geeignete Massnahmen zu planen um den Stand des Systems bezüglich Verständlichkeit, Bedienbarkeit, und Robustheit verfolgen und dokumentieren zu können. Hierbei dürfen

auch Priorisierungen getroffen werden. So ist zum Beispiel eine umfangreiche grafische Umsetzung nicht zentral. Die Darstellung der Daten, sowie das Körpers kann in der ersten Ausbaustufe minimal gehalten werden. Diese Vereinfachungen sollten aber nicht die grundsätzliche Machbarkeit des System in anderen Bereichen in Frage stellt. Der Erreichungsgrad des Systems bezüglich der Kriterien Verständlichkeit, Bedienbarkeit, und Robustheit ist vorgängig und in Absprache mit dem Betreuer zu operationalisieren und der Stand bei Abgabe der BA zu dokumentieren.

1.2.1 Durchführung

Mit dem HSR-Betreuer finden in der Regel wöchentliche Besprechungen statt. Zusätzliche Besprechungen sind nach Bedarf durch die Studierenden zu veranlassen.
Da das Technologierisiko relativ hoch ist (z.B. kein offizielles SDK's zu Kinect) werden die Zielsetzungen der Arbeit während dem Projekt in Absprache mit dem Dozenten laufend angepasst.
Alle Besprechungen sind von den Studenten mit einer Traktandenliste vorzubereiten und die Ergebnisse in einem Protokoll zu dokumentieren das dem Betreuer E-Mail zugestellt wird.
Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. An Meilensteinen gemäss Projektplan sind einzelne Arbeitsresultate in vorläufigen Versionen abzugeben. Über die abgegebenen Arbeitsresultate erhalten die Studierenden ein vorläufiges Feedback. Eine definitive Beurteilung erfolgt aufgrund der am Abgabetermin abgelieferten Dokumentation. Die Evaluation erfolgt aufgrund des separat abgegebenen Kriterienkatalogs in Übereinstimmung mit den Kriterien zur BA Beurteilung. Es sollten hierbei auch die Hinweise aus dem abgegebenen Dokument „Tipps für die Strukturierung und Planung von Studien-, Diplom- und Bachelorarbeiten“ beachtet werden.

1.2.2 Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen. Die zu erstellenden Dokumente sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollten den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Die Dokumentation ist vollständig auf CD/DVD in 2 Exemplaren abzugeben. Zudem ist eine kurze Projektresultatdokumentation im Wiki von Prof. Stolze zu erstellen dies muss einen Link auf einen YouTube Video enthalten welche das Resultat der Arbeit dokumentiert.

1.3 Umfeld der Arbeit

1.3.1 Rahmenbedingungen

1.3.1.1 Aufwand

Die zwei Studenten Felix Egli und Michael Schnyder bearbeiten gemeinschaftlich die Aufgabenstellung. Es wird mit einem Aufwand von durchschnittlich ca. 21h pro Person und Woche gerechnet. Diese Zahl ergibt sich aus der Rechnung: 12 ECTS-Punkte à 30h Aufwand pro ECTS-Punkt. Die Arbeit dauert 17 Wochen (14 Wochen während dem Semester, 1 Woche Osterferien, 2 Wochen während den Semesterferien). Das Wochenpensum entspricht demnach 360h / 17 Wochen $\approx$ 21h/Woche. Lohnkosten sind über die gesamte Projektdauer keine vorgesehen.

1.3.1.2 Betreuung, Partner

Die Arbeit wird von Herrn Dr. Prof. Markus Stolze, Partner am Institut für Software IFS an der Hochschule Rapperswil, betreut. Bei Fragen rund um die Ausstellungsstücke stehen dem Projektteam folgende Kontaktpersonen beim Swiss Science Center Technorama zur Seite:

- Herr Thorsten Künnemann (Direktor)
- Herr Michel Junge (Ausstellungsleiter)

1.3.1.3 Wichtige Termine

Datum	*Termin*
11. April 2011	Präsentation der Möglichkeiten die sich mit Kinect bieten, den Körper auszumessen und Fixierung der definitiven Aufgabe (Requirements Workshop)
18./19. Mai 2011	Prototypentest im Technorama
17. Juni 2011	Abgabe der BA

Tabelle (1.1) Wichtige Termine

1.3.2 Infrastruktur

1.3.2.1 Hardware

Gerät	*Beschreibung*
2x Workstations	Windows 7 x64 Englisch, HSR-Domain
1x MS Kinect	OpenNI Driver (V. 1.0.0 Build 25)
1x Vernier ForcePlate	V. 2.41.0.0
1x HW-Taster für serieller Port	Selbst gebaut

Tabelle (1.2) Hardware

1.3.2.2 Entwicklungsumgebung

Komponente	*Beschreibung*	*Version*
IDE	Visual Studio 2010 Ultimate	10.0.30319.1
SDK	GoIO Treiber & dll's für ForcePlate	2.41.0.0
SDK	Microsoft .NET Framework	4.0
SDK	Primesense OpenNI	1.0.0.25 (Unstable Build for Windows)
OpenNI-Plugin	Kinect Plugin for OpenNI (SensorKinect)	5.0.0, unstable 07.01.2011
OpenNI-Middleware	Primesense NITE-Middleware	1.3.0.18 (Unstable Build for Windows)
Library	LibUsbDotNet	2.2.8
Tool	MeshLab	1.3.0
Tool	Tortoise SVN	1.6.10
Tool	Ankh SVN	2.1.101.29.17
Tool	Enterprise Architect	8.0.863.13
Tool	Adobe Photoshop CS5 (Trial)	CS5
Tool	Paint.Net	3.5.6
Tool	NDepend (Evaluation-Licence)	3.8.1.57.05
Tool	Resharper (Evaluation-Licence)	5.1.3000.12
Tool	Process Explorer	14.12
Tool	.NET Memory Profiler (Evaluation-License)	3.5
Dokumentation	MiKTeX	2.8
Dokumentation	Office 2007 Suite	10
Dokumentation	TeXnicCenter	2.0 Alpha 3 Build 1118
Dokumentation	JabRef (für bibtex)	2.6
Dokumentation	Sandcastle Help File Builder	1.9.3.0
Dokumentation	Sandcastle - Documentation Compiler	2.6.1062.1

Tabelle (1.3) Entwicklungsumgebung

1.3.2.3 Organisation

System	*Beschreibung*
trac	Wiki, Taskverwaltung und Sourcecode-Browser
Visual-SVN	SVN-Server (in Active Directory von Michael Schnyder integriert)

Tabelle (1.4) Organisations-Systeme

1.3.3 Glossar

Für einen kompakten Glossar siehe „Glossar auf Seite 139“.

1.3.4 Abkürzungsverzeichnis

Abkürzungen jeweils sind unterhalb der Seite ausgeschrieben. Siehe auch Abkürzungsverzeichnis auf Seite 141

1.3.5 Abgrenzungen

Nachfolgend Bereiche oder Funktionalitäten welche bei der Durchführung der Arbeit keine Relevanz besitzen. Somit werden die im folgenden genannten Funktionen explizit von der fertigen Arbeit nicht verlangt.

1.3.5.1 Fertig entwickeltes Ausstellungsobjekt

Es war bereits zu Beginn des Projekts absehbar, das die Entwicklung eines voll funktionstüchtigen Ausstellungsobjekt mitsamt der Erfüllung aller Anforderungen nicht möglich sein würde. Desshalb ist dies bereits in der Aufgabenstellung explizit ausgeschlossen worden.

1.3.5.2 Grafische Umsetzung

Die grafische Umsetztung soll sich auf einem minimalen Level befinden. Ziel ist ein Prototyp in welchem die Angaben verständlich anzeiegt werden. Animationen, etc. können in einem zweiten Schritt hinzugefügt werden.

1.3.5.3 Personas

Es wurde in Absprache mit dem Betreuer festgelegt, dass keine Personas erstellt werden. Vielmehr wurde der Fokus auf die technische Realisierbarkeit der Software gelenkt.

1.4 Vorgehen, Aufbau der Arbeit

Die Arbeit ist in mehrere Phasen und Teilaufgaben aufgeteilt. Einen ersten Teil nimmt die Einarbeitung in die Kinect-Technologie ein. Die Grundlage für die Forschung mit der Kinect gab das erste Meeting mit Herrn Junge und Herrn Künnemann vom Technorama. Erste Ideen wurden bereits diskutiert. Danach begann die Einarbeitung in die Technologie und das Erarbeiten von weiteren Ideen (siehe *Evaluation* auf Seite 17). Diese wurden mittels geeigneter Prototypen (siehe *Prototypen* auf Seite 67) auf technische Realisierbarket und Genauigkeit getestet. Die Lösung Kinect-Spezifischer Probleme war oft schwierig. Die Technologie und das OpenNI-Framework war neu und so standen oft keine verlässlichen

Informationen zur Verfügung. Es wurde in dieser ersten Phase der Arbeit Risiko-getrieben gearbeitet. Auftauchende Risiken wurden umgehend mit dem Erstellen von Prototypen minimiert. Zeitgleich bedeute es Aufwand, sich in die Thematik des „Vermessen(d)en Menschen“ einzulesen um Ideen für das Ausstellungsobjekt zu sammeln. Die Ergebnisse dieser Phase konnten wir in einem zweiten Meeting mit dem Partner besprechen und erste Vorschläge für ein Ausstellungsobjekt präsentieren. Aus dieser Liste von Ideen und Möglichkeiten wurde dann zusammen mit dem Partner ein Ausstellungsobjekt kreiert, welche die technischen Möglichkeiten ausschöpft. Bereits während dem Meeting konnten sich die Teilnehmer auf ein Storyboard einigen. Die Definition des Ausstellungsobjekts leitete gleichzeitig das Ende der ersten Phase ein.

In der zweiten Phase wurde der definierte Prototyp des Ausstellungsobjekts implementiert. Die Implementation berücksichtige dabei die Resultate des früh durchgeführten Paperprototype-Test und die technischen Prototypen, welche in der ersten Phase erstellt wurden. Mittels zwei Architekturprototypen konnte die finale Architektur ermittelt werden. Diese Architektur berücksichtige dabei den Kinect-Sensor wie auch die Waage und abstrahierte die verwendeten APIs. Für das UI wurden die ersten Controls entwickelt und der Benutzungsprozess entworfen. Die Entwicklung war begleitet durch ständige Benutzertest der Entwickler, da zu Beginn nur so die Interaktion mit dem Kinect-Sensor getestet werden konnte. Später konnten jedoch für Laufzeittests Mock-Objekte verwendet werden. Der Abschluss dieser Phase bildete ein zweitägiger Systemtest bei welchem das Ausstellungsobjekt unter realen Bedingungen getestet wurde (siehe *Auswertung der Interviews* auf Seite 105).

Im Anschluss an den Benutzertest standen abschliessende Arbeiten an. Die detaillierte Auswertung brachte einige Unschönheiten zum Vorschein, welche die Darstellungsform der Resultate betrifft. Insgesamt verlief der Test erfolgreich, es konnten viele spannende Szenen beobachtet werden. Beispielsweise entblösste sich ein Besucher teilweise um die sichtbare Oberfläche zu verringern. Beim Systemtest mussten wir jedoch auch feststellen, dass die Stabilität noch nicht optimal war. Darauf richteten wir in der verbleibenden Zeit ein besonderes Augenmerk und konnten so die Stabilitätsprobleme mittels Tests erfolgreich eingrenzen und lösen. Nach dem Projekt-Go des Partners werden die grafischen Elemente verbessert mit dem Ziel das Ausstellungsobjekt ab August 2011 bis Ende 2012 auszustellen.

2 Stand der Technik

2.1 Hardware: Kinect-Sensor

Der Kinect Sensor (ehemals bekannt unter dem Codenamen „Natal“) wurde von Microsoft Anfangs November 2010 auf den US-Markt gebracht. Der Sensor ist für die Verwendung mit der XBox-Spielkonsole von Microsoft gedacht. Tage später waren jedoch bereits die ersten Treiber entwickelt, welche es ermöglichten die Kamera, bzw. insbesondere den Tiefensensor unter Windows und Linux auszulesen. Am 3. Januar 2011 wurden bereits mehr als 10 Millionen Kinect-Sensoren verkauft[1]. Damit ist der Kinect-Sensor das am schnellsten verkaufte IT-Gadget weltweit und schlägt damit sogar die Verkaufszahlen von iPhone und iPad[2].

Mit der Verfügbarkeit der ersten Treiber wurde von der Opensource Community erste Frameworks geschaffen, um anderen Entwicklern einen leichteren Einstieg zu ermöglichen. Etwa zeitgleich stellte *Primesense* einen Treiber mitsamt einem Modul zur Skeleton-Erkennung für das OpenNI-Framework zur Verfügung. *Primesense* ist für die Referenzimplementation[3] verantwortlich auf welcher der Kinect-Sensor aufbaut. *OpenNI* bietet eine Schnittstelle für alle Möglichen Arten von *NUI*[4] und bietet Erweiterungsmöglichkeiten um z.B. aus einem Benutzer das Skelett zu identifizieren.

Abbildung (2.1) Kinect Sensor

1 Microsoft Press Release (http://www.xbox.com/en-US/Press/archive/2011/0308-Ten-Million-Kinects)
2 Guiness World Records, Press Release (http://community.guinnessworldrecords.com/_Kinect-Confirmed-As-Fastest-Selling-Consumer-Electronics-Device/blog/3376939/7691.html)
3 Von Prime Sense patentiert. Siehe http://www.freepatentsonline.com/20100118123.pdf
4 *Natural User Interface, engl. für Natürliche Benutzungsoberfläche*

2.1.1 Technische Details

Nachfolgend eine kurze Einführung in die im Kinect-Sensor vorhandenen Kameras, Sensoren und Steuerungsmöglichkeiten. Siehe nachfolgendes Bild mit allen Einzelteilen[1].

Abbildung (2.2) Kinect Sensor Einzelteile

2.1.1.1 Tiefensensor

Die ist das Herzstück des Kinect Sensors. Dieser Sensor besteht aus der Kombination einer 830nm Laser Diode und einem Tiefensensor CMOS welcher die reflektierten Infrarotstrahlen empfangen kann. Aus der Laufzeit kann so die Tiefe der einzelnen Punkte berechnet werden.

Aktuell sind über den Tiefensensor die folgenden Daten bekannt[2][3]:

1 Quelle: http://www.ifixit.com/Teardown/Microsoft-Kinect-Teardown/4066, Aufgerufen am 14.04.2011
2 http://openkinect.org/wiki/Hardware_info
3 http://openkinect.org/wiki/Protocol_Documentation

- Auflösung: 640x480 Pixels
- Tiefenbereich: 0.5m - 10m, Ideal 2m-2.5m
- Tiefengenauigkeit: 1cm

Ein Tiefenbild besteht somit aus 640x480 Pixeln mit je einer Tiefeninformation pro Pixel. Diese Tiefeninformation bewegt sich in einem Bereich von 50cm bis 1000cm mit einer Auflösung von 1cm. Pro Sekunde liefert der Kinect-Sensor 30 Tiefenbilder.

2.1.1.2 Kamerasensor

Nebst dem Tiefensensor besitzt die Kinect eine normale RGB-Kamera. Diese löst mit 640x480 Pixeln auf[1].

2.1.1.3 Audio

Der Kinect-Sensor ist mit 4 Mikrofonen ausgestattet. Mit diesen ist eine 3D Lokalisierung möglich. Der Zugriff auf die Audio-Daten ist jedoch noch mit keinem Framework möglich und es sind auch keine weiteren Informationen bekannt.

2.1.1.4 Neigesteuerung

Der Kinect-Sensor beinhaltet einen 3D-Neigesensor und einen Motor mit welchem der Neigungswinkel gegen vorne angepasst werden kann. Der Neigungswinkel beträgt zwischen -31° und + 30° relativ zum Standfuss.

2.1.1.5 LED-Steuerung

Das Integrierte LED kann ebenfalls angesteuert werden. Folgende Modus sind möglich:

- LED Ausgeschaltet
- Grün
- Rot
- Geld (Orange)
- Gelb (Orange) blinkend
- Grün
- Gelb (Orange)-Rot blinkend

1 Gemäss OpenKinect http://openkinect.org/wiki/FAQ#What_is_the_frame_size.2Fbitrate_of_the_rgb.2Fdepth.2FIR_stream_contained_in_the_isochronous_usb_tranfers_etc..3F

2.2 Aktuelle Frameworks

2.2.1 OpenNI

Mit OpenNI besteht ein Framework, welches unterschiedliche NUI Geräte ansprechen kann, sofern dafür Treiber vorliegen. Das OpenNI-Framework stellt die Schnittstelle zwischen Anwendungssoftware und Treiber für einzelne *NUI* Geräte zur Verfügung. Daneben lässt sich das Framework mit Middleware erweitern, um z.B. Skeleton Algorithmen einzubauen. Einer dieser Middleware Komponenten ist der Skeleton-Algorithmus NITE[1] von PrimeSense.

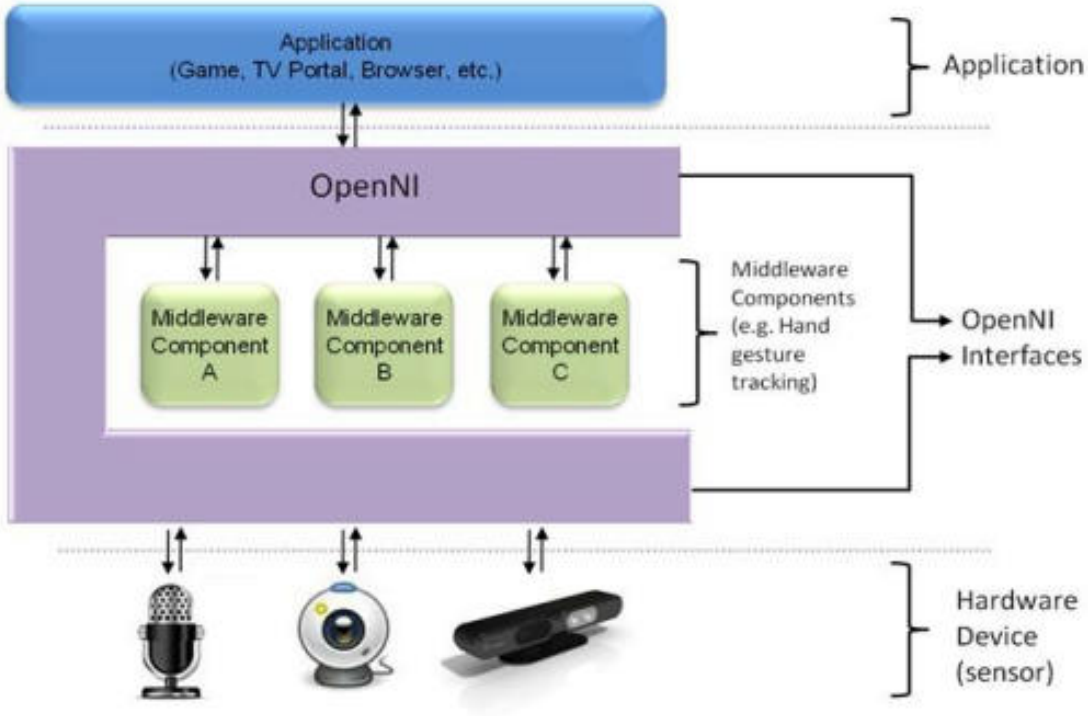


Abbildung (2.3) OpenNI Schema

Das OpenNI-Framework ist in C++ geschrieben und steht für alle gängigen Plattformen (Windows, Linux/Ubuntu, OSX) zur Verfügung. Für Windows besteht zusätzlich ein Wrapper in C#.

Für den Softwareentwickler stehen die folgenden Komponenten des OpenNI-Frameworks im Vordergrund:

- **Context**: Definiert des Umfeld des Sensors. Dies Beinhaltet z.b. die Auflösung der Sensoren, bzw. die Sicht auf die ganze Szene als solches.

1 NITE von PrimeSense http://www.primesense.com/?p=515

- **DepthGenerator**: Erzeugt für den Softwareentwickler ein kontinuierliches Signal welches aus den Dazen des Tiefensensors gespiesen wird.
- **UserGenerator**: Liefert Informationen zu aktuell erkannten Benutzern

Mit *NITE* sind in *OpenNI* User-Detection und -Tracking bereits einhalten. Auch einfache Gesten werden von der NITE-Middleware erkannt und können auf der Applikationsschicht direkt verwendet werden. Mit der NITE-Middleware kann OpenNI zudem so erweitert werden, dass zu einer erkannten Person die Gelenkpositionen errechnet werden.

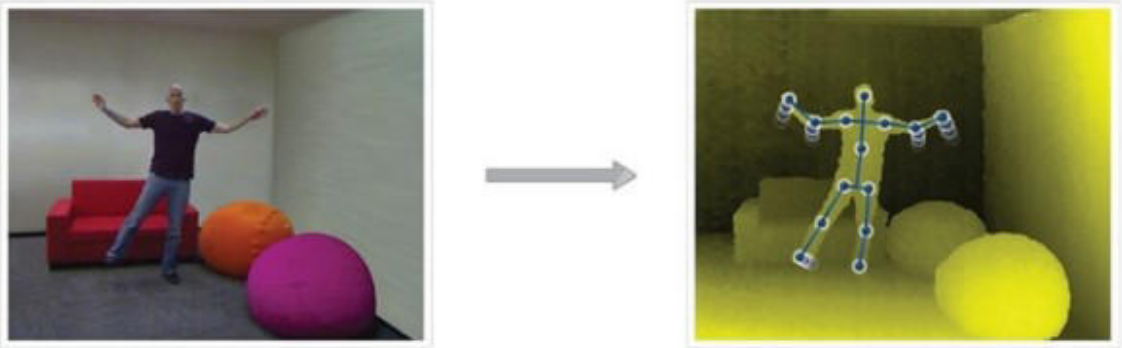


Abbildung (2.4) Funktionsbeispiel NITE Middleware

Wir haben uns auf Grund des grossen Funktionsumfangs und weiteren wichtigen Kriterien zur Verwendung dieses Frameworks entschieden (siehe *Kinect Framework* auf Seite 30).

2.2.2 libfreenect

Das Projekt „libfreenect“ beinhaltet Treiber und eine Schnittstelle um auf den Kinect-Sensor zuzugreifen. libfrenect ist aus OpenKinect enstanden und wurde von der OS-Comunity entwickelt. Es stehen Wrapper für C# und andere Programmiersprachen zur Verfügung. Diese Library abstrahiert jedoch nur die einzelnen Sensoren inner halb Kinect und stellt einen gemeinsamen Zugriffspunkt zu diesen zur Verfügung. Nicht enthalten sind User-Tracking, Gesture-Matching oder Informationen über einzelne Gelenke einer Person.

Das Projekt macht den Anschein, als ob es im Moment nicht weiterentwickelt wird. Eine Roadmap ist zwar vorhanden, jedoch ist gemäss github nur jeweils 1 Entwickler massgeblich an der Weiterentwicklung beteiligt[1].

1 Projektseite auf github.com: https://github.com/surflightroy/libfreenect

2.2.3 Microsoft Kinect SDK

Das offizielle Windows SDK für Kinect wurde von Microsoft am 20.02.2011 für den Frühling 2011 angekündigt[1]. Zur MIX2011 (12. April 2011 bis 16. April 2011 in Las Vegas) wurden weitere Details über den Umfang des offiziellen SDKs bekannt[2]:

> The latest advances in audio processing, which include a four-element microphone array with sophisticated acoustic noise and echo cancellation for crystal clear audio. Sound source localization for beamforming, which enables the determination of a sound's spatial location, enhancing reliability when integrated with the Microsoft speech recognition API. Depth data, which provides the distance of an object from the Kinect camera, as well as the raw audio and image data, which together open up opportunities for creating richer natural user interface experiences. Highly performant and robust skeletal tracking capabilities for determining the body positions of one or two persons moving within the Kinect field of view. Documentation for the APIs and a description of the SDK architecture. Sample code that demonstrates how to use the functionality in the SDK.

Aktuell (am 14.04.2011) wird vermutet[3], dass das Kinect SDK ab dem 16. Mai 2011 erscheinen soll. Dies ist leider aber bis jetzt (12.06.2011) nicht der Fall.

2.2.4 Aktuelle Arbeiten

Eine Grosse Anzahl von Arbeiten mit Kinect sind auf den folgenden zwei Webseiten zu finden:

- Kinect-Hacks.com (http://www.kinect-hacks.com/)
- KinectHacks.net (http://KinectHacks.net/)

1 The Official Microsoft Blog, http://blogs.technet.com/b/microsoft_blog/archive/2011/02/21/kinect-for-windows-sdk-to-arrive-spring-2011.aspx
2 http://research.microsoft.com/en-us/um/redmond/projects/kinectsdk/
3 Blog-Eintrag auf http://www.i-programmer.info/news/91-hardware/2303-microsofts-kinect-sdk.html

3 Evaluation

3.1 Einführung

Dieses Kapitel listet alle wesentlichen Entscheidungen und Konzepte auf, welche während der Arbeit erarbeitet wurden. Diese Entscheidungen betreffen im ersten Teil des Kapitels die Analyse und das Design des Ausstellungsobjekts.

Im zweiten Teil sind Entscheidungen aufgelistet, welche technischer Natur sind. Umsetzungsstechnische Entscheidungen bzw. Prototypen der spezifischen Problemstellungen finden sich im Kapitel Prototypen (siehe *Prototypen* auf Seite 67) und sind nicht Teil dieses Kapitels.

3.2 Entwicklung des Ausstellungsobjekts

Nachfolgende Grafik gliedert die untersuchten Ideen in ihrer zeitlichen und logischen Abhängigkeit (in Pfeilrichtung). Konzepte, welche in die Endlösung eingeflossen sind, sind grün eingefärbt.

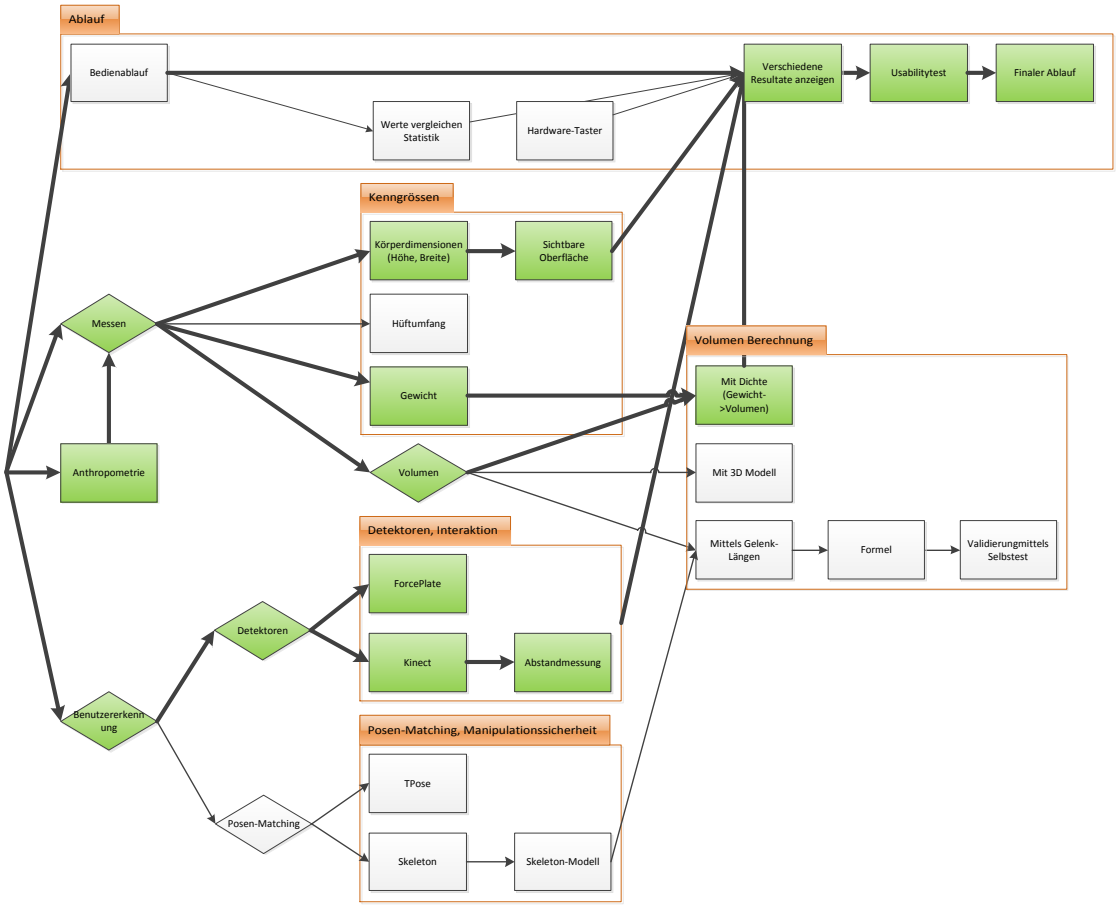


Abbildung (3.1) Übersicht der verfolgten Ideen, welche zum Ausstellungsobjekt führten

Leider konnten nicht alle Ideen in das Ausstellungsobjekt übernommen werden. Dies hat mehrere Gründe, welche in den nachfolgenden Seiten ausgeführt werden.

3.2.1 Ablauf

Der Bedienablauf wurde von den implementierten Funktionen beeinflusst. Es war jedoch von Beginn an klar, dass sich dieser Ablauf in einer Applikation mit mehreren Seiten manifestieren wird. So wurde bereits in einer ersten Fassung des Ausstellungsobjekts der

Ablauf anhand eines Paperprototypen getestet (siehe *Paper-Prototyp Nr. 1 - 14.4.2011* auf Seite 96). Kurzzeitig konnte mittels einem Hardware-Taster (siehe *Prototyp „ComPortButtonTest“* auf Seite 77) der Ablauf gesteuert werden. Dieser Hardware-Taster ist jedoch nicht in die Endlösung eingeflossen, da durch die Detektoren und automatische Umschaltung zwischen den Seiten eine manuelle Interaktion überflüssig wurde (siehe *Generelle State-Machine* auf Seite 59). Dies erleichterte die Bedienung durch den Benutzer und verminderte die Gefahr, dass der Taster nicht als Interaktionsmöglichkeit gesehen wird.

3.2.2 Kenngrössen

In der Gruppe Kenngrössen sind jene Grössen eines Menschen zusammengefasst, welche allgemein bekannt sind. Die Ideen bzw. Möglichkeiten haben folgende Quellen:

- Internes Brainstorming mit Verifikation anhand erstellter Prototypen
- Anschliessend externes Brainstorming im Technorama

Siehe auch Brainstorming (siehe *Internes Ideen Brainstoming* auf Seite 181) und Sitzungsprotokoll Technorama vom 11. April 2011 (siehe *Sitzungsprotokolle* auf Seite 185).

3.2.2.1 Körperdimensionen (Höhe, Breite)

Diese Funktion wurde mit dem Prototypen „HeightMeasureTest“ validiert. Wir konnten verifizieren, dass die Körperhöhe mit einer Toleranz von ca. 3% gemessen werden kann (siehe *Prototyp „HeightMeasureTest“: Messungen, Architekturprototyp* auf Seite 69). Die Abweichung war jedoch für das Technorama zu hoch. Die Messmethode konnte jedoch in der bestehenden Form für die Messung der sichtbaren Oberfläche verwendet werden.

3.2.2.2 Gesamte Körperoberfläche

Die gesamte Körperoberfläche wird auf Basis von Grösse und Gewicht einer Person berechnet. Hierfür gibt es verschiedene erprobte Formeln. Wir verwenden die Formel von Mosteller[1]:

$$S = \sqrt{\frac{L \times M}{3600}}$$

Dabei sind

- **S**: Körperoberfläche in m^2
- **L**: Körpergrösse in cm

1 http://de.wikipedia.org/wiki/Mosteller-Formel

- **M**: Körpermasse in

3.2.2.3 Sichtbare Körperoberfläche

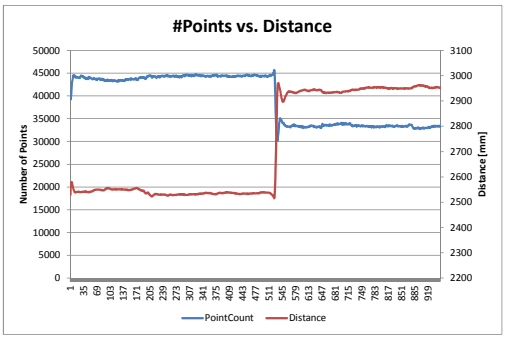
Die Sichtbare Oberfläche wird auf Basis von Körperhöhe und -breite und Auflösung das Tiefenbildes berechnet. Diese Berechnung erfolgt über mehrere Schritte:

1. Identifizieren der äussersten Punkte oben, unten, links und rechts. Hierbei steht der Benutzer am besten so breit und hoch wie möglich hin, damit für die weiteren Schritte möglichst viel Punkte verwendet werden können.
2. Aus diesen äussersten Punkten können die Breite und Höhe der Person berechnet werden.
3. Anzahl Punkte auf der X- und Y-Achse. Es werden die Punkte gezählt, welche die Person in der Breite und Höhe repräsentieren.
4. Mittels der Anzahl Punkte in vertikaler, bzw. horizontaler Richtung und der Höhe und Breite kann die Auflösung eines einzelnen Messpunkts erfasst werden.
5. Daraus lässt sich die Fläche eines einzelnen Pixels des Tiefenbildes im 3D-Raum berechnen.
6. Multipliziert mit der Anzahl Messpunkte, welche den Benutzer erfasst haben, entspricht dies der sichtbaren Oberfläche.

Diese Funktion wurde im „HeighMeasureTest“-Prototypen implementiert (siehe *Prototyp „HeightMeasureTest“: Messungen, Architekturprototyp* auf Seite 69) und ist in die Endfassung eingeflossen.

Die Schwachstelle dieser Idee ist die Anzahl Punkte, welche den Benutzer erfasst haben. Diese Anzahl wird durch die Distanz der Person zur Kamera beeinflusst. Steht eine Person weit von der Kamera entfernt, ist die Auflösung niedriger und die Angaben über Grösse und Höhe der Person sind somit ungenauer. Dadurch nimmt die Fläche eines einzelnen Pixels zu und die Ungenauigkeit steigt zusätzlich. In unseren Tests konnten wir jedoch diese Bedenken ausräumen.

Die folgenden Daten wurden bei einem Test aufgezeichnet, bei welchem die Testperson zuerst mit einem Abstand von ca. 2.5 m zum Kinect-Sensor stand. Danach verändert die Person Ihren Abstand auf 3 m.



(a) Verhältnis Anzahl 3D-Punkte vs. Abstand zum Sensor

(b) Fläche eines einzelnen Pixels

Abbildung (3.2) Veränderung Berechnungsgrundlage bei Abstandsänderung

Die oberhalb gezeigten Diagramme stellen die Veränderung der Anzahl Punkte bei ändernder Distanz zum Kinect-Sensor dar. Dadurch nimmt die Höhe und Breite eines einzelnen Messpunkts (=Pixel) zu. Dies ist auf der rechten Seite dargestellt. Die Veränderung der Anzahl Punkte und der Pixelgrösse egalisiert sich jedoch wieder. Die gerechnete Oberfläche bleibt während der kompletten Messdauer konstant.

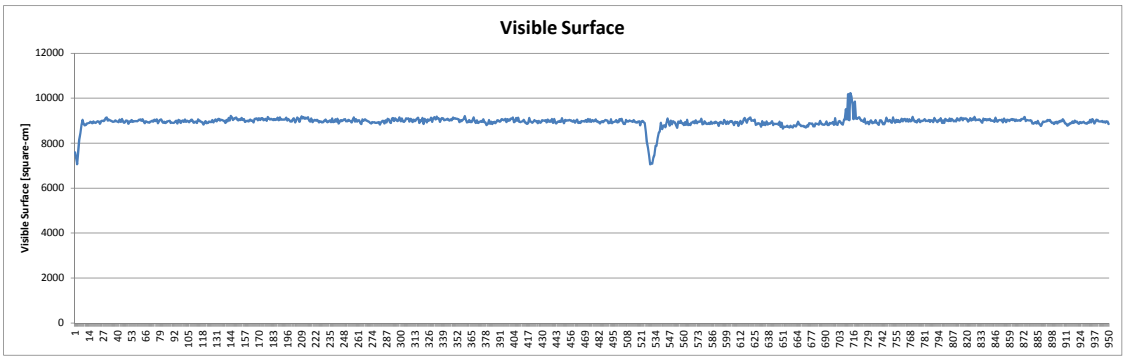


Abbildung (3.3) Die errechnete sichtbare Oberfläche bleibt über die ganze Aufzeichnung konstant.

3.2.2.4 Hüftumfang

Das Ziel dieser Idee war, eine Aussage über die Kleidergrösse einer Person treffen zu können. Alternativ hätte auch der Hüftumfang einen Anfang machen können. Diese Funktion wurde in einem eigenen Prototypen erfolgreich implementiert (siehe *Prototyp „FindHip“: Hüfte-Breite berechnen* auf Seite 74). In der Endfassung fand diese Funktion jedoch keine Anwendung.

3.2.2.5 Gewicht

Das Gewicht bildet die Grundlage für die Berechnung des Volumens über die Dichte. Die Daten hätten auch zur Berechnung vom BMI verwendet werden können. Daneben wird das Gewicht auch benötigt, um die Komplette Körperoberfläche zu berechnen (übersichtshalber nicht im Diagramm dargestellt). Für die Gewichtsdaten wurde eine Waage mit USB-Anschluss verwendet (siehe *Hardware zur Gewichtsmessung* auf Seite 33). Mittels zweier Prototypen (siehe *Prototyp „ForcePlateAPITest“: Waage ansprechen* auf Seite 73) wurde das Wissen über die Hard- und Software erarbeitet.

3.2.3 Anthropometrie

Wir bewegen uns mit unserer Arbeit im Bereich der Körpervermessung. Aus diesem Grund war es wichtig, nebst der Einarbeitung in die Kinect-Technologie auch ein möglichst grosses Wissen im Bereich der Anthropometrie[1] anzueignen.

3.2.3.1 Europamensch

Auf dnm Europamensch beziehen sich die von der deutschen Bundesanstalt für Arbeitsschutz und Arbeitsmedizin (Baua) herausgegebenen „Arbeitswissenschaftlichen Erkenntnisse - Forschungsergebnisse für die Praxis“[2]. In rund 170 Veröffentlichungen sind die wichtigsten Erkenntnisse aus der Arbeits- und Technikforschung lösungsorientiert für die betriebliche Praxis aufbereitet.

1 http://de.wikipedia.org/wiki/Anthropometrie
2 http://www.baua.de/de/Publikationen/AWE/Band3/AWE108.pdf?__blob=publicationFile&v=4

Körpermeßwerte des Europamenschen

Maß-Nr. (lt. Abb.)	Beschreibung des Maßes	Perzentile		
		5	50	95
1	Körperhöhe	1530	1719	1880
2	Augenhöhe	1420	1603	1750
3	Schulterhöhe	1260	1424	1570
4	Ellenbogenhöhe	960	1078	1190
5	Brustkorbtiefe (Brustbein bis Wirbelsäule)	170	215	250
6	Schritthöhe (entspricht dem Schneidermaß »im Schritt«)	709	816	890
7	Tibialhöhe	397	472	530
8	Hüftbreite	300	359	400

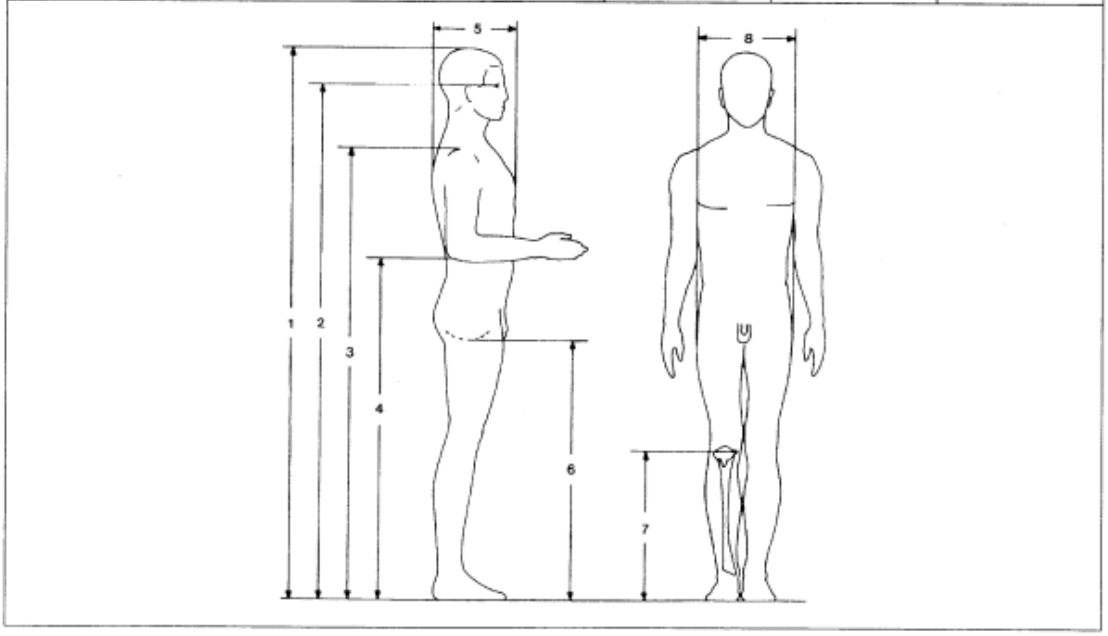


Abbildung (3.4) Der vermessene Europamensch

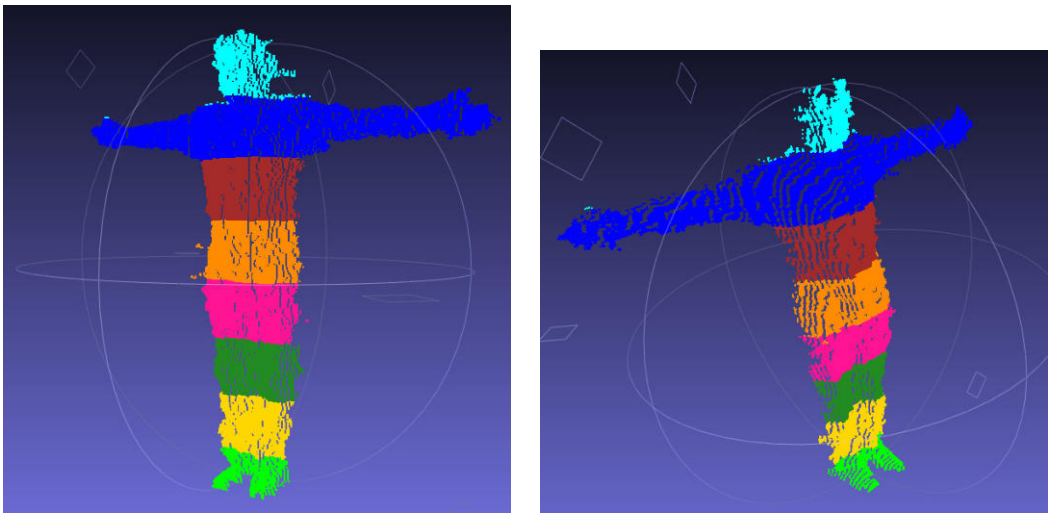
In diesen Veröffentlichungen wird auch auf die Anwendung der einzelnen *Perzentil*en eingegangen. Aufgrund internationalen praktischen Erfahrungen wird auch festgelegt, dass in den meisten Fällen das 5. und das 95. Perzentil ausgeschlossen werden kann. Weiter ist es nicht möglich, aus dem Perzentil*Perzentil* der Körpergrösse auf die Körpermasse zu

schliessen. Gemäss Baua besteht hier keine Korrelation.[1]

3.2.3.2 Körperproportionen beim Menschen

Im Bereich der *Allometrie* gibt es einige interessante Abhängigkeiten. Z.B. ist für uns die Beziehung der Körpergrösse zu weiteren biologischen Grössen interessant. Nachfolgend ein Beispiel der Beziehung zwischen Körpergrösse und Kopfgrösse:

Der Körper eines erwachsenen Menschen entspricht im Allgemeinen dem 7 bis 8-Fachen seiner Kopfhöhe.[2].



(a) Seitlich von rechts **(b)** Sicht von links oben

Abbildung (3.5) Aufteilung des Menschen in 7.5-Kopf-Einheiten als 3-D Modell

Dabei lässt sich auch gut beobachten, dass z.B. die Hüfte eines Mannes im Normalfall ein bisschen breiter als sein Kopf ist. Gleichzeitig wird durchschnittliche Hüfte einer Frau mit zwei Kopfbreiten angegeben.

1 http://www.baua.de/de/Publikationen/AWE/Band3/AWE108.pdf?__blob=publicationFile&v=4

2 http://de.wikipedia.org/wiki/K%C3%B6rperproportionen, oder http://maramattiaart.blogspot.com/2009/01/lesson-11-linear-perspective-in-drawing.html und http://www.suite101.com/content/drawing-human-sketches-a9848

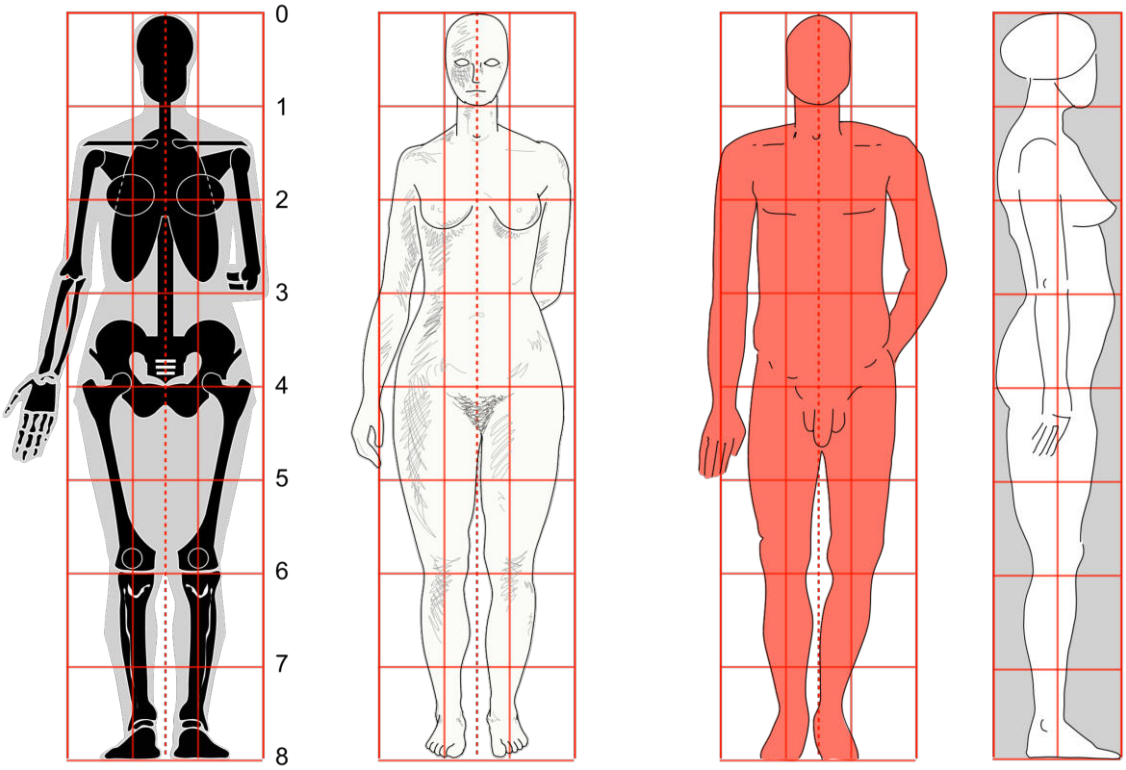


Abbildung (3.6) Körperproportionen

3.2.3.3 Gestenerkennung

Nebst diesem Wissen über die Anthropometrie erachteten wir auch das Wissen um Gesteerkennung im 3D Raum für sinnvoll. All dieses Wissen ermöglichte uns erst Abschätzungen bezüglich der Komplexität und Genauigkeit einer möglichen Umsetzung.

