TEC Technik

Hochschule Reutlingen
Reutlingen University

Hochschule Reutlingen

Reutlingen University

– Studiengang –

Mechatronik Bachelor

– Wahlfach –

Interaktive mobile Roboter

SCITOS Gesten Steuerung mit KineticSpace

Autor:	Christoph Prinz [722568]
Professor:	Prof. Matthias Rätsch
Betreuer:	Felix Ostertag
Zeitraum:	Sommersemester 2015

Inhaltsverzeichnis

1 Aufgabenstellung **1**

2 Gestenanalyse in KineticSpace **1**

2.1 Benutzeroberfläche von KineticSpace 2

2.1.1 Gesten Ansicht 2

2.1.2 Labor Ansicht 3

2.2 Funktionsweise der Kostenmatrix 4

2.2.1 Dynamic Time Warping Algorithmus 4

2.2.2 Aufbau der Kostenmatrix 4

2.2.3 Interpretation der Kostenmatrix 6

3 KineticSpace öffnen und Nutzen **8**

3.1 Starten von KineticSpace und des Rechners 8

3.2 Wichtige Verzeichnisse auf dem Computer 9

Literatur **11**

1 Aufgabenstellung

Im Verlauf des Wahlfaches *Interaktive mobile Roboter* wurde auf die beiden Projekte [3, 4] Aufgebaut. Dabei ging es um die Aufgabe, den *SCITOS*[1] Roboter zu steuern, unter Zuhilfenahme von Software zur Gestenerkennung. Zu diesem Zweck kommt die Software *KineticSpace*[2] zum Einsatz.

Der Fokus lag nicht auf der Implementierung neuer Funktionen, sondern im Verständnis der Gestenanalyse, wie sie in KineticSpace genutzt wird. Darauf soll auch in dieser Ausarbeitung genauer eingegangen werden. Wie KineticSpace gestartet und genutzt wird, behandelt Kapitel 3 auf Seite 8 nochmals in kürze, die meisten Informationen sind jedoch bereits in den oben erwähnten Ausarbeitungen enthalten.

In Abb. 1 ist der Aufbau des Gesamtsystems mit dem Computer, welcher die Daten der Kinect auswertet und über das Client Programm im lokalen Netzwerk zur Verfügung stellt. Auf der anderen Seite ist der SCITOS mit dem Serverprogramm, welches auf erkannte Gesten des Client Computers wartet, um diese in entsprechende Fahrbewegungen umzusetzen.

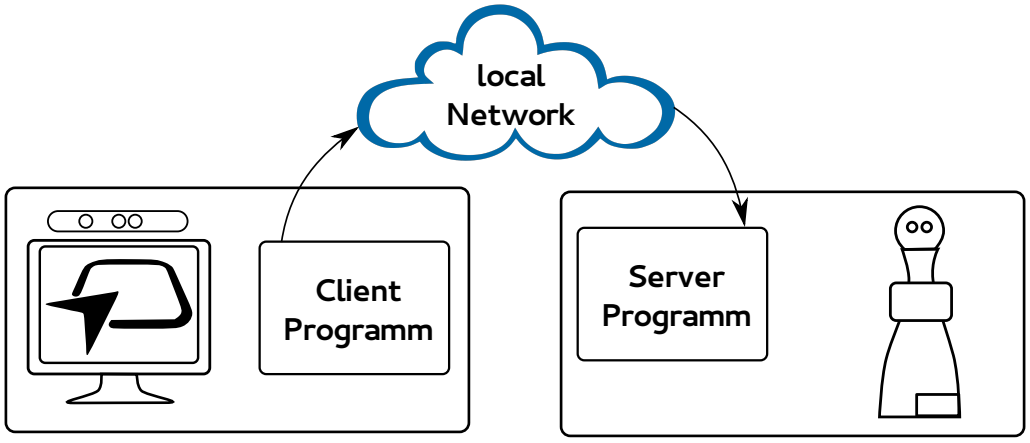

Abb. 1: Prinzipieller Aufbau des Systems mit Client und Server

2 Gestenanalyse in KineticSpace

In diesem Abschnitt wird zuerst nochmal in kürze die Oberfläche von KineticSpace beschrieben, um anschließend einen genaueren Blick auf die Kostenmatrix zu werfen.