Bei unserer Recherche sind wir auf verschiedene Papers gestossen, welche sich mit der 3D Verarbeitung von *PointClouds* befassen. Weiter sind wir auf Papers gestossen, welche sich mit der Vermessung des Menschen befassen. Erstere Papers waren für uns sehr wichtig, um den Implementationsaufwand aufwändigerer Funktionen besser abschätzen zu können.

Es hat sich gezeigt, dass die meisten Papers nicht direkt in die Arbeit einfliessen können. Es ist jedoch spannend, welche Forschungsergebnisse auf dem Bereich der *NHI*[1] gemacht wurden. Zu beachten gilt es aber, dass die meisten Arbeiten nicht auf rohen 3D Daten basieren, sondern diese aus mehrere 2D Bildern errechnet werden. Dies wird sich mit der Verbreitung von Kinect und weiteren Devices bestimmt ändern. Jedoch besitzt der Kinect Tiefensensor wiederum nicht eine identisch grosse Genauigkeit wie reine 3D Kameras.

1 *Natural Human Interaction, engl. für Natürliche Benutzungsinteraktion*

- Real Time Hand Tracking And 3D Guesture Recognition For Interactive Interfaces Using HHM [KEA03]
- Hand Pose Estimation for Vision-based Human Interface [UMIO01]
- Human Body Model Acquisition and Tracking Using Voxel Data [MTHC03]
- Human Motion Analysis: A Review [AC97]
- Mathematical Model of Human Body Surface Area [SZGY00]
- Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review [PSH97]

Es hat sich durch diese Arbeiten gezeigt, dass eine eigene Umsetzung von Gesten den Umfang der Arbeit sprengen würde. Auf den sehr interessanten Bereich wurde deshalb verzichtet und dieser wurde darum auch nicht Teil eines Entwurfs.

3.2.4 Volumen Berechnung

Für die Volumenberechnung einer Person wurde zuerst versucht, diese ohne Waage zu realisieren. Ziel war es, mit den Daten aus der Kinect das Volumen zu bestimmen. Die ersten zwei Ansätze führten jedoch nicht zum gewünschten Erfolg. Die Erkentnisse aus dem zweiten Ansatz konnten jedoch im finalen Ansatz einfliessen.

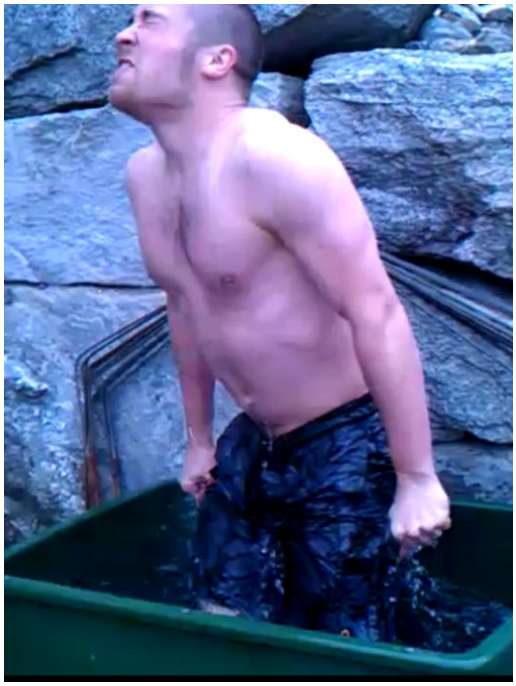
3.2.4.1 Mittels Gelenklängen

Die Idee, das Körpervolumen auf Basis des Skeletts zu berechnen, entstand beim Experimentieren mit dem Skeleton-Tracking des NITE-Frameworks.

Umsetzung: In einer ersten Phase galt es eine Formel zu entwickeln, welche die einzelnen Gelenklängen als Basis für die Volumenberechnung verwendet. Um dies in der Praxis auszuprobieren, haben wir uns mit einem Messband ausgemessen. Konkret mittels Kreisformel haben wir Gelenklängen und Gelenkumfang gemessen. Aus dem Gelenkumfang haben wir auf die Gelenkfläche geschlossen (dies ergibt einen kleinen Fehler, da die Glieder nicht rund sind). Nun wurde Gelenklänge mal Gelenkfläche gerechnet, um das Volumen eines Körperteils zu erhalten. Schlussendlich haben wir alle Körperteile zusammengezählt und daraus resultierte eine Annäherung an das Volumen unseres Körpers.

Validierung: Die ermittelte Formel wurde mit einem Test verifiziert. Um möglichst genaue Daten über das Volumen eines Menschen zu erhalten, führten wir einen Selbsttest durch. Ziel war es, eine Aussage über die Genauigkeit der entwickelten Volumenformel zu erstellen. Der Selbsttest führten mit dem Teammitglied Felix Egli durch. Anhand der verdrängten Wassermasse in einer Tonne konnte das Volumen sehr genau ermittelt werden.

(a) Volumen der Tonne ausrechnen



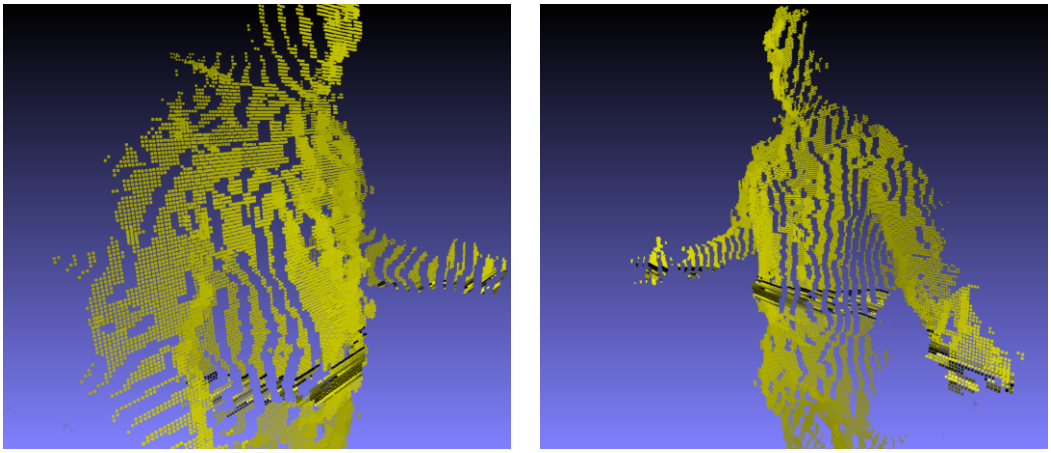
(b) Durchführung

Abbildung (3.7) Validierung der Volumenformel

Felix Egli verdrängte in eisiger Kälte rund 76 l Wasser. Die von uns entwickelte Formel lag annähernd bei diesem Wert. Trotzdem mussten wir die Idee verwerfen, da die Vermessung der einzelnen Körperteile zu kompliziert geworden wäre. Das gemessene Volumen konnten wir jedoch für die Validierung des Berechnungsversuchs „Mit 3D Modell“ verwenden.

3.2.4.2 Mit 3D Modell

Mit einer weiteren Idee versuchten wir aus den 3D-Daten (*PointCloud*) eines Benutzers das Volumen zu errechnen. Als Basis diente uns die Berechnung der sichtbaren Oberfläche (siehe *Sichtbare Körperoberfläche* auf Seite 20). Aufgrund dieser Berechnungen konnten wir die Fläche eines einzelnen Pixels bzw. der Vorderseite eines 3D-Punkts berechnen. Diese diese einzelnen Flächen verwendeten wir zur Bildung von Türmen auf der Z-Achse. Als Boden dieser Türme diente uns eine Ebene, welche durch den hintersten Punkt (auf der Z-Achse) der *PointCloud* bestimmt war. Addiert man die Volumen all dieser Türme, kann das Volumen approximiert werden.



(a) Sicht von Links **(b)** Sicht von Rechts

Abbildung (3.8) Visualisierung Pixel der Pointcloud eines Benutzers

Die genaue Umsetzung scheiterte leider an den folgenden Punkten.

- Die Position der Grundfläche ist sehr schwierig zu bestimmen. Es gibt starken Noise in der Z-Achse, welcher das Problem verschlimmert.
- Die Auflösung ist möglicherweise ungenau.

Wir konnten in einigen, nicht repräsentativen Fällen eine Aussage zum Volumen und damit auch zum Gewicht machen. Jedoch mussten wir dazu die Position der Grundfläche für die „Türme“ stark anpassen.

3.2.4.3 Herleitung über Gewicht

Die einfachste Möglichkeit, das Volumen eines Menschen zu bestimmen, besteht darin, dieses mittels Dichte und Gewicht zu berechnen. Wir gehen bei unserer Berechnung von einer spezifischen Dichte von 1 g/1 cm^3 aus. Diese Annahme konnten wir unter anderem durch unseren Selbstversuch validieren. Sie lässt sich ebenso über bekannte Grössen des menschlichen Körper validieren. Über 60% des menschlichen Körpers bestehen aus Wasser [Rov92]. Maximal 15% bis 22% bestehen aus Fett [Bai91]. Der Rest sind grösstenteils Muskeln und Knochen. Die Dichte von Fett und Muskeln können mit 1.1 g/1 cm^3 (Muskeln) und 0.9 g/1 cm^3 (Fett) angenommen werden[KM77][Wil77]. Der Anteil von Muskeln oder Fett ist jedoch zu klein, um die Dichte des Menschen stark beeinflussen zu können.

3.2.5 Detektoren, Interaktion

Die Bedürfnisse bezüglich der Interaktion mit dem Ausstellungsobjekt veränderten sich während dem Entwicklungsprozess. Ziel war immer, die notwendige Interaktion möglichst

gering zu halten. Dies führte uns zur Entwicklung von einfachen „Detektoren“ (siehe *Detektoren* auf Seite 90), welche z.B. Personen auf der Waage oder im Sichtbereich der Kinect-Kamera verarbeiten können. So startet beispielsweise der Messprozess, sobald eine Person von der Waage und vom Kinect-Sensor erkannt wird.

3.2.5.1 Abstandsmessung

Die Abstandsmessung wurde eingeführt, um diejenige Person sicher detektieren zu können, welche aktuell auf der Waage steht. Sollte sich die aktive Person von der Wage entfernen und sich eine andere Person bereits auf der Waage befinden, kann dies von der Waage nicht detektiert werden. Auch die OpenNI kann nicht aktive Personen ermitteln, da die Anzahl Personen im Sichtbereich möglicherweise identisch geblieben ist. Einzig mit der Abstandsmessung kann diejenige Person ermittelt werden, welche am nächsten zur Waage steht. Sollte ein solcher Wechsel stattfinden, muss die Messung neu angestossen werden (siehe *Generelle State-Machine* auf Seite 59).

3.2.6 Posen-Matching, Manipulationssicherheit

Zu Beginn der Arbeit war die Anforderung gegeben, auf Manipulationssicherheit zu achten. Hierfür haben wir mehrere Erkennungsmethoden spezifiziert und teils auch entwickelt. Im Gespräch mit dem Technorama wurde schnell klar, dass diese Erkennungsmethoden nicht benötigt werden. Vielmehr war sogar erwünscht, dass die Besucher das System manipulieren können.

3.2.6.1 TPose

Der Benutzer sollte während der kompletten Messzeit in der T-Pose verbringen. Damit kann vorallem beim Messen der Höhe und Breite sichergestellt werden, dass die gemessenen Werte nicht (z.B. durch Hochstrecken der Arme) manipuliert werden (siehe *Eigenentwicklung „T-Pose“* auf Seite 32).

3.2.6.2 Skeleton

Durch die Gelenkinformationen hätte während dem Messprozess sichergestellt werden können, dass der Benutzer die Gelenke in einem definierten Körper hält. Dieser Körper hätte anhand Körpergrösse und einzelnen Körperteillängen bestimmt werden können. Dies wurde jedoch nicht implementiert, da für die Gelenkinformationen das Skeleton-Modell benötigt worden wäre. Dieses wiederum wurde aus verschiedenen Gründen nicht eingesetzt (siehe *Skeleton Tracking* auf Seite 30).

3.3 Technische Entscheide

3.3.1 Kinect Framework

Eine Evaluation über das Kinect-Framework, welches wir in unserer Arbeit verwenden möchten, wurde in Absprache mit Herrn Dr. Prof. Stolze nicht detailliert durchgeführt. Das Framework hat nur vorübergehende Relevanz, da geplant ist, auf das offizielle Framework von Microsoft zu wechseln, sofern zeitlich möglich. So wurde auf Basis der folgenden Kriterien das Framework „OpenNI“ ausgewählt.

- Verfügbarkeit .NET-Wrapper
- Kompatibilität mit Hardware
- Umfang (Personenerkennung, Gesten, etc.)
- Stabilität
- Bekanntheitsgrad, Community
- Preis, Kosten

Mit „OpenNI“ konnten diese Anforderungen erfüllt werden. OpenNI stammt vom selben Entwickler, welcher das Referenzdesign für Kinect entwicklet hat. So ist die Kompatibilität automatisch gegeben. Auch punktet dieses Framework mit dem Umfang, welcher bereits mehrere C#-Beispiele umfasst. Das Framework beinhaltet auch bereits Komponenten zur Benutzer- und Gestenerkennung.

3.3.2 Skeleton Tracking

Das OpenNI-Framework bietet mit *NITE* (siehe *Funktionsbeispiel NITE Middleware* auf Seite 15) eine Komponente an, welche zu einer erkannten Person in real-time ein so genanntes Skeleton liefert. Ein Skeleton definiert sich durch die Punkte der Gelenke des Benutzers und besteht meistens aus den folgenden Punkten (die Gelenknamen gelten sinngemäss auch für die linke Seite):

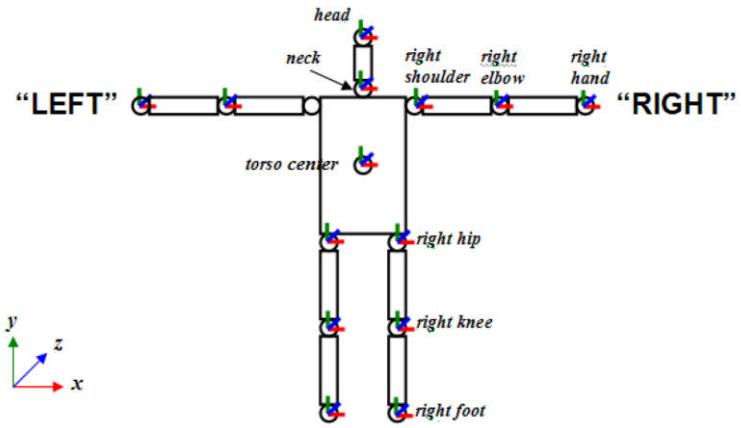


Abbildung (3.9) Erkannte Gelenke von NITE nach Kalibrierung [Pri10]

Für die Erkennung der Gelenke ist der Benutzer angehalten, während 1-5 s die folgende Pose einzunehmen [Pri10]:

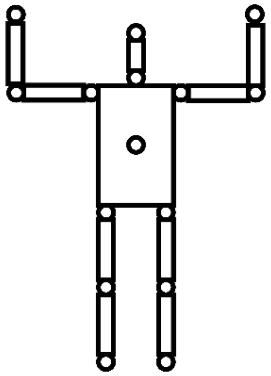


Abbildung (3.10) Kalibrierungspose

Wir haben uns gegen die Verwendung des integrierten Skeleton-Tracking entschieden. Dies hatte die folgenden Gründe:

3.3.2.1 Erkennungsgeschwindigkeit und false-positive Erkennungen

Der Skeleton-Algorithmus hatte in unseren Tests mehrmals Schwierigkeiten, eine Person innerhalb von 5 s zu erkennen. Gleichzeitig konnten wir auch feststellen, dass die gleiche Person sehr schnell in <2 s erkannt wurde. Es gab auch Fälle, in welchen die Person erst nach massiv mehr als 5 s erkannt wurde oder die Erkennung ganz ausblieb. Für uns war das ein Killer-Kriterium, das Skeleton-Feature nicht zu verwenden. Der Benutzer möchte nicht unnötig warten.

3.3.2.2 Verwendung der Punkte

Die erkannten Punkte konnten wir nicht direkt zur Messung verwenden. So entspricht der Abstand der Punkte nicht dem realen Schulterabstand, da die Gelenke für den Schulterabstand verwendet werden. Vielmehr interessieren die äussersten Punkte zur Vermessung. Bei der Körpergrösse ergibt sich die selbe Problematik.

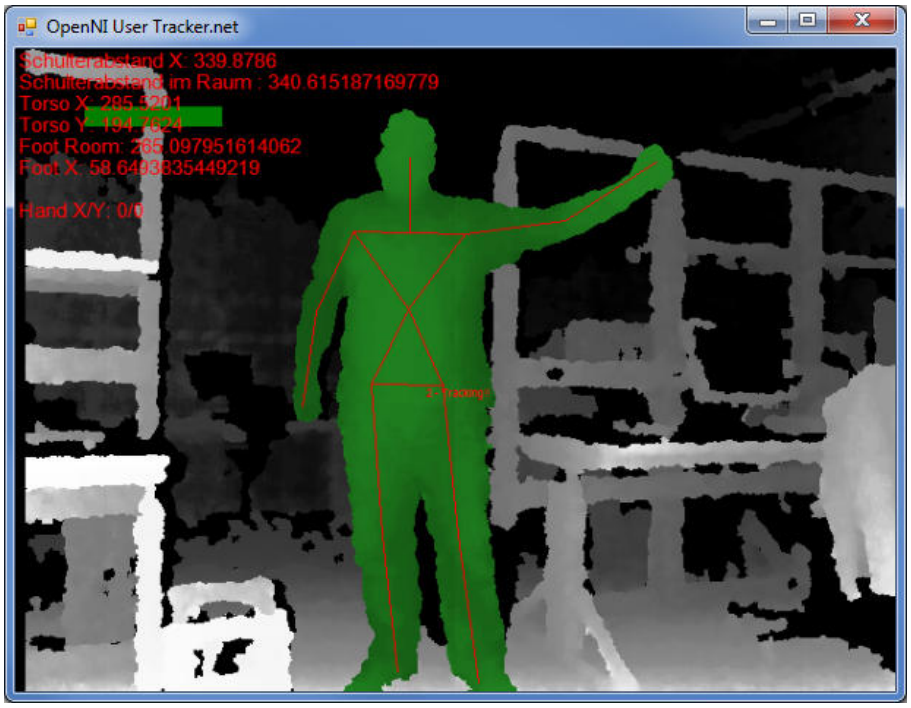


Abbildung (3.11) Skeleton-Tracking im erweiterten OpenNI User Tracker

3.3.2.3 Eigenentwicklung „T-Pose“

Die Probleme mit dem Skeleton-Tracking Algorithmus führten uns zu einer Eigenimplementation der so genannten „T-Pose“. Diese Erkennungsmethode wurde anhand der folgenden Kriterien implementiert, wurde jedoch im finalen Ausstellungsstück **nicht berücksichtigt**.

- Person muss äusserste die Punkte links / rechts parallel zu Kinect halten (=gleicher Z-Abstand)
- Beide äussersten Punkte müssen zudem auf gleicher Höhe sein (Y-Achse)
- Armspannweite sollte ungefähr der Höhe der Person entsprechen

Weiterentwicklung: Die Formel könnte mittels Datamining verbessert werden. Durch die Verarbeitung von 50-100 T-Pose Tiefenbildern könnten die benötigten Parameter genauer definiert werden.

3.3.3 Hardware zur Gewichtsmessung

Für die Erfassung des Gewichts wurde eine Waage notwendig. Die Hauptkriterien zur Beschaffung waren wie folgt:

- Auslesen der Messdaten auf dem PC
- Wenn Möglich C# Programmierschnittstelle
- Preis
- Robustheit

Folgende Möglichkeiten zur Beschaffung einer Waage, welche den oben genannten Kriterien entspricht, wurden eruiert.

3.3.3.1 Consumer Gerät mit WLAN-Adapter

Es gibt verschiedenen Heimanwender-Waagen, welche einen WLAN Adapter besitzen. Diese können das gemessene Gewicht ins Internet stellen. Diese Schnittstelle hätte mit ein wenig Protokoll-Sniffing verwendet werden können. Preislich eine sehr interessante Variante, jedoch hinsichtlich Robustheit und Implementationsaufwand klar ein Nachteil.

3.3.3.2 Industriewaage

Die Industriewaage kam auf Grund des hohen Preises nicht in Frage. Bei den Schnittstellen waren mindestens die RS323-Schnittstelle vorhanden.

3.3.3.3 Vernier ForcePlate (für Schulzwecke)

Im Technorama werden bereits mehrere dieser ForcePlates eingesetzt. Mittels der C#-Api kann auf die sehr präzise Waage zugegriffen werden. Diese lässt sich mit einem Adapter via USB an den PC anschliessen. Mit einem Preis von unter CHF 500.- ist diese Waage sehr preiswert. Bei unseren Test konnten wir uns von der Robustheit der Waage überzeugen. Sie hielt Belastungen von über 3500N stand.

4 Resultate

4.1 Zielerreichung

Ausformuliert auf (siehe *Resultate* auf Seite 113)

4.2 Persönliche Berichte

4.2.1 Felix Egli

4.2.1.1 Projektverlauf

Unsere Idee der Bachelorarbeit generierte sich kurz vor Ende der Studienarbeit. Die Microsoft Kinect für die XBox war nun in den Verkaufsgeschäften erhältlich. Bereits standen erste Videos, von selbst entwickelten Programmen mit der Kinect, im Internet. Angetrieben von diesen Ideen, suchten wir schnellstmöglich einen Anwendungsfall für die Kinect. Denn damit wollten wir uns, in unserer Bachelorarbeit befassen. Wir wurden fündig: Statt einer geplante Microsoft-Surface-Applikation für das Technorama, durften wir dank unserem Betreuer Dr. Prof. M. Stolze einen Prototypen für ein Ausstellungsstück in der Ausstellung „Der vermessen(d)e Mensch“ im Technorama erstellen.

Nach einem ersten Meeting mit den Verantwortlichen im Technorama wurde klar, dass wir uns in einer ersten Phase mit der Technologie und den verfügbaren Libraries herumschlagen mussten. Dies, um abklären zu können, was mit dieser Technologie überhaupt möglich ist. Das war sehr interessant und aufregend. Denn die Dokumentation die bis zu diesem Zeitpunkt verfügbar war, war äusserst lückenhaft. Auch konkrete Berichte im Internet, über andere Projekte waren zu Beginn nicht vorhanden. Aber die Anzahl Berichte stieg mit dem Start unserer Bachelorarbeit täglich an. Dies hatte aber auch den Nachteil, dass wir nur langsam vorwärts kamen. Kaum hatten wir etwas entwickelt, gab es schon etliche Webseiten über das Thema.

Dennoch war unsere Arbeit speziell: Wir benutzen die Kinect für etwas, für das sie nicht direkt entwickelt worden war. Der Fokus unserer Arbeit liegt ja nicht darin Gesten zu erkennen, sondern Menschen auszumessen. Wir konnten mit Hilfe des OpenNI-Framework innert kurzer Zeit Erfolge erzielen. Mit diesem Framework war es möglich Personen erkennen und auf deren 3D-Punkte zugreifen (sogenannte PointCloud). Aus diesem ersten Prototyp enstanden umgehend weitere Ideen. Diese Ideen und Erkenntnisse wurden meist umgehend in Prototypen umgesetzt. Eine weitere wichtige Messgrösse, zur Berechnung von wichtigen Körpereigenschaften, war das Gewicht einer Person. Damit wir darüber verlässliche Aussagen machen konnten, haben wir uns eine Waage angeschafft, welche auch

mittels Programmcode angesprochen werden kann.

Als erstes wollten wir zur Berechnung des Körpervolumens eine komplizierte Formel entwickeln, mit der es möglich sein sollte, über markante Grössen auf das Körpervolumen schliessen zu können. Doch diese Idee mussten wir nach einem Selbsttest verwerfen. Der Selbsttest führte wir mit einer Tonne voll kaltem Regenwassen durch. Bestimmt eine ungeahnte Aktion während einer Bachelorarbeit in Informatik! Um das Körpervolumen trotzdem berechnen zu können schaffte die Waage uns hier Abhilfe. Mit einer theoretischen Dichte von 1:1 von Masse zu Volumen, konnten wir einfach das Gewicht als Volumen verwenden.

Als wir genügend Erfahrungen mit der Vermessung des Menschen gesammelt haben und auch erste Prototypen dazu fertig gestellt hatten, machten wir uns ein weiteres Mal auf den Weg ins Technorama. Wir präsentierten die Prototypen den verantwortlichen Personen. Diese konnten sich nun ein grobes Bild über die Möglichkeiten der Kinect machen. Zu unserem Erstaunen waren Ihnen die Messresultate zu ungenau. Als Produkt dieses Meetings konnte ein Storyboard für ein konkretes Exponat erstellt werden, welches Aufschluss über den Ablauf/Interaktion mit diesem gibt.

In der nächsten Phase hiess es nun, die erfassten Anforderungen umzusetzen. Die nichtfunktionalen Anforderungen wie Stabilität und Robustheit mussten immer im Hinterkopf behalten werden, denn bei einem Ausstellungsstück sind diese sehr wichtig.

Nach etlichen Tests, welche aus Benutzer-, Unit- und Systemtests bestanden, konnte der Prototyp auf einen sicheren Stand gebracht werden, mit welchen wir den Prototyp dann auch zwei Tage im Technorama zeigen konnten. Diese Gelegenheit nutzen wir gleichzeitig um die Benutzer zu befragen, sie zu beobachten oder zum Benutzen unsers Ausstellungsstücks zu motivieren. Dies war besonders Interessant, da wir echte Benutzer hatten und die meisten auch Bereit waren konstruktives Feedback zu liefern.

Nun ging es darum, letzte Fehler auszumerzen und die Applikation technisch auf einen stabilen Stand zu bringen. Ein Memoryleak konnte repariert werden und mittels Endurance-Tests konnten wir die Stabilität über längere Zeit sicherstellen.

Mit dem guten Gefühl, einen guten Prototypen zu haben, machten wir uns an die weniger lustige Arbeit, das Dokumentieren. Hier stellte sich als schwierig heraus, die ganzen Abhängigkeiten der Entscheidungen welche in der ersten Phase gemacht wurden, nochmals sauber auf Papier zu bringen.

4.2.1.2 Rückblick

Rückblickend darf ich sagen, dass wir wieder zu spät mit der Dokumentation angefangen haben! Es war zwar besser als in der Semesterarbeit, aber immer noch zu spät.

Diesmal war ich besonders froh darüber, dass wir einen realen Kunden als Partner hatten. Wir konnten sehr selbständig mit diesem kommunizieren. Der Umstand, dass wir das Geschaffene später im Technorama ausstellen dürfen, gab uns zusätzlich Ansporn.

Der Höhepunkt für mich war der Test im Technorama. Wir konnten vom realen „Endkunden“ unser Produkt testen lassen und hier sehr viele wertvolle Erfahrungen sammeln. Auch die Offenheit der Technorama Mitarbeiter, die uns diese zwei Tage unterstützt haben war einfach genial!

Schön war auch, dass wir C# einsetzen konnten. Denn für die Auswertungen von 3D-Punkten konnten wir z.B. mit LINQ verwenden. Dadurch mussten wir zum finden bestimmte Punkte (z.B. der Höchste Punkt) nicht etliche Schlaufen programmieren, sondern konnten uns stärker auf das eigentliche Problem fixieren.

4.2.1.3 Gelerntes

Ich habe am meisten in der Projektplanung gelernt. Gerade wenn es um ein noch nicht klares Ziel gibt, auf welches man hinarbeitet. So muss man sich trotzdem Ziele setzen um schnell vorwärts zu kommen. Interessant war auch, dass wir dazu unser eigenes Vorgehen entwickeln und benutzen konnten. Zudem bei einem solchen Vorgehen, lernte ich die Protokolle der vergangen Sitzungen sehr schätzen. So konnte über bereits besprochene Angelegenheiten zurückgegriffen werden.
Auch im Bereich der Anthropometrie konnte ich so manches dazulernen. Denn dieses Thema interessiert mich auch ausserhalb des Projekts und ist nicht Standartbestandteil einer Bachelor-Arbeit.
Zudem konnte ich die .NET Technologien (WPF, MVVM, LINQ usw.) in einem realen Projekt anwenden und wertvolle Erfahrungen sammeln. Die geforderte Stabilität war bis jetzt für mich in diesem Umfang noch nicht so ein grosses Thema. Hier war ich in dieser Arbeit gefordert. Wir konnten bereits ein Memory Leak identifizieren und beheben.

4.2.1.4 Fazit

Es war ein sehr abwechslungsreiches Projekt. Ein realer Kunde war involviert und es wurde etwas entwickelt was später definitiv ausgestellt wird! Einen tollen Team-Partner, der weiss von was er spricht und was er tut! Der perfekte Mix, für ein wunderschönes Projekt!

4.2.2 Michael Schnyder

4.2.2.1 Projektverlauf

Mit der Verfügbarkeit der Kinect entstand von uns der Wunsch, in der Bachelorarbeit mit dieser Technologie zu arbeiten. Auf offenes Ohr stiessen wir bei unserem Betreuer, Prof. Dr. Markus Stolze. Durch seine Kontakte konnte auch mit dem Industrieparter, dem Swiss Science Center Technorama in Winterthur, ein erstes Meeting vereinbart werden. An diesem Meeting wurde der Grundstein unserer Arbeit gelegt: Das Technorama war sehr interessiert, ein Ausstellungsstück mit Kinect entwickeln zu lassen. In die kommende Sonderausstellung „Der vermessen(de) Mensch“ würde sich ein solches Ausstellungsstück sehr gut integrieren lassen. Erste Ideen im Bereich der Vermessung des Menschen wurden diskutiert. Noch konnten wir jedoch nicht abschätzen, welche Möglichkeiten die Technologie bietet - Die Opensource-Community hatte es gerade erst geschafft einen Treiber für den Tiefensensor zu entwickeln. Offizielle Angaben waren (und sind es bis jetzt) nicht vorhanden.

Zu Beginn unseres Projekt war also der Kinect-Sensor – gekauft im PC-Shop vis-a-vis für weniger als 200 Fr. Etwas verzögert folgte die Opencource-Community und der Hersteller der Kinect, Prime Sense. Durch die Opensource-Community und Prime Sense wurden Frameworks zur Verfügung gestellt, mit welchen die Kinect mit dem PC angesteuert werden kann. Darauf konnten wir aufbauen. Die Wahl des richtigen Frameworks und der Aufbau für das Verständnis von 3D Informationen bedurfte Zeit. Ziel des ersten Teils unseres Projekts war darum klar die Minimierung der technischen Risiken und die Einarbeitung in die Technologie. Während diesem Prozess konnten wir viele Ideen generieren, was man mit diese Technologie alles anstellen könnte. Dieser explorative Prozess war für das Gelingen des Projekts sehr wichtig, denn nur so konnten wir umsetzbare Anwendungsfälle für das Ausstellungsstück finden. Unsere Idee, mit der Kinect eine Person zu vermessen, war neu und bisher nicht erforscht. Dadurch waren keine Informationen über bisherige Arbeiten und Probleme, bzw. Lösungen vorhanden. So konnten wir durch die Verwendung einer Waage als zusätzliches Messinstrument gewisse Probleme umgehen. Nebst dem Suchen und Testen von möglichen Anwendungsfällen, eigneten wir uns Wissen im Bereich der Körpervermessung an. All dieses Wissen verwerteten wir in mehreren Prototypen. Ein Teil der Ideen haben es dann sogar in die Endlösung geschafft.

In einem Workshop mit dem Technorama konnten wir unsere bisherigen Prototypen und Ideen präsentieren. Wir mussten feststellen, dass die Genauigkeit der Höhenmessung nicht den Anforderungen des Technoramas entsprach. Auf Basis der bisherigen Prototypen und der Evaluation möglicher Funktionen konnten wir in einem äusserst kreativen Prozess den Umfang des zukünftigen Ausstellungsobjekts eruieren. Als Teil dieses Prozesses konnte sogar bereits der Bedienprozess anhand einem Storyboard definiert werden.

Durch das erarbeitete Wissen in der Kinect-Technologie und dem verwendeten Framework

konnte auf Basis von Prototypen eine Systemarchitektur entwickelt und getestet werden. Wir verfolgten das Ziel, den Zugriff auf die Kinect mittels OpenNI (Framework von Prime Sense) zu abstrahieren. So kann das Framework später durch das offizielle SDK von Microsoft ersetzt werden. Diese Architektur beinhaltet eine Hardware-Abstraktionsschicht für die Kinect und die Waage. Im Rahmen dieser Kapselung wurden auch mehrere Controls entwickelt. Das restliche UI basiert dabei auf dem MVVM-Pattern.

Nach intensiver Vorbereitung, testeten wir während zwei Tagen den Prototypen unseres Ausstellungsobjekts im Technorama. Dieser Test fand unter realen Bedingungen statt. Bis zu diesem Zeitpunkt war der Prototyp soweit entwickelt, dass die komplette Interaktion getestet werden konnte. Die benutzerfreundliche Bedienung ist ein sehr wichtiges Qualitätsmerkmal eines Ausstellungsstücks, welches es zu erfüllen galt. Dies validierten wir mit eigenen Beobachtungen und gezielten Interviews. So konnten wir den guten Eindruck der Besucher festhalten. Wir erlebten jedoch eine Überraschung: Nach rund 2h, oder ca. 20 Benutzungen stockte das System stark. So stark, dass es nicht mehr bedienbar war. Dieses Problem mussten wir unbedingt lösen, denn auch Robustheit ist bei einem Ausstellungsstück ein viel beachtetes Attribut.

Nach dem Benutzertest hatten wir noch rund 4 Wochen Zeit um die Arbeiten vorläufig abzuschliessen. Es war zu Beginn der Arbeit klar, dass es unmöglich sein wird, die Arbeit innert der geforderten Zeit fertig zu stellen. So konnten wir uns dann auch auf die wirklichen Arbeiten konzentrieren: Stabilität. Die Suche nach dem Problem erwies sich zum Glück einfacher als Gedacht: Durch weak-References blieb ein der Teil UI-Elemente im Memory bestehen und konnte vom Garbage-Collector nicht abgeräumt werden. Dies führte zu einem erhöhten Speicherverbrauch, welcher der Garbage-Collector immer häufiger versuchte zu optimieren. Natürlich ohne Erfolg, im Gegenteil: Durch die vielen Garbage-Collector-Aufrufe wurde das System ausgebremst. Die Probleme konnten bei der Wurzel gepackt und gelöst werden. Mit dem Finden der Probleme ist eine Monitoring-Komponente entstanden. Diese stellt einerseits dem Betriebssystem wichtige Performance-Daten zur Verfügung. Andererseits werden diese Daten gespeichert, was eine spätere Fehlersuche enorm erleichtert.

4.2.2.2 Rückblick

Das Projekt war herausfordernd. Wir waren zu Beginn stark auf uns selbst gestellt. Durch das Unwissen über die Technologie war es uns nicht möglich in dieser Zeit eine verlässliche Planung zu erstellen. Mit dem Ziel, in der Mitte der Zeit einen Prototypen zu präsentieren konnten wir dennoch eine Grobplanung erstellen. Wir arbeiteten explorativ, was mir sehr viel Spass bereitete.

Nach dem Zwischenmeeting mit dem Technorama war die Motivation noch höher als zu Beginn. Die Herren Künnemann und Junge verstanden es, die bisher erstellten Prototypen

in einen sinnvollen Prototypen zusammenzufügen, wenn auch nur auf dem Papier. Der Partner signalisierte da auch schon die Bereitschaft, den fertigen Prototypen in die Ausstellung aufzunehmen. Das spornte uns weiter an.

Am Benutzertest stellten wir fest, dass wir uns in die richtige Richtung bewegt hatten. Wir konnten mit grossem Stolz feststellen, dass das Ausstellungsstück auch während unseren Abwesenheiten ausgiebig genutzt und getestet wurde. Die gefunden Probleme konnten wir in der restlichen Zeit beheben, so dass nun nur noch grafische Anpassungen durchzuführen sind.

4.2.2.3 Gelerntes

Die Arbeit war sehr abwechslungsreich. Teil dieser Abwechslung waren Bereiche, mit welchen ich mich bis jetzt noch nicht beschäftigt hatte. Mit der Körpervermessung ist ein interessantes Themengebiet in unsere Arbeit eingeflossen. Dass es sogar eine Wissenschaft über das Verhältnis einzelner Körpereigenschaften (Alliometrie) gibt, war mir neu. Doch fast spannender war für mich die Kinect-Technologie und das Ausprobieren, das Suchen, und vor allem auch das Finden von Informationen. Das Denken im 3D-Raum war lange Zeit nicht mehr gefordert worden und musste erst wieder reaktiviert werden. Die Entwicklung eines Ausstellungsobjekts war ebenfalls neu für mich. Zwar kenne ich mit Usability aus, doch ein Ausstellungsobjekt stellt viel höhere Anforderungen im Bezug auf die Benutzbarkeit, als eine Alltagssoftware. Aus diesem Entwicklungsprozess kann ich auch die Aussage von Herr Junge mitnehmen: Er versicherte uns, dass ein Ausstellungsstück nie fertig entwickelt sei. Im Bezug auf unser Vorgehen konnte ich auch wichtige Erkenntnisse machen. So ist es für das Gelingen des Projekts nicht unbedingt notwendig, dass zu Beginn der Arbeit schon definiert ist, was das Endresultat sein sollte. Wir genossen das Vertrauen unseres Betreuers und hatten so freie Hand.

4.2.2.4 Fazit

Herr Junge sollte mit seiner Aussage vorläufig Recht behalten. Das Ausstellungsstück ist nicht fertig entwickelt. Da dies für uns nie das Ziel war, betrübt mich das nicht wenn ich auf die vergangenen 17 Wochen zurückblicke. Wir haben es geschafft, ohne das Budget gross zu sprengen, einen funktionstüchtigen Prototypen zu erstellen. Dieser wird von uns in den Sommerferien fertiggestellt. Das „Projekt-Go!“ erhielten wir in den vergangenen Tagen vom Industriepartner.

Ich freue mich auf kommende Zeit, um zusammen mit Felix Egli das Projekt abzuschliessen. Felix zeichnet sich durch seine ausgeglichene und motivierende Art aus. Zusammen pflegen wir mittlerweile ein mehr als ein kollegiales Verhältnis, in welchem das Arbeiten sehr angenehm ist. Das gegenseitige Verständnis war uns beiden sehr wichtig. So konnten wir Probleme immer auf einem hohen Level diskutieren und Lösungen finden.

4.3 Lessions Learned

Zusammenzug aus den oben genannten Problemen und Lösungen.

- **Realer Kunde**: Ein realer Kunde benötigt viel mehr Zeit um Abklärungen zu machen. Denn diese Abklärungen müssen meist durch ganzen internen Prozesse des Kunden laufen, welches zusätzlich Zeit benötigen.
- **Concurrency**: Durch die Verwendung von mehreren Threads für ForcePlate und Kinect, steigt die Gefahr, sich in einem Deadlock oder komplexen Ablaufproblemen zu verstricken. Durch Endurance-Tests kann dies geprüft werden.
- **Themengebiet**: Das Themengebiet über das Ausmessen des Menschen ist ein sehr interessantes und komplexes Gebiet und wir durften im Rahmen unserer Arbeit viel Wissen in dem Bereich erarbeiten.
- **OpenNI**: Verwendung von OpenNI-Framework mit den Userdetectors und Verarbeitung von PointCloud's.
- **Dispose implementieren**: Bei einer „Realtime"-Umgebung, die über eine längere Zeit laufen soll, ist es wichtig, dass alle Objekte sauber via Dispose aufgeräumt werden. Damit kann auch der Garbage-Collector seine Arbeit wie gewünscht verrichten.
- **Object vs. Struct**: Initialisierung von Objekten braucht sehr viel Zeit. Bei einem Array mit 300'000 Einträgen pro 33ms fällt dies sehr ins Gewicht. Es sollten besser Structs verwendet werden! Das ist ein X-faches schneller.
- **Testing**: Wie stellten fest, dass Testing für verschiedene Qualitätsanforderungen (z.B. Usability, System und Performance) ein sehr wertvolles Instrument ist.
- **Planung**: Die Planung war in diesem Projekt extrem schwierig. Da neue Technologie sehr viele Risiken bergen. Diese lassen sich, weil eben die Technologie sehr neu ist, schwer in konkrete Zeiten einteilen.
- **Prototypen**: Für ein Forschungsprojekt, wie es zu Beginn der Arbeit der Fall war, unabdingbar. Ohne solche ist es unmöglichen Risiken einzuschätzen und zu kontrollieren.

Teil II

SW-Projektdokumentation

5 Anforderungsspezifikation

5.1 Grundlage

Als Grundlage der Anforderungsspezifikation dient das Meeting mit Herrn Künnemann und Herrn Junge vom 11. April 2011 im Technorama in Winterthur (siehe *Sitzungsprotokolle* auf Seite 185).

5.2 Szenarios

Susi Sorglos besucht am Mittwochmorgen mir ihrer Klasse, der dritten Oberstufe aus Horgen, das Technorama in Winterthur. In der Biologielektion haben sie schon einige wichtige Fakten über den menschlichen Körper erfahren. Nun haben sie die Möglichkeit diese Fakten in der Ausstellung „Der vermessen(d)e Mensch“ selber auszuprobieren. Vor dem Ausstellungsstück „Kinect Bodyscanner“ bleibt Susi stehen und betrachtet das Exponat. Sie steht auf eine markierte Fläche. Nun erfährt sie mehr über ihr Körpervolumen, welches in Liter angezeigt wird. Anschliessend erfährt sie etwas über ihre gesamte Körperoberfläche und die aktuell sichtbare Körperoberfläche in Quadratmetern und A4-Blättern.

5.3 Akteure & Stakeholders

5.3.1 Primär Akteure

Technoramabesucher: Besucher, welcher sich im Technorama aufhält.
Hausdienst: Hausdienst, welcher für die Geräteinstallation verantwortlich ist.

5.4 Use Cases

5.4.1 Diagramm

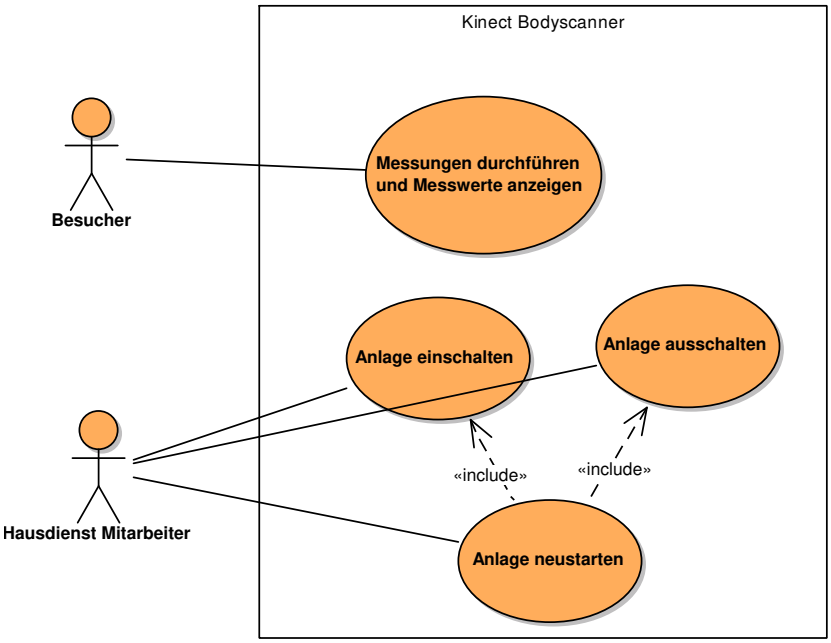


Abbildung (5.1) Use Cases Diagramm

5.4.2 UC01: Messungen durchführen und Messwerte anzeigen

Der Besucher stellt sich auf die vorgegebene Markierung. Nun wird er vom System erkannt und der Messprozess beginnt. Anschliessend werden die gemessenen Werte in geeigneten Messwerten angezeigt (z.B. 1-Liter Milchflaschen/DIN-A4 Blätter). Messwerte beinhalten das Körpervolumen und die Körperoberfläche.

5.4.2.1 UC01b: System aus-tricksen/spielen

Versuch das System auszutricksen. D.h Messungen werden versucht zu verfälschen.

5.4.3 UC02: Anlage in Betrieb nehmen

Mitarbeiter des Technoramas schliesst die Anlage an das Stromnetz an. Der angeschlossene Computer und Kinect starten. Nach wenigen Minuten ist die Anlage benutzbar und es kann der UC01 durchgeführt werden.

5.4.4 UC03: Anlage ausschalten

Mitarbeiter zieht den Stecker an dem die Anlage angehängt ist.

5.4.5 UC04: Anlage neustarten

UC02 durchführen, anschliessend UC03.

5.5 Nicht funktionale Anforderungen

5.5.1 Benutzerfreundlichkeit/Usability

Jeder Benutzer, der an dem Ausstellungsstück interessiert ist, sollte ohne Erklärung mit dem Ausstellungsstück interagieren können.

5.5.2 Abschalten/Hochfahren des Systems

Durch das Ausschalten des Systems darf kein Schaden am System entstehen, denn das System wird über ein Jahr lang jeden Tag eingeschaltet und am Abend auch wieder ausgeschaltet.

5.5.3 Dauerbetrieb

Das System muss ohne Unterbruch, d.h. während acht Stunden am Tag sieben Tage die Woche, laufen.

5.5.4 Leistung

5.5.4.1 Reaktionszeiten

Das System muss während der Betriebszeit stehts innerhalb von 200 ms reagieren können.

5.5.4.2 Durchsatz

Die Framerate der Anzeige darf nicht unter 10 Frames pro Sekunde fallen.

5.5.4.3 Verfügbarkeit

Während den acht Stunden Laufzeit sollte das System zu 98% der Zeit verfügbar sein.

5.5.5 Wartbarkeit

5.5.5.1 Konfigurierbarkeit

Die wichtigsten Parameter müssen einfach konfiguriert werden können.

5.5.6 Schnittstellen

5.5.6.1 Benutzerschnittstelle

Als Benutzerschnittstelle wird eine grafische Benutzeroberfläche gewählt, in welcher die gemessenen Werte in geeigneter Form angezeigt werden können.

5.5.6.2 Hardwareschnittstelle

Es müssen Schnittstellen zu Sensoren, welche die sichtbare Körperoberfläche und das Gewicht der Person messen, vorhanden sein.

6 Analyse

6.1 Domain Model

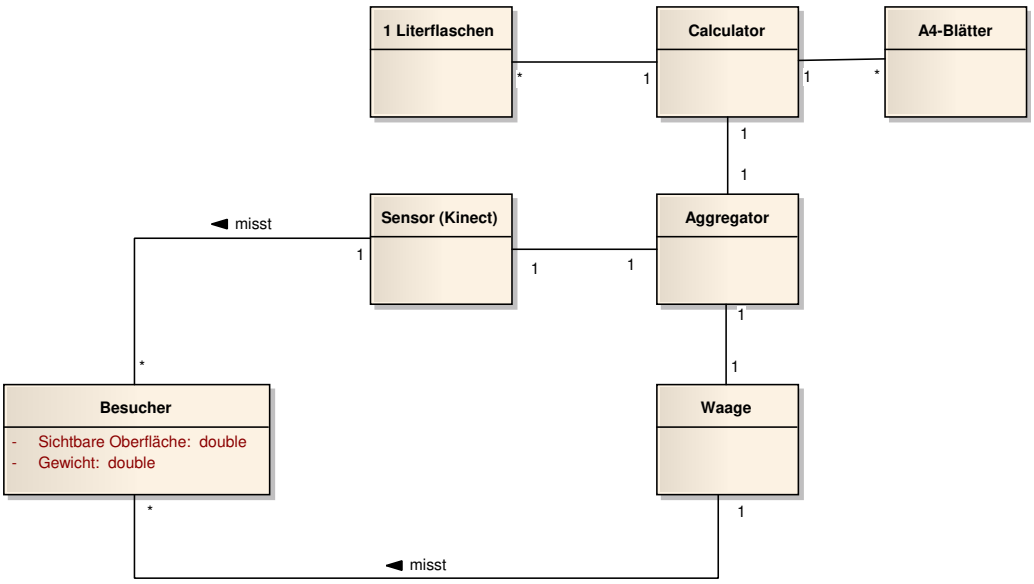


Abbildung (6.1) Domain Model

6.2 Objektkatalog

6.2.1 Besucher

Dieses Objekt repräsentiert den Technorama-Besucher. Dieser besitzt ein spezifisches Gewicht, sowie eine sichtbare Oberfläche.

6.2.2 Sensor (Kinect)

Dies ist der Sensor, mit welchem die sichtbare Oberfläche erfasst werden kann.

6.2.3 Waage

Die Waage dient dazu, das spezifische Gewicht des Besuchers zu messen.

6.2.4 Aggregator

Der Aggregator ist die Komponente, welche die Messdaten des Besuchers aggregiert und dem Calculator zur Verfügung stellt.

6.2.5 Calculator

Der Calculator erhält die Messdaten vom Aggregator und errechnet daraus die Anzahl A4-Blätter und 1-Literflaschen, welche die Daten repräsentieren.

6.2.6 1 Literflaschen

Stellt eine 1-Literflasche dar.

6.2.7 A4-Blätter

Stellt ein DIN-A4-Blatt dar.

7 Design

7.1 Architektur

7.1.1 Systemübersicht

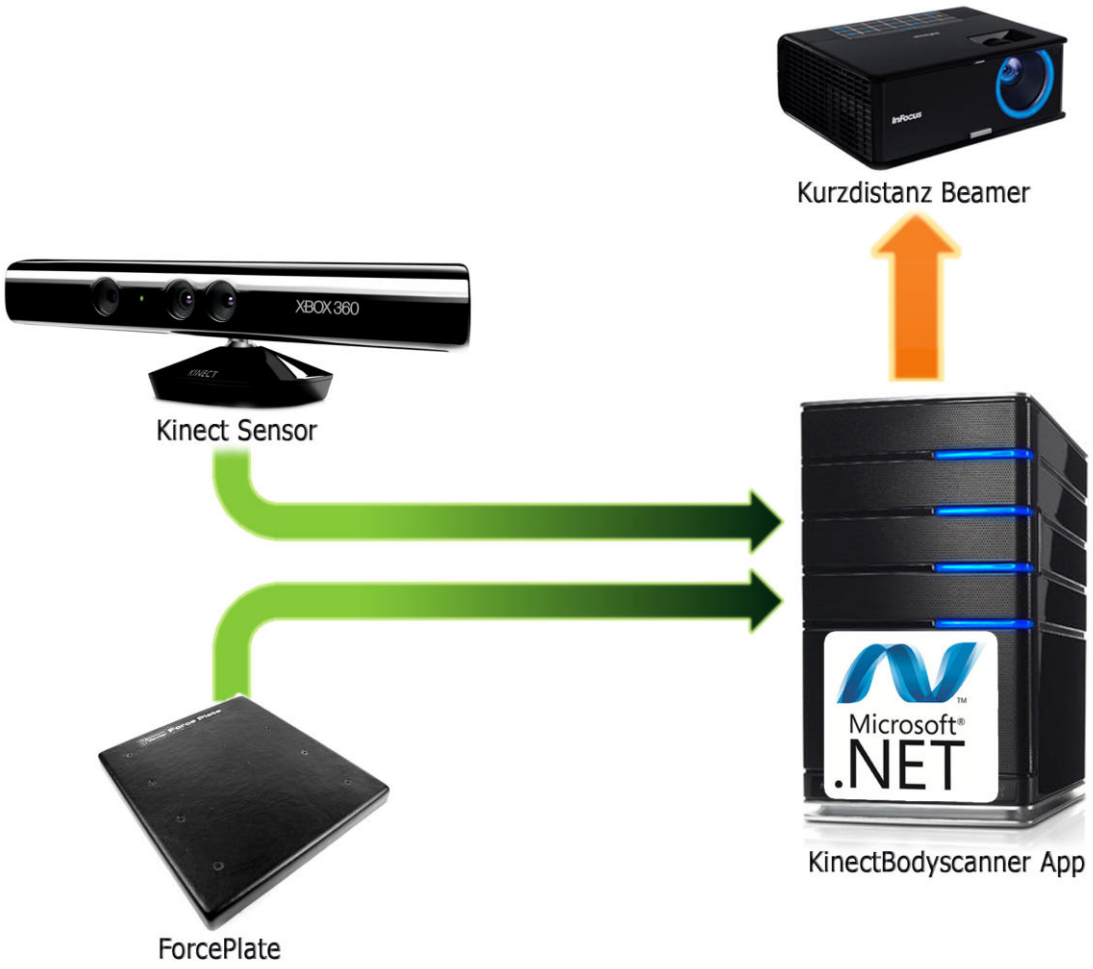


Abbildung (7.1) Systemübersicht mit KinectSensor, ForcePlate, Applikation und Beamer

Aus der oben ersichtlichen Grafik gehen die wesentlichen Komponenten des Systems hervor. Zentrales Element ist der **Kinect Sensor**. Zusammen mit der **ForcePlate** ist er für die Vermessung der Person und die Interaktion mit dem System zuständig. Mittels den Daten dieser beiden Sensoren wird über die **Bodyscanner Applikation** die Benutzeroberfläche

gesteuert und die Berechnungen werden durchgeführt. Die Ausgabe erfolgt über einen herkömmlichen **Beamer** oder Monitor. Es wurde ein Kurzdistanzbeamer gewählt, da dieser für die Darstellung einer Person in realer Grösse ohne grossen Platzbedarf auskommt.

Im weiteren wird hauptsächlich auf die Architektur und Funktionsweise der „KinectBodyscanner"-Applikation eingegangen.

7.1.2 Schichtung

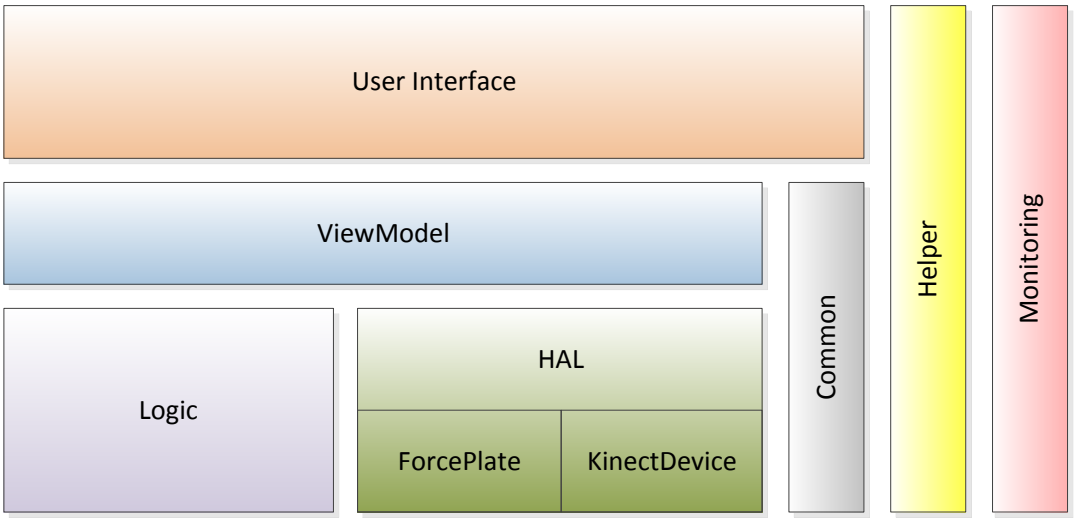


Abbildung (7.2) Verschiedene Layer innerhalb des KinnectBodyscanners

Die Applikation besteht neben den üblichen Schichten für UI und Businesslogik aus Schichten für die Hardwareabstraktion. Nebenbei existieren durchgängige Schichten für allgemeine Aufgaben (z.B. Helper) oder Monitoring, welche von allen Layers verwendet werden.

Die nachfolgenden Seiten geben Aufschluss darüber, welche Layers für welche Funktionen verantwortlich sind.

7.1.2.1 UI-Layer

Im UI-Layer befinden sich diejenigen Elemente, welche für den Benutzer als UI-Element sichtbar sind. Hierzu gehören insbesondere Windows, Pages, sowie eigene Controls.

7.1.2.2 View Model

Das ViewModel steuert die Eingaben und Aktionen auf dem User Interface. „ViewModel“ ist ein Begriff aus dem MVVM-Pattern[1]

7.1.2.3 Logic

Der Logic-Layer beheimatet die Programmsteuerung und Hintergrundberechnungen.

7.1.2.4 HAL

Im *HAL*[2] sind Abstraktionen der Hardware Devices zusammengefasst. Dieser Layer stellt den darüberliegenden Layern eine einheitliche Schnittstelle zur Verfügung, so dass die oben liegenden Layer kein Wissen über die konkrete Ansteuerung der Hardware benötigen.

7.1.2.5 Common

Der Common-Layer dient den untersten Schichten und beinhaltet oft verwendete Funktionen aller Art. Teil dieses Common-Layers ist die DeviceFactory (siehe *Device Factory* auf Seite 92).

7.1.2.6 Helper

Funktionen, welche über die gesamte Architektur regelmässig Anwendung finden, sind in diesem Layer zusammengefasst. So findet sich hier etwa eine Klasse, um einheitlich auf die Anwendungskonfiguration (app.config) zugreifen zu können.

7.1.2.7 Monitoring

Der Monitor-Layer beinhaltet verschiedene Methoden zur Überwachung der einzelnen Komponenten im System. Ebenso ist es die Aufgabe des Monitoring-Layers, diese Daten über definierte Schnittstellen, anderen Schichten zur Verfügung zu stellen.

1 Siehe best Practices MVVM-Pattern von Microsoft (http://msdn.microsoft.com/en-us/library/gg405484(v=PandP.40).aspx)

2 *Hardware Abstraction Layer*

7.2 Assemblies und Namespaces

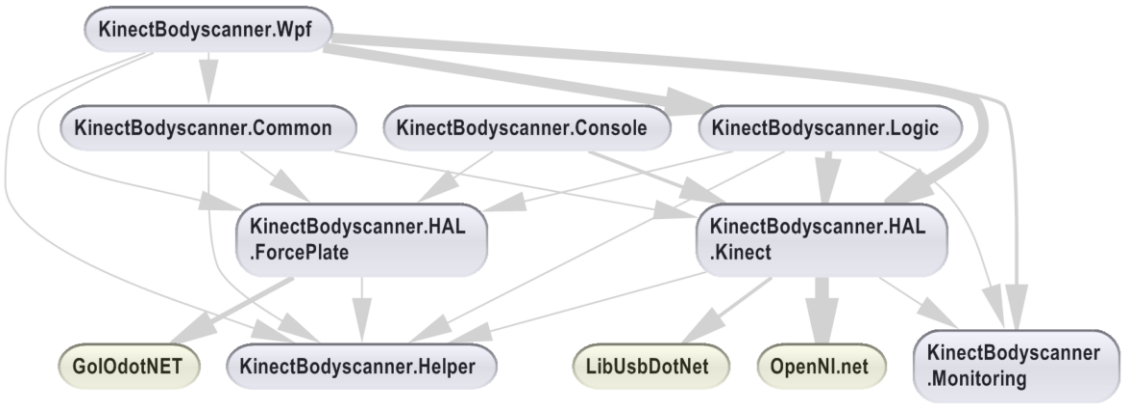


Abbildung (7.3) Assemlbyübersicht mit Abhängigkeiten mit NDepend erstellt

7.2.1 Assembly „KinectBodyscanner.Wpf“

Der gesamte Code, welcher zur Darstellung der Applikation verwendet wird, ist in diesem Assembly kompiliert. Das Assembly dient zudem als Startassembly und beinhaltet die dafür notwendige Konfiguration als App.xaml und das einzige UI-Window (BodyscannerWindow.xaml).

Das Assembly beinhaltet zusätzlich die folgenden Namespaces:

Suffix	*Zweck*
.Controls	Hier befinden sich alle WPF Controls wie z.B. jene für die Videoanzeige, ProgressIndicator, A4-Darstellung usw. (siehe *UI-Controls* auf Seite 79).
.Converters	Converter, um vom ViewModel in die View konvertieren zu können und umgekehrt.
.Pages	Die einzelnen Views während dem Bedienvorgang sind als Pages in diesem Namespace gespeichert
.Util	Diverse kleine Methoden und Klassen für das UI
.ViewModels	Speicherort der Viewmodels gemäss MVVM-Design-Pattern

Tabelle (7.1) KinectBodyscanner.Wpf-Subnamespaces

7.2.2 Assembly „KinectBodyscanner.Common“

Das Common Assembly beinhaltet im Wesentlichen nur eine Factoryklasse, um Objekte für die einzelnen HW-Devices zu erzeugen (siehe *Device Factory* auf Seite 92). Die Factory wird über die *app.config* gesteuert.

7.2.3 Assembly „KinectBodyscanner.Console“

Alternative Startmethode um die Treiber der Kinect Installation zu überprüfen. Zudem kann über die Pfeiltasten der Motor des Kinect-Sensors gesteuert werden. Für den Entwickler bietet sich dieses Projekt für Tests mit dem Gerät an.

7.2.4 Assembly „KinectBodyscanner.Logic“

Dieser Namespace beinhaltet Berechnungsalgorithmen und Algorithmen zur Identifizierung von Personen und der korrekten Selektion eines bestimmten Benutzers. Die Berechnung der Körpereigenschaften sind ebenfalls Aufgaben innerhalb dieses Namespaces.

Suffix	*Zweck*
.Calculation	Hier befinden sich Algorithem zur Berechnung einzelner Körpereigenschaften.
.Detector	Detektoren, welche auf Basis der Messwerte von Kinect und Forceplate reagieren (siehe *Detektoren* auf Seite 90).
.InteractionFlow	Für den Ablauf der Messung relevanter Status.

Tabelle (7.2) KinectBodyscanner.Logic-Subnamespaces

7.2.5 Assembly „KinectBodyscanner.HAL.ForcePlate“

Dieses Assembly kümmert sich um die Abstraktion der Forceplate von Vernier[1] bzw. der .NET-API dazu (siehe *Externe Library „GoIOdotNET“* auf Seite 56).

7.2.6 Assembly „KinectBodyscanner.HAL.Kinect“

Mittels diesem Assembly wird der Zugriff auf die einzelnen Komponenten eines Kinect-Sensors möglich. Es abstrahiert den Kinect-Sensor mit zwei verschiedenen Frameworks und stellt deren Funktionen anderen Assemblies zur Verfügung. Die meisten Klassen dieses Assemblies dienen als Wrapper für dahinterliegende APIs (siehe *Externe Library „OpenNI.net“* auf Seite 56).

1 http://www.vernier.com/probes/fp-bta.html

Suffix	*Zweck*
.	Direkt im Hauptverzeichnis befinden sich die verschiedenen KinectDevices. Je eines für Verwendung eines aufgezeichneten Videos und eines zur Verwendung des Hardware-Devices sind vorhanden.
.Components	Einzelne Abstraktionsklassen für die Hardware-Komponenten eines Kinect-Sensors.
.Data	Datenhaltung der Kinect-Frames (siehe *KinectFrame* auf Seite 88).
.Filter	Filter, welche das Bildsignal der Kamera verbessern.

Tabelle (7.3) KinectBodyscanner.HAL.Kinect-Subnamespaces

7.2.7 Assembly „KinectBodyscanner.Helper“

Stellt für alle Assemblies eine Schnittstelle zu den Anwendungseinstellungen unter *app.conf* (bzw. beim Starten des WPF-Projekts unter *Kinect.Bodyscanner.Wpf.exe.conf*) zur Verfügung.

7.2.8 Assembly „KinectBodyscanner.Monitoring“

Dieses Assembly stellt Methoden zur Überwachung des Systems zur Verfügung (siehe *Performance Monitoring* auf Seite 92).

7.2.9 Externe Library „GoIOdotNET“

Dieses Assembly stammt von Vernier, dem Hersteller der ForcePlate und ist Teil des Go! I/O SDKs[1]. Dieses Assembly dient nur der Umsetzung der Aufrufe in die dahinterliegende C++ Implementierung. Das Assembly übernimmt somit eine Wrapperfunktion für die native C++ API.

7.2.10 Externe Library „LibUsbDotNet“

Dient dem direkten Ansteuern der Komponenten Motor und LED von Kinect (siehe *Abstraktion Kinect Sensor* auf Seite 86). Die Assembly mit der Library stammt von Travis Robinson und darf frei verwendet werden[2].

7.2.11 Externe Library „OpenNI.net“

Bei OpenNI handelt es sich um ein OpenSource-Framework (siehe *Kinect Framework* auf Seite 30), mit welchem sich der Tiefensensor und der Kamerasensor von der Kinect

1 Go! I/O Software Development Kit from Vernier, http://www.vernier.com/downloads/gosdk.html
2 LibUsbDotNet auf sourceforge.net http://libusbdotnet.sourceforge.net/

ansprechen lassen. Weiter bietet das Framework auch Routinen zur Personen- und Gestenerkennung. Die Library ist in C++ geschrieben und stellt einen Wrapper für .NET zur Verfügung.

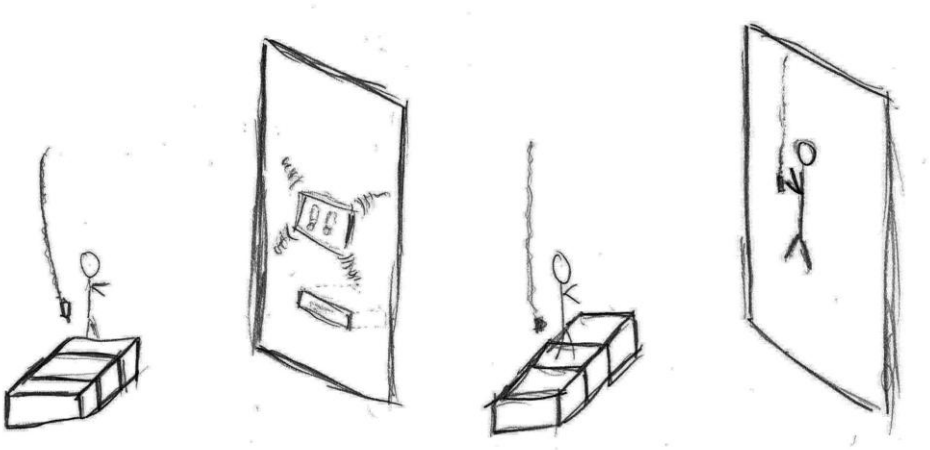
7.3 Bedienprozess

Als Teil der Lösungsfindeung ist der Bedienprozess entstanden. Dieser wurde zusammen mit dem Technorama ausgearbeitet und umfasst die Interaktion mit dem System und der Benutzeroberfläche der Anwendung.

7.3.1 Storyboard

Das Storyboard vermittelt einen Eindruck davon, wie ein erfolgreich durchgeführtes Szenario ablaufen kann.

7.3.1.1 Schritt 1: Messprozess beginnen

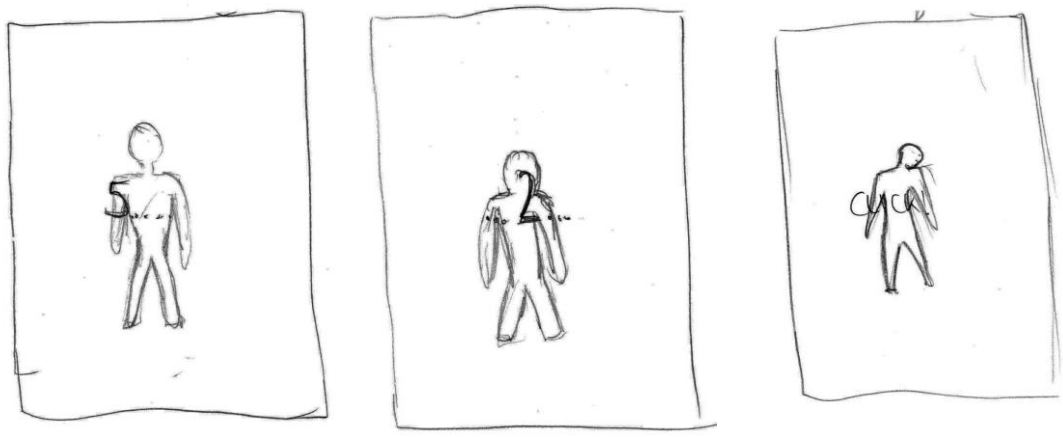


(a) Person nicht auf ForcePlate **(b)** Person auf ForcePlate

Abbildung (7.4) Starten des Messprozesses

Der Messprozess wird dadurch gestartet, dass der Benutzer vollständig von der Kinect und der Forceplate erkannt wird. Erst wenn diese beiden Bedingungen erfüllt sind, kann der Messprozess gestartet werden. Ausgelöst werden kann dieser Prozess durch den Benutzer selbst, über einen Konpf oder auch direkt über die Detektion, dass eine Person auf der ForcePlate ist.

7.3.1.2 Schritt 2: Countdown und Messung



(a) Start des Countdowns **(b)** Noch 2 Sekunden **(c)** Messung!

Abbildung (7.5) Ablauf des Countdowns und Messung

Noch während dem Countdown wird die Messung durchgeführt.

7.3.1.3 Schritt 3: Resultate

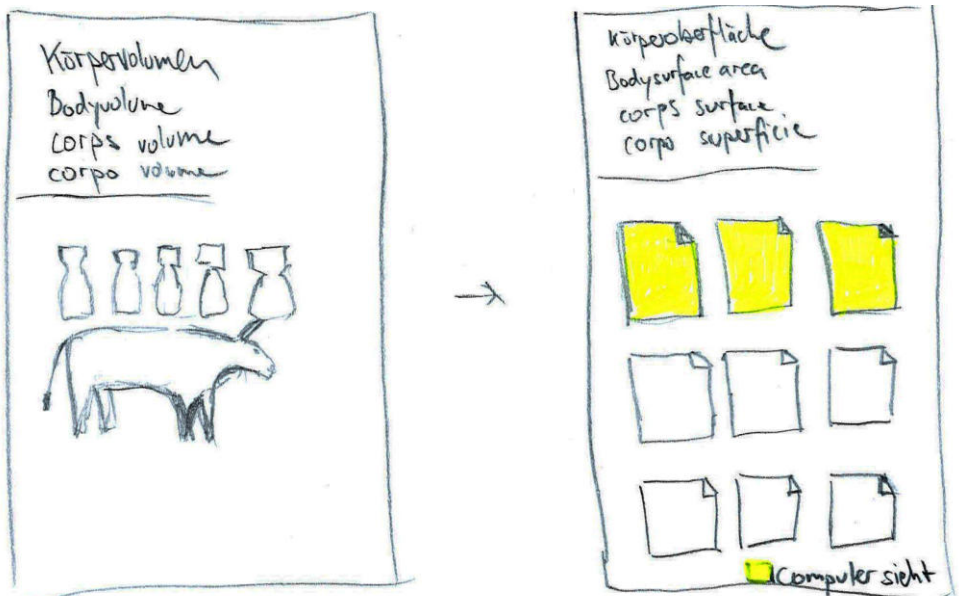


Abbildung (7.6) Resultat als Volumen und als Fläche mit sichtbarem Anteil

Nach der Messung werden die Resultate aufgeteilt nach Volumen und Fläche nacheinander dargestellt. Die gelbe Fläche passt sich während Bewegungen des Benutzers laufend an, so dass ein „Spielen“ mit dem Resultat möglich wird.

7.3.2 State-Machine

Die folgenden State-Machines geben Auskunft darüber, in welchen Status die Applikation sich befinden kann und unter welchen Ereignissen zwischen den einzelnen Status gewechselt wird. Hierzu eine Übersicht:

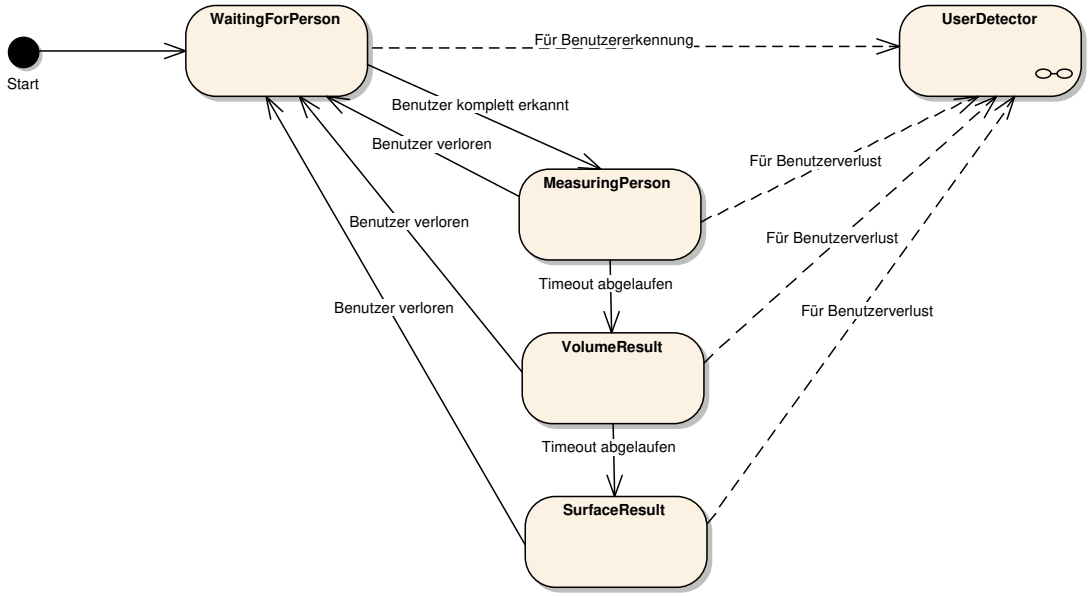


Abbildung (7.7) Generelle State-Machine

Auf Basis der oben ersichtlichen Status wird das UI gesteuert. Viele der Status hängen direkt von der UserDetector-Statemachine ab, welche nachfolgend ebenfalls dargestellt ist.

7.3.2.1 UserDetector-Statemachine

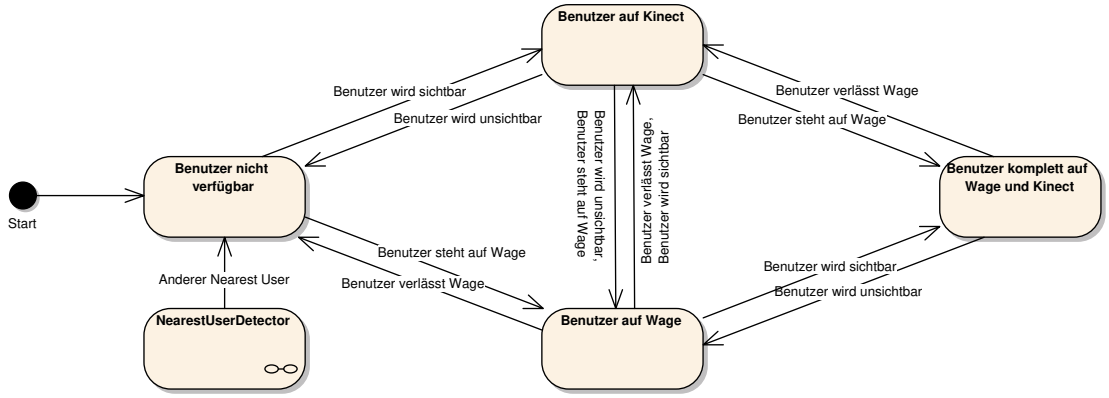


Abbildung (7.8) Userdetector State-Machine

Der UserDetector stellt der generellen State-Machine die zwei wichtigsten Status, „Benutzer nicht verfügbar“ und „Benutzer komplett auf Wage und Kinect“, zur Verfügung. Die restlichen Status sind für die Applikation nicht relevant. Der UserDistanceDetector prüft regelmässig, welcher Benutzer aktuell zu einem definierten Punkt im Raum am nächsten steht. Wird der bisher am nächsten stehende Benutzer von einem anderen Benutzer abgelöst, muss die UserDetector-State-Machine den Zustand der Person wieder neu definieren.

7.3.2.2 NearestUserDetector

Diese Statemachine kennt selbst direkt keine Status, doch wird sie vom UserDetector verwendet, um bei Änderungen des nächsten Benutzers reagieren zu können (siehe *Detektoren* auf Seite 90).

7.4 UI-Design

7.4.1 Allgemein

Unsere Benutzeroberfläche basiert auf WPF[1]. Es kam fast auschliesslich (ausser Teile der Controls) das MVVM-Pattern[2] zur Anwendung, um die Daten darzustellen. Weiter haben wir verschiedene Controls entwickelt (siehe *UI-Controls* auf Seite 79), um bestimmte Aufgaben zu delegieren.

7.4.2 Organisation

Die Organisation gliedert sich folgendermassen. Hauptsächlich übernimmt das *StatusFrame*-Control (siehe *StatusFrame* auf Seite 81) die Navigation über die einzelnen Seiten. Dieses *StatusFrame*-Control ist in einem Hauptfenster eingebettet. Auf den einzelnen Frames sind jeweils einzelne Controls eingebettet.

7.4.3 GUI-Map

Durch die unten aufgeführte Grafik wird ersichtlich, welcher Interaktionsstatus welcher Seite zugeordnet ist. Diese Zuordnung wird in den Eigenschaften zum *StatusFrame*-Control definiert.

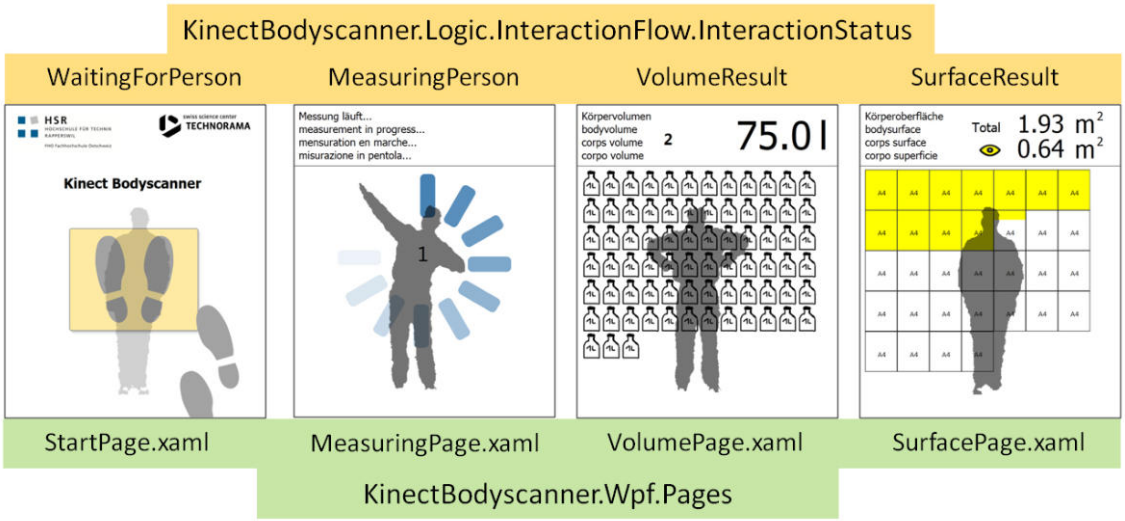


Abbildung (7.9) GUI Map

1 http://msdn.microsoft.com/de-de/netframework/aa663326.aspx
2 http://en.wikipedia.org/wiki/Model_View_ViewModel

7.4.4 Countdown-Timer

Die Countdown-Timer werden von einzelne Seiten (Messseite, Volumenresultat) verwendet, um auf untere Seiten weiter zu schalten. Dies geschieht mittels Command-Binding (siehe *Steuerung mit ViewModel* auf Seite 83).

7.4.5 Guidelines

Die Guidelines, nach welchen wir uns richteten, bestanden hauptsächlich aus Aussagen (z.B. Mehrsprachigkeit usw.) abgestützt auf die Meetings mit dem Technorama (siehe *Sitzungsprotokolle* auf Seite 185). Auf Farbenblindheit wurde in diesem Prototyp nicht weiter eingegangen.

7.5 Prozesse und Threads

Das folgende Kapitel stellt eine Übersicht über die in KinectBodyscanner tätigen Threads dar. Es existiert nur der Hauptprozess *KinectBodyscanner.Wpf.exe*, weshalb hier auf eine separate Auflistung der Prozesse verzichtet wird. Zur besseren Übersichtlichkeit sind alle Threads (und asynchronen Aufrufe) innerhalb dieses Prozesses in der folgenden Tabelle dargestellt.

Name	*Typ*	*Namespace*	*Start*	*Aufgabe*
Event-Dispatching	Async	Diverse	Diverse	Events Async verteilen
Countdown-Timer	Async	Wpf.Pages	Constructor	Countdown für Pages auslösen
Kinect Videocontrol	BG-Worker	Wpf.Controls	Kinect-Init	Aktualisierung des UI
Kinect Videocontrol Stats	Thread	Wpf.Controls	Prop-Change	Zählen der Frames für UI
KinectDevice (Startup)	Thread	HAL.Kinect	Constructor	Nicht-blockierendes Initialisieren
DephtSensor (KinectDevice)	Thread	HAL.Kinect.Com	Nach Init	Daten von OpenNI-API verarbeiten
ForcePlate	Thread	HAL.ForcePlate	Nach Init	Daten von GoOI-API verarbeiten
UserDistanceDetector	Thread	Logic.Detector	Constructor	Aktuell nächste Person an 3D-Punkt ermitteln
MeasuringProjectionThread	Thread	ViewModel	State-Change	Sichtbare Oberfläche berechnen
PerformanceMonitor	Thread	Monitoring.Perfc	Appstart	Performance-Daten abrufen und verarbeiten

Tabelle (7.4) Threads in KinectBodyscanner

Anmerkung: Bis auf die asynchrone Eventverarbeitung, den Countdown-Timern und

den *MeasuringProjectionThread* werden die obigen Threads nur einmalig gestartet und bleiben während der kompletten Laufzeit aktiv, auch wenn die entsprechenden Instanzen während der kompletten Laufzeit verwendet werden. Es handelt sich bei allen um benötigte Threads, welche zwingend im Hintergrund laufen müssen.

7.5.1 Allgemein

Die folgenden Threads oder asynchronen Aufrufe werden von der .NET-Laufzeitumgebung verwaltet. Der Vollständigkeit halber sind sie jedoch ebenfalls aufgelistet.

7.5.1.1 Async Event Handling

Es werden, wo immer möglich, die EventHandler asynchon aufgerufen. Damit wird der Produzent des Events entlastet, da durch den ansyonchronen Aufruf der Produzent nicht blockiert wird.

7.5.1.2 Page Countdown-Timer

Einige UI-Elemente bzw. Pages besitzen einen Countdown, nach welchem sie das ViewModel anweisen, auf die nächste Seite zu wechseln. Diese sind als Timer implementiert, welche nach Ablauf des Timers wieder deaktiviert werden.

7.5.2 KinectVideo-Control

Namespace: *KinectBodyscanner.Wpf.Controls*

Das KinectVideo-Control bildet die Schnittstelle zwischen der Kinect-HAL und WPF. Dieses Control beinhaltet mehrere Threads, welche die folgenden Aufgaben haben. Die nachfolgenden Threads werden gestartet, sobald das mit diesem Control verbundene KinectDevice initialisiert ist (siehe *KinectVideo-Control* auf Seite 79).

7.5.2.1 Frames zeichnen

Dieser Thread ist als Background-Worker implementiert, welcher das jeweils aktuellste Frame aus der Kinect.HAL abruft und mittels Bitmap dem UI zur Verfügung stellt. Das Bitmap wird jeweils vom UI gezeichnet. Der Prozess des Neuzeichnens wird jedoch vom Background-Worker angestossen.

7.5.2.2 Frame Statistiken

Dieses Control kann die Frameraten des KinectDevices und die eigene Framerate auf dem Videobereich einblenden. Hierzu wird ein Thread genutzt, welcher alle 1000 ms die Werte auf dem UI aktualisiert.

7.5.3 KinectDevice

Namespace: *KinectBodyscanner.HAL.Kinect*

Das KinectDevice stellt die Abstraktion der Kinect dar (siehe *Abstraktion Kinect Sensor* auf Seite 86).

7.5.3.1 StartupDevice

In der Startmethode von KinectDeviceBase (von welcher alle KinectDevices ableiten) wird der Initialisierungprozess als separater Thread gestartet. Dies, um Verzögerungen, welche durch das OpenNI Framework und die Hardware entstehen, zu vermindern. Der Thread beendet sich nach der Initialisierung automatisch.

7.5.3.2 DepthSensor

Ein DepthSensor wird beim Starten eines KinectDevices erstellt. Direkt im Konstruktor des DepthSensors wird ein zusäztlicher Thread gestartet, welcher permanent Frames vom OpenNI-Framework abholt und in einen Buffer schreibt. Der Thread wird beendet, sobald der DephtSensor beendet wird.

7.5.4 ForcePlateDevice

Namespace: *KinectBodyscanner.HAL.ForcePlate*

Nach dem Instanziieren der Klasse läuft im Hintergrund ein Thread, welcher die Verbindung zur ForcePlate herstellt und danach schläft. Sobald mit *StartMeasuring()* die Messung gestartet wird, liest dieser Thread jede Sekunde die Daten mittels API von der ForcePlate (siehe *Abstraktion ForcePlate* auf Seite 89).

7.5.5 UserDinstanceDetector

Namespace: *KinectBodyscanner.Logic.Detector*

Der UserDistance-Detector wird vom ViewModel erstellt und dadurch automatisch gestartet und steht somit ab Applikationsstart zur Verfügung. Der UserDistance-Detector überprüft laufend, welcher Kinect-User sich aktuell am nächsten zu einem definierten Punkt im 3D-Raum befindet. Dazu ruft er regelmässig das aktuellste Frame vom KinectDevice ab und vergleicht die Distanzen der User. Bei einer Änderung werden entsprechende „Abonnenten“ via Events informiert.

7.5.6 MeasuringProjectionThread

Namespace: *KinectBodyscanner.ViewModels.BodyscannerViewModel*

Der *MeasuringProjectionThread* wird vom ViewModel jeweils während der Anzeige der Körperoberfläche gestartet und beim Wechsel in einen anderen Status wieder beendet. Der Thread rechnet die aktuell sichtbare Körperoberfläche aus und stellt das Resultat dem ViewModel zur Verfügung. Datenbasis bilden die KinectDephtFrames.

7.5.7 Performance Monitor

Namespace: *KinectBodyscanner.Monitoring.Performance*

Der Performance Monitor prüft als einzelner Thread jede Sekunde die registrierten Performance Meters (siehe *Performance Monitoring* auf Seite 92). Dazu wird der Thread alle 1000 ms vom System aufgeweckt. Der Thread startet, sobald sich ein erster Performance Meter registriert hat.

8 Implementation

8.1 Prototypen

Die nachfolgenden Prototypen wurden in chronologischer Reihenfolge erstellt. Sie wurden soweit ausgearbeitet, dass Unklarheiten und Risiken ausgeschlossen werden konnten. Auch dienten die Prototypen dem Technologiestudium.

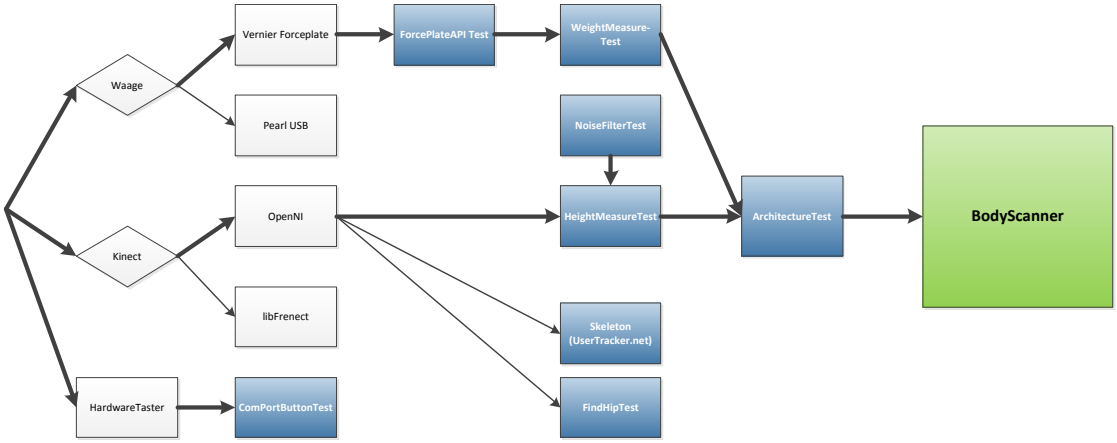


Abbildung (8.1) Übersicht der erstellten Prototypen als Abhängigkeitsgrafik

Auf der oben ersichtlichen Grafik sind die technischen Zusammenhänge und logischen Abfolgen der einzelnen Prototypen zusammengefasst. Daraus lässt sich aus den hervorgehobenen Pfeilen ablesen, welche Prototypen ins finale Endprodukt eingeflossen sind.

8.1.1 Prototyp „UserTracker.net“: Kinect-Sensor Ansprechen

8.1.1.1 Ziele

- Ansprechen der Kinect über den Computer
- Erste Beispiele von OpenNI zum Laufen bringen
- Den C# Wrapper für die OpenNI-Library verwenden
- Distanzen messen (auch im Raum)
- Personen erkennen und das grafisch darstellen
- PointCloud des Bildes abspeichern
- PointCloud visualisieren

8.1.1.2 Resultate

Die folgenden Resultate konnten bezogen auf die oben genannten Ziele erreicht werden.

- **Ansprechen der Kinect über den Computer**
  Dies konnte nach der korrekten Abfolge der Treiber-Installation ohne Probleme erreicht werden.
- **Erste Beispiele von OpenNI zum Laufen bringen**
  Die Beispiele, die OpenNI mitlieferte, konnten direkt im Visual Studio 2010 geöffnet und verwendet werden. Nur die Konfigurationsfiles mussten auf die Kinect angepasst werden.
- **Den C# Wrapper für die OpenNI-Library verwenden**
  Hierfür gabs auch schon Beispiele im Standartpaket von OpenNI.
- **Distanzen messen (im Raum)**
  Hier wurde über die Kalibrierung das Skelett einer Person erkannt. Somit konnten anhand der beide getrackten Vektoren Kopf und Füsse erkannt werden.
  Weiter konnte die Differenz dieser beiden Vektoren[1] gebildet und anschliessend die Länge des Vektors [2] berechnet werden.
- **Personen erkennen und grafisch darstellen**
  Die OpenNI-Library stellt diese Funktion auch schon in einem ihrer Beispiele zur Verfügung. Dieses konnten wir für einen ersten Test übernehmen.
- **PointCloud des Bildes abspeichern**
  Jeder Punkt des Tiefenbildes kann in einen 3D-Punkt mit allen drei Komponenten (x,y,z) umgewandelt werden. Für erste Tests implementierten wir eine Export-Funktion, welche auf Tastendruck das aktuelle Tiefenbild im Polygon File Format[3]abgespeichert.
- **PointCloud visualisieren**
  Mit MeshLab[4] können Dateien im Polygon File Format importiert und in einer 3D Ansicht betrachtet werden.

1 http://de.wikipedia.org/wiki/Vektor
2 http://de.wikipedia.org/wiki/Vektor#L.C3.A4nge.2FNorm_eines_Vektors
3 http://en.wikipedia.org/wiki/PLY_(file_format)
4 http://meshlab.sourceforge.net/

8.1.1.3 Entscheide

Weitere Schritte waren, anhand der *PointCloud* markante Punkte zu finden, die zur Berechnung des Körpervolumens Anhaltspunkte liefern sollten. Danach sollte versucht werden, anhand der *PointCloud* das Volumen zu berechnen.

8.1.2 Prototyp „HeightMeasureTest“: Messungen, Architekturprototyp

Dieser Prototyp wurden zum Erreichen der folgenden Ziele erstellt: Erstellen einer ersten Abstraktion zum Kinect-Sensor via OpenNI, so dass der Zugriff auf den Sensor in eine Klasse gekapselt wird. Zweitens: Ausrechnen und Darstellen der Höhe des Benutzers und allenfalls der Breite. Alle Ziele wurden erreicht. Nebst diesen Zielen diente der Prototyp auch der Entwicklung weiterer Algorithmen, welche zuerst in einzelnen Prototypen getestet wurden. Nachfolgend eine Übersicht der implementierten Funktionen und den daraus erfolgten Erkenntnissen.

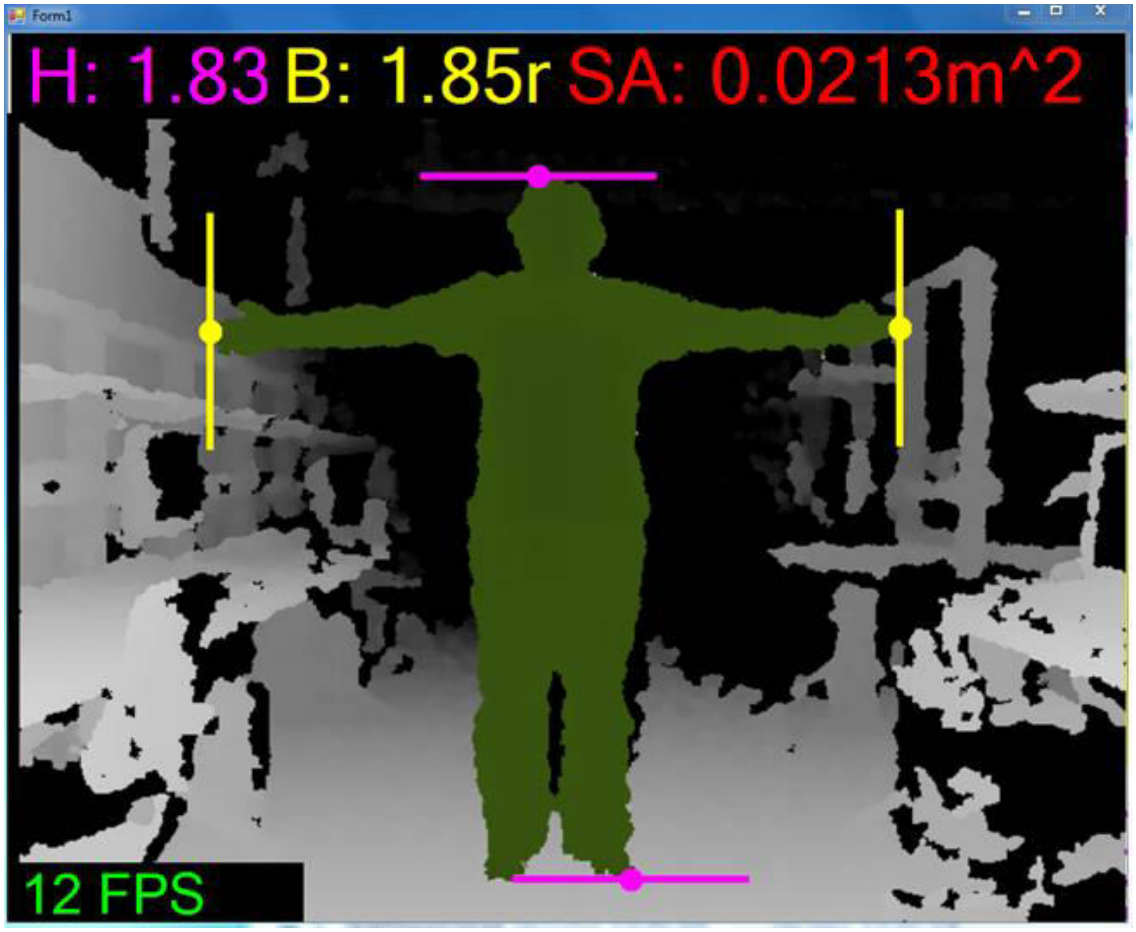


Abbildung (8.2) Screenshot Prototyp „HeightMeasureTest“

8.1.2.1 Architektur

Die Kapselung des Kinect-Sensors geschieht in einer separaten Klasse und einer separaten Assembly. Die Sensor-Daten werden in diesem Prototyp mittels Event-Delegates an interessierte Klassen weitergegeben. Es ist wichtig zu beachten, dass der Event-Handler UI-Anpassungen dem UI-Thread übergibt, da nur der UI-Thread diese durchführen darf. Weiter gibt es Performance-Probleme, wenn zu viele Berechnungen durchgeführt werden bzw. der Benutzer sich sehr nahe am Kinect-Sensor befindet. Bei ausreichender Distanz zum Sensor (ist gegeben, wenn der Benutzer komplett vom Sensor erfasst werden soll) beträgt die Frame-Rate noch knapp 14 FPS.