[1] Mobieler Roboter der Firma Metralabs
http://metralabs.com/index.php?option=com_content&view=article&id=67&Itemid=66
[2] text

2.1 Benutzeroberfläche von KineticSpace

Die Oberfläche von KineticSpace besteht im Grunde aus zwei Ansichten, welche wahlweise angezeigt werden können.

2.1.1 Gesten Ansicht

Die erste Ansicht ist in Abb. 2 abgebildet und wird nach dem Programmstart standardmäßig angezeigt. Diese stellt eine Übersicht der gespeicherten Gesten zur Verfügung. Es können immer zehn Stück gleichzeitig angezeigt werden. Durch die vor- und zurück Pfeile in der oberen, linken Ecke kann die Seite geblättert werden. Unter jeder Geste ist ein Balken, welcher je nach Übereinstimmung der momentanen Bewegung länger oder künzer ist. Wir ein bestimmter Threshold überschritten, wird der Balken grün und signalisiert damit eine Übereinstimmung.

Im oberen Bereich des Screens wird die gefundene Bewegung mit dem dazugehörigen Namen und der relativen Geschwindigkeit zu der gespeicherten Geste angezeigt. In dem Fenster auf der rechten Seite wird das Livebild der Kinect Kamera dargestellt. Wird eine Person davor erkannt, wechselt die Anzeige automatisch den Modus und stellt nur noch das Bonemodel der Person dar.