Das gewählte Design ist problembehaftet: Mit 30 FPS, welche der KinectSensor bzw. das OpenNI-Framework liefert, werden pro Sekunde viel zu viele Events „gefeuert“. Da die Events synchron ablaufen, verringerte sich durch die Berechnung der Höhe die Framerate unserer Abstraktion des Sensors. Dies führte zu Verzögerungen des UI-Threads, welcher

nebst den Events keine Möglichkeit besitzt, auf die Sensor-Daten zuzugreifen.

Viel Zeit musste auch aufgewendet werden, um die Sensor-Daten performant auf dem Bildschirm anzeigen zu können. Mit einer Auflösung von 640x480 Pixeln und 30FPS müssen pro Sekunde fast 10 Mio. Punkte gezeichnet werden, um das Bild flüssig erscheinen zu lassen. Dies konnte erreicht werden, indem für das Zeichnen des UI direkt auf das interne Bitmap zugegriffen wird.

8.1.2.2 Livebild

Durch die vorgegeben Architektur wurde vorausgesetzt, dass sich der UI-Thread am „Daten-Event“ des Kinect-Sensors registrieren musste. Somit bestanden folgende zwei Schwierigkeiten:

- **Übergeben der Daten an UI-Thread**: Die Sensor-Daten wurden vom Event-Handler lediglich gebuffert und danach das UI invalidiert. Nach der Invalidation des UIs konnten die Daten jeweils vom UI-Thread aus dem Buffer gelesen werden. Dies war notwendig, da nur der UI-Thread Zugriff auf die UI-Elemente hat.
- **Performante Darstellung auf Form**: Für die Darstellung der rund 300’000 Bildpunkte pro Frame wird direkt auf das Bitmap des Panels zugegriffen. Nur so konnte eine flüssige Framerate (>24 FPS ohne eingeschaltete Algorithmen) erreicht werden.

8.1.2.3 Höhe / Breite Messen

Nach der Implementation des Noise-Filters (siehe *Prototyp „NoiseFilterTest“: Filter für Z-Achse* auf Seite 73) konnte das Frame auf den höchsten und den niedrigsten Wert in der Y-Achse hin untersucht werden. Diese Werte werden im 3D-Raum gesucht. Dazu ist es notwendig, die Kameradaten in Punkte im 3D-Raum umzurechnen. Die Benutzererkennung wird bereits vom OpenNI-Framework erledigt, diese Funktion musste nicht implementiert werden. So konnten wir bereits mit denjenigen Punkten rechnen, die eindeutig einem Benutzer zugeordnet werden konnten.

Der höchste 3D-Punkt in der Y-Achse wurde über die genannte Liste mit 3D-Punkten gesucht. Dies geschieht über LINQ und mit dem Befehl `userPoints3D.Max(p => p.Y)` bzw. `.Min(p => p.Y)`. Anschliessend ist der höchste Y-Wert bekannt. Mit diesem wird eine Anfrage auf den höchsten 3D-Punkt mit dem entsprechenden Y-Wert gemacht. So kann der gesuchte Punkte gefunden werden.

Nachfolgend ein Auszug aus dem Quellcode:

```
1  double maxY = userPoints3D.Max(r => r.Y);

   Point3D maxHeight3DPoint = (from p in userPoints3D
                               where p.Y == maxY
5                              select p).First();
```

Nun enthält die Variable `maxHeight3DPoint` den Punkt, welcher sich an oberster Stelle befindet. Um die Breite zu messen, werden die höchsten und tiefsten-Werte aus der X-Achse zur Bewertung verwendet.

8.1.2.4 Motor ansteuern / LED setzen

Mit Reverse Engineering und der Arbeit der Community[1] konnte ermittelt werden, wie andere Projekte[2] auf den Motor und auf die LED-Anzeige zugreifen können. Das LED wurde zur Signalisation verwendet. Blinken bedeutete: Kein User erkannt. Sobald ein Benutzer erkannt wurde, wechselte das LED auf Grün. Der Motor konnte mittels den Pfeiltasten gesteuert werden.

8.1.2.5 Oberfläche, Projektion messen, Volumen

Diese Funktion ermöglicht aus der *PointCloud* des Benutzers, die von vorne sichtbare Fläche (*Projektion*) zu berechnen (siehe *Sichtbare Körperoberfläche* auf Seite 20) und daraus mit Hilfe eines fixen Faktors das Volumen des Körpers zu approximieren (siehe *Volumen Berechnung* auf Seite 26). Es war wichtig festzustellen, dass sich die Werte beim Entfernen vom Sensor nicht massiv verändern, auch wenn die Auflösung kleiner wird. Je weiter hinten sich eine Person im Raum befindet, desto kleiner wird jedoch die Auflösung des Kinect Sensors.

8.1.2.6 Verifikation der Gemessenen Werte

Die gemessene Oberfläche wurde mehrmals mit unterschiedlichen Abständen zum Kinect-Sensor getestet(siehe *Sichtbare Körperoberfläche* auf Seite 20).

8.1.2.7 Offene Probleme

- Event-Delegates sind ungeeignet für die Datenübermittlung, da der Thread-Context ändert. Besser (Circular-)Buffer verwenden.
- Die Umrechnung von der Projektion in 3D Koordinaten dauert lange und kann evtl. auf mehrerer Threads aufteteilt werden.
- Berechnungsalgorithmen beeinflussen UI-FPS, hier sollten ebenfalls Berechnungsthreads für einzelne Masse erstellt werden.

1 Open Kinect http://openkinect.org/wiki/Protocol_Documentation
2 KinEmote-Projekt http://www.kinemote.net/

8.1.3 Prototyp „NoiseFilterTest“: Filter für Z-Achse

Die Messung auf der Z-Koordinate wurde oftmals durch starkes Rauschen erschwert. Dies war vor allem im Bereich der Füsse ein Problem, da gerade in diesem Bereich der tiefste Punkt für die Höhenmessung der Person gesucht wurde. Mit diesem Prototyp wurde das Ziel verfolgt, die *PointCloud*, welche auch weiteren Algorithmen als Basis dient, von unnötigen und störenden Punkten zu säubern.

Mit dem Berechnen der Standardabweichung konnte ein Bereich ausgehend vom Körpermittelpunkt berechnet werden, in welchem sich die 3D-Punkte befinden sollten. Die Distanz in der Z-Achse zum Körpermittelpunkte sollte maximal das Doppelte der Standardabweichung aller Z-Abstände zum Körpermittelpunkt betragen.

Die Berechnung gliedert sich in folgenden Schritte:

1. Source-File (.ply) mit den 3D-Punkten auslesen
2. 3D-Punkte in eine Liste mit 3D-Punkt Klassen abfüllen
3. Alle Punkte erstmals von ungültigen 0,0,0 Punkten säubern
4. Anschliessend auf der Z-Achse die Standabweichung ausrechnen. Ausgehend vom CoM-Punkt des Benutzers wird gegen vorne und hinten ein Bereich definiert, welcher in jede Richtung so gross wie das 2-Fache der Standabweichung ist.
5. Alle Punkte, welche ausserhalb dieser 4x-Standabweichung liegen werden gelöscht

Danach ist das grobe Säuberungsverfahren abgeschlossen.

8.1.4 Prototyp „ForcePlateAPITest“: Waage ansprechen

Mit diesem Prototypen wurde das Ziel verfolgt, die Vernier Force Plate[1] über C# anzusprechen. Vorgängig konnte sichergestellt werden, dass die Waage mittels eines USB-Adapters[2] und dem dazugehörigen *SDK*[3] des Herstellers[4] ansprechbar ist. Mit dem SDK wurde bereits schon eine sehr umfangreiche Demo-Version in VB.NET ausgeliefert. Diese wurde auf das Wesentliche reduziert und in C# portiert.
Die Erkenntnisse aus diesem Prototypen sind in einen weiteren Prototypen und in die Endfassung eingeflossen *Abstraktion ForcePlate* (siehe Seite 89) .

1 http://www.vernier.com/probes/fp-bta.html
2 Go!Link USB-Adapter http://www.vernier.com/go/golink.html
3 *Software Development Kit*
4 Go! I/O Software Development Kit from Vernier http://www.vernier.com/downloads/gosdk.html

8.1.4.1 Resultate

- Ansprechen der Waage mittels .NET
- Kapselung des Zugriffs in separater Klasse in C#
- Verständnis über Schnittstelle zum *SDK* und der Waage

8.1.5 Prototyp „WeightMeasureTest“: Waage kapseln

In diesem Prototypen ging es darum, die vorgegebene ForcePlateAPITest umzubauen, damit sie über C# ansprechbar ist und mit minimalem Codeaufwand die aktuellen Messwerte liefert. Das periodische Abfragen der Messresultate konnte somit ebenfalls in einen separaten Thread ausgelagert werden.

8.1.6 Prototyp „FindHip“: Hüfte-Breite berechnen

In diesem Prototypen wurde versucht, die Hüfte einer Person zu finden. Wir gingen nach dem 7.5-8 Kopf-Modell (siehe *Körperproportionen beim Menschen* auf Seite 24) aus. Wir gingen zudem davon aus, dass sich die Hüfte im vierten Kopf von oben befindet. In diesem Bereich des vierten Kopfes wird die linke und die rechte Seite als Linie erkannt. Anschliessend wird der Minimalabstand in der X-Achse zwischen den beiden Linien ermittelt. Dieser entspricht der Hüfte mit dem Abstand.

Zu Beachten: Bei Kindern wird dies nur teilweise funktionieren, denn bis ca. 7 Jahren unterscheiden sich die Proportionen von denen eines erwachsenen Menschen.

8.1.7 Prototyp „ArchitectureTest“: Weiterer Architekturprototyp

In diesem Prototyp ging es darum, die einzelnen Komponenten sauber in einem Prototypen zu vereinen. Auf der einen Seite die Waage, auf der anderen Seite die Kinect. Beide müssen immer die aktuellen Daten auslesen und an verschiedenen Events weiterleiten.

Das Klassendiagramm ermöglicht eine Übersicht über die involvierten Klassen und deren Abhängigkeiten.

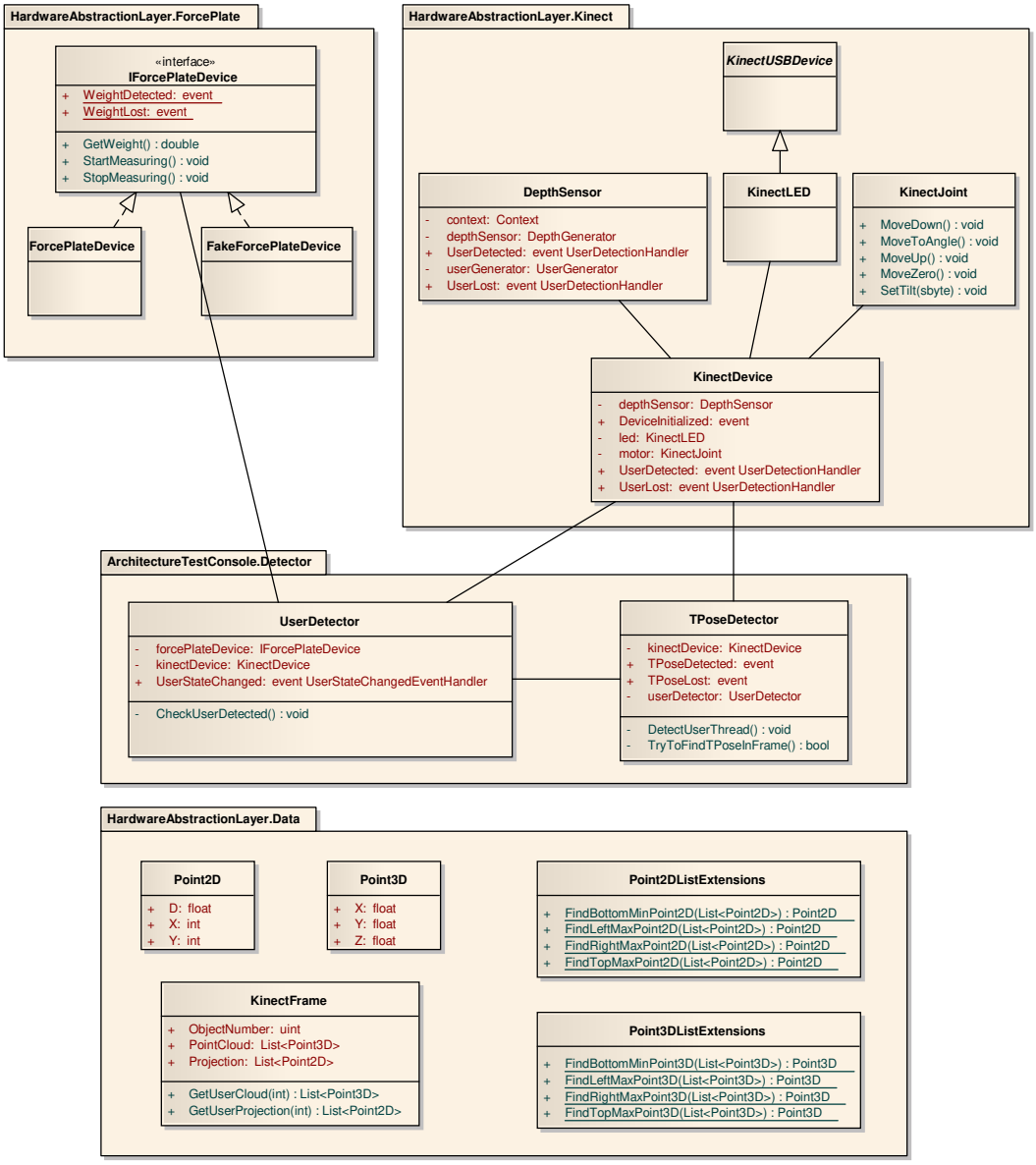


Abbildung (8.3) Class Model von ArchitectureTest-Prototyp

Nachfolgend eine Übersicht über die implementierte State-Machine. Daraus wird klar, welche Bedingungen erfüllt sein müssen, damit z.B. das Suchen der T-Pose gestartet wird.

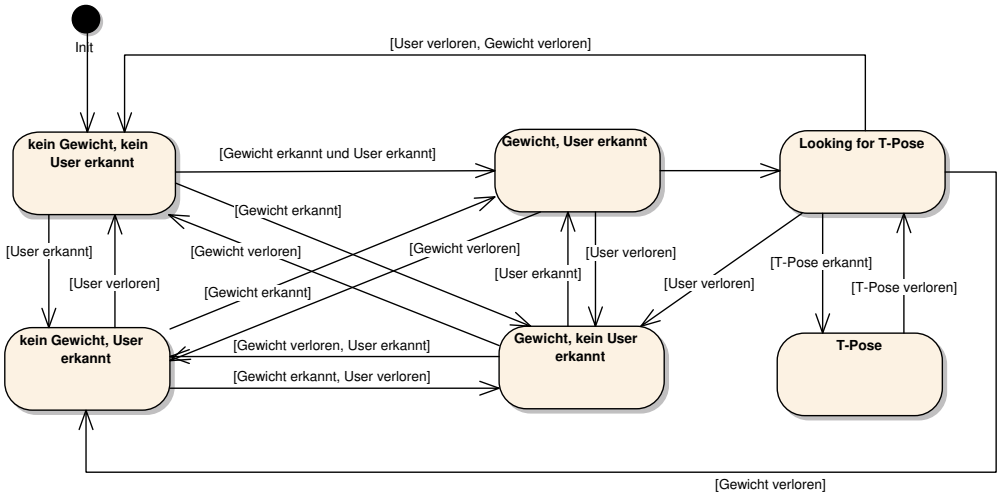


Abbildung (8.4) StateDiagramm von ArchitectureTest-Prototyp

Hier ist der Ablauf beschrieben, wie und wo sich die Komponenten registrieren, um gegenseitig Events konsumieren zu können.

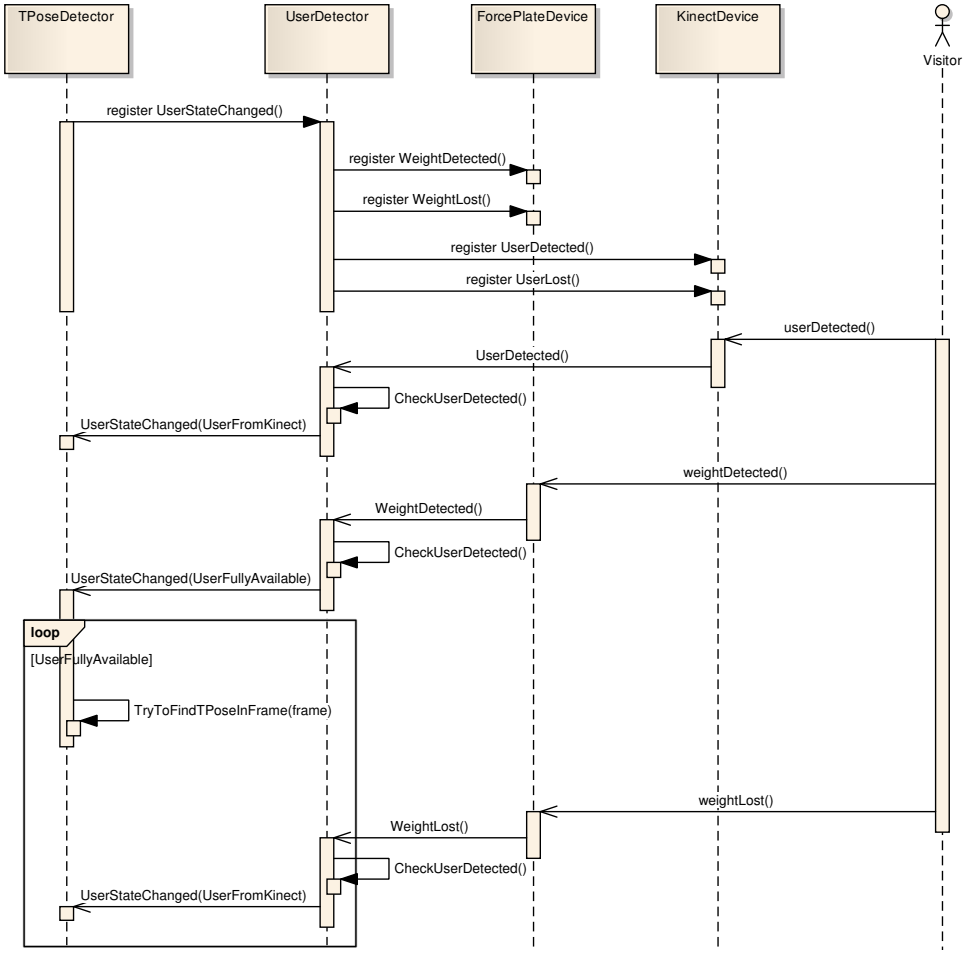


Abbildung (8.5) System Sequenz Diagram von ArchitectureTest-Prototyp

8.1.8 Prototyp „ComPortButtonTest“

Der Prototyp wurde entwickelt, um zu Testen ob und wie ein Taster am seriellen Port eines PCs dazu verwendet werden kann, Events in der Applikation auszulösen. Die erstellte Library stellt somit eine Abstraktion von Hardware-Taster und Applikation zur Verfügung. Der Hardware-Taster fiel im Laufe der Entwicklung wieder weg, deshalb findet diese Library im Endprodukt momentan keine Anwendung mehr.

8.1.8.1 Schema Hardware-Taster

Der Hardware-Taster wurde von uns entwickelt. Als Schnittstelle wurde RS232 gewählt. Hier stehen 2 uncodierte 2 Bits (CTS[1] und RNG[2]) als Eingang zur Verfügung. Wir benutzen einen Schalter, welcher die Pins vom CTS-Ping mit dem Ground verbindet.

1 *Clear to Send-Signal eines Modems, sobald Daten gesendet werden können*
2 *RING-Signal eines Modems, bei einem eingehenden Telefonanruf*

8.2 UI-Controls

Die folgenden WPF-Controls wurden im Rahmen der Arbeit entwickelt. Diese befinden sich alle im Namespace *KinectBodyScanner.Wpf.Controls* des Assemblies *KinectBodyScanner.Wpf*.

8.2.1 KinectVideo-Control

Das KinectVideo-Control ist für die Darstellung der Kinect Tiefeninformationen zuständig. Für den Benutzer ist dieses Control sehr wichtig, da er so nachvollziehen kann, wie das System seine Bewegungen interpretiert hat. Es handelt sich hierbei um ein WPF-Control. Das Control unterstützt zwei verschiedene Modus, um die Daten aus den KinectFrames darzustellen. Daneben existieren weitere Eigenschaften, welche die Darstellung zusätzlich beeinflussen.

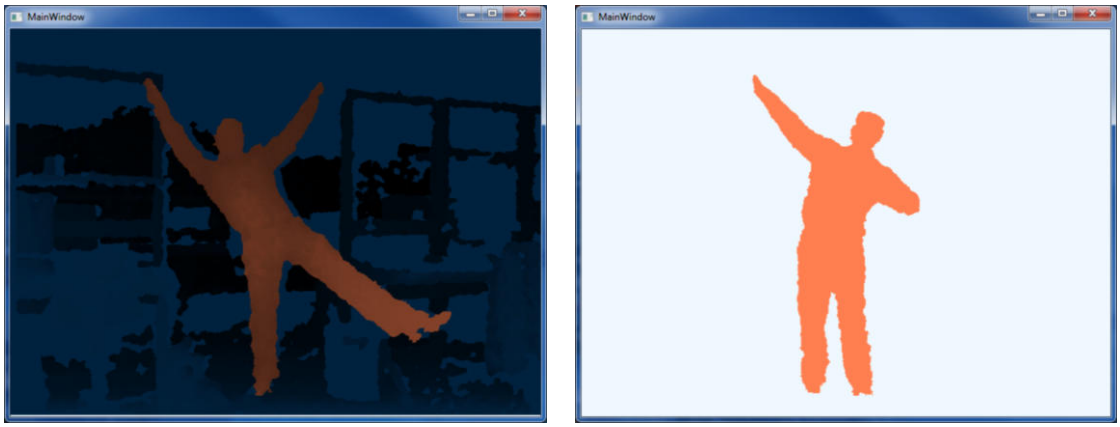
8.2.1.1 Wichtigste Eigenschaften

Mit den folgenden Eigenschaften lässt sich das Control an die eigenen Bedürfnisse anpassen:

Name	*Werte / Datentyp*	*Zweck*
KinectSourceDevice	KinectDeviceBase	Kinect-Object des genannten Datentyps
VideoMode	DephtImage | Projection	VideoModus siehe unten
SelectedUser	1-4	Bestimmte Person hervorheben
Foreground	SolidColorBrush	Farbe des selektierten Benutzers
CrowdColor	Color	Farbe der nicht hervorgehobenen Personen
ShowScene	Boolean	Anzeige der Szene (alles ausser die Personen) Ja/Nein
ShowUserCrowd	Boolean	Anzeige der nicht selektierten Personen Ja/Nein
ShowFrameInfo	Boolean	Framerate von Control und Kinect anzeigen

Tabelle (8.1) Haupteigenschaften des KinectVideo-Controls

8.2.1.2 Videomodus



(a) Tiefenmodus („DepthImage“-Modus) **(b)** Projektion („Projection“-Modus)

Abbildung (8.6) Unterschiedliche Darstellungsoptionen der Tiefeninformationen

Im „DepthImage“-Modus wird die Tiefeninformation in Form von unterschiedlichen Farben und Intensitäten dargestellt, während im „Projection“-Modus nur die Projektion einzelner Personen dargestellt wird. Je nach Modus macht es zusätzlich Sinn, die Szene (und damit alles ausser die erkannten User) auszublenden.

8.2.1.3 Video-Mode/Optionen-Kombinationen

Nicht bei jedem VideoMode ergeben die restlichen Einstellungsmöglichkeiten Sinn. Möglicherweise führen falsch eingestellte Eigenschaften zu einem Fehlverhalten oder zu Performance-Einbussen. Nachfolgend eine Matrix, welche VideoMode mit welchen restlichen Einstellungen verwendet werden sollen und können.

Einstellung	*DephtMode*	*Projection*
SelectedUser (1-4)	OK	OK
Foreground beliebig	OK	OK
CrowdColor beliebig	OK	OK
ShowScene An	Möglich	Nicht empfohlen
ShowUserCrowd An	Möglich	OK
ShowFrameInfo An	OK	OK

Tabelle (8.2) Matrix für sinnvolle Einstellungskombinationen

8.2.2 StatusFrame

Das StatusFrame-Control leitet vom *System.Windows.Controls.Frame* ab und besitzt zwei neue Eigenschaften: CurrentStatus und StatusPages. Durch diese neuen Eigenschaften lässt sich durch das Setzen eines Status und der Hinterlegung einer Page für genau diesen Status effizient zwischen verschiedenen Pages wechseln.

Beispielcode in XAML:

```
<c:StatusFrame CurrentStatus="{Binding Path=Status}">
  <c:StatusFrame.StatusPages>
    <c:StatusPage Status="WaitingForPerson" Page="StartPage.xaml" />
    <c:StatusPage Status="MeasuringPerson" Page="MeasuringPage.xaml" />
    <c:StatusPage Status="VolumeResult" Page="VolumePage.xaml" />
    <c:StatusPage Status="SurfaceResult" Page="SurfacePage.xaml" />
  </c:StatusFrame.StatusPages>
</c:StatusFrame>
```

Das Control übernimmt zusätzlich das Databinding. So können die einzelnen Pages durch die Eingenschaft „DataContext“ auf den DataContext des StatusFrames zugreifen.

8.2.2.1 Eigenschaft „StatusPages“

Dies ist eine Auflistung von StatusPages, welche nebst dem Namen für die Page auch einen Status zugewiesen haben. Siehe obiges XAML-Beispiel.

8.2.2.2 Eigenschaft „CurrentStatus“

Aufgrund diesem Wert schaltet das Control auf die in der Eigenschaft „StatusPages“ hinterlegte Page um.

8.2.3 MilkBottle-Control (MilkBottle und MilkBottleGrid)

Das Milkbottle-Control besteht aus einem MilkBottleGrid und mehreren MilkBottles. Hierbei passt sich das MilkBottleGrid immer der Grösse des umgebenden Controls an. Ändert sich die Anzahl der Milkbottles oder die Grösse das MilkBottleGrids, werden die MilkBottles in ihrer Grösse angepasst. So ist der Bildschirmbereich immer optimal ausgenutzt.

8.2.4 A4-Control (A4Page und A4PageGrid)

Das A4-Control funktioniert mit einer Abweichung identisch wie das MilkBottle-Control. Über die zusätzliche Eigenschaft „NumberOfFilledPages“ kann gesteuert werden, vieviele der A4-Seiten gelb ausgefüllt werden sollen.

8.2.5 ProgressIndicator

Der ProgressIndicator basiert auf dem Project „WPF Circular Progress Indicator“ von http://www.codeproject.com/KB/WPF/rounder_progressBar_ctrl.aspx und wurde an einigen Stellen angepasst.

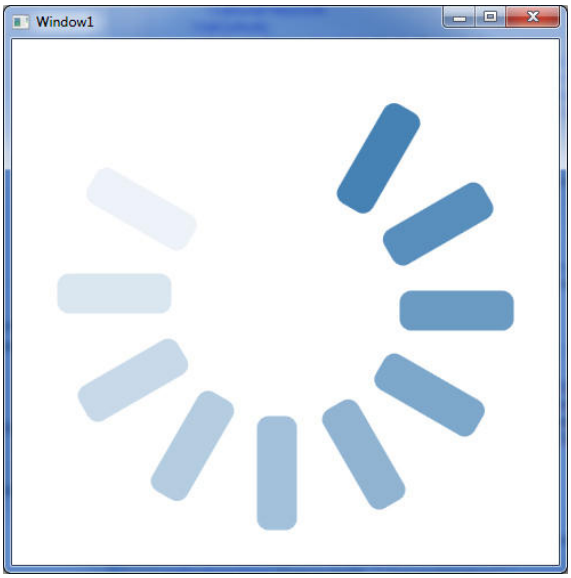


Abbildung (8.7) ProgressIndicator-Control

Die Animation läuft während der ganzen Zeit und kann nur durch Entladen des Controls gestoppt werden. Dies ist in unserem Fall jedoch nicht notwendig.

8.3 Konzepte

8.3.1 Steuerung mit ViewModel

Die Steuerung des UI's und damit auch ein grosser Teil der Applikationslogik befindet sich im ViewModel *BodyscannerViewModel*. Dieses ist gemäss dem MVVM-Pattern implementiert und leitet dadurch vom auch *System.Windows.DependencyObject* ab. Die Aufgaben des ViewModels sind wie folgt gegliedert.

8.3.1.1 Properties für UI

Das ViewModel stellt verschiedene Properties für die UI-Controls bereit. Diese sind alle als DependencyProperty implementiert. Für die Verwendung von DependencyProperty spricht das integrierte Handling von Change-Notifications. Die folgenden Properties sind für das UI von besonderer Bedeutung.

- **Status**: Das Property Status führt immer den aktuellen Status des Messvorgangs (siehe *Generelle State-Machine* auf Seite 59).
- **PersonWeight**: Gewicht der aktuellen Person.
- **PersonSurfaceArea**: Oberfläche der aktuellen Person.
- **PersonProjectiveArea**: Sichtbare Oberfläche der aktuellen Person.
- **CurrentKinectUserId**: Beinhaltet die Kinect-Benutzer-ID derjenigen Person, welche am Nächsten zur Waage steht.

Nicht nur das UI, sondern auch das ViewModel registriert sich auf Änderungen der einzelnen Properties (*Status* und *CurrentKinectUserId*). So können z.B. Aufgaben, welche bei Statuswechseln abzuarbeiten sind, in einen EventHandler ausgelagert werden.

8.3.1.2 Commands für UI

Das ViewModel offeriert den Command[1] *NextStatusCommand* ebenfalls als DependencyProperty. Dieser Command kann vom UI verwendet werden um auf den nächsten Status umzuschalten. Dies wird häufig von den Countdown-Seiten verwendet. Hierbei ist wichtig zu beachten, dass das UI keinen Einfluss darauf hat, welcher Status als nächstes angezeigt werden soll. Darum befindet sich diese Logik im ViewModel.

1 Command-Pattern auf Wikipedia (http://en.wikipedia.org/wiki/Command_pattern)

8.3.1.3 Initialisierung

Die Initialisierung des ViewModels geschieht im Konstructor. Das hat den Vorteil, dass das ViewModel vom UI direkt in XAML initialisiert werden kann. Während der Initialisierung durchläuft der Prozess die folgenden Phasen:

- Laden von *KinectDevice* und *ForcePlateDevice* über die *DeviceFactory*
- Initialisieren von *UserDetector* und *UserDistanceDetector*. Hinzufügen von eigenen EventHandlern zum *UserDetector*.
- Hinzufügen von Event-Handlern für die eigenen Properties *Status* und *CurrentKinectUserId*

8.3.1.4 Status Transformationen

Nach den folgenden Ereignissen ändert das ViewModel seinen Status (in Klammer der Ursprung des Events).

- UserDetector ändert in den Status *UserFullyAvailable* (via EventHandler)
- UserDetector ändert **nicht** in den Status *UserFullyAvailable* (via EventHandler)
- Der *NextStatusCommand* wird im Status MeasuringPerson aufgerufen (via UI)
- Der *NextStatusCommand* wird im Status VolumeResult aufgerufen (via UI)

Durch die oben aufgeführten Transformationen ändert das Property Status seinen Wert. Nebst anderen EventHandlern wird nach jeder diese Änderungen auch der eigene EventHandler des ViewModels für dieses Property aufgerufen. Damit wird sichergestellt, dass im neuen Status die benötigten Berechnungen durchgeführt werden oder Threads gestartet/gestoppt werden.

Folgende Grafik verdeutlicht die Registrierung der EventHandler am UserDetector und die Reaktion auf Ändeungen.

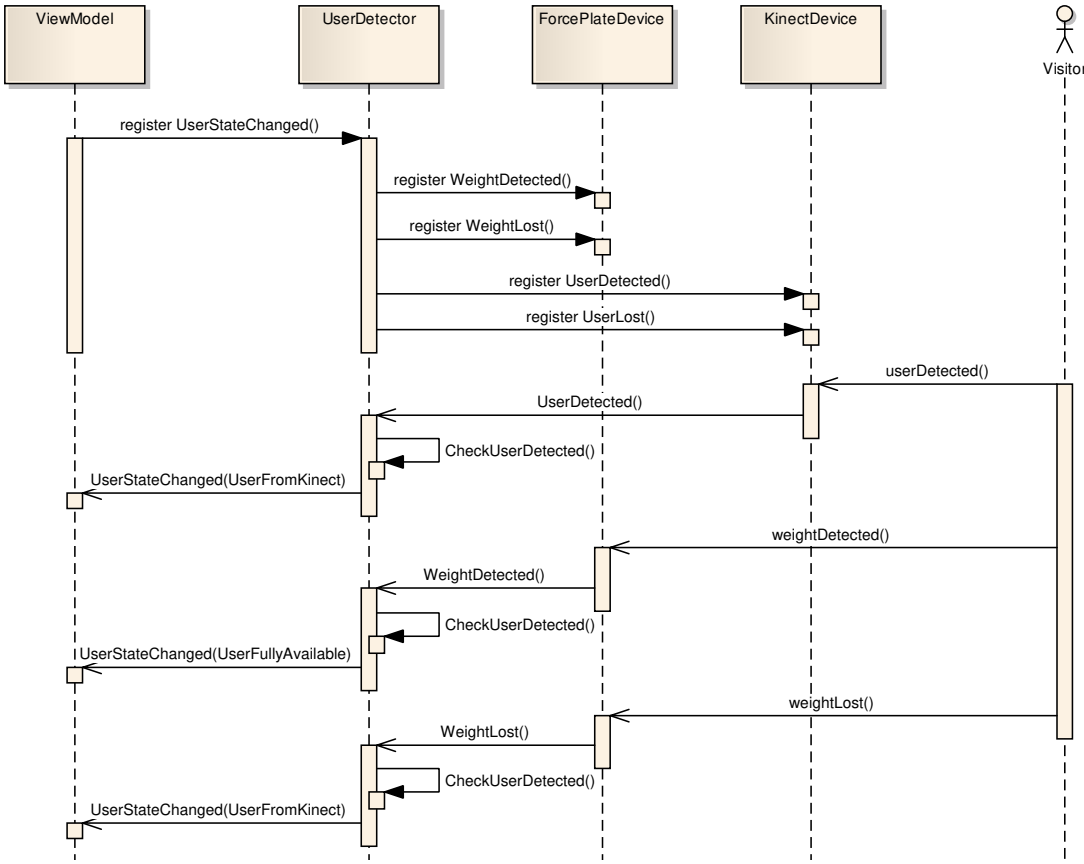


Abbildung (8.8) SSD vom ViewModel mit UserDetector

8.3.1.5 Verfügbarkeit der Daten

Welche Daten im ViewModel verfügbar sind, hängt vom Status ab. Ausser bei der Sichtbaren Oberläche handelt es sich um Momentaufnahmen, welche beim Welchsel in den jeweiligen Status neu berechnet werden. Nachfolgende Tabelle gibt Aufschluss über die verfügbaren Daten je Status.

Status	*Gewicht*	*totale Oberfläche*	*sichtbare Oberfläche*
WaitingForPerson	Veraltet	Veraltet	Veraltet
MeasuringPerson	Veraltet	Veraltet	Veraltet
VolumeResult	Snapshot	Snapshot	Aktuell
SurfaceResult	Veraltet	Veraltet	Aktuell

Tabelle (8.3) Verfügbare Messdaten pro Status

Die mit Snapshot bezeichneten Werte werden bei jedem Wechsel in den entsprechenden Status ausgelesen oder allenfalls berechnet (siehe *Gesamte Körperoberfläche* auf Seite 19). Die sichtbare Oberfläche wird im Hintergrund permanent berechnet (siehe *Sichtbare Körperoberfläche* auf Seite 20). Die Berechnung startet schon einen Status vorher. Damit wird sichergestellt, dass bei Wechsel in diesen Status bereits Daten zur Verfügung stehen.

8.3.2 Abstraktion Kinect Sensor

Namespace: *KinectBodyscanner.HAL.Kinect*

Die Abstraktion des OpenNI-Frameworks und der Kinect hat den Vorteil, dass die Lösung unabhängig vom verwendeten Framework entwickelt werden kann. Es ist anzunehmen, dass das zugrunde liegende Framework durch das offizielle Framework von Microsoft ausgetauscht wird. Ziel ist, die Anpassungen möglich gering zu halten.

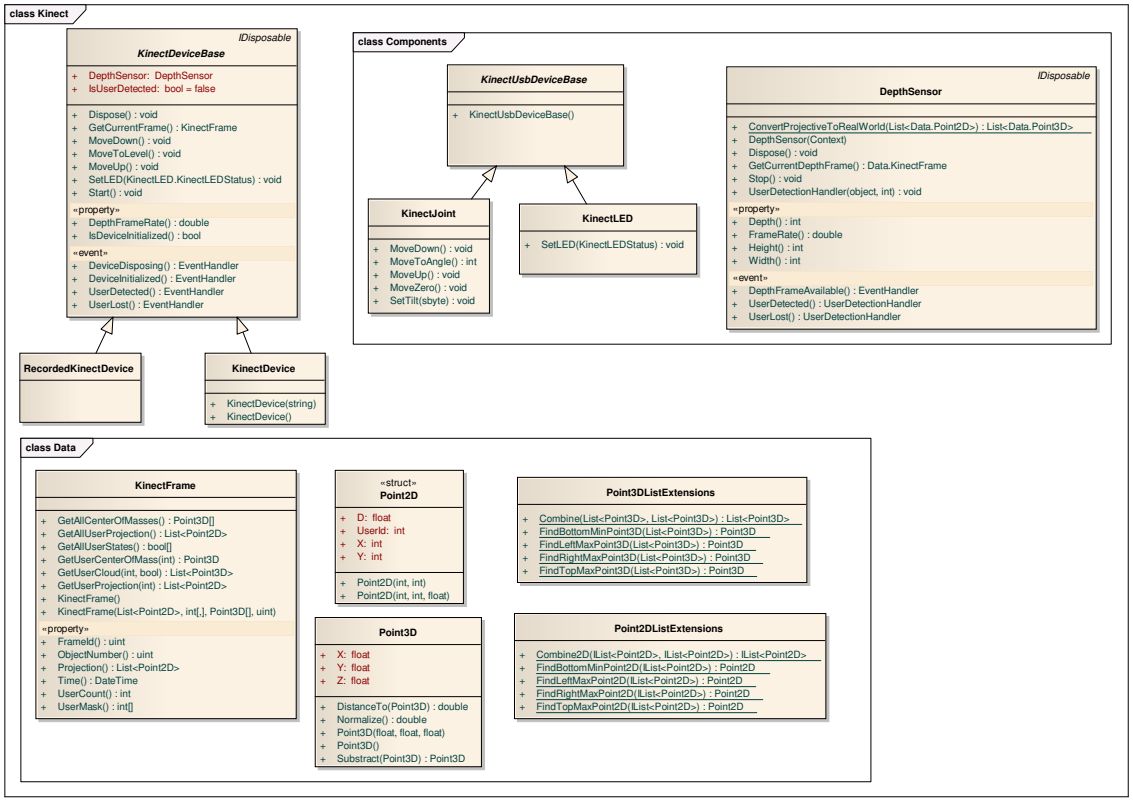


Abbildung (8.9) Klassenübersicht Kinect Abstraktion

Die Kinect-Abstraktion besteht aus mehreren Komponenten, welche nachfolgend beschrieben sind.

8.3.2.1 KinectDevice

Namespace: *KinectBodyscanner.HAL.Kinect.KinectDevice*

Mit dem *KinectDeviceBase* steht eine abstrakte Klasse zur Verfügung, welche die erforderlichen Methoden gegenüber der Applikation definiert. Aus dieser Klasse wurden die konkreten KinectDevices *KinectDevice* und *RecordedKinectDevice* abgeleitet. Wichtigste Methode ist die *GetCurrentFrame* welche das aktuell letzte *KinectFrame* aus dem Tiefensensor zurückliefert. Je nach Verwendugszweck sind auch die Events von starker Bedeutung.

8.3.2.2 Tiefensensor (DepthSensor)

Namespace: *KinectBodyscanner.HAL.Kinect.Components.DepthSensor*

Der Tiefensensor ist eine Komponente der Kinect und besteht somit auch hier eine einzelne Klasse. Diese Komponente ist zuständig für die Datenbeschaffung aus dem OpenNI-Framework. Hierzu werden die Roh-Tiefeninformation kontinuierlich abgerufen, in ein *KinectFrame* gespeichert und im internen zyklischen Buffer abgelegt. Die Roh-Tiefereninformation kann sich wie ein Foto vorgestellt werden. Die Tiefe des einzelnen Pixels entspräche dann der Farbe eines Pixels.

8.3.2.3 Motor (KinectJoint)

Namespace: *KinectBodyscanner.HAL.Kinect.Components.KinectJoint*

Die Klasse *KinectJoint* stellt den Neigemotor dar. Die Klasse spricht mit den Methoden der Superklasse *KinectUsbDeviceBase*, die Kinect direkt an und verwendet nicht das OpenNI-Framework. Die *KinectUsbDeviceBase*-Klasse verwendet hierbei die Library „LibUsbDotNet“ (siehe *Externe Library „LibUsbDotNet“* auf Seite 56).

8.3.2.4 LED-Steuerung (KinectLED)

Namespace: *KinectBodyscanner.HAL.Kinect.Components.KinectLED*

Das LED der Kinect kann ebenfalls direkt über USB gesteuert werden. Hierzu wird wiederum die Klasse *KinectUsbDeviceBase* verwendet, welche sich um die Adressierung und den Verbindungsaufbau kümmert. Folgende Farben stehen für das LED zur Auswahl:

- Ausgeschaltet
- Grün
- Rot

- Gelb
- Gelb blinkend
- Grün
- Gelb-Rot blinkend

8.3.2.5 Lagesensor

Wurde nicht in Software abstrahiert, da nicht verwendet.

8.3.2.6 KinectFrame

Namespace: *KinectBodyscanner.HAL.Kinect.Data.KinectFrame*

Das *KinectFrame* ist primär für die Datenhaltung eines einzelnen Frames aus dem Tiefensensor der Kinect zuständig. Nebst dieser Aufgabe stellt das *KinectFrame* Methoden zur Verfügung, um aus den gespeicherten Tiefeninformationen umgerechnete 3D-Daten zu erhalten. Über die Eigenschaften können die Rohdaten jedoch weiterhin performant ausgelesen werden.

- **GetAllCenterOfMasses()**: Gibt eine Liste von *CoM*[1]-Punkten von allen erkannten Personen zurück.
- **GetAllUserProjection()**: Liefert die Roh-Tiefeninformation von allen erkannten Benutzern, jedoch ohne Szene.
- **GetAllUserStates()**: Returniert die eine Liste mit der Information welche User von der Kinect erkannt wurden.
- **GetUserCenterOfMass()**: Gibt den *CoM* einer bestimmten Person zurück.
- **GetUserCloud()**: Liefert eine Liste von 3D-Punkten eines bestimmten Benutzers
- **GetUserProjection()**: Liefert die Roh-Tiefeninformation eines bestimmten Benutzers als Liste von 2D-Punkten.
- **Projection**: Entspricht dem kompletten *KinectFrame*, d.h. die komplette Tiefeninformation aller Personen und der Szene.
- **UserCount**: Anzahl erkannte Benutzer
- **UserMask**: Array mit der Information, an welchem Pixel welcher Benutzer erkannt wurde.

1 *Center Of Mass ist die englische Bezeichnung für den Mittelpunkt eines Körpers*

8.3.2.7 Point2D und Point3D

Dieses Struct bzw. Klasse wird verwendet, um die Informationen zu einem einzelnen Punkt als Projektion mit Tiefeninformation (*Point2D*) oder als Punkt im 3D-Raum (*Point3D*) zu speichern. Beim *Point2D* besteht jeder Punkte aus der X- und Y-Koordinate mit dem dazugehörigen Wert vom Tiefenbild. Dem Gegenüber steht der *Point3D*, welcher auf den selben Daten basiert, diese jedoch als Punkt im 3D-Raum repräsentiert. Die Umrechnung zwischen diesen beiden Formaten geschieht mittels dem OpenNI-Framework.

8.3.3 Abstraktion ForcePlate

Namespace: *KinectBodyscanner.HAL.ForcePlate.ForcePlateDevice*

Die Abstraktion der ForcePlate ermöglicht die Kapselung der spezifischen Vernier-API Methoden innerhalb einer Klasse. Hierfür wurde ein Interface verwendet, welches von der realen Abstraktion des Waage und auch von der Fake-Waage (siehe *Fake Hardware Devices* auf Seite 91) implementiert wird. Die Events *WeightDetected* und *WeightLost* sind insbesondere für die Erkennung von Personen auf der Waage relevant.

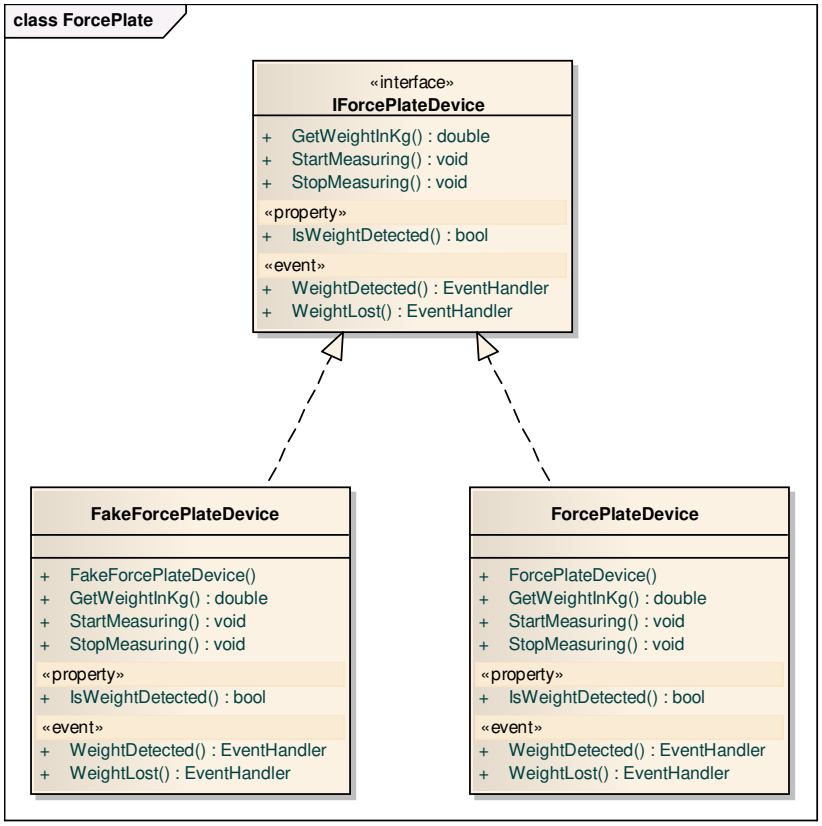


Abbildung (8.10) Klassenübersicht ForcePlate Abstraktion

8.3.4 Detektoren

Die folgenden Detektoren aggregieren Statusänderungen von Kinect und ForcePlate und stellen daraus resultierende Statusänderungen (siehe *Bedienprozess* auf Seite 57) zur Verfügung. Diese können mit Eventlisteners abonniert werden.

8.3.4.1 UserDetector

Namespace: *KinectBodyscanner.Logic.Detector.UserDetector*

Der „*UserDetector*“ ist gemäss der State-Machine implementiert (siehe *Generelle State-Machine* auf Seite 59). Die Benachrichtigung der abonnierten StateChange-Listeners geschieht über EventHandler. Der UserDetector kann hierbei die folgenden Status annehmen:

- **UserFullyAvailable**: Benutzer steht auf Wage und ist von Kinect erfasst

- **UserNotAvailable**: Es ist kein Benutzer vorhanden
- **UserFromKinect**: Der Benutzer wurde von der Kinect erfasst
- **UserFromPlate**: Der Benutzer steht auf der Wage

Der *UserDetector* registriert sich zudem an einem *UserDistanceDetector*. Wechselt der aktuell am definierte 3DPunkt nächsten stehende Benutzer, wechselt der *UserDetector* in den *UserNotAvailable*-Status und prüft danach, ob ein weitere Statusübergang angebracht ist.

8.3.4.2 UserDistanceDetector

Namespace: *KinectBodyscanner.Logic.Detector.UserDistanceDetector*

Der *UserDistanceDetector* wird mit einem 3DPunkt initialisiert. Dieser Punkt ist im Konfigurationsfile definiert. Ändert fortan ein Benutzer seine Distanz zu diesem Punkt und steht dadurch näher als alle anderen Benutzer, reagiert der *UserDistanceDetector*. Er benachrichtigt alle registrierten EventHandler über dieses Ereignis.

8.3.5 Fake Hardware Devices

Die Kinect und die Waage können auch mit aufgezeichneten Daten betrieben werden. Hierfür wurde für jedes der Geräte eine Fake-Implementation erstellt, welche vorher aufgezeichnete Daten abspielen kann.

8.3.5.1 RecordedKinectDevice

Beim *RecordedKinectDevice* handelt es sich um ein *KinectDeviceBase* welches vorher aufgezeichnete „*.oni“-Files abspielen kann[1]. Die „*.oni“-Files können mit OpenNI aufgezeichnet werden. Der Pfad zum abspielenden File wird über die Eigenschaft *KinectRecordedFile* in der Konfigurationsdatei gesteuert.

8.3.5.2 FakeForcePlate

Die FakeForcePlate verhält sich ähnlich wie eine *ForcePlateDevice*. Unterschied ist einzig die Herkunft der Daten. Bei dieser Implementation werden die Gewichtsdaten sekündlich und zeilenweise aus einem Textfile geladen. Der Pfad zu diesem Textfile wird wiederum über die Konfigurationsdatei gesteuert. Es ist der Eintrag mit dem klingenden Namen *ForcePlateRecordedFile*.

1 OpenNI/Samples/NI-Viewer.exe - Mit der Taste „S“ kann eine Aufzeichnung gestartet werden

8.3.6 Device Factory

Namespace: *KinectBodyscanner.HAL.Kinect*

Über die *DeviceFactory* werden Instanzen von *KinectDeviceBase* oder *ForcePlateDeviceBase* bzw. deren Ableitungen instanziiert. Die für die Instanziierung notwendigen Informationen wie Klassennamen und Assembly sind in der Konfigurationsdatei gespeichert. Die relevanten Einträge lauten wie folgt:

- **KinectDeviceType**: Klassenname und Assembly für *KinectDevice*
- **ForcePlateDeviceType**: Klassenname und Assembly für *ForcePlateDevice*

Der voll qualifizierte Klassenname und Assembly-Name sind dabei als einzelnen String mit Komma als Trennzeichen gespeichert. Beispiel für ein *RecordedKinectDevice*: „KinectBodyscanner.HAL.Kinect.RecordedKinectDevice, KinectBodyscanner.HAL.Kinect“. Das Assembly muss sich dabei entweder im *GAC*[1] oder im Execution-Path[2] befinden.

8.3.7 Performance Monitoring

Namespace: *KinectBodyscanner.HAL.Kinect*

Mit dem Performance Monitoring wird erreicht, dass sich jede Komponente überwachen lassen kann. Zentraler Punkt ist der PerformanceMonitor, bei welchem alle PerformanceMonitors registriert sind. Mit einem Intervall von 1 Sekunde werden diese Monitors abgerufen und die Werte dieser in ein Performance-Log geschrieben. Gleichzeitig werden die Werte Windows zur Verfügung gestellt. Diese können dadurch auch von externen Programmen (wie z.B. *perfmon*) überwacht werden.
Alle PerformanceMonitor-Implementationen leiten von *PerformanceMonitorBase* ab. Dadurch ist die Schnittstelle eines PerformanceMeters automatisch gegeben.
Je nach Anwendungsfall instanziiert und bedient eine zu überwachende Komponente eine der folgenden PerformanceMeter.

8.3.7.1 CountingPerformanceMeter

Der *CountingPerformanceMeter* setzt sich bei jedem Auslesen durch den PerformanceMonitor wieder auf null. Er eignet sich dadurch sehr gut, um die Anzahl Aktionen pro Sekunde zu zählen. Ein Beispiel hierfür sind die Frameraten der verschiedenen Komponenten.

1 *Global Assembly Cache, Zentraler Ort für Assemblies unter .NET*
2 Verzeichnis von welchem die Applikation gestartet wurde

```
// Neue Instanz mit Namen und Einheit
var meter = new CountingPerformanceMeter("KinectVideoControl", "FPS");

// Eintrag erhöhen, z.b. jedes Mal, wenn ein Frame gezeichnet wurde.
meter.Increment();
```

8.3.7.2 DelegatePerformanceMeter

Der *DelegatePerformanceMeter* ist universell einsetzbar. Mit diesem Meter können beliebige Werte im Sekundentakt an den PerformanceMonitor übermittelt werden. Beim Konstruktor muss dafür ein Delegate mitgegeben werden, welches einen double-Wert zurückliefert. Dieses Delegate wird dann vom PerformanceMonitor jede Sekunde aufgerufen.

```
// Neue Instanz mit Namen und Einheit
var meter = new DelegatePerformanceMeter("GCTotalMemory", "MB",
        (m) => (double)GC.GetTotalMemory(false) / 1024 / 1024); })
    );

// Wert setzen fällt dadurch weg, da der Wert immer
// über das Delegate gefunden wird.
```

9 Testing

9.1 Unit-Tests

9.1.1 Allgemein

Unser Fokus lag nicht darauf unser komplexes „Echtzeitsystem“ bis ins letzte Detail mit Unit-Tests zu testen. Denn ansonsten wäre unser Prototyp nicht fertig geworden.

Vielmehr war uns die Stabilität, welche wir am Ende durch einen ausgedehnten Performance und Endurance-Test überprüft haben, sehr wichtig.

Das ***KinectBodyscanner.Test***-Projekt beinhaltet alle unsere Unittests.

Um die Unit-Tests durchzuführen muss im Testprojekt das ***app.config*** entsprechend angepasst werden.
Nachfolgend eine Übersicht über die getesteten Komponenten.

9.1.2 Kalkuratoren

Hier werden die Klassen in ***KinectBodyscanner.Logic.Calculation*** getestet. Diese haben die Aufgabe zu überprüfen, ob die `Calculate`-Funktion einen bekannten und korrekten Wert als Ergebnis liefert.

9.1.3 Detektoren

Hier werden die Detektoren in ***KinectBodyscanner.Logic.Detector*** getestet. Diese überprüfen das Erkennen von Personen. Hierzu musste eine *TestableForcePlate* und eine *TestableKinectDevice* erstellt werden, diese können über eine `Fire...()`-Methode einen entsprechenden Event auslösen, welcher simuliert als ob eine Person auf die Waage stehen würde. Somit kann jeder beliebige Event kontrolliert augerufen auf den korrekten Status geprüft werden.

9.1.4 Kinect Device

Hier wird die wird die Kinectabstraktion anhand eines aufgezeichneten Files getestet (*app.config* anpassen). Diese befinden sich in ***KinectBodyscanner.HAL.Kinect***. Von der Device werden periodisch die aktuellen Frames abgeholt und gezählt. Unterschreiten die Anzahl Frames pro Sekunde und die Gesamtanzahl dem vorgegebenen Wert, so ist er nicht erfolgreich.

9.2 Usability Test

Da die Usability bei unserem Produkt sehr wichtig ist, haben wir uns schon sehr früh mit dem Thema auseinandergesetzt. Sobald die Aufgabe konkret formuliert wurde, haben wir bereits erste „Paper"-Prototypen erstellt. Weitere Verbesserungen entwickelten sich aus spontanen Tests unserer *BA*[1]-Zimmergenossen oder von uns selbst. Die Krönung der Usability-Tests war der zweitägige Test im Technorama.

9.2.1 Paper-Prototyp Nr. 1 - 14.4.2011

9.2.1.1 Vorgehen

Für das Testen eines interaktiven Prototypen eignete sich Papier nicht, so mussten wir eine alternative Variante zum testen des Ausstellungsstücks entwickeln. Wir bauten die verschiedenen Screens in Photoshop. Für jeden einzelnen Screen benutzen wir eine Layergruppe. Anschliessend konnten wir nach jeder Benutzerinteraktion die einzelnen Layergruppen ein- oder ausblenden.

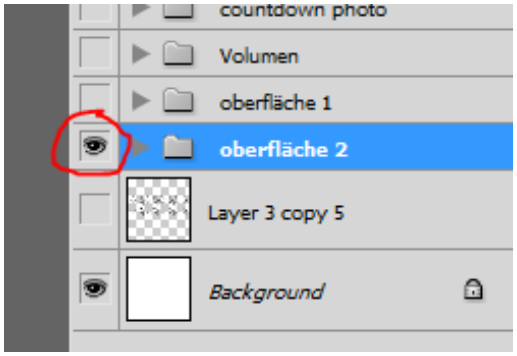


Abbildung (9.1) Photoshop Layers dynamisch ein- und ausblenden

1 *Bachelorarbeit*

9.2.1.2 Testumgebung

(a) Aufbau **(b)** Testperson

Abbildung (9.2) Testumgebung

Das Bild wurde mit einen Beamer auf die Orginalgrösse wie sie im Technorama wäre projektiert (Bildhöhe ca. 1.30m).
Die ForcePlate wurde mit einem Zettel versehen, auf dem steht „orange“, denn auf dem Bildschirm war die blinkende Platte auch „orange“.

9.2.1.3 Getestetes Szenario

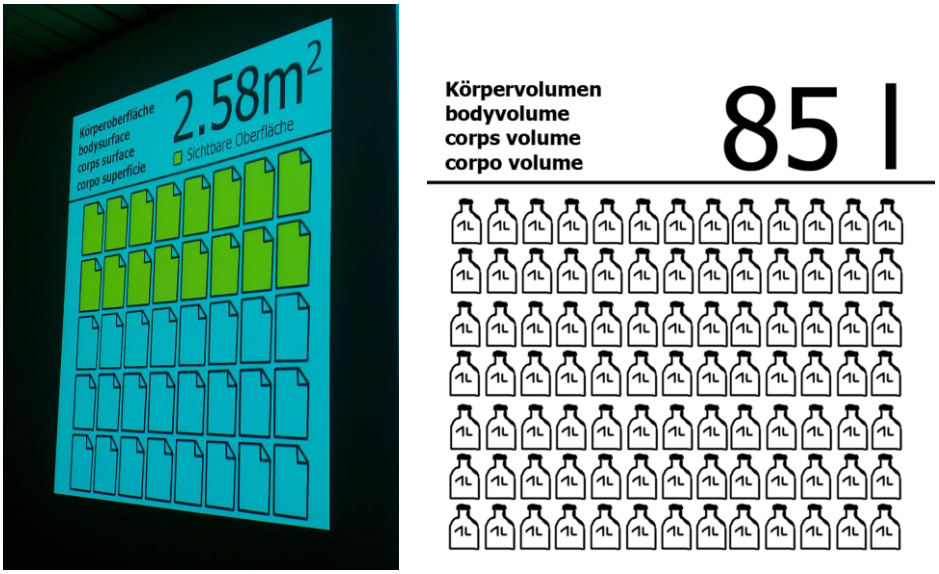
Der Tester wurde entsprechend informiert, dass er sich in der Ausstellung „Der vermessen(d)e Mensch“ im Technorama, vor einem interessanten Ausstellungsstück befindet, welches er gerne genauer betrachten/ausprobieren würde.

(a) Titelscreen

Abbildung (9.3) Titelscreen Szenario

(a) Countdown **(b)** Flash

Abbildung (9.4) Getestetes Szenario



(a) Oberflächen-Resultat **(b)** Volumen-Resultat

Abbildung (9.5) Auswertung

9.2.1.4 Probleme

Folgende Probleme sind aufgetreten:

- Auslöser zum starten des Vorganges, dass ein Foto aufgenommen wird wurde ein Bild mit einer Hand verwendet, dies ist nicht intuitiv
- Wieso muss überhaupt ein Foto manuell ausgelöst werden? Denn dies könnte ja ohne Interaktion direkt beim betreten der ForcePlate geschehen.
- Sichtbare Oberfläche nur in Deutsch beschriftet
- Es wurde nicht bei jedem Screen der Schatten der Person im Hintergrund angezeigt => Konsistenz verletzt

9.2.1.5 Redesign-Entscheide

Fotoauslöser weglassen
Der automatische Fotoauslöser wurde mit Beschluss vom Weeklymeeting W09 am 20. April gestrichen. Denn dies benötigt zusätzlich Aufwand vom Besucher, welcher aber nichts studieren will.
Wir müssen einen Countdown einbauen, welchen den Benutzer darauf hinweist, dass er aktuell vermessen wird. Zu Beginn war dieser auf zwei Sekunden eingestellt. Doch nach einigen Tests mit weiteren Personen wurde schnell klar, dass es nicht möglich ist, nur so

kurz eine Meldung anzuzeigen. Denn für den Benutzer geht dies viel zu schnell, wenn so kurz eine Meldung angezeigt wird. Er kann in dieser kurzen Zeit diese Information gar nicht verarbeiten. Also haben wir den Timer auf fünf Sekunden erhöht.

Die sichtbare Oberfläche wurde nur in Deutsch beschriftet
Das wäre für nicht deutsch-sprechende Personen ein Problem, wie sie auch oft im Technorama vorkommen, ein Problem. Als Lösung verwenden wir neu ein Icon. Zudem wird die sichtbare Oberfläche auch mit einem Quadratmeter-Wert angegeben. Steht dieser direkt unter der Gesamtoberfläche.

9.3 Systemtests

Am Ende der Entwicklungsphase haben wir uns mit den Systemtests auseinandergesetzt. Denn eine Grundstabilität für den Test im Technorama musste sichergestellt werden.

9.3.1 Lokale Testumgebung

Der Test fand auf einer folgenden Installation statt:

- Microsoft Windows 7 Enterprise (Service Pack 1)
- Microsoft Visual Studio 2010 Ultimate (V. 10.0.30319.1)
- GoIO_DLLdll (V. 2.41.0.0) / Vernier ForcePlate
- GoIOdotNET.dll (V. 1.0) / Vernier ForcePlate .NET Wrapper
- LibUsbDotNet.dll (V. 2.2.8.104) / Usb-Library (Kinect-Motor)
- OpenNI.dll (V. 1.0.0 Build 25) / Kinect-Treiber
- OpenNI.net.dll (V. 1.0.0.25) / Kinect-Treiber .NET Wrapper

Hinweise zur Installation siehe Anhang. (siehe *Installations Anleitung Kinect Treiber* auf Seite 201)

9.3.1.1 Umfeld

Die Tests fanden unter folgenden Bedingungen statt, damit das Fehlverhalten der Kinect eingeschränkt werden kann:

- Abgedunkelten Raum statt, ohne Sonneneinstrahlung
- ForcePlate wurde 2.5m entfernt von dem Kinect-Sensor aufgestellt, damit der Benutzer vollständig erkannt werden kann

9.3.2 Computerinstallation

Auf folgendem Computer wurde das System getestet.

- Fujitsu CELSIUS W380
- Prozessor: Intel(R) Xeon(R) CPU X3450 @ 2.67GHz, 2661 Mhz, 4 Core(s), 8 Logical Processor(s)
- Installed Physical Memory (RAM): 8.00 GB
- BIOS Version/Date FUJITSU // Phoenix Technologies Ltd. 6.00 R1.20.2917.A1, 18.08.2010

9.3.3 Vorlage

Die Tests basieren alle auf einer Vorlage, welche im Anhang aufgeführt ist. (siehe *Systemtest Vorlage* auf Seite 171)

9.3.4 Systemtest - 11.05.2011

(siehe *Systemtest 17.05.2011* auf Seite 174)

9.3.4.1 Resultate

Die Resultate waren im Grossen und Ganzen waren zufriedenstellend. Das Hauptproblem war, dass der Benutzer teilweise nicht richtig erkennt wird. Somit kann es sein, dass ein Benutzer auf der Waage steht und von der Kinect nicht erkannt wird, aber dennoch die Messung/Anzeige weiter läuft.
Genauers ist im oben genannten Protokoll aufgelistet.

9.3.4.2 Lösungsansatz

Die Lösung dafür liegt darin, wenn ein Benutzer auf der Kinect die ForcePlate verlässt, das nicht der aktive Benutzer auch „abgemeldet“ wird. Es sollte jedesmal beim verlassen der ForcePlate überprüft werden welcher Benutzer mit welcher Benutzer-ID die ForcePlate verlässt.

9.3.5 Systemtest - 04.06.2011

Dies war der finale Systemtest, nach dem Refactoring und Performance-Verbesserungen eingeflossen waren. Zur Verifizierung, damit sich nicht weitere Fehler eingeschlichen hatten (siehe *Systemtest 04.06.2011* auf Seite 177).

9.4 Technorama Besucher-Test - 18./19.05.2011

Wir hatten die Möglichkeit, unsere Software von realen Besuchern im Technorama testen zu lassen. Damit bis dahin ein gewisser Qualitätsstandard erreicht wurde, haben wir wie oben erwähnt die Systemtests durchgeführt.

Als erstes haben wir versucht die Besucher zu motivieren unser Ausstellungsstück zu benutzen. Bei grösseren Problemen während der Benutzung haben wir eingegriffen oder kleine Tipps gegeben. Zum Schluss haben wir die Freiwilligen um ein Feedback angefragt. Um das Feedback strukturiert zu analysieren haben wir mehrere Umfragebogen vorbereitet. Diese wurden verwendet, um die Besucher nach der Benutzung des Ausstellungsstückes zu befragen. Weiter haben wir die Besucher beobachtet und uns Notizen dazu gemacht.

9.4.1 Aufbau

(a) Ankündigung **(b)** Ankündigung Prototyp

Abbildung (9.6) Ankündigungen im Technorama

Abbildung (9.7) Testaufbau im Technorama

(a) Anzeige der drei Jungen

(b) In Aktion

Abbildung (9.8) Drei Jungen beim Benutzen der Anlage

Abbildung (9.9) Technorama Team

9.4.2 Umfragebogen 1

Dieser war hautpsächlich in offenen Fragen formuliert. Somit musste der Interviewer selbst das Gespräch leiten. Am ersten Tag, wurde mit diesem Fragebogen die Umfrage gemacht. (siehe *Benutzerfragebogen 1* auf Seite 159)

9.4.3 Umfragebogen 2

Dieser war ein Fragebogen, bei welchem der Interviewte spezifisch auf die Fragen in sechs verschiedenen Intensitäten (trifft voll zu bis trifft überhaupt nicht zu) antworten musste. Dieser wurde am zweiten Tag verwendet. (siehe *Benutzerfragebogen 2* auf Seite 161)

9.4.4 Auswertung der Interviews

Diese befinden sich im Anhang. (siehe *Auswertung und Aussagen Benutzerbefragung Technorama* auf Seite 163)

9.4.5 Resultate

Die wichtigsten Resultate nachfolgend.

9.4.5.1 Usability

Erwähnt in (siehe *Auswertung der Interviews* auf Seite 105).

9.4.5.2 Technisch

Folgende Probleme sind aufgetreten:

- Nach ca. 40 ganzen Benutzungen (jede Anzeige einmal angezeigt) wurde der Messbildschirm immer langsamer und stockte kurz vor Ende
- Nach ca. 40 ganzen Benutzungen (jede Anzeige einmal angezeigt) stockte das Schattenbild der Person enorm.
- Nach ca. 40 ganzen Benutzungen (jede Anzeige einmal angezeigt) kam es teilweise an verschiedenen Teilen im Code zu einer Out-Of-Memory-Exception

Analysiert unter (siehe *Memoryleak* auf Seite 106).

9.5 Performance-/Endurancetests

Da sich die Performance nach dem Technorama-Test als sehr grosses Problem herausgestellt hatte, mussten wir etwas dagegen unternehmen.

9.5.1 Vorgehen Endurancetest

Damit wir nicht andauernd auf der Waage und vor dem Kinect-Sensor hin- und her springen mussten, haben wir folgende Methode entwickelt.
Als erstes haben wir über das OpenNI-Viewer Programm, welches mit der OpenNI-Library mitgeliefert wird, ein komplexeres Szenario mit drei bis vier Personen aufgezeichnet. Dieses diente zusammen mit einem Simulationsfile für die Waage als Grundlage für unsere Tests.

Nun konnten wir über Nacht und übers Wochenende dieses automatische Testkonfiguration laufen lassen Anschliessend über das Logfile die Memorykonsumation und Performanceindikatoren der einzelnen Controls auswerten. Wurde vom 23. Mai 2011 bis am 6. Mai 2011 jede Nacht ein Endurancetest durchgeführt. Nach der Behebung des Memory-Leaks wurden keine Exceptions oder Performance-Probleme festgestellt.

9.5.2 Memoryleak

Der Indikator dafür, dass es sich um ein Memoryleak handelte war die mehrfach aufgetretene Out-Of-Memory-Exception. Nun wird im Code jede Sekunde geloggt, wieviel Memory aktuell verwendet wird. Dies schwankt sehr stark. Denn die KinectFrames, welche für eine Sekunde in einem Ringbuffer zwischengespeichert werden, benötigen sehr viel Memory. Darum sind die Memoryschwankungen sehr hoch, da der Garbage-Collegtor dementsprechend viel aufräumen muss.

9.5.2.1 Memoryusage - Problem

Wie wir in folgenden Grafiken erkennen, steigt die Auslastung des Memory rasant über die Anzahl der Messungen an. Bei ca. 1200 Megabytes liegt die Grenze, an welcher der Garbage-Collector immer öfters aufgeräumt wird. Schlussendlich an der 1200 MB Grenze, kann der Garbage-Collector nicht mehr abräumen und somit auch nicht mehr Speicher allozieren.
Nachfolgend haben wir mehrere Messungen aufgezeigt, welche das Beschriebene verdeutlichen.

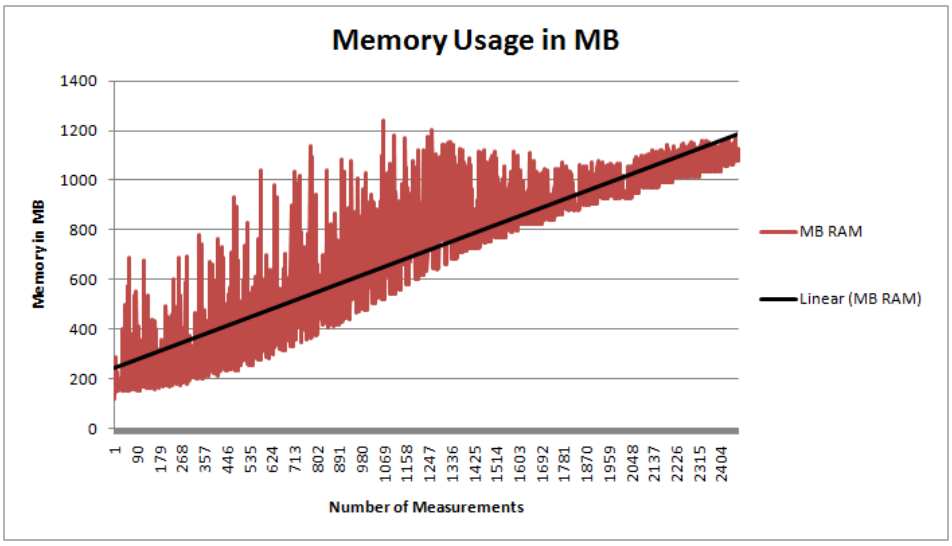


Abbildung (9.10) Memory Leak beim Versuch 1

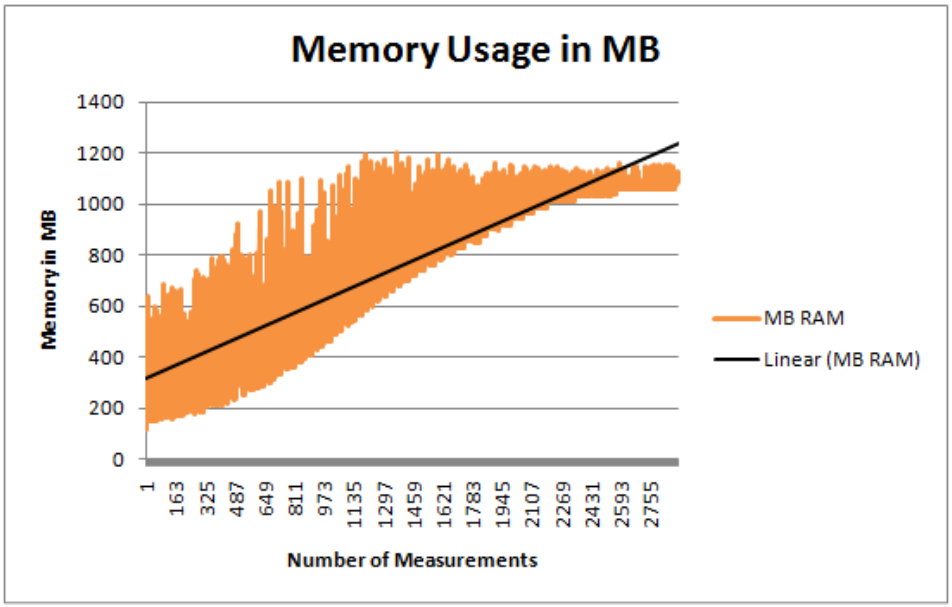


Abbildung (9.11) Memory Leak beim Versuch 2

9.5.2.2 Anzahl Instanzen

Nun mussten wir den Grund für die Memoryverschwendung finden. Hierzu haben wir den Memory Profiler 3.5[1] verwendet. Mit diesem konnten wir feststellen, dass die Total Instanzen stetig steigen. Daraus folgte die Erkenntnis war klar, dass gewisse Objekte nicht sauber abgeräumt wurden. Auf unsere Objekte herunter gebrochen zeigte sich, dass nur Objekte vom WPF-Projekt nicht sauber aufgeräumt wurden. Beispielsweise die A4-Pages und die Milchflaschen.

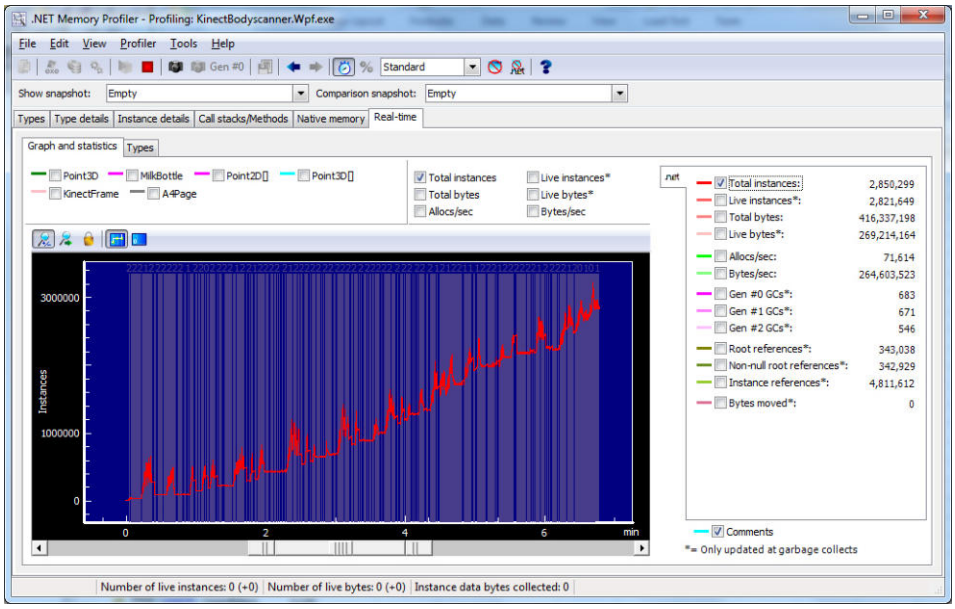


Abbildung (9.12) Number of Total Instances

1 Memory Profiler 3.5 http://memprofiler.com/

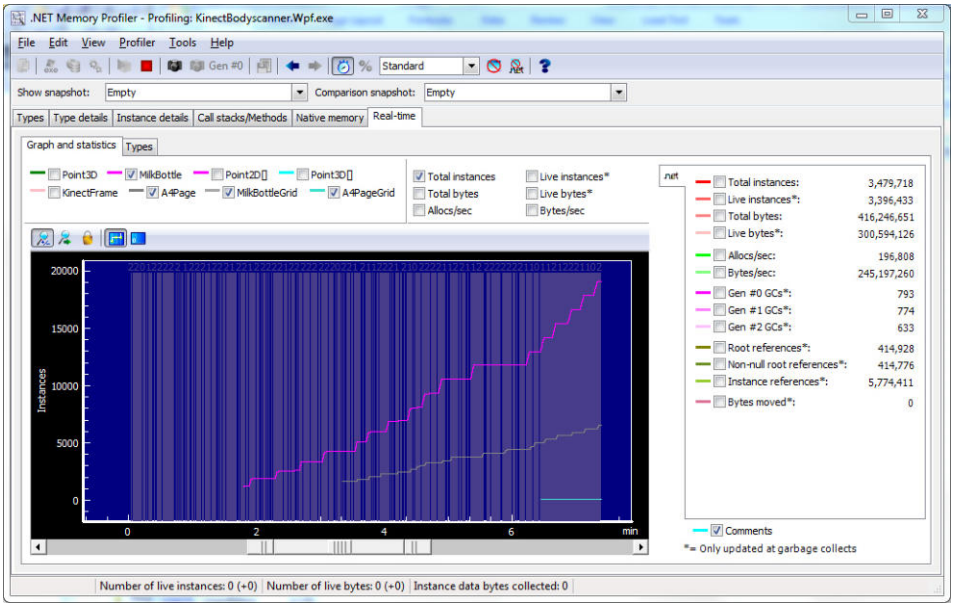


Abbildung (9.13) Number of specific Bodyscanner Instances

9.5.2.3 Lösung

Das Hauptproblem liegt darin, dass wir uns verschiedentlich auf ValueChanges bei ViewModels registriert haben. Die Deregistrierung stellten wir jedoch nicht sicher.
Dadurch blieben Referenzen auf EventHandler bestehen und bereits nicht mehr benötigte Controls blieben im Speicher vorhanden. Gleichzeitig wurden diese immer ordentlich mit Events versorgt und lasteten durch die Verarbeitung der Events die CPU zusätzlich aus. Weiter gab es noch einige Countdowns, die nicht gestoppt wurden, wenn die Seite verlassen wurde.
Wir haben nun alle Dispose-Methoden sauber implementiert, damit alle Countdown gestoppt werden und die Eventhandler für Änderungen der Dependency Properties ordentlich de-registriert. Nun rufen wir auch manuell, wenn das Objekt nicht mehr verwendet wird die Dispose-Methode auf.
Somit konnten wir, wie unten dargestellt die Anzahl Total Instanzen und die Totalen Bodyscanner spezifischen Instanzen minimal halten. Wie wir auch bei anschliessenden Memory-Messungen feststellen konnten, geht das Memory immer wieder auf den Ausgangswert zurückgeht.

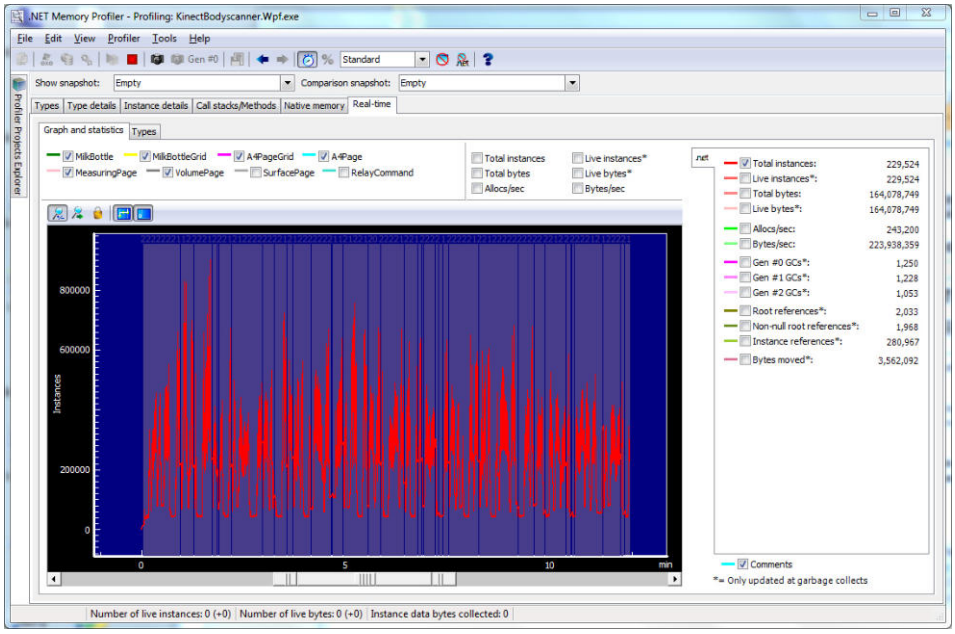


Abbildung (9.14) Number of Total Instances

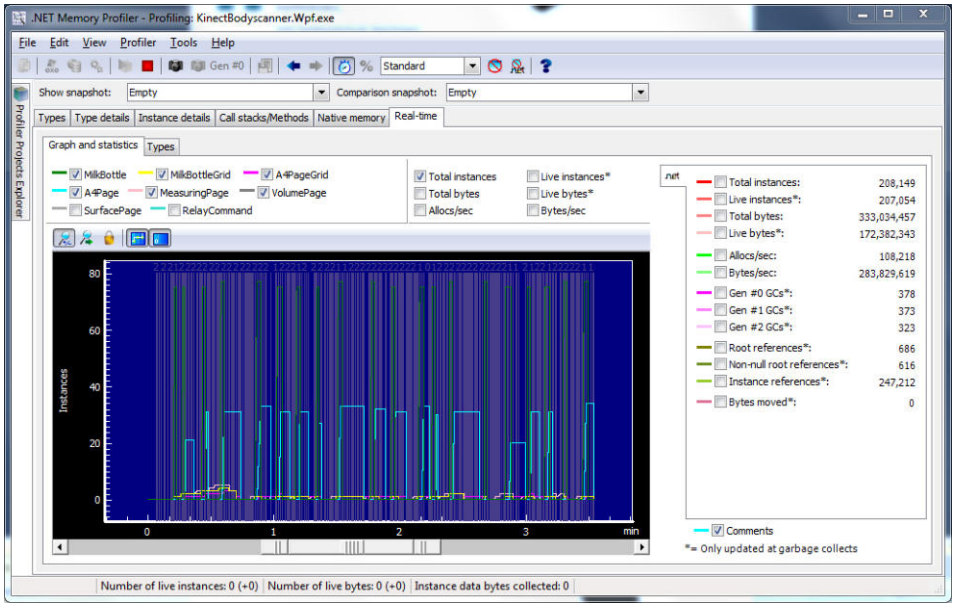


Abbildung (9.15) Number of specific Bodyscanner Instances

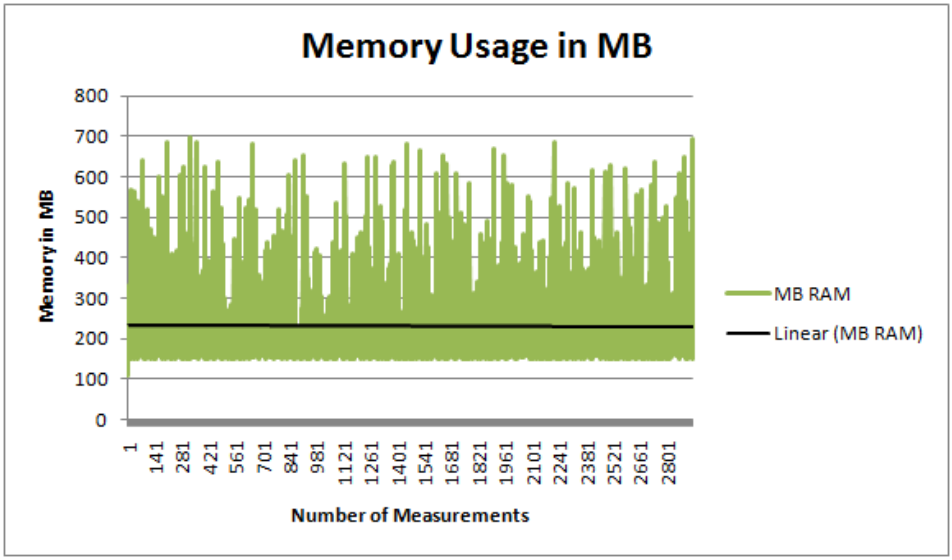


Abbildung (9.16) Memory Leak gelöst

9.5.3 Performancemonitor

Um die Leistungseigentschaften aktuell einzusehen, stellen wir dem Performance Monitor[1] die Messewerte zur Verfügung. (siehe *Performance Monitoring* auf Seite 92)

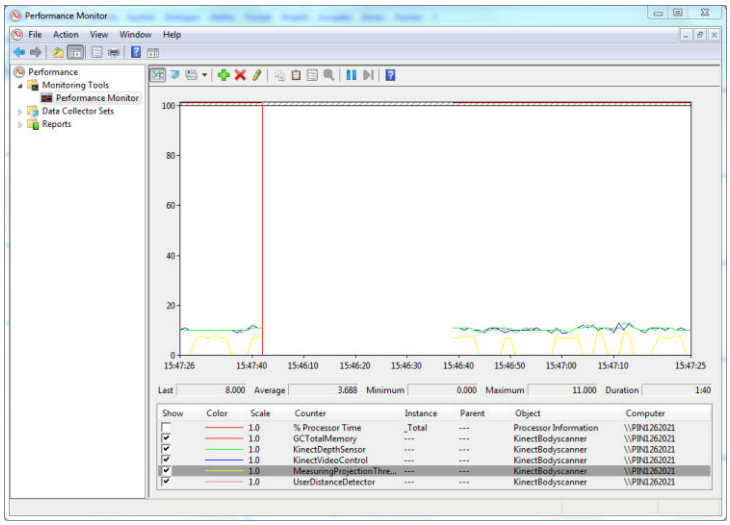


Abbildung (9.17) Performance Monitor

1 perfmon.exe - Standard Windows Komponente

10 Resultate und Weiterentwicklung

10.1 Resultate

10.1.1 Kurzfassung der Resultate

Es wurde in den folgenden Bereichen Resultate erzielt.

Technologie Möglichkeiten
Wir haben festgestellt, dass mit der MS Kinect nicht auf wenige Millimeter genau gemessen werden kann. Umwelteinflüsse, wie Sonnenlicht, Kleidung haben einen grossen Einfluss auf die Messgenauigkeit.
Das OpenNI-Framework bietet bereits Möglichkeiten, Benutzer zu erkennen und über mehrere Bilder zu verfolgen. Somit waren wir nun in der Lage eine Person dauerhaft auf Grösse und Spannweite auszumessen und weitere Berechnungen, wie zum Beispiel die sichtbare Oberfläche, durchzuführen.

Naturwissenschaftliches Wissen
Wir haben uns umfangreiches Wissen über den menschlichen Körper und dessen Vermessmethoden angeeignet. Auch über die Proportionen welche auch in der Malerei verwendet werden. Zudem konnten wir in Selbstversuchen Annahmen, über den menschlichen Körper, bestätigen oder verwerfen. Beispielsweise haben wir versucht das Volumen des menschlichen Körpers über messen der einzelnen Körperteile, zu ermitteln. Anschliessend haben wir versucht, dies in einen Selbstversuch zu verifizieren, in welchem wir in ein Wassertonne gestiegen sind und das verdrängte Wasser gemessen haben.

Prozess
In verschiedenen Phasen konnten wir mit unterschiedlichen Methodiken strukturiert und gezielt vorgehen. Es war nicht ein Prozess, wie er im Lehrbuch stand. Vielmehr war es ein Prozess den wir selber entwickelt und angewendet haben.

Ausstellungsobjekt
Das Hauptziel war es ein Prototyp eines Ausstellungsobjekts zu erstellen. Ein solcher wurde erstellt und sogar im Technorama getestet, eine Benutzerbefragung durchgeführt, in welcher wir genau herausfinden konnten, wo noch Verbesserungspotential vorhanden ist. Dem Benutzer steht soweit ein Ausstellungsstück zur Verfügung, welches ihm im Verlauf von mehreren Anzeigen erlaubt sein Körpervolumen in 1l Milchflaschen zu sehen. Weiter ist das Verhältnis von seinem aktuell sichtbaren, zu seiner gesamten Körperoberfläche in DIN-A4 Blätter zu sehen.

Software

Wir konnten nebst dem oder den verschiedenen Prototypen auch das Ausstellungsstück bauen. Wir konnten uns in verschiedenen Technologien im .NET-Bereich (z.B. WPF, Linq, usw.) Wissen aneignen. Es wurden zwei bzw. drei Hardware-Geräte angesprochen, welche parallel miteinander kommunizierten. Zudem sind auch Monitoring-Komponenten eingebaut, welche jederzeit ermöglichen den aktuellen Status des Systems zu ermitteln. Die Software wurde durch Endurance- und Performancetest auf Memoryleaks und Rechenleistung geprüft.
Leider konnten nicht alle unsere angedachten Möglichkeiten umgesetzt werden.

10.1.2 Funktionale Anforderungen

Auf Basis der ausgearbeiteten Szenarios wurden die gesetzten Ziele erreicht. Durch die daraus resultierenden Use Cases konnten die implementierten Funktionen wie folgt bewertet werden.

Die folgenden Use Cases wurden gemäss der Anforderungsspezifikation umgesetzt:

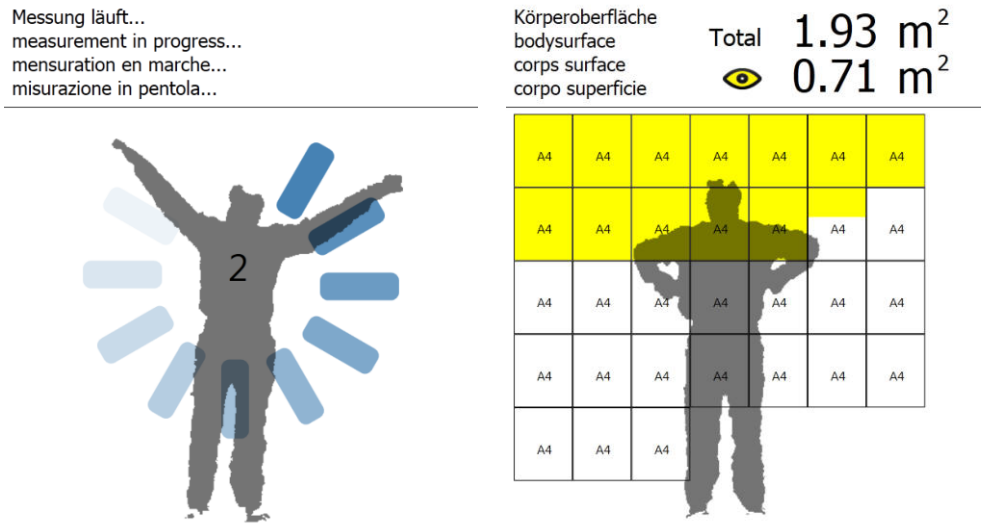
Use Case	*Erfüllungsgrad*
Messen durchführen und Messwerte anzeigen	erfüllt
System aus-tricksen/spielen	erfüllt
Anlage in Betrieb nehmen	Im Rahmen des Protoypen nicht speziell getestet
Anlage ausschalten	Im Rahmen des Protoypen nicht speziell getestet
Anlage neustarten	Im Rahmen des Protoypen nicht speziell getestet

Tabelle (10.1) Zielereichung Use Cases

Wie in der Tabelle ersichtlich, konnten alle geforderten Use Cases implementiert werden. Zusätzlich wurden folgende Funktionalitäten umgesetzt, welche in der Anforderungsspezifikation nicht eindeutig definiert wurden.

10.1.2.1 Messen durchführen und Messwerte anzeigen

Dies sind Ausschnitte der Anzeigen in welchen demonstriert wird, wie gemessen wird und anschliessend die Resultate angezeigt werden.

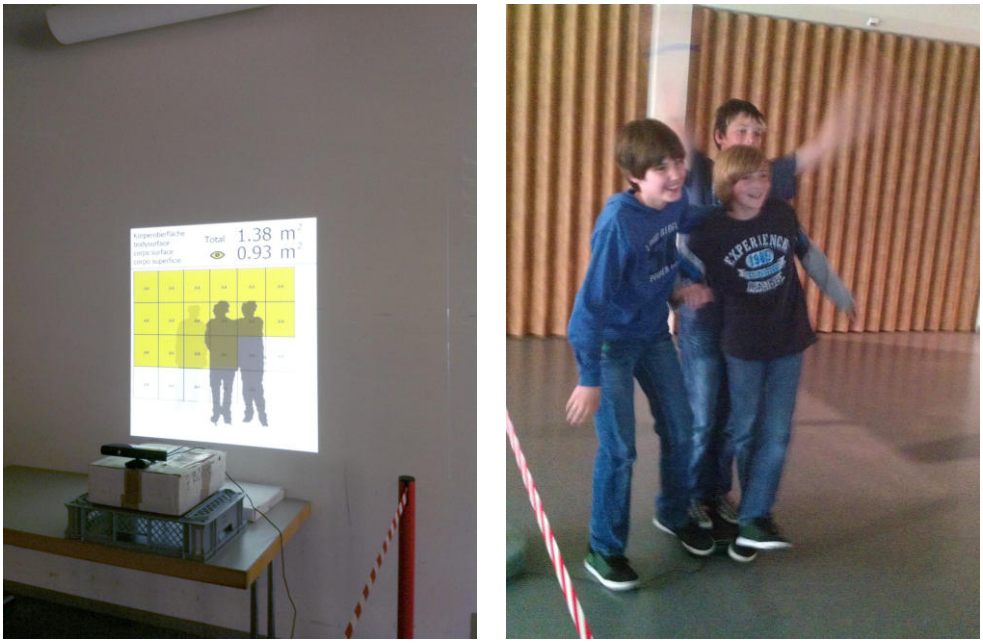


(a) Messung läuft... **(b)** Darstellung der Körperoberfläche

Abbildung (10.1) Messung und Darstellung der Resultate

10.1.2.2 System aus-tricksen/spielen

Der UseCase konnte vollumgänglich erfüllt werden, hier als Beispiel drei Jungen, die im Technorama mit dem System spielen und versuchen es zu überlisten.