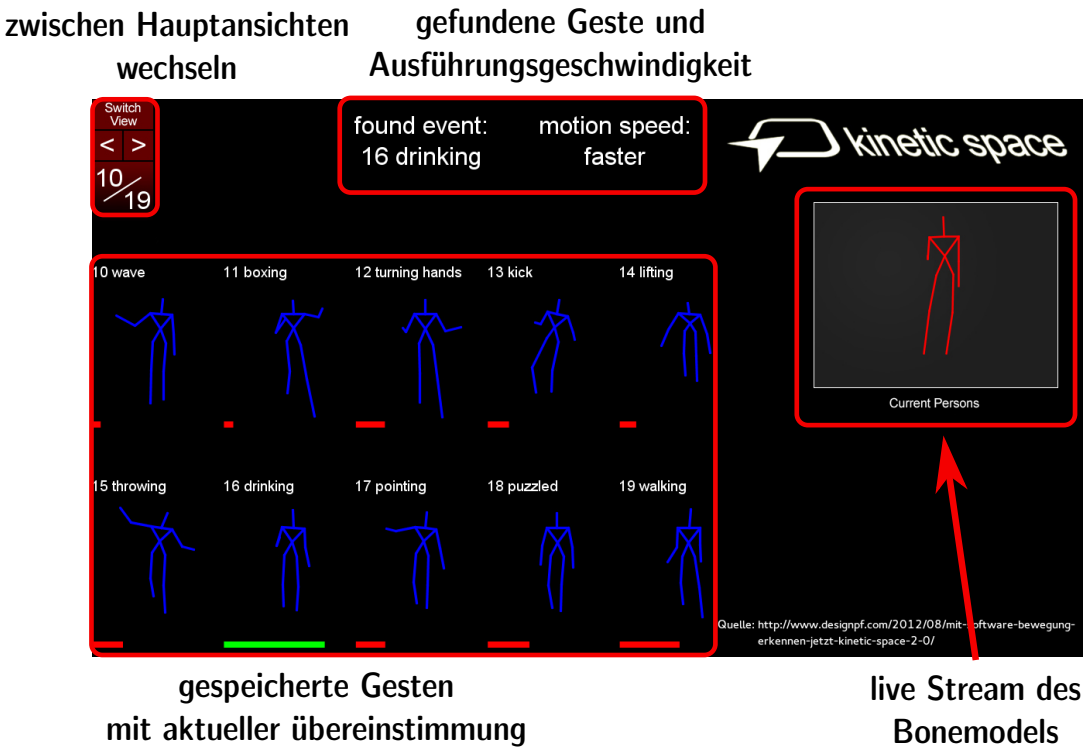

Abb. 2: Die Gesten Ansicht mit den verschiedenen Bereichen

2.1.2 Labor Ansicht

In diese zweite Ansicht gelangt der Benutzer durch drücken der Schaltfläche *Switch View* in der oberen, linken Ecke. Dadurch erhält man die in Abb. 3 dargestellte Oberfläche, welche zum Feinjustieren und Analysieren der einzelnen Gesten gedacht ist.

Unten, links in der Ecke finden sich allgemein Verwaltungsaufgaben, wie das Löschen oder Aufnehmen einer Geste. Auf der rechten Seite ist das selbe Livebild wie auch in der Gesten Ansicht. Darunter befindet sich dieses mal zusätzlich noch die Kostenmatrix, auf welche in Kapitel 2.2 auf der nächsten Seite noch genauer eingegangen wird. In der Mitte wird die aktuell ausgewählte Geste in einer Dauerschleife wiedergegeben. Hier kann auch der Name und die Anzahl der Frames editiert und eingestellt werden.