(a) Sie haben es geschafft, dass zwei Personen als eine angezeigt wird

(b) Zu dritt auf der Waage

Abbildung (10.2) Drei Jungen beim Austricksen des Systems

10.1.3 Erfüllte nicht-funktionale Anforderungen

Die **Benutzerfreundlichkeit/Usability** konnten wir durch den Paperprototypen, wie auch durch den Test im Technorama sicherstellen. Das **Abschalten/Hochfahren des Systems** haben wir im Rahmen des Prototypen nicht berücksichtigt, stellt aber erfahrungsgemäss kein grosses Problem dar. Der **Dauerbetrieb** wurde erstmals vollumfänglich im Technorama getestet, erste Indikatoren auf Memoryleaks wurden verfolgt und Probleme beseitigt. Durch einen automatisierten Endurancetest und umfassende Logging-Informationen kann das System laufend überwacht werden. Zudem wird die **Reaktionszeiten/Durchsatz/Verfügbarkeit** gemessen, welche im Windows Performance Monitor hinzugefügt werden können.
Das System verfügt über eine **Konfigurationsdatei**, durch welche die wichtigsten Parameter der Applikation steuern lassen. Die **Hardwareschnittstelle** wurde soweit abstrahiert, damit diese möglichst einfach Austauschbar ist, um beispielsweise später dann das offizielle MS-Kinect-Framework einzusetzen. Die **Benutzerschnittstelle** ist aktuell in WPF umgesetzt. Diese hat sich auch sehr gut bewährt, da Microsoft-Technologien sehr gut zusammenspielen.

10.2 Weiterentwicklung

Anhand des Benutzertests im Technorama und Feedback des Ausstellungsleiters, konnten wir folgende Punkte identifizieren. Diese sind sehr detailliert und konkret anwendbar. Basiert auf dem Meeting vom 19. Mail 2011 (siehe *Sitzungsprotokolle* auf Seite 185)

10.2.1 Grafisch

- Ausnutzung der ganzen Projektionsfläche (wird ca. 2.62m Breit x 1.85m Hoch)
- DIN-A4 Blätter in Orginalgrösse darstellen
- Einfliege-Animation der DIN-A4 Blätter
- Nummerierung der DIN-A4 Blätter
- Klare Legende um Sichtbare/Unsichtbare/Totale Oberfläche darzustellen

10.2.2 Konzeptionell

- **Volumenanzeige entfernen, weil**
  - Verwirrt viele Benutzer
  - Ausstellungsstück wirkt zu überladen
  - Bei der Beschriftung des Ausstellungsstückes kann schon auf die Oberfläche und die DIN-A4 Blätter eingegangen werden

10.2.3 Technisch

- **Refactoring des UserDetectors**
  - Viele Aufgaben sind verteilt über UserDetector und UserDistanceDetector, welches klarer getrennt werden sollte
- **Businesslogik aus ViewModel in separate Komponente kapseln**

10.2.4 Fertigstellung

Das Ausstellungsstück wird in den Semesterferien vom 3. Juli bis Anfang August zum Ausstellungsstück weiterentwickelt. Das fertige Ausstellungsstück ist dann von Ende August 2011 bis Ende 2012 im Technorama in der Ausstellung „Der vermessen(d)e Mensch“ zu besichtigen.

11 Projektmanagement

11.1 Prozess

Der Prozess war keiner der direkt so im Lehrbuch steht, er setzte sich mehreren zusammen. Im folgenden Abschnitten wird erläutert, wie unser Prozess funktioniert hat und wie gearbeitet wurde.

Dieser Prozess war sehr agil, denn es flossen laufend Änderungen ein, welche priorisiert und angepasst werden mussten.

11.1.1 Zeitmanagement/Meilensteine

Das Zeitmanagement des Projektes bezieht sich auf die wichtigen 3 Meilensteine, vor jedem Meilenstein befindet sich jeweils eine Phase, welche anschliessend genauer beschrieben ist. (siehe *Projektplan* auf Seite 155)
Meilensteine

- **11. April 2011** Präsentation der Möglichkeiten die sich mit Kinect bieten, den Körper auszumessen und Fixierung der definitiven Aufgabe (Requirements Workshop)
- **18./19. Mai 2011** Prototypentest im Technorama
- **17. Juni 2011** Abgabe der BA

Die jeweilige Zeit vor den oben beschriebenen Meilensteine wurde dazu aufgewendet um an den jeweiligen Präsentationen/Tests, das bestmögliche Resultat zu erzielen.

11.1.1.1 Planung

Die Planung wurde mindestens wöchentlich geführt. Es wurden die offenen Punkte besprochen, ein Zeitfenster gesetzt, bis wann, was fertig sein soll. Dies hat meist folgendermassen Ausgesehen:

Abbildung (11.1) Vorgehem Planung

Working Items Das wichtigste der Planung, welche „Working Items“ gibt es, und wie ist deren aktuelle Stand.

Wichtige Punkte Hier wurden Notizen gesetzt welche bei der Arbeit besonders zu beachten sind.

Wochenplanung Hier wurde aufgezeichnet wer/wann am Arbeitsplatz ist, und was er dann machen wird.

11.1.2 Phase 1 - Forschung

Das Ziel dieser Phase ist, die grössten Risiken aus dem Weg zu schaffen um anschliessend konkrete Anforderungen an das Ausstellungsobjekt formulieren zu können.

Der Prozess wird auf den Risiken abgestützt. D.h. wird ein Risiko identifiziert, werden sofort nach Lösungen oder möglichen Workingitems gesucht um das Risiko zu minimieren.

Diese werden dann wie bei (siehe *Planung* auf Seite 119) beschrieben hinzugefügt und aufgeteilt.

Die Resultate dieses Prozesses sind unter (siehe *Evaluation* auf Seite 17) zu finden. Die Resultate der Prototypen (siehe *Prototypen* auf Seite 67).

11.1.3 Phase 2 - Prototypenentwicklung

Nach einem Requirements-Workshop, steht ein grober Ablauf für das Ausstellungsstück fest. Das Ausstellungstück wird in zwei Teilbereiche eingeteilt (Benutzeroberfläche/Architektur) in diesen Bereichen müssen Protoypen entwickelt. Auf folgende wichtige Bestandteile sollte grossen Wert gelegt werden.

- Usabilitytest damit die Benutzer wissen, mit was sie es zu tun haben
- Grobarchitektur
- Testing des Prototyp für stabilen Dauereinsatz

Der Abschluss dieser Phase stellt ein zweitägiger Benutzertest in der Liveumgebung dar, welcher auch eine Benutzerbefragung beinhaltet.

11.1.4 Phase 3 - Auswertung/Dokumentation

Nach dem Benutzertest werden Workingitems definiert um die folgenden Verbesserungen zu erzielen und die Übergabe des Produktes vereinfachen.

- Auswertung der Benutzerbefragung
- Analysieren der grafischen Verbesserungsmöglichkeiten
- Analysieren der Stabilität/Performance (z.B. Memoryproblem/Performance)
- Dokumentation der aktuellen Lösung

11.1.5 Weeklymeeting

Wichtige Entscheide und Arbeiten wurden jeweils am Weeklymeeting am Mittwoch morgen um 10:00 im IFS Rapperswil durchgeführt.

11.1.5.1 Protokollierung

Wichtige Entscheide und Protokolle sind alle im Anhang in den Sitzungsprotokollen hinterlegt. (siehe *Sitzungsprotokolle* auf Seite 185)

11.1.6 Standupmeeting

Standupmeetings wurden jeweils Montag und bei Bedarf Mittwochs durchgeführt. Dabei wurden die Tasks für die Woche aufbereitet und verteilt. (Dies wurde jeweils nur auf Papier festgehalten.)

11.2 Risiken

Anbei die Risiken, welche vor allem im ersten Teil des Projektes sehr wichtig waren. Diese wurden wöchentlich überarbeitet.

Risikoname	Device über Framework nicht ansprechbar			**ID**	R001
Risikostufe	Sehr Hoch	**Max. Schaden**	k.o.	**Erkannt**	21.02.2011
Status (Historie)	erledigt			**Abgeschlossen**	21.02.2011
Beschreibung	Es ist nicht sicher, ob das Device über ein in Qualität ausreichendes Framework ansprechbar ist. Nebst dem Framework selbst betrifft dies auch die Treiber und deren Qualität.				
Massnahmen	Es wurde das weit verbreitete OpenNI-Framework getestet und die Ansteuerung der Tiefensensor-Kamera des Kinect-Devices getestet. Weitere Frameworks wurden durch das baldige erscheinen des offiziellen SDK von Microsoft nicht evaluiert.				
Folgeaufgaben	W001 Funktionierendes Framework mit Treibern suchen				
Folgerisiken	R002 Device nur in C++ ansprechbar R004 Abstraktion in C# in-komplett oder zu langsam				

Risikoname	Device nur in C++ ansprechbar			**ID**	R002
Risikostufe	Sehr Hoch	**Max. Schaden**	150h	**Erkannt**	22.02.2011
Status (Historie)	erledigt			**Abgeschlossen**	23.02.2011
Beschreibung	Es gibt viele Frameworks mit nur C++ Schnittstelle. Wir benötigen jedoch ein Framework, welches mit C# ansprechbar ist.				
Massnahmen	Das gewählte Framework auf dieses Kriterium hin untersuchen und versuchen anzusprechen				
Folgeaufgaben	W002 Device mit C# ansprechen				
Folgerisiken	R004 Abstraktion in C# in-komplett oder zu langsam				

Risikoname	Keine Programmierbeispiele vorhanden, Schlechte Dokumentation			**ID**	R003
Risikostufe	Mittel	**Max. Schaden**	80h	**Erkannt**	21.02.2011
Status (Historie)	erledigt			**Abgeschlossen**	15.03.2011
Beschreibung	Das Framework liefert zu wenig genaue Beispiele und/oder ist schlecht dokumentiert. Es ist viel Reverse-Engineering notwendig.				
Massnahmen	Ersten Prototyp mit Tiefensensor-Kamera erstellen				
Folgeaufgaben	W003 Tiefensensor ansprechen, Demo erstellen				
Folgerisiken	R004 Abstraktion in C# in-komplett oder zu langsam				

Risikoname	Abstraktion in C# in-komplett oder zu langsam			**ID**	R004
Risikostufe	Mittel	**Max. Schaden**	50h	**Erkannt**	27.02.2011
Status (Historie)	erledigt			**Abgeschlossen**	19.05.2011
Beschreibung	C# oder das FW ist zu langsam um Live-Berechnungen durchführen zu können.				
Massnahmen	Prototyp in C# erstellen				

Risikoname	Waage mit C# ansprechen für Gewichtsmessung			**ID**	R005
Risikostufe	Mittel	**Max. Schaden**	20h	**Erkannt**	21.02.2011
Status (Historie)	erledigt			**Abgeschlossen**	21.02.2011
Beschreibung	Es muss eine geeignete Wage gefunden werden, welche mit C# angesprochen werden kann.				
Massnahmen	Prototyp in C# erstellen				
Folgeaufgaben	W004 Waage suchen und evaluieren W005 Waage mit C# ansprechen				

Risikoname	Messdaten sind ungenau (Kinect)			**ID**	R006
Risikostufe	Hoch	**Max. Schaden**	60h	**Erkannt**	21.02.2011
Status (Historie)	erledigt			**Abgeschlossen**	19.05.2011
Beschreibung	Es gibt zu viele Störfaktoren, welche eine exakte Messung de Höhe, Breite eines Benutzers mit dem Kinect-Device verunmöglichen.				
Massnahmen	Prototyp in C# erstellen Menschen vermessen und gegen testen *Hier wurde die Messanzeige so gewählt, dass dies bei möglicher Ungenauigkeit so gewählt, dass es nicht hundert Prozentig nachvollzogen werden kann des Besuchers.*				
Folgeaufgaben	W006 Höhen-, Breitenmessprototyp erstellen				

Risikoname	Entwickelte Volumenformel nicht anwendbar			**ID**	R007
Risikostufe	Mittel	**Max. Schaden**	20h	**Erkannt**	27.01.2011
Status (Historie)	eingetroffen - erledigt			**Abgeschlossen**	09.03.2011
Beschreibung	Die vom Team entwickelte Volumenformel ist zu ungenau.				
Massnahmen	Volumen über Gewicht errechnen und gegen prüfen. Volumen mit Test (Wasserverdrängung in Tonne) messen -> Genauer Volume <=> 1 zu 1				
Folgeaufgaben	W007 Personentest mit Wasserverdrängung durchführen.				

Risikoname	Anwendung (Ausstellungsobjekt) nicht intuitiv			**ID**	R008
Risikostufe	Mittel	**Max. Schaden**	20h	**Erkannt**	21.02.2011
Status (Historie)	teilw. eingetroffen			**Abgeschlossen**	19.05.2011
Beschreibung	Die Benutzer der Ausstellungsobjekts müssen dieses erfolgreich bedienen können innerhalb einer bestimmten Zeit.				
Massnahmen	Usability-Test Bentzertest im Technorama, Änderungen dokumentiert				
Folgeaufgaben	W008 Usability Test				

Risikoname	Nicht genügen Testpersonen / Zeit für umfangreichen Funktionstest (Genauigkeit der Angaben) vorhanden			**ID**	R009
Risikostufe	Mittel	**Max. Schaden**	20h	**Erkannt**	21.02.2011
Status (Historie)	erledigt			**Abgeschlossen**	19.05.2011
Beschreibung	Genug grosse Gruppen von Personen finden, mit welchen getestet werden kann. Diese müssen danach auch von Hand ausgemessen werden (das ist sehr zeitaufwändig)				
Massnahmen	Bentzertest im Technorama, Änderungen dokumentiert				

Risikoname	Grosser Testaufwand			**ID**	R010
Risikostufe	Mittel	**Max. Schaden**	40h	**Erkannt**	01.03.2011
Status (Historie)	erledigt			**Abgeschlossen**	19.05.2011
Beschreibung	Der Kinect-Sensor kann nicht gemockt werden. Es muss eine eigene Testumgebung geschaffen werden.				
Massnahmen	*Es wurde nicht komplett Automatisiert, nur die wichtigen Abläufe siehe Endurance Test*				

Risikoname	Missbrauchsfälle finden			**ID**	R011
Risikostufe	Niedrig	**Max. Schaden**	20h	**Erkannt**	01.03.2011
Status (Historie)	erledigt			**Abgeschlossen**	19.05.2011
Beschreibung	Es müssen verbreitete Missbrauchsfälle zur Umgehung der Messung gefunden werden. Das Team muss sehr kreativ sein. (Grosser Mantel, Höpfen um Grösser zu werden, Hand hochstrecken, etc.)				
Massnahmen	Workshop mit Studienkollegen durchführen, Testpersonen genau beobachten. *Im Technorama hat das Objekt jeglichen Missbräuchen standgehalten*				

Risikoname	Betriebsumgebung sub-optimal			**ID**	R012
Risikostufe	Mittel	**Max. Schaden**	20h	**Erkannt**	19.03.2011
Status (Historie)	erledigt			**Abgeschlossen**	25.03.2011
Beschreibung	Es müssen konstante Betriebsbedingungen (z.B. keine direkte Sonneneinstrahlung) herrschen, damit der Sensor gut funktioniert. Auch die Position des Sensors ist wichtig, dass das Bild möglichst nicht verzogen ist.				

Risikoname	Performance der Berechnung beeinträchtigt Frame-Rate stark			**ID**	R013
Risikostufe	Mittel	**Max. Schaden**	50h	**Erkannt**	01.03.2011
Status (Historie)	erledigt			**Abgeschlossen**	19.05.2011
Beschreibung	Durch die Berechnung der verschiedenen Distanzen, bzw. insbesondere die Erkennung der korrekten Punkte im Körper ist sehr rechenintensiv. Dadurch sinkt die Frame-Rate für die Darstellung der Person am Bildschirm. Die Usability sinkt.				
Massnahmen	Berechnung nur in Standbildern machen und kein Realtime anzeigen der Grössen Brechung auf mehrerer Threads aufteilen *Wird lediglich noch die Oberfläche Realtime gerechnet, Performance keine Probleme*				

Risikoname	3D Modell ungenau, Punkteerkennung schwierig			**ID**	R014
Risikostufe	Mittel	**Max. Schaden**	20h	**Erkannt**	27.01.2011
Status (Historie)	erledigt			**Abgeschlossen**	19.05.2011
Beschreibung	Das 3D Modell ist stark verrauscht, was es schwierig macht einzelne Punkte als repräsentative Punkte zu verwenden. Geeignete Mathematische Lösungen sind für 2D Rauschen konzipiert oder sehr komplex.				
Massnahmen	*Wird nicht mehr verwendet, da immer der Durchschnitt über mehrere Frames verwendet wird*				

Risikoname	Volumenberechnung Mittels Projektion ungenau			**ID**	R015
Risikostufe	Hoch	**Max. Schaden**	20h	**Erkannt**	23.03.2011
Status (Historie)	eingetreten			**Abgeschlossen**	19.05.2011
Beschreibung	Für die Berechnung des Volumens wird die Oberfläche verwendet, jedoch ist die Auflösung zu niedrig oder die Berechnung nicht möglich.				
Massnahmen	*Wird nicht mehr verwendet, da das Volumen 1:1 zum Gewicht durch die Waage verwendet wird*				

Risikoname	Ansynchronitätsprobleme			**ID**	R016
Risikostufe	Hoch	**Max. Schaden**	20h	**Erkannt**	08.04.2011
Status (Historie)	teilweise eingetreten			**Abgeschlossen**	19.05.2011
Beschreibung	Das asynchrone Auslösen der Events bei der Abstraktion des Kinect-Sensors / Force-Plate bereitet den registrierten Event-Handlern ungeahnte Probleme.				
Massnahmen	*Benötigte mehr Zeit für die Analyse des Problems*				

11.3 Q-Massnahmen

Folgende Q-Massnahmen werden durchgeführt. Diese sind alle in der Dokumentation protokolliert.

Massnahme	*Umsetzung*
Codereview	Wöchentlich während der Planung werden Codeerfahrungen und aktuelle Probleme besprochen. (nicht explizit Protokolliert)
Codeanalysetool	NDepend wird eingesetzt um Codemetriken und zirkuläre Abhängigkeiten usw. zu testen.
Risikomanagement	Jede Woche wird die Risikoliste neu bewertet und aktualisiert. (siehe *Risiken* auf Seite 122)
Unittests	Für die wichtigsten Komponenten (Algorithmen) werden Unittests geschrieben. (siehe *Unit-Tests* auf Seite 95)
Systemtests	Zweimal werden Systemtests durchgeführt. (siehe *Systemtests* auf Seite 100)
Papierprototypen	Damit unser Interface benutzerfreundlich wird, werden Papierprotoypen erstellt. (siehe *Paper-Prototyp Nr. 1 - 14.4.2011* auf Seite 96)
Interviews und Anforderungsworkshops	Damit die Anforderungen für unser Projekt nicht aus der Luft gegriffen werden, wurden Interviews und Anforderungsworkshops durchgeführt. (siehe *Sitzungsprotokolle* auf Seite 185)
Usabilitytests	Mehrmals während des Projekts werden Usabilitytests durchgeführt. (siehe *Usability Test* auf Seite 96)
Endurance/Performance-Tests	Da das Programm viel mit Echtzeit und längere Zeit laufen muss wurden Endurance & Performance-Tests durchgeführt. (siehe *Performance-/Endurancetests* auf Seite 106)
Livetest im Technorama	Testen des Prototyps im Technorama, ob Verständlichkeit und Bedienbarkeit des Ausstellungsstückes gewährleistet ist. (siehe *Technorama Besucher-Test - 18./19.05.2011* auf Seite 102)

Tabelle (11.1) Q-Massnahmen

11.4 Team und Verantwortlichkeiten

Das Team besteht aus zwei Personen. Michael Schnyder (m2) und Felix Egli (fx).

(a) Felix Egli **(b)** Michael Schnyder

Abbildung (11.2) Projektteam

Bereich/Dokument	*Verantwortlich*
Projektplan Excel	m2
Zeiterfassung	m2
Q-Massnahmen	fx
LaTex	fx
Konfigurationsmanagement	m2
Prototypen	beide
Technorama-Test	fx
Anforderungen	fx
Technologieabklärungen	beide
Architektur	m2
GUI-Implementation	fx
GUI-Prototypen	fx
Performance/Endurance-Tests	beide
Usability-Tests	fx
Dokumentation	beide

Tabelle (11.2) Verantwortlichkeiten

12 Projektmonitoring

12.1 Soll-Ist-Zeitvergleich

Folgend der Soll-Ist Zeitvergleich. Die Sollwerte orientieren sich am wöchentlichen Durchschnittszeit die aus den ECTS-Punkten abgeleitet werden kann. Die Ist-Zeiten überragen deutlich die Soll-Zeiten.

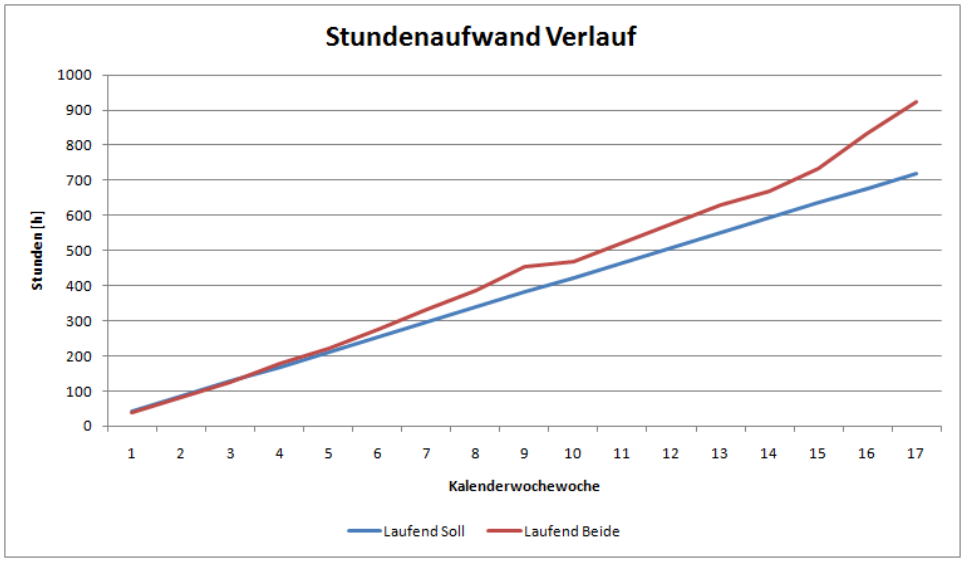


Abbildung (12.1) Stundenaufwand kumuliert

Wie hier unschwer erkennbar ist, handelt es sich hier um die geleisteten Stunden der einzelnen Teammitglieder. Felix Egli, hat in der Gesamtzeit etwas weniger Zeit aufgewendet, da er noch eine Woche während der Arbeit, Militärdienst leisten musste.

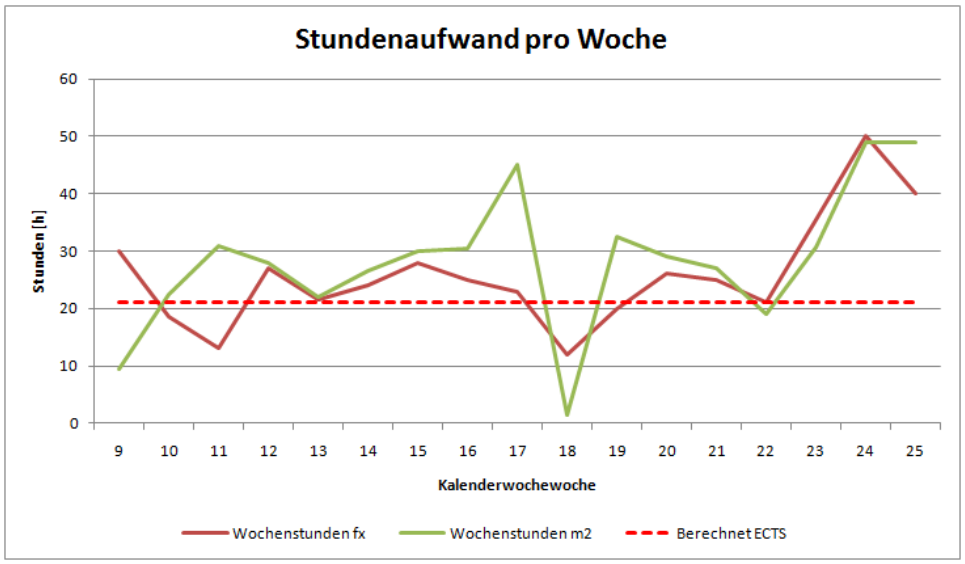
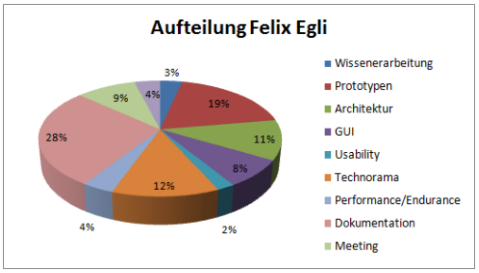
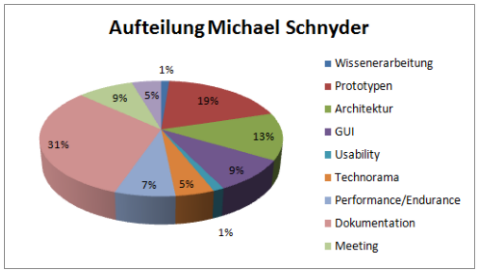


Abbildung (12.2) Stundenaufwand pro Woche

12.1.1 Haupttätigkeiten



(a) Haupttätigkeiten Felix Egli



(b) Haupttätigkeiten Michael Schnyder

Abbildung (12.3) Haupttätigkeiten

Wie aus den obigen Diagrammen hervorgeht, gibt es bei den Zeiten die in den dargestellten Bereichen, keine markanten Unterschiede. Was ziemlich heraus-sticht, ist die Dokumentation, welche ca. 30% der ganzen Arbeitszeit ausmacht. Weiter die grössten Pakete waren Prototypen, Architektur und GUI. In diesen Paketen steckt auch die Hauptentwicklungszeit des Projektes.

12.2 Codestatistik

Folgende Codemetriken wurden mit dem *NDepend* erstellt.

Kinect-Bodyscanner Solution

Beschrieb	*Anzahl*
Assemblies	8
Namespaces	25
Types	90
Methods	555
Fields	331
Classes	81
Abstract classes	3
Interfaces	1
C# source files	81
IL instruction	11,294
Lines of code	1,391
Lines of comment	728

Tabelle (12.1) Kinect-Bodyscanner Codemetriken

Die komplette Auswertung befindet sich auf der CD. Gestartet wird es mit Extras/NDepend_Auswerung/NDependReport.html.

12.3 Protokolle

Die Protokolle sind im Anhang abgelegt. (siehe *Sitzungsprotokolle* auf Seite 185)

13 Softwaredokumentation

13.1 Installation

Die für die folgende Installation benötigten Dateien befinden sich im Ordner „drivers“ auf der CD. Sofern nicht anders beschrieben befindet sich die benötigten Dateien in diesem Verzeichnis.

Für die Verwendung der Software wird empfohlen einen PC mit min 4 CPU-Core's und 8 GB Speicher zu verwenden. Genauere Angaben siehe (siehe *Lokale Testumgebung* auf Seite 100).

13.1.1 Kinect Treiber und Frameworks

Die Installation ist in im Anhang beschrieben (siehe *Installations Anleitung Kinect Treiber* auf Seite 201)

13.1.2 ForcePlate

How to Install Vernier Force Plate:

- Download the LoggerLite from Vernier or use the seupt delivered with this installation guide
- Download the Go! Software Development Kit if you need samples (is not required by the Program). When installed, the LED will change from Orange to Green

13.1.3 Einrichtung in Visual Studio 2010

Die Anleitung im Anhang (siehe *Einrichtungsanleitung Solution* auf Seite 207) beschreibt die korrekte Einrichtung des Projekts unter Visual Studio 2010 sowie den Buildvorgang.

Literaturverzeichnis

[AC97] Aggarwal, J.K. und Cai, Q.: Human motion analysis: a review, in: *Nonrigid and Articulated Motion Workshop, 1997. Proceedings., IEEE*, S. 90 –102

[Bai91] Bailey, Covert: *The New Fit or Fat*, Boston: Houghton-Mifflin (1991), iSBN 0395272181 / 9780395272183 / 0-395-27218-1

[KEA03] Keskin, C.; Erkan, A. und Akarun, L.: Real time hand tracking and 3d gesture recognition for interactive interfaces using hmm, in: *In Proceedings of International Conference on Artificial Neural Networks*

[KM77] Katch, F. und McArdle, W.: *Nutrition, Weight Control, and Exercise*, Boston: Houghton-Mifflin (1977)

[MTHC03] Mikic, Ivana; Trivedi, Mohan; Hunter, Edward und Cosman, Pamela: Human Body Model Acquisition and Tracking Using Voxel Data. *International Journal of Computer Vision* (2003), Bd. 53: S. 199–223, URL http://dx.doi.org/10.1023/A:1023012723347, 10.1023/A:1023012723347

[Pri10] Primesense: *Prime Sensor™NITE 1.3 Algorithms notes* (2010)

[PSH97] Pavlovic, V.I.; Sharma, R. und Huang, T.S.: Visual interpretation of hand gestures for human-computer interaction: a review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* (1997), Bd. 19(7): S. 677 –695

[Rov92] Rovin, Jeff: *Laws of Order: A Book of Hierarchies, Rankings, Infrastructures*, Ballantine Books (1992)

[SZGY00] Sheikh-Zade, Yu. und Galenko-Yaroshevskii, P.: Mathematical model of human body surface area. *Bulletin of Experimental Biology and Medicine* (2000), Bd. 129: S. 304–305, URL http://dx.doi.org/10.1007/BF02433914, 10.1007/BF02433914

[UMIO01] Ueda, E.; Matsumoto, Y.; Imai, M. und Ogasawara, T.: Hand pose estimation for vision-based human interface, in: *Robot and Human Interactive Communication, 2001. Proceedings. 10th IEEE International Workshop on*, S. 473 –478

[Wil77] Wilmore, Jack H.: *Athletic Training & Physical Fitness: Physiological Principles & Practices of the Conditioning Process*, Boston: Allyn & Bacon (1977)

Glossar

Allometrie	Befasst sich mit den Abhängigkeiten von Körpergrössen untereinander
Center Of Mass	Als Körpermittelpunkt (engl. Center of Mass) wird der Punkt eines Körpers bezeichnet, welcher sich genau in der Mitte des Körpers befindet
CTS	Mit dem Signal CTS auf der RS232-Schnittstelle signalisiert ein Gerät dem PC, dass es für die Übertragung von Daten bereit ist.
Global Assembly Cache	Im Global Assembly Cache lassen sich Assemblies registrieren, welche danach Systemweit von allen .NET-Applikation verwendet werden können, unabhängig von dem Pfad der Applikation.
HAL	Eine Hardwareabstraktionsschicht (von engl. Hardware Abstraction Layer) isoliert die zugrundeliegende Hardware durch eine einheitliche API
NDepend	NDepend is a Visual Studio tool to manage complex .NET code and achieve high Code Quality. With NDepend, software quality can be measured using Code Metrics, visualized using Graphs and Treemaps, and enforced using standard and custom Rules.
NITE	Middleware für OpenNI um Benutzer und Gesten zu erkennen.
NUI	Unter NUI versteht man ein eine Interaktionsform mit dem PC ohne Maus und Tatsatur
OpenNI	Das OpenNI-Framework von Prime Sense stellt eine API zur Verwendung unterschiedlicher NUI-Sensoren zur Verfügung.
perfmon	Mit dem Performance Monitor kann unter Windows der Systemzustand anhand verschiedener Indikatoren überprüft werden.

Perzentil	Durch Perzentile (lat. 'Hundertstelwerte'), auch Prozentränge genannt, wird die Verteilung in 100 gleich große Teile zerlegt.
PointCloud	Als Pointcloud wird eine Menge von 3D-Punkten bezeichnet, welche durch ihre X-, Y- und Z-Komponenten räumlich dargestellt werden können
Primesense	Prime Sense ist der Hersteller des Referenzdesigns der Kinect und Initiant des OpenNI-Frameworks
Projektion	Als Projektion bezeichnet man die Sicht auf eine 3D-Darstellung von einer bestimmten Richtung aus. Hierbei geht eine Dimension verloren.

Abkürzungsverzeichnis

BA Bachelorarbeit

CoM Center Of Mass ist die englische Bezeichnung für den Mittelpunkt eines Körpers

CTS Clear to Send-Signal eines Modems, sobald Daten gesendet werden können

GAC Global Assembly Cache, Zentraler Ort für Assemblies unter .NET

HAL Hardware Abstraction Layer

IDE Integrierte Entwicklungsumgebung (engl. integrated development environment)

NHI Natural Human Interaction, engl. für Natürliche Benutzungsinteraktion

NUI Natural User Interface, engl. für Natürliche Benutzungsoberfläche

RNG RING-Signal eines Modems, bei einem eingehenden Telefonanruf

SDK Software Development Kit

Abbildungsverzeichnis

Tabellenverzeichnis

Anhang A

Aufgabenstellung

Aufgabenstellung BA Michael Schnyder & Felix Egli

Ausstellungsobjekt für das Technorama mit MS Kinect

0. Auftraggeber und Betreuer

Praxispartner und informeller Auftraggeber diese Bachelorarbeit ist das Technorama Winterthur.

Ansprechpartner Auftraggeber:
Thorsten D. Künnemann TKuennemann@technorama.ch
Direktor Swiss Science Center Technorama
Tel.: +41(0)52 244 08 44

Betreuer HSR:
Prof. Dr. Markus Stolze, Institut für Software mstolze@hsr.ch

1. Ausgangslage

Das Technorama startet im Frühjahr 2011 eine Ausstellung „Der vermessene Mensch". Als Teil dieser Ausstellung wollte das Technorama ursprünglich einen kommerziellen 3D Scanner nutzen um die Körpermasse von Besuchern zu erheben und Informationen wie Körpervolumen, Oberfläche und Grösse anzuzeigen. Aufgrund der hohen Kosten kommerzieller Laser-Scanner wurde dieser Plan vom Technorama verworfen.

Aus dieser Situation ergab sich die Gelegenheit für die HSR eine Bachelorarbeit auszuschreiben in der die Verwendbarkeit des Microsoft Kinect Sensors für die 3D Analyse von Besuchern getestet wird.

Der Kinect Sensor wurde von Microsoft für die Gebärden-Steuerung von XBox Spielen entwickelt. Seit Anfang 2011 sind für diesen Sensor Open Source Libraries verfügbar welche es erlauben die 3D Informationen an einem angeschlossenen PC auszulesen.

2. Ziele der Arbeit

Für das Technorama soll prototypisch ein Ausstellungsobjekt entwickelt werden, welches dem Besuchern Informationen zu seinem Körper anzeigt.

Die hierfür notwendigen Informationen sollen mittels einem Microsoft Kinect Sensor (und ggf. weitere technische Mittel wie Waage, Distanzmesser) gesammelt werden.

Für das Ausstellungsobjekt sind gute Verständlichkeit der dargestellten Informationen, einfache Bedienbarkeit und hohe technische Robustheit wichtige Eigenschaften. Das Ausstellungsobjekt soll so ausgelegt sein, dass im Regelfall Technorama Besucher im Alter von 7 bis 70 Jahren in der Lage sein sollen ohne

Lesen einer Anleitung die Funktion des Ausstellungsobjektes zu erfassen und innerhalb von 4 Minuten eine aus ihrer Sicht bereichernde Interaktion mit dem Ausstellungsobjekt abschliessen können. Besucher sollen, soweit sinnvoll, zur korrekten Bedienung des Systems angeleitet werden und auf die vom System getroffene Annahmen hingewiesen werden. Weiterhin kann auf Möglichkeiten zur Verbesserung der Resultate hingewiesen werden (Manipulationssicherheit). Das Ausstellungsobjekt soll zudem für den Dauerbetrieb ausgelegt sein. Dies bedeutet, zum Beispiel, dass 8 Stunden Betrieb ohne Absturz überstanden werden, und das Objekt durch Schalten der zentralen Stromzufuhr im Technorama an- und abgeschaltet werden kann (niedriger Wartungsaufwand im Betrieb).

Es ist wahrscheinlich, dass das System welches im Rahmen der BA erstellt wird nicht alle diese Eigenschaft vollständig erreichen kann. Es sind aber trotzdem im Projektablauf geeignete Massnahmen zu planen um den Stand des Systems bezüglich Verständlichkeit, Bedienbarkeit, und Robustheit verfolgen und dokumentieren zu können. Hierbei dürfen auch Priorisierungen getroffen werden. So ist zum Beispiel eine umfangreiche grafische Umsetzung nicht zentral. Die Darstellung der Daten, sowie das Körpers kann in der ersten Ausbaustufe minimal gehalten werden. Diese Vereinfachungen sollten aber nicht die grundsätzliche Machbarkeit des System in anderen Bereichen in Frage stellt. Der Erreichungsgrad des Systems bezüglich der Kriterien Verständlichkeit, Bedienbarkeit, und Robustheit ist vorgängig und in Absprache mit dem Betreuer zu operationalisieren und der Stand bei Abgabe der BA zu dokumentieren.

3. Zur Durchführung

Mit dem HSR-Betreuer finden in der Regel wöchentliche Besprechungen statt. Zusätzliche Besprechungen sind nach Bedarf durch die Studierenden zu veranlassen.

Da das Technologierisiko relativ hoch ist (z.B. kein offizielles SDK's zu Kinect) werden die Zielsetzungen der Arbeit während dem Projekt in Absprache mit dem Dozenten laufend angepasst.

Alle Besprechungen sind von den Studenten mit einer Traktandenliste vorzubereiten und die Ergebnisse in einem Protokoll zu dokumentieren das dem Betreuer E-Mail zugestellt wird.

Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. An Meilensteinen gemäss Projektplan sind einzelne Arbeitsresultate in vorläufigen Versionen abzugeben. Über die abgegebenen Arbeitsresultate erhalten die Studierenden ein vorläufiges Feedback. Eine definitive Beurteilung erfolgt aufgrund der am Abgabetermin abgelieferten Dokumentation. Die Evaluation

erfolgt aufgrund des separat abgegebenen Kriterienkatalogs in Übereinstimmung mit den Kriterien zur BA Beurteilung. Es sollten hierbei auch die Hinweise aus dem abgegebenen Dokument „Tipps für die Strukturierung und Planung von Studien-, Diplom- und Bachelorarbeiten" beachtet werden.

4. Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen. Die zu erstellenden Dokumente sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollten den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Die Dokumentation ist vollständig auf CD/DVD in 2 Exemplaren abzugeben. Zudem ist eine kurze Projektresultatdokumentation im Wiki von Prof. Stolze zu erstellen dies muss einen Link auf einen YouTube Video enthalten welche das Resultat der Arbeit dokumentiert.

5. Termine

Siehe auch Terminplan auf
https://www.hsr.ch/Termine-Diplom-Bachelor-und.5142.0.html

21.02.2011	Beginn der Studienarbeit, Ausgabe der Aufgabenstellung durch die Betreuer.
Mai 2011	Fotoshooting. Genauere Angaben erteilt die Rektoratsassistentin rechtzeitig.
10.06.2011	Die Studierenden geben den Abstract/Kurzfassung für die Diplomarbeitsbroschüre zur Kontrolle an ihren Betreuer/Examinator frei. Die Studierenden erhalten vorgängig vom Studiengangsekretariat die Aufforderung zur Online-Erfassung der Kurzfassung. Die Studierenden senden per Email das A0-Poster zur Prüfung an ihren Examinator/Betreuer. Vorlagen stehen unter den allgemeinen Infos Diplom-, Bachelor- und Studienarbeiten zur Verfügung.

15.06.2011	Der Betreuer/Examinator gibt das Dokument mit den korrekten und vollständigen Abstract/Kurzfassung zur Weiterverarbeitung an das Studiengangsekretariat frei.
17.06.2011	Abgabe des Berichtes an den Betreuer bis 12.00 Uhr. Fertigstellung des A0-Posters bis 12.00 Uhr.
17.06.2011	HSR-Forum, Vorträge und Präsentation der Bachelor- und Diplomarbeiten, 13 bis 18 Uhr
08.08. - 27.08.2011	Mündliche BA-Prüfung
30.09.2011	Nachmittag Bachelorfeier und Ausstellung der Bachelorarbeiten

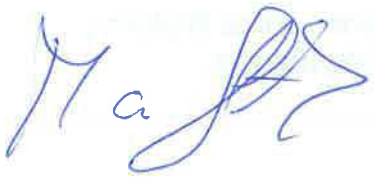
6. Beurteilung

Eine erfolgreiche BA zählt 12 ECTS-Punkte pro Studierenden. Für 1 ECTS Punkt ist eine Arbeitsleistung von ca. 25 bis 30 Stunden budgetiert. Entsprechend sollten ca. 350h Arbeit für die Studienarbeit aufgewendet werden. Dies entspricht ungefähr 25h pro Woche (auf 14 Wochen) und damit ca. 3 Tage Arbeit pro Woche.

Für die Beurteilung sind der HSR-Betreuer verantwortlich unter Einbezug des Feedbacks des Auftraggebers.

Die Bewertung der Arbeit erfolgt entsprechend der verteilten Kriterienliste

Rapperswil, den 6. April 2011



Prof. Dr. Markus Stolze
Institut für Software
Hochschule für Technik Rapperswil

Anhang B

Nutzungsvereinbarung

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

Vereinbarung

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Studienarbeit KinectBodyscanner von Felix Egli und Michael Schnyder, nachfolgend auch Autoren genannt, unter der Betreuung von Dr. Prof. Markus Stolze geregelt. Als Auftraggeber ist das Swiss Science Center Technorama, im folgenden auch Partner genannt, unter der Direktion von Herr. T. Künnemann am Projekt beteiligt.

2. Urheberrecht

Das Urheberrecht steht den beiden Autoren zu.

3. Partner

Die Autoren gestatten dem Partner die unentgeltliche nicht exklusive Nutzung des Quellcodes. Insbesondere sind Anpassungen gestattet, welche den Betrieb als Exponat positiv beeinflussen. Der Quellcode sowie damit verbundene Dokumente dürfen nicht ausserhalb des Swiss Science Center Technorama weiterverbreitet und verwendet werden.
Die Nennung des Autoren-Teams und der HSR als Schule am Exponat ist wünschenswert. Geeignete Texte/Bilder stellt das Autoren-Team zur Verfügung. Gleichfalls wird auf die Mitarbeit des Swiss Science Center Technorama hingewiesen.

4. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von den Autoren und der HSR nach Abschluss der Arbeit verwendet und weiter entwickelt werden. Die Überführung in ein kommerzielles Produkt durch die HSR, beteiligte Partner, das Technorama oder den Autoren ist nur nach gegenseitiger Absprache möglich.
Das Ausstellungsobjekt darf während 5 Jahren und im Umkreis von 250km um Winterthur nicht ein zweites Mal gezeigt werden. Davon ausgenommen ist die Präsentation des Exponats durch die HSR oder die Autoren zur Leistungsschau.

Rapperswil, den 11.05.2011 .....
Die Studentin/der Student

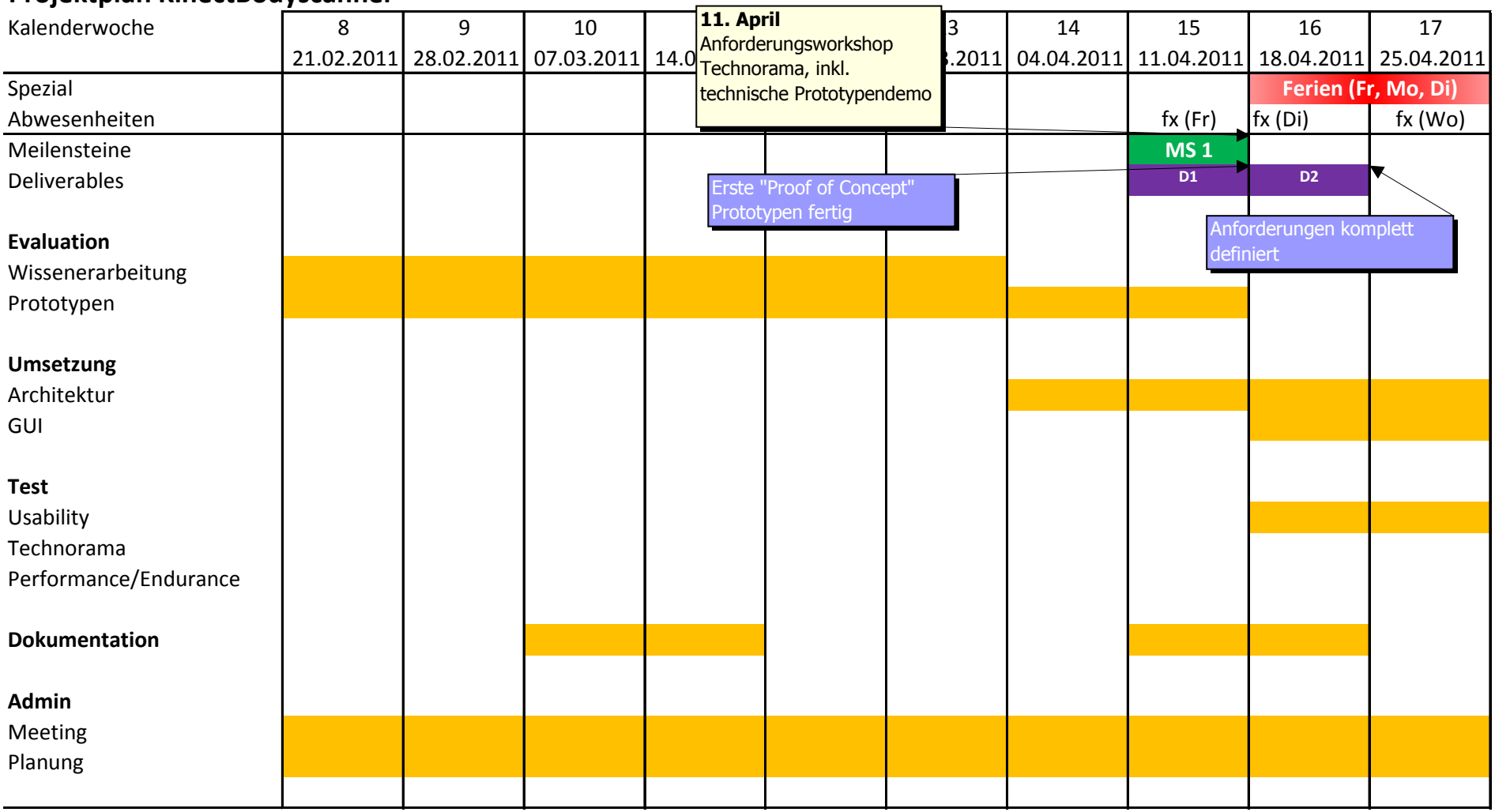
Winterthur, den 26.5.2011 .....
Der Partner

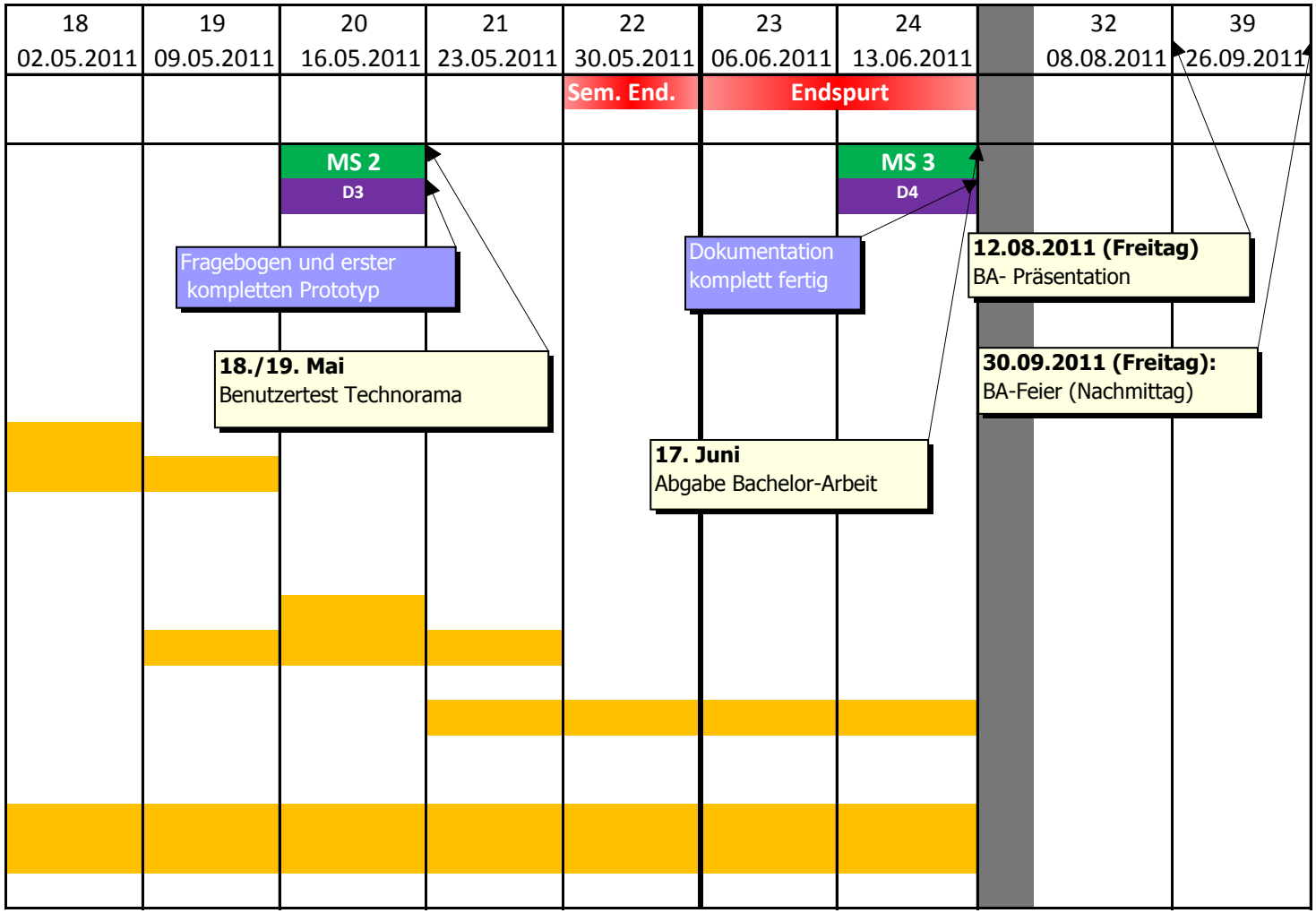
Rapperswil, den 11.05.11 .....
Der Betreuer / die Betreuerin der Studienarbeit

Anhang C

Projektplan

Projektplan KinectBodyscanner




18
02.05.2011
19
09.05.2011
20
16.05.2011
21
23.05.2011
22
30.05.2011
23
06.06.2011
24
13.06.2011
32
08.08.2011
39
26.09.2011
Sem. End.
Endspurt
MS 2
D3
MS 3
D4
Fragebogen und erster kompletten Prototyp
Dokumentation komplett fertig
12.08.2011 (Freitag)
BA- Präsentation
18./19. Mai
Benutzertest Technorama
30.09.2011 (Freitag):
BA-Feier (Nachmittag)
17. Juni
Abgabe Bachelor-Arbeit

Anhang D

Benutzertest im Technorama

D.1 Benutzerfragebogen 1

Benutzerbefragung Technorama

Was hat das Ausstellungsstück auf sie für einen Eindruck gemacht?(Optisch, interessant?)