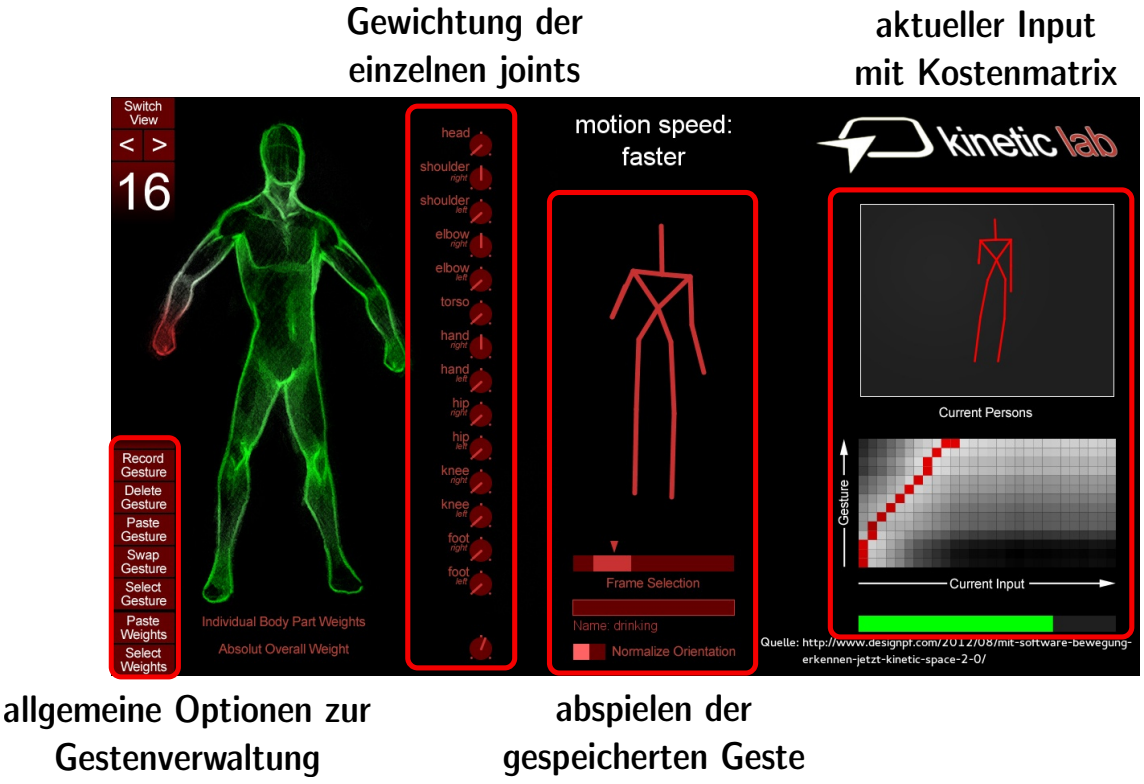

Abb. 3: Die Labor Ansicht, mit den Einstellungen zu der ausgewählten Geste

Daneben befinden sich viele Einstellrädchen für jeden einzelnen Joint des Bonemodels. Über die Einstellsymbole kann die Gewichtung eingestellt werden, also wie viel Einfluss dieser Verbindungspunkt auf die Erkennung der Geste hat. Fährt man mit dem Cursor über die einzelnen Einstellelemente, öffnet sich ein kleines Fenster mit weiteren einstellmöglichkeiten für die x, y und z Achse. Soll beispielsweise eine Winkgeste optimiert werden, können alle Joints die nichts mit dem Arm oder der

Schulter zu tun haben, komplett heruntergestellt werden und haben damit auch keine Auswirkungen mehr für die Erkennung der Geste.

2.2 Funktionsweise der Kostenmatrix

Dieses Kapitel beschäftigt sich mit der Kostenmatrix, wie sie in der Labor Ansicht zu sehen ist. Dabei handelt es sich um eine grafische Darstellung des Dynamic Time Warpings (DTW), welches im folgenden genauer angeschaut werden soll.

2.2.1 Dynamic Time Warping Algorithmus

Dieser Algorithmus ist dafür ausgelegt, die Ähnlichkeit zwischen zwei Signalverläufen zu ermitteln. Dabei ist einer der Signalverläufe ein abgespeicherter Referenzverlauf und das zweite Signal der aktuelle Input, welcher untersucht werden soll.

Der Algorithmus löst im wesentlichen das Problem, dass die beiden Verläufe oft eine unterschiedliche Zeitabhängigkeit besitzen. In der Praxis bedeutet dies einfach, dass ein ausgesprochenes Wort im einen Fall schnell und im nächsten langsam ausgesprochen wird. Trotzdem ist es das selbe Wort und durch ein Verzerren der Zeitachse erhält man das selbe Signal.

Eine häufige Verwendung findet das Dynamic Time Warping beispielsweise bei Spracherkennung, oder wie in unserem Fall bei der Gestenerkennung. In beiden Fällen wird versucht, den Input, also das Sprachsignal vom Mikrofon, oder das ermittelte Bonemodel mit einem zeitlichen Signal, welches zuvor aufgenommen und gespeichert wurde zu ermitteln.

2.2.2 Aufbau der Kostenmatrix

Auf der y-Achse der Matrix wird das zeitdiskrete, gespeicherte Signal aufgetragen und auf der x-Achse der aktuelle Input. Im nächsten Schritt wird zwischen jedem Inputwert und Vergleichswert die Differenz, also die Kosten gebildet und in die Matrix eingetragen. Dadurch entsteht eine zweidimensionale Matrix, wie sie in Abb. 4 auf der nächsten Seite zu sehen ist.

Im nächsten Schritt wird der günstigste Weg von links unten, nach rechts oben durch die Matrix gesucht. Dabei ist zu beachten, dass der Weg nie zurück gehen kann, also von links unten kann man nur nach rechts, nach oben oder diagonal

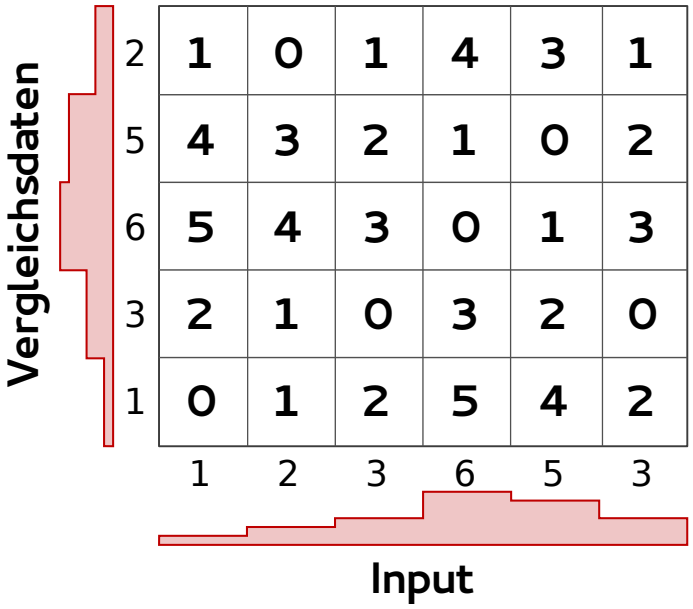

Abb. 4: Kostenmatrix mit der Differenz zwischen allen Datenwerten

nach rechts oben gehen. Dabei wird der Weg mit den geringsten Kosten gesucht. Dies kann am einfachsten über ein backtracking Verfahren realisiert werden, welches alle Möglichkeiten durchprobiert und den Weg mit den geringsten Gesamtkosten auswählt. In Abb. 5 ist der Weg für das Beispiel dargestellt.

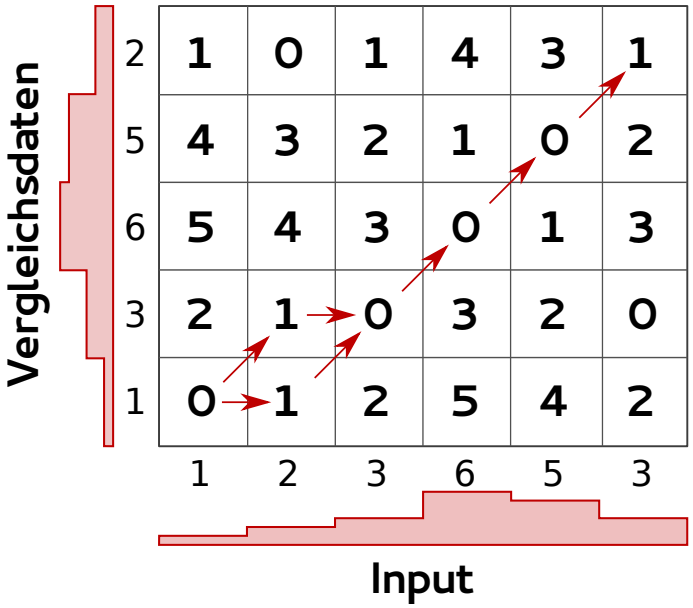

Abb. 5: Die Kostenmatrix mit dem Weg der geringsten Kosten

Um nun eine ähnliche Darstellung wie in KineticSpace zu erhalten, müssen nur noch alle Felder entsprechend der Kosten eingefärbt werden, wobei ein dunkleres Feld höhere Kosten darstellt. Zudem kann der Pfad noch rot markiert werden. Das

Beispiel sieht nun Vergleichbar mit der Darstellung in KineticSpace aus.