Was haben Sie vom Ausstellungsstück erwartet? Wurde es dieser Erwartung gerecht?

Um was ist es in diesem Ausstellungsstück gegangen?

- Optional: Konnten Sie ihr Körpervolumen sehen?

- Optional: Konnten Sie Ihre Körperoberfläche sehen?

- Optional: Konnten Sie sehen, wieviel von der Körperoberfläche für einen menschen maximal sichbar ist?

Wie fanden sie die Bedienung? Schnell/Langsam/Unklar?

- Optional: Hätten sich zwischen den einzelnen Resultatseiten umschalten wollen?

- Optional: Wie hätten Sie sich das vorgestellt? Hand, Taster, Neu beginnen?

Können sie mit der Zahl der Quadratmeter etwas anfangen? (Anzahl A4-Blätter in Werten darstellen?)

- Optional: Können Sie sich noch an Ihre Werte erinnern?

Was nehmen Sie an Erfahrung mit diesem Ausstellungsstück nach Hause?

D.2 Benutzerfragebogen 2

Benutzerbefragung Technorama (Fragebogen 2)

Aussage / Frage	**Trifft überhaupt nicht zu**	**Trifft voll-ständig zu**
1. Die Interaktion mit dem System machte Spass …… …… ……	☐☐☐☐☐☐☐	
2. Ich habe alle Anzeigen des Systems verstanden …… …… ……	☐☐☐☐☐☐☐	
3. Das System hat nichts Unerwartetes gemacht …… …… ……	☐☐☐☐☐☐☐	
4. Ich habe immer genau gewusst was ich als nächstes machen muss …… …… ……	☐☐☐☐☐☐☐	
5. Wenn ich einen Kollegen/Freund beim Museums Eingang treffen würde ich ihn/sie auf dieses System hinweisen …… …… ……	☐☐☐☐☐☐☐	
6. Wie würden Sie Ihrem Kollegen das System beschreiben …… …… …… …… ……		
7. Sonstiges …… …… …… …… ……		

D.3 Auswertung und Aussagen Benutzerbefragung Technorama

Auswertungen anhand Aussagen bei der Benutzerbeobachtung/-Befragung Technorama 18. / 19. Mai 2011

Allgemein

„Hier wir mein Gewicht nicht angezeigt? Sonst stehe ich nicht hier drauf!“
-- Etwas zurückhaltend junge Dame

„Is this about my body fat? “
-- ca. 50 jährige Besucherin

Besucher: *Zückt Notizzettel und notiert sich sofort alle Werte die auf dem Bildschirm erscheinen.*

„Ah, das isch jetzt sVolume vo mir, wenn ich verschnätzlet wirde und i 1 Literfläsche abgfüllt wirde und das isch mini Körperoberflächi in DIN A4 Siite, sichtbar vo vorne und nöd sichtbar.
-- 14 jähriger Oberstüfler

Felix: „Was steht denn oben auf diesem Bildschirm?“
Besucher: „Lesen?“

Auswertung

Wenn man die Leute gefragt hat um was, dass es hier überhaupt ginge, konnten die meisten nur entweder Oberfläche oder Volumen nennen. Nur etwa 50% konnte klar beide Funktionalitäten aufzählen. Die hängt sicher auch damit zusammen dass nur wenige Leute die Texte lesen, welche angezeigt werden.

Volumenanzeige

„Aah, das sind jetzt die lääre Fläsche.“ *Seite wechselt auf die Körperoberfläche Anzeige.* „Gaht echli schnell“
-- Besucher Technorama

Teilweise wurde schon während dem Countdown auf der ersten Anzeige, die Messplatte verlassen. Somit wurde der zweite Bildschirm nicht angezeigt.

„Wieso sind die Fläsche ned gfüllt?“
-- Technorama Mitarbeiterin

Beim Volumenscreen: „Aah mein Gewicht!“
-- Besucher Technorama

„Das Gerät sieht mein Volumen, obwohl es mich nicht ganz sieht!“

Auswertung

Das Volumen zu verstehen war für die wenigsten Besucher ein Problem. Dennoch war der Wechsel zwischen den zwei Anzeigen (Volumen/Oberfläche) ziemlich schwierig. Der dargestellte Countdown, bis zum Seitenwechsel, wurde selten als ein solcher wahrgenommen.
Schade war auch, als wir abwesend waren, dass rund 1/5 der Benutzer unserer Applikation nach der ersten Anzeige die Messplatte verlassen hatten (dies stammt aus den Applikationslogs).
Vielen war die Assoziation zwischen Gewicht und Volumen sofort klar.

Oberflächenanzeige

Felix: „Versueched Sie mal Ihri sichtbari Oberflächi chlinner zmache.“
Besucher: *Zieht Hemd und T-Shirt aus!*

„Gang stand du mal ane und lueg wie chli dich chasch mache.“
-- Mutter zu ihrem kleinen Sohn

„Ah so viel DIN A4 Blätter bruuchts um mich einzuwickeln. Da chan ich mir guet öbbis vorstelle.“

„Ah jetzt ist es einfach zu verstehen, wenn es erklärt wurde, das mit den A4-Blättern“
-- 150l und 14 sichtbare Blätter Mann

„Ahh soo.. dann kann ich so und so….“ *Bückt sich, dreht sich und spielt mit dem System*

„Ahh, das Gelbe ist der Inhalt der schwarzen Silhouette.“

„Ah das sind richtige A4-Blätter?“

Besucher: „Ja das gelb markierte ist meine Körperoberfläche?“
Felix: „Und was sind die weissen Blätter?“
Besucher. „Hmm… Das ist ein Raster für alle Blätter!“

„Was ist das Gelbe? Ein Auge? Für was sollte das stehen?“

3er-Gruppe Jungs turnt auf der Messplatte herum und versuchen das System auszutricksen, dass alle drei nebeneinander als Person erkannt werden.

„Es ist sehr ungewohnt, dass in der horizontalen sehr unterschiedliche Anzahl A4-Blätter sind. Da muss ich jedes einzeln zählen.“

„Das mit den A4 ist lustig, da kann ich ja den Regenwald retten, wenn ich mich klein mache!“

„Die Kleidung hat ja hier einen ziemlich grossen Einfluss auf meine sichtbare Oberfläche.“

Auswertung

Grösster unklarer Punkt, dass das Gelbe auf dieser Anzeige die sichtbare Oberfläche ist. Die Assoziation mit A4-Blättern war ca. 60% der Besucher klar.

Wenn die Besucher nach der Beschreibung komplett alles verstanden haben. Hatten 90% eine riesen Freude mit dem System zu spielen und verschiedene Möglichkeiten auszuprobieren. Fantasielosen Besuchern konnte mit einigen kleinen Tipps auf die Sprünge geholfen werden. Wie zum Beispiel bücken, drehen usw.

Die Anzahl der A4 Blätter wurde oft als allgemein maximal (wie vorgegebenes Raster) erreichbares Total aufgefasst. Nicht als eigene totale Oberfläche. Der Bezug zur Quadratmeterzahl wurde meistens nicht gemacht.

Das Auge wurde nicht als die sichtbare Oberfläche erkannt.

Fazit

Das Ausstellungsstück ist definitiv zu überladen. Es sollte nur möglich sein die Körperoberfläche zu messen und damit zu spielen. Denn der Wechsel, wie auch die zwei komplett unterschiedlichen

Dinge passen nicht wirklich zusammen. Durch die Vereinfachung kann bei der Beschriftung des Ausstellungsstückes direkt auf die Oberfläche und die A4-Blätter eingegangen werden.

D.4 Protokoll zum Benutzertest im Technorama

Protokoll zum Benutzertest im Technorama

Gedanken:

- A4 Darstellung oftmals unklar, da nicht als solches erkannt
- Wenn die Darstellung verstanden wird, spielen die Benutzer auch mit der Fläche
- Sichtbare/Totale OberflächeA4 nicht erkannt oder ausgefüllte/leere Blätter falsch oder nicht interpretiert
- Durch die Volumenanzeige wird das Ausstellungsobjekt aufgeblasen und es kann Verwirrung entstehen
- Die Anzahl der A4 Blätter wurde oft als allgemein maximal erreichbares Total aufgefasst und nicht als eigene totale Oberfläche. Der Bezug zur Quadratmeterzahl wurde meistens nicht gemacht.
- Fläche in der Breite besser ausnutzen.
- Was macht eigentlich dieses Auge? Ist es ein Auge?

Besprochen mit Hr. Künnemann und Moor

- Volumen weglassen
- A4 Blätter
- Titel des Ausstellungsstücks anpassen
    - „Ihre Körperoberfläche jetzt messen“
    - „Deine Körperoberfläche in .... DIN A4 Blättern“
    - Weiteres
- Allgemein Beschriftungen
    - Evtl. Legende verwenden
- Messung läuft / Screen weglassen
    - Pro: Ist schön, kein harter Umbruch, etc.
    - Con: Kann bei mehrmaligem Benutzer langweilen
- HW-Anforderungen:
    - PC wird wenn normalerweise einfach vom Strom gekappt. Schwierig bei Schreiboperationen. Noch nicht getestet.
    - Beamer:
        - Kurzdistanz EPSON EW460 (Wärmekamera): 2,62m Breit x 1.85 Hoch
        - Weiterer EPSON EW440W (Bodymirror): Keine Angabe

Grafisches

- Raufzählen der Blätter und schrittweises erscheinen
- Nummerierung der Blätter
- Saubere Legende um klar zu machen, dass die markierte Fläche ein Teil des Ganzen ist
- Titel evtl. Ganzflächig anzeigen und erst dann die Werte dazu (dann Titel wieder kleiner)

Technisches

- Wir haben Laggs nach ca. 2 Stunden/ 20? Benutzungen. Vielleicht hängt es mit folgendem Zusammen:
- Nach ca. 2 Stunden / 20? Benutzungen ist das Memory ca. 30% höher als zu Beginn. Es steigt dann stetig.
- Gleichzeitig ist dann auch ein CPUKern vollständig ausgelastet
- CenterOfMass Exception
-

Unklar betreffend Grösse und Skalierung wegen Bodymirror und so.

Gemäss Herrn Moor muss der projizierte Mensch nur etwa 60% sein, weil die Augen des Menschen oben und nicht in der Mitte des Körpers sind. Uns ist das jedoch noch etwas unklar, da ja in diesem Fall auch die A4 Blätter skalierten werden müssen. Wird dies jedoch gemacht, sind die A4 Blätter auf der Projektion selber zu klein und somit nicht direkt mit einem realen A4 Blatt vergleichbar.

Anhang E

Systemtests

E.1 Systemtest Vorlage

Systemtest

Datum: _____
Examinator: _____

Langzeittests

Test	Resultat
Halbtag laufen lassen (4h) Unser Programm wird für vier Stunden laufen gelassen *Alternative: Die Applikation wird einen ganzen Tag über ein bereits aufgezeichnetes File gemacht werden.*	Anzahl Abstürzte Memory Consumtion als Grafik
20 Benutzungen Test Die Applikation muss mit mindestens 20 Benutzungen hintereinander durchführen können können ohne abzustürzen. *Alternative: Verschiedene Aufzeichnungen hintereinander laufen lassen. (Technisch möglich)*	Anzahl Abstürzte Memory Consumtion soll konstant bleiben
Neustart Die Applikation startet wieder ohne Probleme wenn der PC‘ neu min 5mal gestartet wurde	5 Neustarts überstanden

Ablauftest

Test	Resultat
Der Ablauf Testen (Screens nacheinander) Der Ablauf bei jedem der 20 Benutzer ist derselbe und läuft nach dem exakt selben Muster ab. (Reihenfolge der Screens)	Anzahl falsche Wechsel zwischen den Screens aufschreiben
Reset Test Nach 20 maligem verlassen der Waage ist der Benutzer nicht mehr markiert und der Startscreen mit allen verfügbaren Benutzer wird angezeigt. Beim wiederdrauftreten wird der Benutzer wieder erkannt und auf den nächsten Screen geschaltet	Anzahl Falsche Abläufe

Funktionstest

Test	Resultat
Aus zwei Personen kann die gewogene Person (die, die aktuell auf der Waage steht) identifiziert und vermessen werden.	Richtige Person gefunden?
Das Flackern der A4-Blätter darf bei nicht bewegen der Person im abgedunkelten Raum nicht über A4-Blatt betragen.	Flackern < ein A4-Blatt
Funktioniert die Waage korrekt? 20 Benutzer hintereinander ohne Absturz messen.	
Zweimal hintereinander die gleiche Person, Werte reproduzierbar? Nur 0.1 m2 Abweichtung maximal?	
Benutzer spring zwanzig mal während des Messen und dem Countdown und beim Wechsel	

Test	Resultat
zum A4-Screen von der Waage weg. Beim Betreten muss die Messsequenz wieder von Vorne beginnen und die Countdowns wieder neu gestartet werden	

Hardware / Umgebung

Test	Resultat
15 kg Kind sollte funktionieren für Messung.	
Eine 2m Person muss durch geeignete Position der Kinect erkannt werden bei der Waage. Der Ablauf muss einmalig durchgeführt werden können.	
Eine 1.7m Person muss durch geeignete Position der Kinect erkannt werden bei der Waage. Der Ablauf muss einmalig durchgeführt werden können.	
Es ist egal an welchem USB-Port die Waage angeschlossen ist. Nach Neustart der Applikation muss diese wieder verwendbar sein.	Einzeltest pro USB Port
Es ist egal an welchem USB-Port die Kinect angeschlossen ist. Nach Neustart der Applikation muss diese wieder verwendbar sein.	Einzeltest pro USB Port

E.2 Systemtest 17.05.2011

Pretestszenarios für Technorama

Datum: 16.5.2011
Examinator: F. Egu, M. Schnyder

Langzeittests

Test	Resultat
Halbtag laufen lassen (4h) Unser Programm wird für vier Stunden laufen gelassen *Alternative: Die Applikation wird einen ganzen Tag über ein bereits aufgezeichnetes File gemacht werden.*	Anzahl Abstürzte Memory Consumtion als Grafik Nicht getestet.
20 Benutzungen Test Die Applikation muss mit mindestens 20 Benutzungen hintereinander durchführen können können ohne abzustürzen. *Alternative: Verschiedene Aufzeichnungen hintereinander laufen lassen. (Technisch möglich)*	Anzahl Abstürzte Memory Consumtion soll konstant bleiben HAT GUT Geklappt!
Neustart Die Applikation startet wieder ohne Probleme wenn der PC' neu min 5mal gestartet wurde	5 Neustarts überstanden Nicht getestet.

Ablauftest

Test	Resultat
Der Ablauf Testen (Screens nacheinander) Der Ablauf bei jedem der 20 Benutzer ist derselbe und läuft nach dem exakt selben Muster ab. (Reihenfolge der Screens)	Anzahl falsche Wechsel zwischen den Screens aufschreiben 3 Falsche wechsel
Reset Test Nach 20 maligem verlassen der Wage ist der Benutzer nicht mehr markiert und der Startscreen mit allen verfügbaren Benutzer wird angezeigt. Beim wiederdrauftreten wird der Benutzer wieder erkannt und auf den nächsten Screen geschaltet	Anzahl Falsche Abläufe Auslösung hat nicht geklappt

Funktionstest

Test	Resultat
Aus zwei Personen kann die gewogene Person (die, die aktuell auf der Waage steht) identifiziert und vermessen werden.	Richtige Person gefunden? Richtige Person gefunden
Das Flackern der A4-Blätter darf bei nicht bewegen der Person im abgedunkelten Raum nicht über A4-Blatt betragen.	Flackern < ein A4-Blatt Richtig / gut
Funktioniert die Waage korrekt? 20 Benutzer hintereinander ohne Absturz messen.	Neue Waage erhalten
Zweimal hintereinander die gleiche Person, Werte reproduzierbar? Nur 0.1 m2 Abweichtung maximal?	Erfolgreich bei 2 Testpersonen
Benutzer spring zwanzig mal während des Messen und dem Countdown und beim Wechsel	

Test	Resultat
zum A4-Screen von der Waage weg. Beim Betreten muss die Messsequenz wieder von Vorne beginnen und die Countdowns wieder neu gestartet werden	Erfolgreich

Hardware / Umgebung

Test	Resultat
15 kg Kind sollte funktionieren für Messung.	nicht getestet
Eine 2m Person muss durch geeignete Position der Kinect erkannt werden bei der Waage. Der Ablauf muss einmalig durchgeführt werden können.	Hat funktioniert
Eine 1.7m Person muss durch geeignete Position der Kinect erkannt werden bei der Waage. Der Ablauf muss einmalig durchgeführt werden können.	Hat funktioniert
Es ist egal an welchem USB-Port die Waage angeschlossen ist. Nach Neustart der Applikation muss diese wieder verwendbar sein.	Einzeltest pro USB Port Nicht getestet
Es ist egal an welchem USB-Port die Kinect angeschlossen ist. Nach Neustart der Applikation muss diese wieder verwendbar sein.	Einzeltest pro USB Port Nicht getestet.
Person mit Jacke	Funktioniert

Es gab Probleme mit den Messwerten der Waage. Zudem hat teilweise das wechseln in den „Mess-Modus" nicht funktioniert, da keine Person vom Algorithmus als aktiv detektiert wurde.
Bildschirm war zu weit weg, daher schwierig für die Testpersonen die Anzeige richtig zu interpretieren.

E.3 Systemtest 04.06.2011

Systemtest

Datum: 04.06.2011
Examinator: F. Egli

Langzeittests

Test	Resultat
Halbtag laufen lassen (4h) Unser Programm wird für vier Stunden laufen gelassen *Alternative: Die Applikation wird einen ganzen Tag über ein bereits aufgezeichnetes File gemacht werden.*	Anzahl Abstürzte Memory Consumtion als Grafik Wurde im vorherein Separat getestet
20 Benutzungen Test Die Applikation muss mit mindestens 20 Benutzungen hintereinander durchführen können können ohne abzustürzen. *Alternative: Verschiedene Aufzeichnungen hintereinander laufen lassen. (Technisch möglich)*	Anzahl Abstürzte Memory Consumtion soll konstant bleiben Wurde mit Recordetem File getestet
Neustart Die Applikation startet wieder ohne Probleme wenn der PC' neu min 5mal gestartet wurde	5 Neustarts überstanden nicht getestet

Ablauftest

Test	Resultat
Der Ablauf Testen (Screens nacheinander) Der Ablauf bei jedem der 20 Benutzer ist derselbe und läuft nach dem exakt selben Muster ab. (Reihenfolge der Screens)	Anzahl falsche Wechsel zwischen den Screens aufschreiben ✓
Reset Test Nach 20 maligem verlassen der Waage ist der Benutzer nicht mehr markiert und der Startscreen mit allen verfügbaren Benutzer wird angezeigt. Beim wiederdrauftreten wird der Benutzer wieder erkannt und auf den nächsten Screen geschaltet	Anzahl Falsche Abläufe ✓

Funktionstest

Test	Resultat
Aus zwei Personen kann die gewogene Person (die, die aktuell auf der Waage steht) identifiziert und vermessen werden.	Richtige Person gefunden? Funktioniert mit bis zu 4 Personen
Das Flackern der A4-Blätter darf bei nicht bewegen der Person im abgedunkelten Raum nicht über A4-Blatt betragen.	Flackern < ein A4-Blatt Deutlich weniger als 1A4-Blatt
Funktioniert die Waage korrekt? 20 Benutzer hintereinander ohne Absturz messen.	kein Problem
Zweimal hintereinander die gleiche Person, Werte reproduzierbar? Nur 0.1 m2 Abweichtung maximal?	funktioniert
Benutzer spring zwanzig mal während des Messen und dem Countdown und beim Wechsel	

Test	Resultat
zum A4-Screen von der Waage weg. Beim Betreten muss die Messsequenz wieder von Vorne beginnen und die Countdowns wieder neu gestartet werden	Funktioniert

Hardware / Umgebung

Test	Resultat
15 kg Kind sollte funktionieren für Messung.	
Eine 2m Person muss durch geeignete Position der Kinect erkannt werden bei der Waage. Der Ablauf muss einmalig durchgeführt werden können.	Funktioniert
Eine 1.7m Person muss durch geeignete Position der Kinect erkannt werden bei der Waage. Der Ablauf muss einmalig durchgeführt werden können.	Funktioniert
Es ist egal an welchem USB-Port die Waage angeschlossen ist. Nach Neustart der Applikation muss diese wieder verwendbar sein.	Einzeltest pro USB Port nicht getestet
Es ist egal an welchem USB-Port die Kinect angeschlossen ist. Nach Neustart der Applikation muss diese wieder verwendbar sein.	Einzeltest pro USB Port nicht getestet

Performance

FRAMES

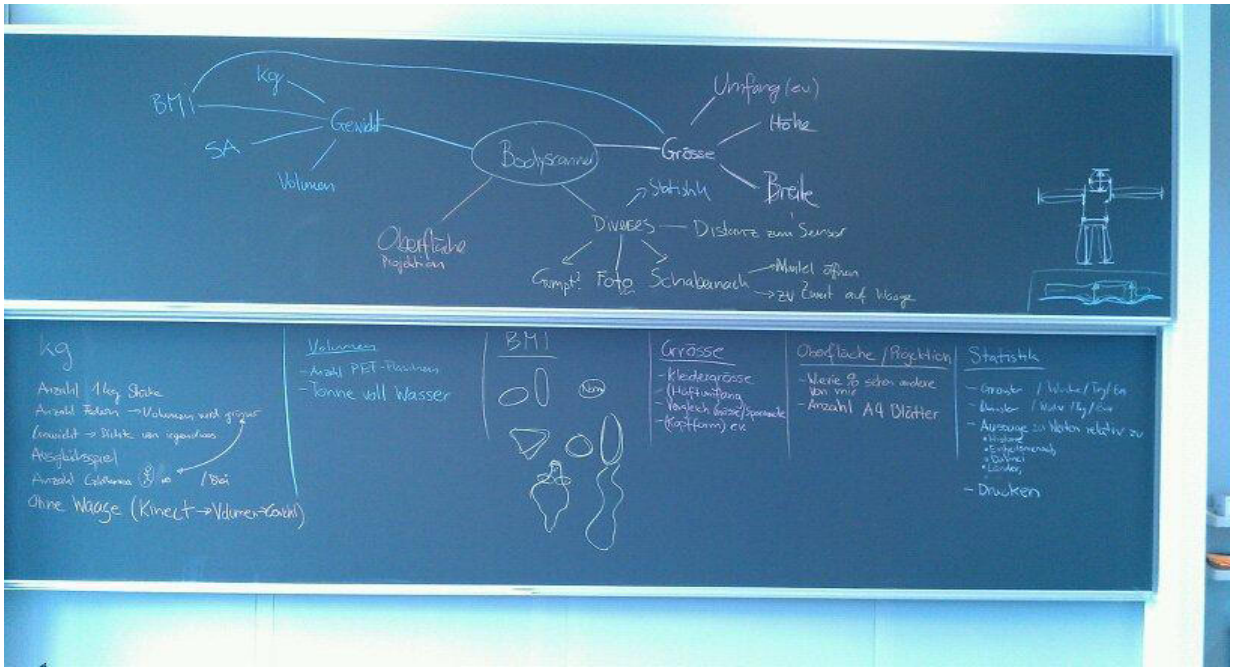
Bei ca. 4 Benutzern kann es kurzzeitig der Fall sein, dass die Framerate unter 10 FPS fällt.

Anhang F

Sitzungsprotokolle

F.1 Internes Ideen Brainstoming

Internes Brainstorming



Gewicht	Volumen	BMI	Grösse	Statistik	Spiel
Anzahl 1 kg Stücke	Anzahl Pet Flaschen	Ovale Form (Höher, Breiter) als Kreis	Wie viel % sehen anderen von mir	Grösster / Woche, Tag, Jahr	Ziel genau erreichen mit Oberfläche gross machen
Anzahl Federn -> Volumen wird Grösser	Tolle Voll Wasse	Norm(Kreis)	Anzahl A4 Blätter	Kleinster / Woche, Tag, Jahr	Longest Arm
				Aussage zu Werten relativ zu	
				- Historie	
				- Einheitsmensch	
				- DaVinci	
Gewicht -> Dichte von urgendwas				- Länder	Fläsche füllen
Anzahl Golfbarren				Ausdrucken	Quiz? Über mich
Ausgleichsspiel mit Waage					
Ohne Waage (Kinect -> Volumen -> Dichte -> Gewicht					

F.2 Anforderungsworkshop Technorama Ablauf

Schedule Montag

Ziel
Erarbeiten der konkreten Anforderungen inkl. grobes GUI des Ausstellungsobjekts.

Beachten
Profil Technoramabesucher!

09:00 Eintreffen / Aufbau Einrichtung

09:30 Begrüssung, Zielvorstellung

09:35 Demo der Demoapplikationen

09:50 Erläuterung technische Möglichkeiten
- Höhe / Breite / Kontur / min. Abstand
- Gewicht
- Teil-Volumen
- Oberfläche (Berechnet)

10:00 Präsentation unserer Ideen
- Interaktiv / Instant Feedback
  - BMI
    - Ovale Form die sich anpasst (bei Norm Kreis)
  - Volumen
    - Badewanne
    - Anzahl PET-Flaschen
  - Gewicht
    - Anzahl 1kg Stücke
    - Federn
    - Gold
    - Kinect rechnet Gewicht ohne Waage, über Volumen
    - Waagespiel (2 Personen vergleichen)
  - Grösse
    - Anzahl A4-Blätter
    - Wieviel % von meinem Körper sichtbar
- Erkennen -> Messen -> Resultate
  - Grösster / Woche, Tag, Jahr
  - Kleinster / Woche, Tag, Jahr
  - Aussage zu Werten relativ zu
    - Einheitsmensch
    - DaVinci
    - Länder (Rassen)
    - Ausdrucken

➢ Interaktion mit Device möglichst gering halten

10:15 Gemeinsames Brainstorming mit weiteren Ideen

11:00 Weiterverfolgen zweier Ideen

➢ Konkretisieren der Ideen

➢ Erste GUI's skizzieren

11:45 Zusammenfassen
- Weiteres Vorgehen besprechen
  - Einführungstest im Technorama-Nebenraum möglich?

F.3 Sitzungsprotokolle

Weekly Meeting Wo17

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum14. Juni 2011, 10:00 Uhr

Ort Rapperswil, IFS

Traktanden

- Protokoll letztes mal
- Vereinbarung -> wird abgeholt
- Wikipage -> ok
- Video -> in Arbeit
- Emailverkehr Künnemann Anhang -> nein nicht, denn Stolze im CC
- Rückmeldung von Künnemann? -> Direkt an Sitzung Mail geschrieben
- Präsentationsslides auch schon abgeben? -> Nein
- BA-Präsi 12. August um 10:00
- Abgabe Freitag zwischen 10 und 11
- Poster abgenommen mit einer kleinen Änderung, die gleich vorgenommen wurde
- Lessons Learned auch in technischer Hinsicht
- Review Doku
  - Kurz grober Ablauf erklärt und als gut empfunden
- Endspurt

Weekly Meeting Wo16

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum8. Juni 2011, 10:00 Uhr

OrtRapperswil, IFS

Traktanden

- Protokoll letztes Meeting
- Anpassungen Kriterienliste
- Review Dokumentation
  - Review Inhaltsverzeichnis anhand Plakat mit Titeln
  - Umsetzungskonzept notwendig -> Nein
  - Klassendiagramme komplett notwendig -> Nein
  - Problemstellung
    - Technorama erwähnen
    - Sonderausstellung auch erwähnen
  - Lession Learned
    - Können durchaus auch technisch sein
    - Lange Füsse, Filtering notwendig
- Poster
  - Logo Technorama
  - Fragestellung besser ausformulieren "Was kann und was ist ein interessantes Ausstellungsobjekt"
  - Bild Vermessen(de) Mensch. Ziel: Vermessen mit Kinect
  - Ablauf: Ziel -> Auafbau -> Lösung
- Youtube-Film
  - Erste Version im Rahmen der BA notwendig
    - Folgt dem Poster
    - Präsentation der zwei Prototypen von hinten gefilmt

        - Prototyp 2 wird umgesetzt
        - Abschlussslide mit Endurancetest etc.
- Zweite Version mit Sequenzen aus Technorama? Nachher während Einsatz im IFS
- Massive Multiple Abstract Writing
    - DAB-Brochüre review (http://www.hsr.ch/6041.89.html?&p=800d597bd6d0757c00804cc857261173&uid1=4008&uid2=1120&no_cache=1&view=pdf0)
    - ePrints auf basis von DAB
    - Für Dokumentation auch auf Basis DAB
    - Mgmt Summary wieder basierend und auf abstract, einfach erweitert mit bildern etc. Ähnlich dem für DAB
- Ausstellung am 17.06.2011
    - Bildschirm FlickrMood? ist ok, oder Multimedia Karrer ginge auch

Ungeklärte Punkte

- Mailverkehr mit Technorama in Anhang?
- Abnahme (aus Zeitgründen nicht abgenommen)
    - Personas und Szenarios
    - Anforderungsspez
    - Domain Analyse bereits ok
    - UI Design inhaltlich
    - Entwurf (Design und Impl)
    - Testskapitel (noch ohne NDepend beweis)

Ausblick

- Letztes Meeting in der letzten Woche am Dienstag
- Abwesenheiten SZE nächste Woche -> ist da. Super!
- Einzelne Abnahmen/Reviews? an SZE jederzeit möglich
- Dokumentation bis So. Abend finish
- Review Mo/Di?
- Erstellen Video Mo, Di.
- Posterdruck Mi/Do?

Weekly Meeting Wo15

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum1. Juni 2011, 10:00 Uhr

Ort Rapperswil, IFS

Tr□kt□nden

- Protokolle letzten 2 mal => Abgenommen
- Was müssen wir Herrn Künnemann nun mitteilen?
    - Mail mit
        - Bis Ende BA Stand Z
        - Weiterentwickelt in Ferien bis Stand X
        - So möglich um einzusetzen?
        - Herr Stolze Bewertung? -> Wird selbst noch nachhacken
- Meilensteine 14 => OK
- Memoryleaks gefixt
    - Diagramme gezeigt mit Objektinstanzen, die jetzt gerade sind
    - MVVM das Deregistrieren der Objekte, ansonsten werden sie nicht aufgeräumt
- Performance Monitor
    - Perfmon.exe in Windows gezeigt
- Codecleanup
    - Refactoring / Kommentare usw.
- Kriterienliste
    - Bitte auch mit fehlenden Punkten ergänzen

    - Aufgabenstellung = Vision -> OK!
    - Punkt 3.1.2 P3 -> Ist ok, erfüllt durch Meeting + Storyboad, Missbrauch ist kreative Benutzung in Dokumentation.
    - Punkt 3.2.2 P5 -> Kurzes Kapitel: Am X ist Microsoft Framework noch nicht publiziert
    - Punkt 3.3.3 P1 -> Ok mit Treibern etc.
    - Punkt 3.3.3 P2 -> Dokumentieren den Stand beim Test im Technorama

Ausblick

- Dokumentation
- Künnemann Nutzungsrechteformular nachfragen

Weekly Meeting Wo14

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum23. Mai 2011, 08:30 Uhr

Ort Rapperswil, IFS

Traktanden

- Protokoll -> Offen
- Meeting von Mittwoch vorgezogen
- Resultate und Meinungen von Herrn Moor und T. Künnemann diskutiert und validiert
- Weiteres Vorgehen für BA definiert
    - Beschluss:
        - Soweit gröbsten Aufräumarbeiten machen
        - BA -> Engineermässig sollten mechanismen vorhanden sein, welche über Laufzeitprobleme hinweisen
            - Wenn möglich gleich Memory & Überlastprobleme beheben
        - Usability -> Probleme sauber dokumentieren und Vorschläge für Verbesserungen
        - Prozess beschreiben
            - Explorativ
            - User Zentriert
                - Benutzertests
                - Pilottest
- 2. Waage als Ersatz erhalten für Test im Technorama, da die andere nicht sauber funktioniert hat
- Microsoft SDK noch nicht erschienen

Ausblick

- Dokumentation

Weekly Meeting Wo13

Allgemeines

AnwesendM. Schnyder (m2), F. Egli (fx), Herr Moor, T. Kuennemann

Zeit/Datum19. Mai 2011, 15:00 Uhr

OrtWinterthur, Technorama

Traktanden

- Volumen weglassen
- A4 Blätter
- Titel des Ausstellungsstücks anpassen
    - "Ihre Körperoberfläche jetzt messen"
    - "Deine Körperoberfläche in .... DIN A4 Blättern"

    - Weiteres
- Allgemein Beschriftungen
    - Evtl. Legende verwenden
- Messsung läuft / Screen weglassen
    - Pro: Ist schön, kein harter Umbruch, etc.
    - Con: Kann bei mehrmaligem Benutzer langweilen
- HW-Anforderungen:
    - PC wird wenn normalerweise einfach vom Strom gekappt. Schwierig bei Schreiboperationen. Noch nicht getestet.
    - Beamer:
        - Kurzdistantz EPSON EW460 (Wärmekamera): 2,62m Breit x 1.85 Hoch
        - Weiterer EPSON EW440W (Bodymirror): Keine Angabe

Grafisches

- Raufzählen der Blätter und schrittweises erscheinen
- Nummerierung der Blätter
- Saubere Legende um klar zu machen, dass die markierte Fläche ein Teil des Ganzen ist
- Titel evtl. Ganzflächig anzeigen und erst dann die Werte dazu (dann Titel wieder kleiner)

Technisches

- Wir haben Laggs nach ca. 2 Stunden/ 20? Benutzungen. Vielleicht hängt es mit folgendem Zusammen:
- Nach ca. 2 Stunden / 20? Benutzungen ist das Memory ca. 30% höher als zu Beginn. Es steigt dann stetig.
- Gleichzeitig ist dann auch ein CPU-Kern vollständig ausgelastet
- CenterOfMass? Exception

Unklar betreffend Grösse und Skalierung wegen Bodymirror undso. Gemäss Herrn Moor muss der projezierte Mensch nur etwa 60% sein, weil die Augen des Menschen oben und nicht in der Mitte des Körpers sind. Uns ist das jedoch noch etwas unklar, da ja indiesem Fall auch die A4 Blätter skalierten werden müssen. Wird dies jedoch gemacht, sind die A4 Blätter auf der Projektion selber zu klein und somit nicht direkt mit einem realen A4 Blatt vergleichbar.

Gedanken

- A4 Darstellung oftmals unklar, da nicht als solches erkannt
- Wenn die Darstellung verstanden wird, spielen die Benutzer auch mit der Fläche
- Sichtbare/Totale? OberflächeA4 nicht erkannt oder ausgefüllte/leere Blätter falsch oder nicht interpretiert
- Durch die Volumenanzeige wird das Ausstellungsobjekt aufgeblasen und es kann Verwirrung entstehen
- Die Anzahl der A4 Blätter wurde oft als allgemein maximal errteichbares Total aufgefasst und nicht als eigene totale Oberfläche. Der Bezug zur Quadratmeterzahl wurde meistens nicht gemacht.
- Fläche in der Breite besser ausnutzen.
- Was macht eigentlich dieses Auge? Ist es ein Auge?

Ausblick

- Besprechung weiteres Vorgehen

Weekly Meeting Wo12

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum11. Mai 2011, 10:00 Uhr

Ort Rapperswil, IFS

Traktanden

- Protokoll der letzten zwei Wochen -> abgenommen
- Meeting Flückiger
    - Eindruck von M. Flückiger
        - Könnte noch etwas mehr "Magie" vertragen.

        - Besprochen, aber nicht zwingend nötig
      - Ev. zwei Säulen um zwischen die Views zu wechseln
        - Besprochen ziemlich sicher nicht nötig wird an der Demo im Technorama abgeklärt
- Feldtest Technorama
  - 18./19. Mai Mittwoch und Donnerstag
    - Junge/Künnemann? nur Teilweise anwesend
  - Autokosten, HSR Auto
    - Kann über Sekretariat organisiert werden
  - Nächste Woche fällt wegen Technorama Meeting fällt aus.
  - Worstcase nach Risiken abklären
  - Interviewfragebogen, Stolze zur übersicht schicken
  - Stabilität der Applikation klären anhand Testcases sicherstellen
  - Was passiert wenn mehrere Leute davor stehen, wie sieht man, welche erkannt wird?
    - Dies über verschiedene Abstufungen der Farbe möglich machen

- Software Mocking
  - Waage gemockt
    - File wird ausgelesen
  - Kinect gemockt (grösstenteils)
    - USB-Motor usw.
    - Recordetes File kann verwendetet werden -> Demo mit aufgezeichnetem File
  - Button noch nicht da Verwendungszweck nicht mehr gegeben (Keine Fotoseite mehr)
- Software Verbesserungen
  - Ablauf komplett fehlerfrei
  - Erkennen wer am nächsten steht
- Anpassungen Vereinbarung
  - Nehmen wir dann ins Technorama mit

Ausblick

- Codereview anschliessend
- Besuch Technorama

Weekly Meeting Wo11

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx), M. Flückiger

Zeit/Datum4. Mai 2011, 10:00 Uhr

OrtRapperswil, IFS

Traktanden

- Meeting Flückiger
  - Aufgabenstellung hinweisen
  - Ablauf erklären
    - 1. Meeting Technorama
    - Prototyp zeigen
      - HeightMeasureTest? (Breite) / Tiefe
      - Noise entfernen
      - Kontextwissen Einheitsmensch/Volume/Fläche/FindHip/ForcePlate?
      - Architektur
  - 2. Meeting Technorama
    - 4 Varianten rauskristalisiert -> 1 Entschieden -> Austellungsobjekt

- Prototypen ("Paper"/Controls/Scene)
- Alles zusammen in einem Projekt vereint

- Ausblick
  - Taster
  - Optimierungen (Button/Countdown?)
  - Design nicht im Fokus
  - Test im Technorama
  - Sauber Dokumentieren

- Eindruck von M. Flückiger
  - Könnte noch etwas mehr "Magie" vertragen.
  - Ev. zwei Säulen um zwischen die Views zu wechseln
- Wie möchten sie die Abgabe?
  - Als CD per Post an das Firmen-Office
- Spezielle Anforderungen?
  - Archidtektur
  - Prototypen-Test, wie war das Testsetting
  - Technorama Benutzertest
    - Methoidik
    - Fragen, evtl. Antworten als Transkript
    - Daraus resultierte Erkentnisse
    - Hoffentlich möglichst authentisch
- Präsentation Aktueller Prototyp, bisherige Prototypen
  - Aufpassen bei mehreren Personen, evtl. nur die eine Person anzeigen, welche sich im Bereich der Waage befindet
  - Fand HeighMeasurePrototype? auch gant gut, evtl. weiterferfolgen???

Weekly Meeting Wo10

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2)

EntschuldigtF. Egli (fx): Militär

Zeit/Datum27. April 2011, 10:00 Uhr

Ort Rapperswil, IFS

Traktanden

- Meeting Glatz
  - Versionshistorie ein/aus? SZE egal
  - Abgabe als CD
  - Thema Nutzwertanalyse
    - Kann für Features verwendet werden
    - "Matrix Konstant ausgebaut mit Prototypen"
    - Eignet sich eher für einzelne Features anstatt für Prototypen, weil diese ja nicht verglichen werden könne. (Zu kleine Gemeinsame basis)
- Aktueller Stand
  - WPF einzelne Pages mit Xaml
  - Eigenes Control für Schatten, DephtImage?, etc.
    - Proformance? LabLog??
  - Eigenes Control für Navigation und ViewModel? mit Status
    - Archotejtur mit M. Gefeller anschauen, da er das bewertet
- Weitere Entwicklungsthemen (Was ist wichtig)
  - Design: Wie wichtig ist dieses?
  - Unit-Testing Mocking
  - Schatten verbessern (Rundungen)
  - Interaktion mit Gerät vs. Hardware-Taster
  - Langzeitbetrieb
  - Usertest
    - Paperprototypen in WPF
    - Testvorbereitung im Technorama
- Mit SZE besprochen. Seine Prio (unter marginalem einfluss von M. Schnyder)
  - 1. Interaktion vs. HW-Taster
  - 2. Mocking (Aufpasse Nutzen/Ertrag?)
  - 3. Langzeitbetrieb
  - 4. Schatten (wird sicher schnell komplex)
  - Ausser Disziplin: Dseign, da nicht unsere Kernkompetent im Rahmen der BA

    - WPF Benutzertest können wir machen, was ist das Ziel? Aufwand/Ertrag?? "Haben ja schon den PP".

- Mails an Künnemann / Flückiger gemacht
- Milestones abgehakt:
    - 4 ok
    - 5 ok
    - 6 ok, ist als Teil der Aufgabenstellung eingeflossen
    - 8 ok, ist Teil von Technorama Meeting #2
    - 9 ok, ist Teil von Technorama Meeting #2
    - 10 ok, wurde angepasst ohne Partner
    - 11 ok , ist Teil von Technorama Meeting #2
    - 12 ok, angepasst zu "MS: Abschluss von Prototypen" anstatt "ABschluiss von Interationen"
    - 13 ok, angepasst zu "MS: Ranking von Tasks wöchentlich"
- Flückiger einladen
    - Nächste Woche Mittwochs
    - Zeit: 10 bis 12:00 oder 13:30 bis 15:00 SZE hat bei diesen Slotz zeit
    - Ablauf: Info aktueller Stand mit anschliessender Demo
    - Teilnehmer: fx, m2, sze, flü

Ausblick

- Brief Künnemann
- Lablog Initialisieren
- Michael Gfeller einladen

Gegenleser Meeting Wo09

Allgemeines

AnwesendE. Glatz, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum20. April 2011, 15:10 Uhr

Ort Rapperswil, Zimmer 1.275

Traktanden

* Ziel der Arbeit: Ausstellungsstück mit Kinect im Technorama haben * Viel Kontextwissen in Arbeit drin * Bisheriger Ablauf

- Vor Beginn der Arbeit im Technorama Ideen gesammelt
- Library / Frameworks finden / Kinect zum laufen bringen
- Berechungsprototypen erstellen
    - Kontext wissen erarbeitet
    - Formeln erarbeitet und getestet (Archimedes)
    - Verschiedene Prototypen
- 2. Gespräch im Technorama
    - Konkret 4 Ideen ausgearbeitet: Gewählt eine mit SZE zusammen
- Umsetzungsstand
    - Archidektur
    - Paperprototype
- Weiterer Verlauf
    - Arbeiten weiterführen
    - EInführungstest in Technorama
- Fragen an E. Glatz
    - Abgabe am 17.06.2011 ok, ist einfach in den Ferien, einfach in Fach legen
    - Abgabe als CD mit Doku und allem OK

- Wichtige Punkte für E. Glatz
    - Systematisches Vorgehen
    - Keine Versionshistorie (ausser wenn von SZE gewünscht)
    - Anforderungsspezifikation muss lösungsneutral, Feststellbar/Messbar? sein
    - Projektmanagement: Schema beschreiben und ggf. erklären
    - Wichtige Entscheidungen deklarieren

- Bewertung von E. Glatz
    - Projektorganisation (Milestones, Zwischenresultate)
    - Bericht

    - Analyse (Problem verstanden?)
    - Entwurf (Läsungssuche und ggf. Auswahl der Lösungsvariante)
    - Realisierung

Ausblick

- Abgabe:
    - CD auch für Glatz
    - Ist in dern Ferien am 17.06.2011

Weekly Meeting Wo09

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum20. April 2011, 10:00 Uhr

OrtRapperswil

Traktanden

- Protokoll letzte Woche
    - Was waren die Tests genau, noch besser Dokumentieren
    - Dataminig nur als Option, nicht weiterverfolgen
    - Dann ists abgenommen
- Doku weitergeführt (viel weitergeführt)
    - Kontext der Arbeit
    - Prototypen: Fragestellung sicherstellen dass immer eine Frage mit dem Prototypen beantwortet werden kann
    - Ev. in Form eines Lab-Log?
    - Kontext der Arbeit besser "Hintergrundwissen" mit kleinem Einführungstext in Kapitel wieso
    - Kapitel für Portierung auf ofizielles Kinect SDK, bis spätestens 1. Juli erhältlich.
- Paperprototyp
- Architekur Waage <--> Kinect
    - In /src überführt und WPF Projekt hinzugefügt
    - WPF Applikation mit Video Output
- "Paperprototyp" durchgeführt (kurz zeigen & erklären)
    - definitive Anpassungen noch offen
    - Wie kann der Milch-Screen nochmals angezeigt werden?
- MS Kinect Library noch offen (ca. 16. Mai)
    - Kapitel für Portierung auf ofizielles Kinect SDK, bis spätestens 1. Juli erhältlich.
- Termin für erstes Testing im Technorama festlegen
- Experten / CoLeser? / Auftraggeber
    - Flücker Mail mit Einladung wenn er will. Aufgabenstellung auch direkt senden
    - Glatz Meeting am Donnerstag

Ausblick

- Abgabe:
    - CD für SZE
    - Mit Glatz separat schauen
- Vereinbarung senden

Weekly Meeting Wo08

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum13. April 2011, 10:00 Uhr

OrtRapperswil

Traktanden

- Protokoll letzte Woche -> OK
- Doku weitergeführt -> erklärt
- Architekur Waage <--> Kinect
  - Testing der einzelnen Algorithmen via Testfile mit zB T-Pose
- Posen erkennen mit DataMining?
  - Wäre eine Möglichkeit
- Sitzung Künnemann/Junge?
  - Sitzung erklärt und Protokoll durchgegangen
  - Weiteres vorgehen besprochen
  - Beschluss -> Idee 2 festgesetzt (Ausstellungsstück fertig inkl. Testing)!
    - Weg zählt für Bewertung, auch wenn nicht alles evaluierte verwendet wird
- Vereinbarung bresprochen und validiert

Ausblick

- Vereinbarung senden und unterschreiben (Gegenseitigkeit)

Partner Workshop

Allgemeines

AnwesendHr. Künnemann, Hr. Junge, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum technorama Winterthur (Gruppenraum 1), 11. April 2011, 09:00 bis 12:00 Uhr

OrtWinterthur

Traktanden

Siehe auch (source:workshops/Schedule%2011.04.2011.docx)

- Einleitung
  - Aufgebaut im Sitzungsraum
  - Demonstrationen der bisherigen Prototypen, waren jedoch zu ungebau
  - Aufzeigen der aktuell möglichen Messungen
- Allgemeines
  - Besucher durfen gerne mit dem Objekt etwas spielen, "Not UseCases?" sind erwünscht
  - Spielerei ist wünschenswert (Besucher kennt die Vermessungsmethode und kann so die Vermessung besser einschätzen)
  - Im definierten Optimalfall soll das System jedoch verlässlich (d.h. keine Abweichung in der Grösse bei Messung, da sich Besucher selber an anderen Exponaten messen können)
  - Konzentration auf Werte, welche der Besucher nicht eindeutig nachprüfen kann
  - Werte als Flaschen oder A4-Seiten darstellen
  - BMI Werte kritisch und spenziell kritisch wenn ungenau
  - Evtl. auch mit 3-4 Kinects arbeiten?
  - Messung (z.B. Oberfläsche) darf ungenau sein, wenn z.B.
- Thema Vergleiche
  - Keine Einteilung in Rassen etc. (Deutschland)
  - Besser mit anderen Besuchern vergleichen als mit "Standard"-Menschen.
  - Alternativ vergleich mit einem Crash-Test-Dummy, da nicht mehr persönlich
  - Vergleich mit Goldbarren (und evtl. Tagespreis?) wäre auch interessant
- Ideen Allgemein:
  - Animationen sind erwünscht (ist aber keine Spielplattform wie die XBOX)
  - Lieber keine nackten Zahlen, sondern interessante Darstellung diesen
- Idee 1: Weiterverfolgen aktuelle Messung
  - Problem: Besucher können sich an anderen Exponaten schon vermessen und wissen Ihre Höhe und Spannbreite sehr genau. Kinect-Vorteil ist nicht ersichtlich.

    - (+) Nutzt Kinect in grossem Umfang
    - (0) Ist aktuell sehr Ungenau, kann jedoh noch verbessert werden
    - (-) Aufwand um Genauigkeit zu verbessern ist ungewiss. evtl. müssen mehrere Kinect-Sensoren verwendet werden

- Idee 2: Schappschuss mit physikalischen od. virtuellen Auslöser und mit Countdown
    - Muss animiert sein
    - Nach dem Countdown wird mit der ForcePlate? das Gewicht gemessen und das Volumen und Oberfläche via Formeln berechnet
    - Die Kinect wird benutzt um anzuzeigen, vieviel % der Oberfläche des Besuchers sichbar ist
    - (-) Kinect wird nur sehr begrenzt eingesetzt
    - (+) Wage dafür umso mehr

- Idee 3: Ausbau der Skeleton-App für Darstellung der einzelnen Joints
    - (+) Ist schnell gemacht, einfach die Abstände zwischen den Gelenken einbauen
    - (0) Aussagekraft ungewiss
    - (-) In Basisausführung nicht genau definiert

* Idee 4: Flügelschläge zählen

- (+) Neuartiger Einsatzzweck für Kinect
- (0) mit 30 FPS evtl. Schwierig

- Einsatz
    - PC Voraussetzungen müssen definiert werden
    - Beschaffungskosten wurden nicht besprochen
    - Beamer und Fläche mit 1.4m - 1.6m soll angesprochen werden. Auflösung?

- SW-Vereinbarung
- Ist im Kern ok, muss jedoch noch überarbeitet werden:
    - Möchen gerne gegenseitiges OK von Autoren und Technorama für Weiterentwicklung, bzw. kommerzielle Weiterentwicklung
    - Konkurrentverbot: im Umkreis von 250km und während 5 Jahren darf Ausstellungsobjekt nicht gezeigt werden. Ausgenommen Leistungsschau HSR und Autoren.
    - Bei Einsatz des Exponats Autoren, Stolze und in Zusammenarbeit mit dem Technorama erwähnen (jeweils gegenseitig)
- Angepasstes Dok. kommt per Mail an Künnemann von Team

Ausblick

- SW-Vereinbarung anpassen
- Einführungstest ankünden, kann direkt in Ausstellung integriert werden (zwei Wochen im voraus anmelden)
- Abklärung Auflösung Beamer???

Weekly Meeting Wo07

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum6. April 2011, 10:00 Uhr

Ort Rapperswil

Traktanden

- Protokoll letzte Woche -> OK
    - Reminder für SZE: Flückiger und Glatz einladen
- Architekur erstellt und Waage <--> Kinect
    - Verschiedene Threads
    - Sauber geschichtet
    - Performance einbussen mit Class

- Einzelne Prototypen abgeschlossen

- Brainstorming -> Bisschen weitergeführt
- Aufgabenstellung unterschrieben

Ausblick

- Künnemann Sitzung 11. April 9:00
  - Laptop testen
  - Vereinbarung mitnehmen
- WIki Seite mit Link zu Trac versehen

Weekly Meeting Wo06

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum30. März 2011, 16:00 Uhr

Ort Rapperswil

Traktanden

- Protokoll letzte Woche -> OK
- Künnemann Stand (Sitzung)
  - Vereinbarung wird an Sitzung mitnehmen
- Unterteilung der Person 7.5 Teile (erklärt)
- Oberfläche messen mit Projektion funktioniert
  - Gfeller gesprochen
- Kontur bilden funktioniert (Minimaler abstand & ähnliches)
- Stand Doku (Risiken/Projektplan?) -> OK
- Einheitsmensch ok, aber auch Kinder dabei und nicht abdriften
  - Viel Research im nicht Informatikbereich
- MS SDK ev. bei MIX
- Brainstorming -> Resultate -> Künnemann Präsentieren am 11. April
  - Print nur Futurework
  - BMI von Stars
- Flückiger & Glatz einladen

Ausblick

- Künnemann Sitzung 11. April 9:00
- Challengeprojekt Anforderungen

Weekly Meeting Wo05

Allgemeines

Anwesend M. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum23. März 2011, 10:00 Uhr

Ort Rapperswil

Traktanden

- Protokoll letzte Woche -> OK
- Waage -> in C# jetzt ansprechbar
  - Bis jetzt nur unter VB.Net
  - Unerklärbare Aussetzter bis jetzt 2mal
  - Kann nun auch verwendet werden
- Stand der Arbeiten
  - Doku kurz angsprochen (Risikos / Plan geändert)
  - Projektplanung
    - Unterteilung in Phasen (Device ansprechen, Bild verarbeiten, Algorithmen, ..., Admin) Sinvoll

        - Unterteilung zusäztlich in Analyse, Design, Implementation und Bericht)
    - Approximation mit PointCloud? (Grid noch etwas schwierig)
        - Problem an M. Gfeller schildern, vlll. hat er eine Idee
    - Wasserexperiment -> erledigt Stand
        - Eigene Formel ist zu ungenau, approximation über Gewicht ist genauer

- Noise bei Personenerkennung -> Problem wurde mit Standardabweichung gemindert

- Weiteres Vorgehen
    - Diskussion ob Gewicht und Grösse + Talienumfang schon ausreichen für Annahmen über Person
        - Z.B. Sehr Athetisch, T-Form, Birnen-Form, etc.
        - Stimmt die Grösse und Bauchumfagn mit Gewicht überein?
        - Meeting (Telefon, oder Vor Ort)
            - Was geht bis jetzt, wo sind die Limitierungen
            - Was soll nun alles genau gerechnet werden?

- Testumgebung eingerichtet, dabei
- Aufgabestellung -> Kritik platziert wegen Anforderungen mit hochen Umsetzungsaufwand
    - Siehe Anmerlungen im Word-Dokument
    - Wird von SZE überarbeitet

Ausblick

- Künnemann schicken -> Vereinbarung / Statusmail
- Meeting (Tel, Ort) Technorama (ohne SZE)
- Spellchecker für LaTeX organisieren

Weekly Meeting Wo04

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum16. März 2011, 10:00 Uhr

Ort Rapperswil

Traktanden

- Protokoll letzte Woche -> OK
- Spellchecker für LaTeX organisieren
- Waage, Peter Bühler
    - Wurde gezeigt (Tool das mitgeliefert wurde und angesprochen über Visual Studio)
    - Lustige Ideen angesprochen (Framerate)
- Stand der Arbeiten
    - Doku kurz angsprochen (OK)
    - Volumen Berechnung -> Real mit Messen <-> mit Wasserdichteapproximation / Approximation mit PointCloud?
        - Wasserexperiment
    - Domain Model -> OK von Stolze
    - Angesprochen Berechnung Risiko / Eventbasiert usw.
- Vereinbarung -> von Stolze OK, Künnemann schicken
- Kriterienliste ok? -> Wurde vorläufig abgenommen, 3 Wochen vor Abgabe nochmals Prüfen, Meeting wurde eingerichtet.
- Aufgabestellung? -> wird auf nächstes mal komplettiert
- Messprototypen wurden gezeigt

Ausblick

- Künnemann schicken -> Vereinbarung / Statusmail

Weekly Meeting Wo03

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2)

Zeit/Datum9. März 2011, 10:00 Uhr

Ort Rapperswil

Traktanden

- Protokoll letzte Woche
- Waage, Peter Bühler
  - Kosten: Wage 372.- Adapter 105.-, SDK gratis, aber möglicherweise nur c++
  - SDK abgeklärt, Zugriff ist möglich -> bestellt am 10.03.2011
- Projektplan -> Soweit geplant ist ok
- Stand der Arbeiten
  - Doku kurz angsprochen
  - Verschiedene Algorithmen
    - Fixe Punkte aus PointCloud? suchen und auswerten
    - 3D Modell aus Cloud-Generieren und dieses verwenden
  - Raum-Sicht
    - 3D Guestures
    - Eigene Kalibration
    - Mehrere Kinects (jedoch dann auch mehrere PCs)
  - Domain Model (Input)
    - Nicht angeschaut
- Vereinbarung
  - Noch nicht besprochen
- MS Library
  - Joller hat kein Kontakt mehr
  - SZE hat auch zwei Kontakte welche noch warten, arbeiten alle mit OpenNI
- Kriterienliste ok? -> noch pendent
- Aufgabestellung? -> noch pendent

Ausblick

- Peter Bühler (Waage abklären günstig/teuer)
- TODO's SZE
  - Kriterienliste review
  - Vereinbarung review
  - Aufgabenstellung review
- Domain Model

Weekly Meeting Wo02

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum2. März 2011, 11:00 Uhr

Ort Rapperswil

Traktanden

- Protokoll letzte Woche --> OK
- Qualität vs. Quantität (Libraries Eval, just what we need?)
  - Mindestens 3 Libraries, zusäztliche die Library von MS
  - Projektentscheidung: Einfach mit der einen weitermachen ohne formale Evaluiert zu haben. Wenn die offizielle kommt, dann vergleichen und dann ggf. umstellen.
  - MS Library portieren, strategisches Vorgehen, sicher vergleichen und dokumentieren
  - Early Adapter via MS?
- Vereinbarung draft / unterschreiben
  - Dürfen von den Studenten, der HSR und dem Technorama Winterthur
  - Die Überführung in ein kommerzielles Produkt von der HSR oder anderen Partnern ist nicht möglich.
  - Wird dann auch vom Partner unterschrieben -> nächstes Statusmail
- Aufgabenstellung draft (:m freude machen)

    - Ist ok, kommt per Mail

- Kriterienliste Costumized v0.1 draft
    - Einhaltung der NFA dokumentiert auch ok
    - Enthält initiale Stakeholderanalyse (z.b. Administrator, Benutzer, Museumsdirektor möchte "der vermessene Mensch" attraktiver machen). Evlt. vorherige System (Product Prospective). Ein bisschen Research, insbesondere wenn es schon im Technorama steht.
    - Minimalbeschreibung (würde auch anhang von Skizzen reichen)
    - Personas werden weggelassen, jedoch Szenarien
    - Storyboard (Skizze von Interaktion mit System, mit mehr als einem Bild)
    - Brief Use-Cases (als erster Satz in Abschnitt Use-Cases). Vieleicht auch noch Miss-Use-Cases
    - Ableitbare NF-Anforderungen sind dokumentiert
    - Accessability (Wir dürfen alles ausschliessen, jedoch explizit)
    - Domain Analyse minimal halten (Teil vom Domain Model kann auch Algorithmus sein), Berechnungsformeln / Alternativen, Buchreferenz
    - Screenmap mit nur einem Screen ist kein Screenmap
    - UI-Guidelines (siehe Dokument K. Gaunt)
    - Schichtenmodell: Ist halt einfach klein
    - UnitTesting?: Kinect nicht mocken, jedoch keine Handstände
    - Abschliessender Unit-Testing: Grosser Fokus, mit Video aufnehmen. Sehr Wichtig!!!! (Eigentlich wäre Usability 90%, Technologie: 10%). Bei uns jedoch zuerst Technologie machen, danach sehen was noch für Usability übrig bleigt
    - Javadoc für MS
    - Michael Gfeller ist unser Assistent (interessiert sich auch für Kinect)
    - Unterschiedliche Dokumentation für Admin, welcher das Ding installiert, SW-Projektdokumentation für Weiterentwicklung

- Algorithmen zur Berechnung
    - Sehr gut gemacht fx, schon analysiert
    - richtiges Vorgehen
    - Eigene Formal entwickelt
    - Experiment durchführen mit 10 Personen

- Risikoliste besprechen
    - Existiert
    - Risiken wurden z.T. schon eliminiert
    - Noch zu wenige Miss-Use-Cases Riskiko

- Warten auf Antwort von Herrn Künnemann betr. Waage
    - http://www.vernier.com/probes/fp-bta.html
    - http://www.pearl.de/a-NC5248-5220.shtml?query=waage+usb

- Stand Gedanken zum Design (grafisch) unserer Lösung (:m)

- Allgemein
    - Statusmail war genau so wie gewünscht.
    - Gegenleser sind immer noch nicht bestimmt.
    - Billete für SBB bei Stolze falls benötigt

Ausblick

- Felix Mittwoch Militär
- Kriterienliste noch per Mail senden
- Aufgabenstellung senden
- Peter Bühler (Waage abklären günstig/teuer)

Weekly Meeting Wo01

Allgemeines

AnwesendM. Stolze, M. Schnyder (m2), F. Egli (fx)

Zeit/Datum14:00 / 21. Februar 2011

OrtRapperswil

Traktanden

- Technische Ziele angesprochen (Ansprechen der HW, Grössen messen, Bewegungen erkennen)

- Nutzen:
  - Werte des Individums werden mit Tabellenwerten verglichen. Anhand dieses Wertes bildliche und vorstellbare vergleiche liefern (z.B. Anzahl Petflaschen für Volumen)
  - Vergleich der letzten Besucher
- Waage abklären
- Technorama als Industriepartner erwähnen und via E-Mails Informationen kommunizieren
- Risikodriven RUP (nur Elab. mit Risikoiterationen) sonst wie Normal.
- Anforderungsanalyse (2-3 Szenarios je gut/schlecht)
- Prototypen (Wie ein Log, was ist dabei Rausgekommen, warum so, was gelernt)
- Endprodukt genau Dokumentiert (und nicht jeder Prototype im Detail)
- Verschiedene Abstufungen GUI
  - minimal ausgearbeitet A2 das erklärt -> Audio (verschiedene Stufen, Ampeln)

Ausblick

- Waage abklären & Status an Herrn Künnemann senden
- Vereinbarung
- Kriterienliste Costumized
-

Weekly Meeting Wo00

Meeting im Technorama Thorsten - D. Künnemann (TKuennemann@technorama.ch), M. Stolze, F. Egli, M. Schnyder

Ziel: Ideenfindung -> Bodyscanner

Anhang G

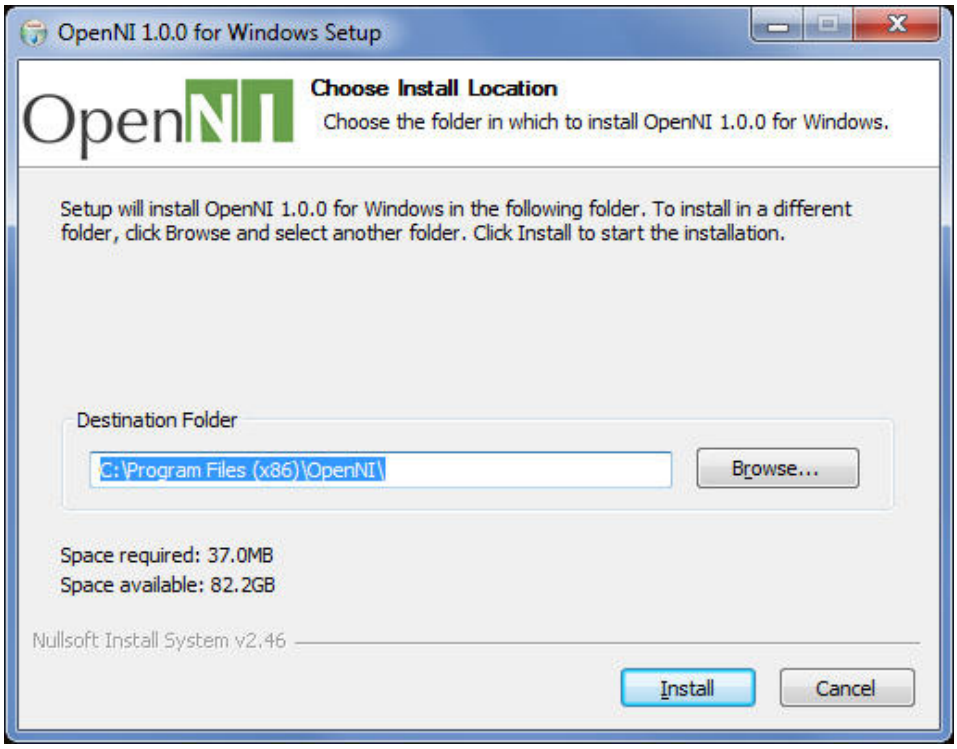
Installationsanleitungen

G.1 Installations Anleitung Kinect Treiber

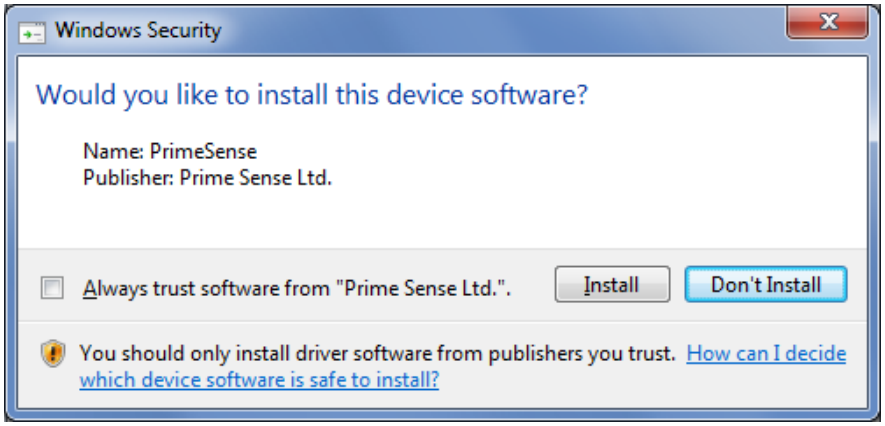
OpenNI (OpenNI Unstable Build for Windows v1.0.0.25)

Download from:
http://www.openni.org/downloadfiles/openni-binaries/latest-unstable/25-openni-unstable-build-for-windows-v1-0-0/download

Step 1



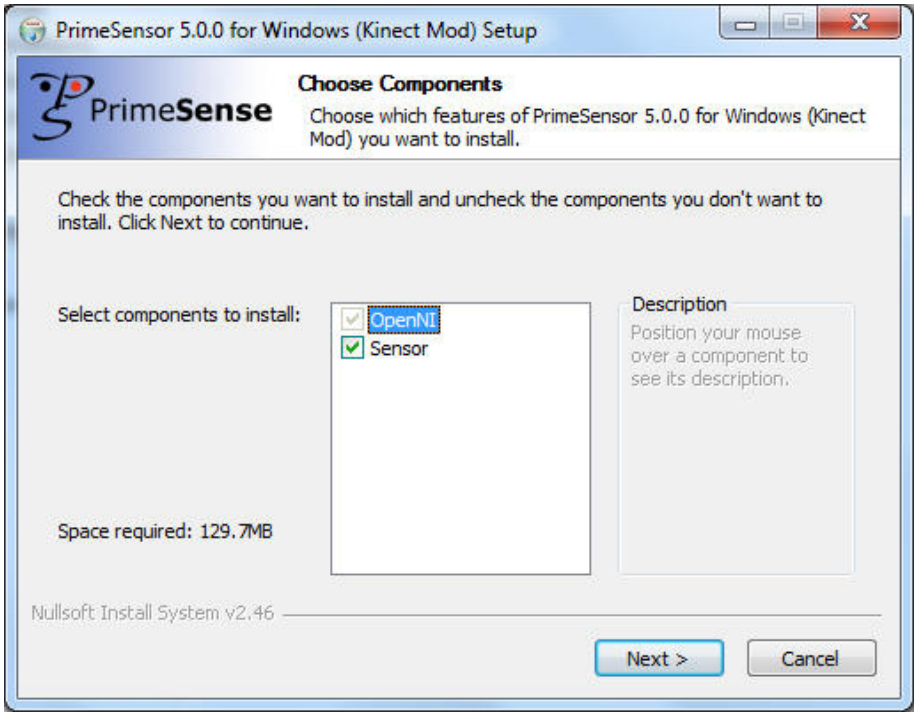
Step 2

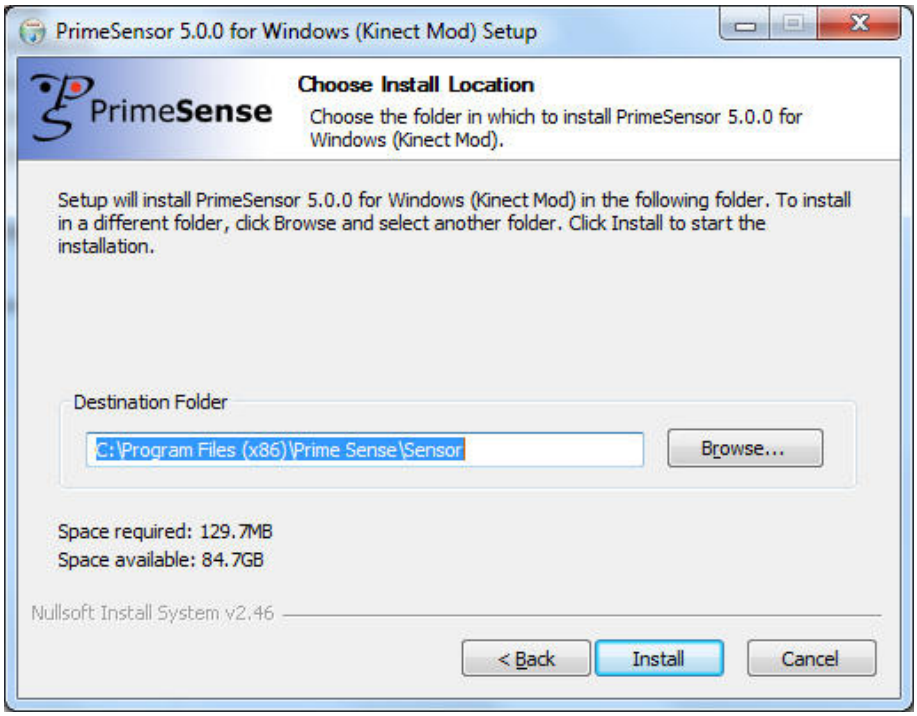


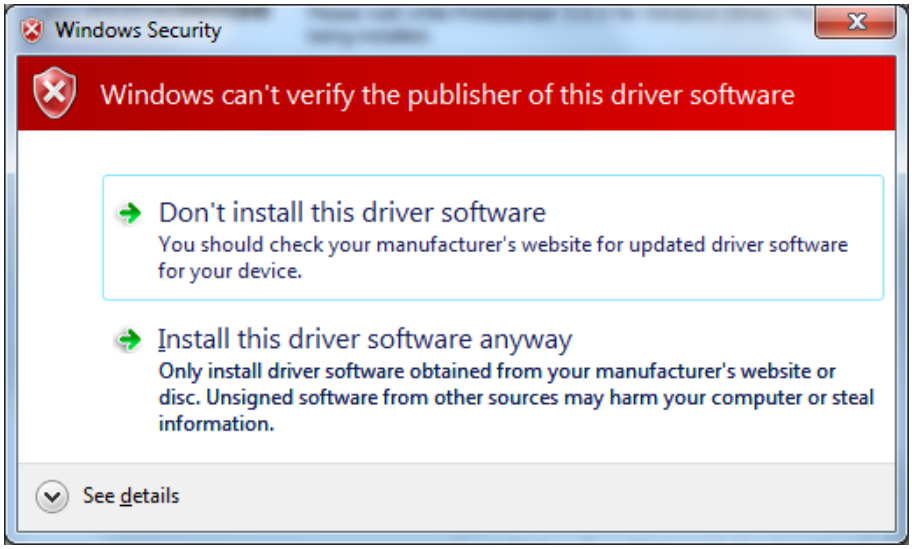
This installs a PrimeSense driver at the end

Kinect Plugin for OpenNI (SensorKinect) (January 07, 2011)

Download from:
https://github.com/avin2/SensorKinect/blob/unstable/Bin/SensorKinect-Win32-5.0.0.exe?raw=true



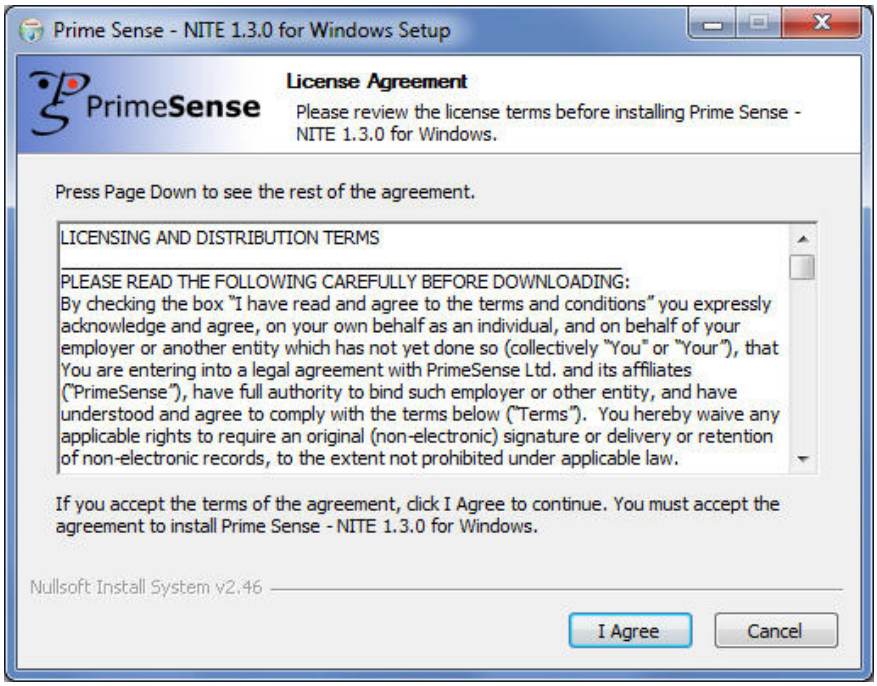




Install anyway

Install the NITE Middleware (PrimeSense NITE Unstable Build for Windows v1.3.0.18)

Download from:
http://www.openni.org/downloadfiles/openni-compliant-middleware-binaries/33-latest-unstable



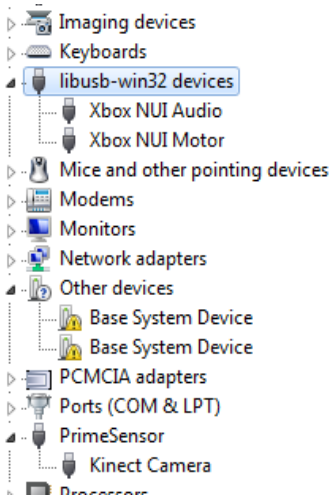
Use The Key **0KOIk2JeIBYClPWVnMoRKn5cdY4=**

AND DOUBLE-CHECK YOUR CODE, PLEASE!

Reapply Motor Driver (http://www.kinemote.net/community/viewtopic.php?f=12&t=24)

Search the KinectMotor-Device and manual install the driver from the driver package (*Kinect_Drivers* - folder). Windows will show an error thus the driver is not digitally signed

The current description for the device is "Xbox NUI Motor" your device manager should look like this:


Imaging devices
Keyboards
libusb-win32 devices
Xbox NUI Audio
Xbox NUI Motor
Mice and other pointing devices
Modems
Monitors
Network adapters
Other devices
Base System Device
Base System Device
PCMCIA adapters
Ports (COM & LPT)
PrimeSensor
Kinect Camera

G.2 Einrichtungsanleitung Solution

Einrichungsanleitung KinectBodyscanner

Folgende Schritte sind notwendig, um die Entwicklungsumgebung so einzurichten, dass KinectBodyscanner gestartet werden kann

Installation der benötigten Libarys/APIs, etc.

Dies ist in der Anleitung „Installations Anleitung Kinect Treiber“ beschrieben

Öffnen der VisualStudio-Solution

Die Solution befindet sich im Source-Code Verzeichnis und lautet „KinectBodyscanner.sln“. Nach dem öffnen kann es sein, dass die benötigten Assemblies vom VisualStudio nicht gefunden werden.
Tritt dieser Fall ein, müssen die Assemblies manuell hinzugefügt werden.

KinectBodyscanner.HAL.Kinect

Verwendet die folgenden Assemblies, welche sich im Projektverzeichnis unter im Ornder \libs befinden:

- LibUsbDotNet.dll (Assemblyname: „LibUsbDotNet“)
- OpenNI.net.dll (Assemblyname „OpenNI.net“)

KinectBodyscanner.HAL.ForcePlate

Verwendet die folgenden Assemblies, welche sich im Projektverzeichnis unter im Ornder \libs befinden:

- GoIOdetNET.dll (Assemblyname: „GoIOdetNET“)
- VSTCoreDefsdotNET.dll (Assemblyname: „ VSTCoreDefsdotNET“)

Damit sollte ein erfolgreicher Debug-Build möglich sein.

Ausführen der Applikation

Als Startapplikation kann KinectBodyscanner.Wpf verwendet verwendet

Vor dem Ausführen der Applikation muss sichergestellt werden, dass sich die folgenden Dateien im Ausführungsverzeichnis befinden.

- GoIO_DLL.dll (Aus dem Verzeichnis „KinectBodyscanner.HAL.ForcePlate\libs“
- OpenNI.xml (Aus dem Hauptverzeichnis der Solution), alternativ kann auch die Configuration von OpenNI verwendet werden.

Hierbei gillt zu beachten, dass die Applikation Schreibrechte in den Pfad benötigt, damit Debug- und Performance-Logfiles gespeichert werden können.

Wichtig: Beim ersten Start registriert die Applikation die PerformanceMonitor bei Betriebssystem. Dafür werden Administrator-Rechte benötigt. Es wir empfohlen, VisualStudio als Administrator zu starten.

Aufgezeichnete Daten für Kinect, Forceplate

Aufzeichnungen für diese beiden Devices befinden sich im Ordner „extra“. Sollen die Devices simuliert werden, müssen die entsprechenden Einträge in der „KinectBodyscanner.Wpf.exe.config“-Datei geändert werden.

G.3 Installationsanleitung LaTeX

Installationsanleitung LaTeX

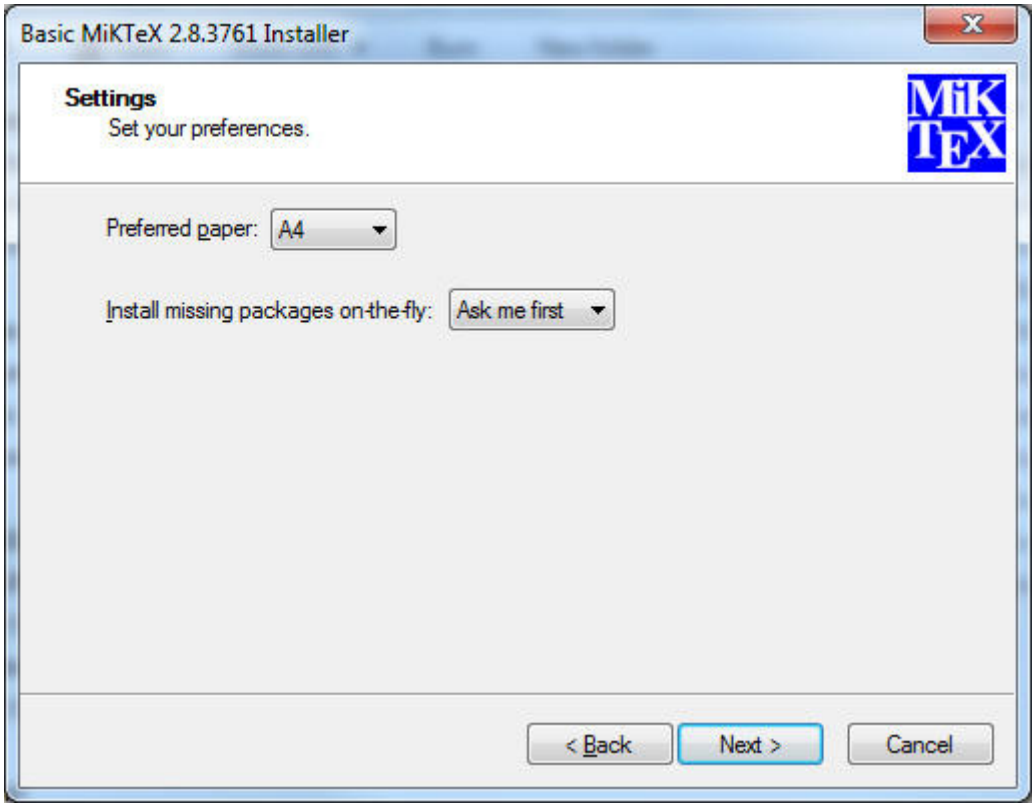
Installation MiKTeX

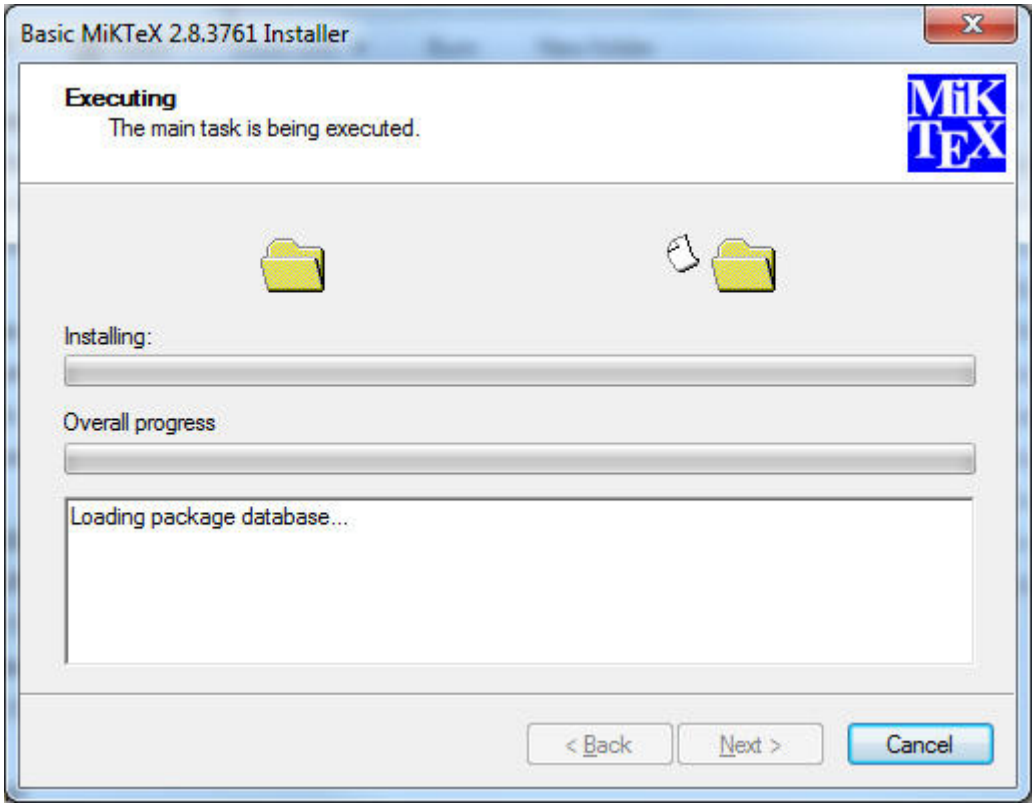
Dieses Packet enthält alle Informationen damit das .tex-Dokument in verschiedenen Schritten in ein PDF umgewandelt werden kann. Stellt die Notwendigen Library's und zusätzlichen Fancy-Addons zur Verfügung.

Downloadlink:

http://miktex.org/2.8/setup

Installieren wie folgt





Installation TEXnicCenter

Dies ist eine grafische IDE um das arbeiten mit dem Textdokument zu erleichtern.

Downloadlink

http://www.texniccenter.org/resources/downloads/29

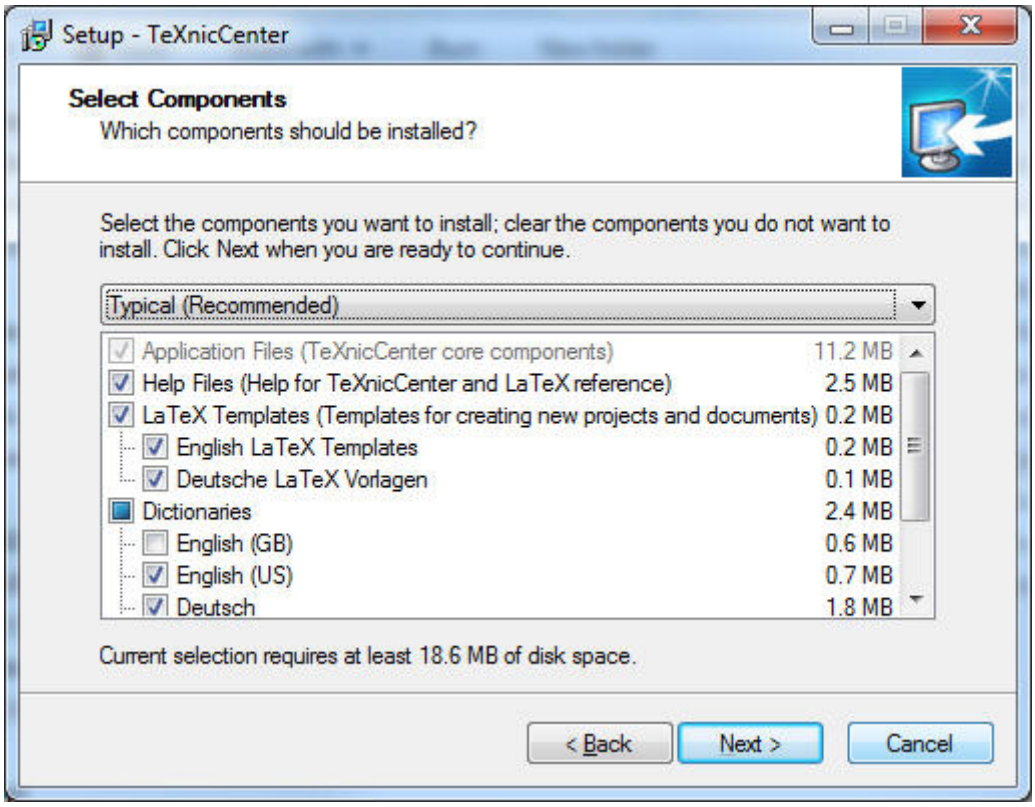
Auswählen

TeXnicCenter Alpha Installer

Direktdownload

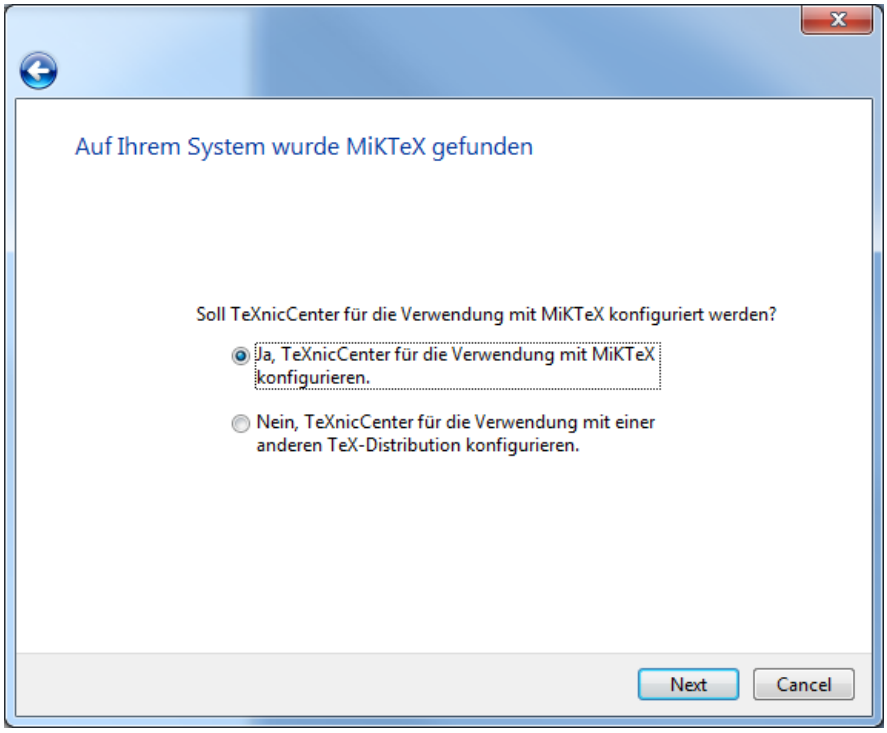
http://sourceforge.net/projects/texniccenter/files/TeXnicCenter%20Alpha/TXCSetup_2Alpha3.exe/download

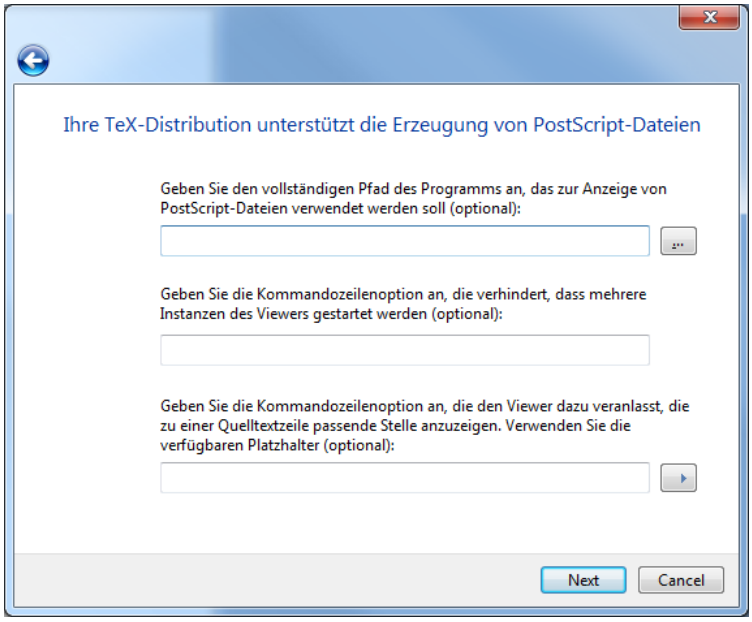
Installieren wie folgt:



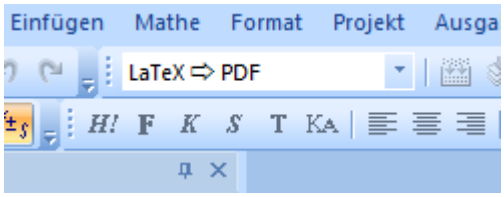
Starten und Anwenden

Nun ist das TeXnicCenter bereit zum Start. Beim ersten Start wird nachgefragt ob nun MiKTeX verwendet werden soll.

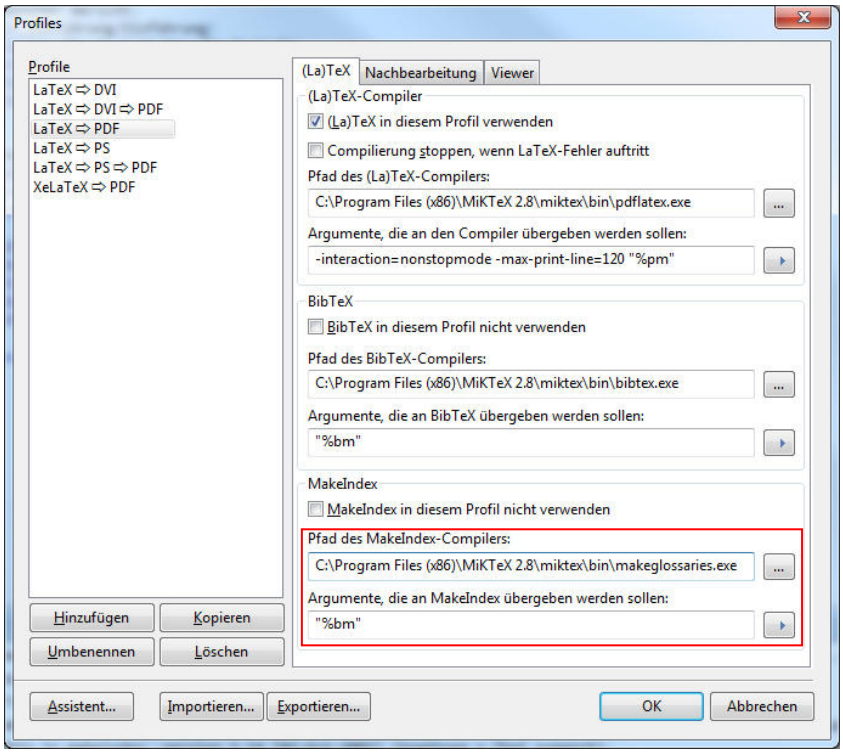




Nun nur noch auswählen, dass ein PDF erzeugt werden soll und alles ist gut!



Ausgabeprofil anpassen (**ALT + F7**)



Für den Glossar muss noch ActivePerl installiert werden.

Download unter:

http://www.activestate.com/activeperl/downloads

Version 5.12.2.1203 für Windows (64-bit, x64)

Installieren & Viel Spass mit LaTeX!