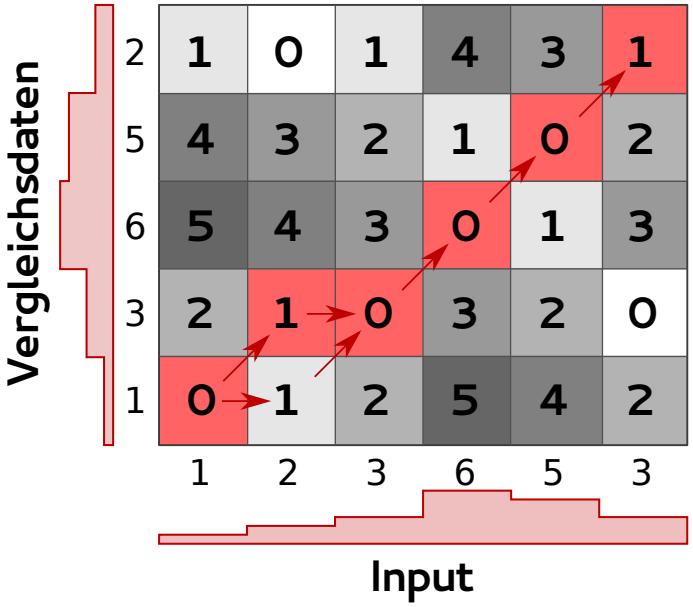

Abb. 6: Durch Einfärben der einzelnen Felder erhält man die selbe Optik wie in KineticSpace

2.2.3 Interpretation der Kostenmatrix

Doch nur eine schöne Grafik hilft nicht weiter, deswegen soll diese im Folgenden interpretiert werden. Zuerst kann natürlich erkannt werden, wie gut die Übereinstimmung der zwei Gesten ist, dies ist schließlich auch die Hauptaufgabe. Verläuft der rote Pfad entlang einem 'weißen' Tal, ist die Ähnlichkeit sehr gut. Dies ist in Abb. 7 sehr schön zu erkennen.

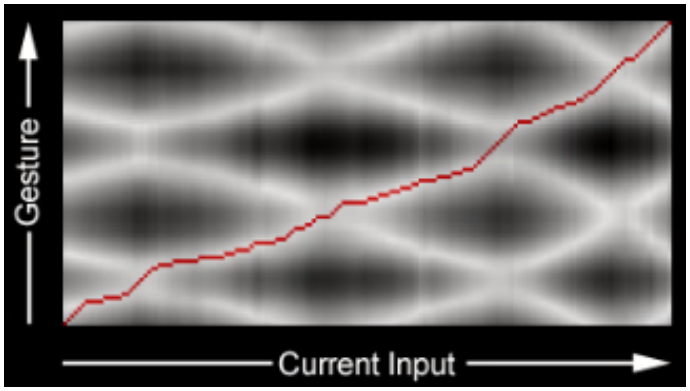

Abb. 7: Die Kostenmatrix, wenn die Geste langsamer ausgeführt wird

Eine weitere Eigenschaft, welche aus der Matrix abgelesen werden kann, ist, ob die Geste schneller oder langsamer ausgeführt wurde. Ist der Pfad im Durchschnitt

flacher als 45° ist die Ausführung der Geste langsamer als die Referenzgeste gewesen, wie es in Abb. 7 auf der vorherigen Seite dargestellt ist. Im Gegensatz dazu zeigt Abb. 8, wie die Matrix aussieht, wenn die Geste schneller ausgeführt wird. Dabei wird der Pfad durchschnittlich steiler als 45°.

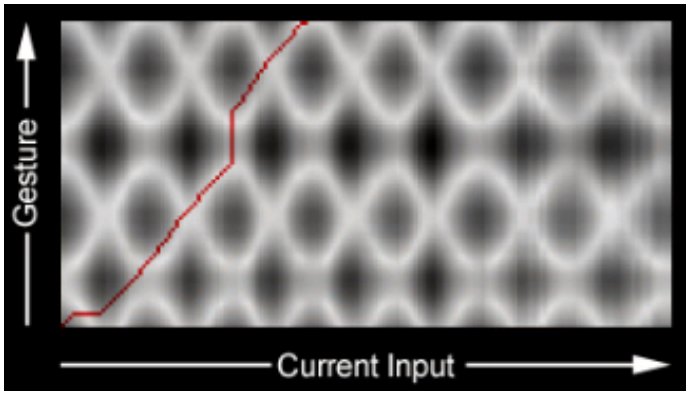

Abb. 8: Bei einer schnelleren Geste

3 KineticSpace öffnen und Nutzen

In den nachfolgenden Zeilen wird kurz beschrieben, wie KineticSpace gestartet wird. Einiges davon wird bereits in der vorherigen Ausarbeitung von Shasindran Poonudurai und Alexander Mielchen[3] erläutert.

3.1 Starten von KineticSpace und des Rechners

Auf dem Rechner `TI-13` wurde ebenfalls KineticSpace installiert. Beim Start ist darauf zu achten, dass es sich um ein Dualboot System handelt, bei dem standardmäßig Windows 7 ausgewählt und gebootet wird. Beim Start muss im Grub-Bootmenü der Eintrag *Ubuntu* ausgewählt werden, wie in Abb. 9 zu sehen.

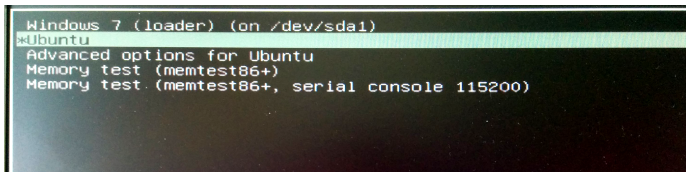

Abb. 9: Das Grub Menü beim Start. Hier muss *Ubuntu* ausgewählt werden

Danach kann als Benutzer `RT-Lions` mit dem Passwort `Scit0s` (mit Null statt 'O')eingeloggt werden. Anschließend befindet man sich auf dem klassischen Ubuntu Desktop. Bevor KineticSpace gestartet wird, ist es ratsam, die Kinect über USB mit dem Computer zu verbinden. Anschließend kann das Shell-Script auf dem Desktop genutzt werden, mit dem KineticSpace direkt gestartet wird. Nach einem Doppelklick öffnet sich wie in Abb. 10 zu sehen ein Fenster in dem wir `Run in Terminal` auswählen. Dadurch öffnet sich ein Terminal Fenster und zusätzlich KineticSpace im Vordergrund.

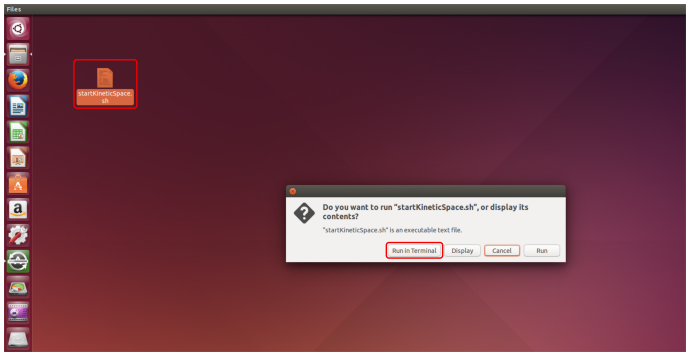

Abb. 10: Das Startskript zum starten von KineticSpace

Das automatische weiterleiten an den Client Prozess wurde in dem Script auskommentiert, da KineticSpace bei den Tests dadurch sehr instabil lief und immer wieder eingefroren ist. Vermutlich bedarf es hier noch an Optimierungsarbeit am Quelltext, oder der Client benötigt zwingend einen erreichbaren Server. Wenn dies trotzdem gewünscht ist, kann die Datei wie in Abb. 11 angepasst werden, indem das Kommentarzeichen `'#'` vor `'| ./client.out'` entfernt wird.

```
#!/bin/bash ohne Weitergabe an den Client Prozess

cd ~/KineticSpace/kineticspace_pro_ver25.linux64/kineticspace_pro_ver25.linux64
./kineticspace_pro_ver25  #| ./client.out


#!/bin/bash mit weitergabe der Gefundenen Gesten an den Client Prozess

cd ~/KineticSpace/kineticspace_pro_ver25.linux64/kineticspace_pro_ver25.linux64
./kineticspace_pro_ver25  | ./client.out
```

Abb. 11: Das Startskript mit und ohne Weiterleitung an den Client Prozess

3.2 Wichtige Verzeichnisse auf dem Computer

Alle für dieses Projekt wichtigen Dateien und Dokumente sind im *Home* Verzeichnis, also dem Benutzerverzeichnis zu finden. In Abb. 12 sind die einzelnen Ordner markiert. Zum einen sind dort noch Dateien welche für die Installation von KineticSpace und den Treibern für die Kinect zu finden, der Programmordner für KineticSpace und die bisherige Dokumentation sind ebenfalls enthalten.

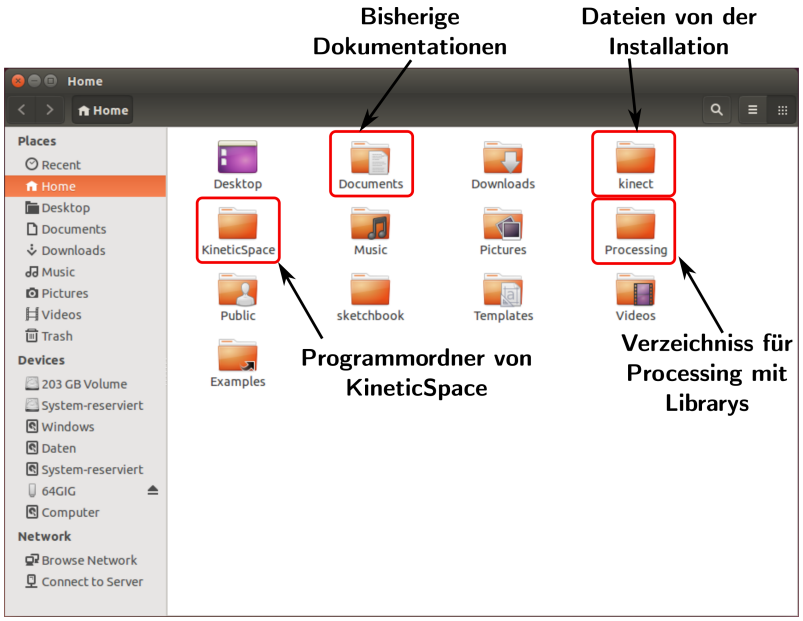

Abb. 12: Die wichtigsten Ordner im *Home* Verzeichnis

Sollte es einmal notwendig sein, KineticSpace zusammen mit den Treibern für die Kinect auf einem andern Rechner zu installieren, kann die Anleitung aus der vorherigen Arbeit[3] genutzt werden, welche auch für diese Installation zum Einsatz kam und problemlos umgesetzt werden konnte. Das direkte kopieren der Befehle funktioniert allerdings nicht, da Unix Systeme casesensitiv sind, also groß und Kleinschreibung in Dateinamen und Pfaden unterscheiden, worauf bei der Anleitung nicht geachtet wurde. Eine große Hilfe beim Eingeben der Pfade und Dateinamen im Terminal ist die Autovervollständigung mit der Tab Taste. Wird ein Ordnername nach den ersten paar Buchstaben vervollständigt, oder durch doppeltes drücken der Tab Taste eine mögliche Auswahl angezeigt, liegt ein Tippfehler vor und sollte überprüft werden.

Quelle

[1] Kinetic space - user manual. 01. 2012.

[2] Dynamic time warping, 07. 2015. URL https://de.wikipedia.org/wiki/Dynamic-Time-Warping.

[3] S. P. und Alexander Mielchen. Gesture recognition with ms kinect sensor. 2014.

[4] J. N. und Philipp Stark. Trainieren von body gestures zur steuerung des scitos powered by kinetic space. 2015.

[5] M. Wölfel. Kinetic space. 09. 2012.