Interaktion in einem Cave Automatic Virtual Environment unter Verwendung mehrerer Tiefensensor-Kameras

Thomas Jung, Franz Simon

HTW Berlin,
Wilhelminenhofstr. 75A, D-12459 Berlin
eMail: t.jung@htw-berlin.de
URL: http://www.f4.htw-berlin.de/~tj

Zusammenfassung. Im Rahmen dieser Arbeit wird ein Natural User Interface (NUI) vorgestellt, das die Interaktion in einem Cave Automatic Virtual Environment (CAVE) ermöglicht. Im Unterschied zu anderen CAVE-Systemen wird der hier beschriebene CAVE durch ein NUI gesteuert. Das NUI basiert zurzeit auf Tiefensensor-Kameras der Firma Microsoft („Kinect"). Im Rahmen dieser Arbeit wird untersucht, ob mit mehreren Kameras die Erkennungsgenauigkeit verbessert werden kann. Desweiteren wird eine einfache Handgestenerkennung auf Basis der relativ niedrig aufgelösten Tiefenbilder beschrieben.

1 Einleitung

CAVE-Systeme [1] werden schon seit den 1990er Jahren zur Erzeugung immersiver Virtueller Umgebungen verwendet, die Interaktion erfolgte dabei zunächst über magnetische Tracking-Systeme später über optische Tracking-Systeme, bei denen Benutzer in der Regel jedoch mit aktiven oder passiven Markern instrumentalisiert werden.
Natural User Interfaces vermeiden weitestgehend sichtbare Bedienelemente, um die Bedienung dadurch natürlicher zu gestalten. Tiefensensor-Kameras, die die Entfernungen zu Bildpunkten bestimmen, ermöglichen dabei die markerlose Erkennung von Gesten. Durch die Entwicklung kostengünstiger Geräte (Microsoft Kinect, Asus Wavi Xtion) gewinnt die Entwicklung von NUIs an Bedeutung.
Die CAVE der HTW besitzt seit 2011 ein NUI basierend auf einer Kinect-Kamera. Die direkt von der Kinect gelieferten Sensordaten sind geeignet, um Head Tracking und Navigationsinterfaces zu realisieren, für Selektionsaufgaben waren die Daten bisher zu ungenau [2].
Im Rahmen dieser Arbeit wird untersucht, ob und ggf. wie die Genauigkeit durch Verwendung einer weiteren Kinect-Kamera verbessert werden kann. Bei hinreichender Genauigkeit können 3D-Gesten zum Greifen und Verschieben von Objekten realisiert werden. Dazu müssen mindestens zwei unterschiedliche Handstellungen (geschlossen, geöffnet) unterschieden werden können, um den Einsatz weiterer Eingabegeräte vermeiden zu können. Im Rahmen dieser Arbeit wird deshalb ein einfacher Algorithmus zur Echtzeit-Erkennung dieser Gesten aus den Tiefendaten beschrieben.

2 Bildgenerierung

In der CAVE der HTW Berlin werden die Wände mit passiver Stereoprojektion betrieben, u.a. damit die Quellentrennung durch leichte Brillen, die herkömmlichen Sonnenbrillen ähneln, erfolgen kann. Dies soll dem Ideal der NUI (keine Controller, keine Instrumentalisierung des Benutzers) entsprechen. Bisher wurde auf die relativ kostengünstige horizontale/vertikale Polarisation gesetzt, da auf der Bodenfläche ohnehin nur Mono-Darstellungen unterstützt werden konnten [2].
Seit kurzem ist eine robuste projektionserhaltende Bodenfläche erhältlich [3], so dass in der CAVE nun alle vier Flächen mit Stereobildern betrieben werden können. Dies ist eine wichtige Voraussetzung, um geeignetes visuelles Feedback für Selektionsaufgaben zu ermöglichen. Da die Kopfposition bei der Betrachtung des Bodens variiert, musste die CAVE auf zirkulare Polarisation umgestellt werden.

3 CAVE der HTW Berlin

Die CAVE der HTW hat eine Grundfläche von 3m x 3m, die Projektionswände sind ca. 2,3m hoch. Es wird auf drei Seitenwände und den Boden in stereo projiziert. Für die Positionierung der Kinect-Kameras kommen im Frontbereich deshalb nur Standpunkte oberhalb von 2,3m in Frage. Bisher wurde nur eine Kamera für das Head-Tracking und für Interaktionsszenarien eingesetzt. Der größte Schwachpunkt dieses Szenarios war die geringe Abdeckung des Interaktionsraums (siehe Abbildung 1 links)

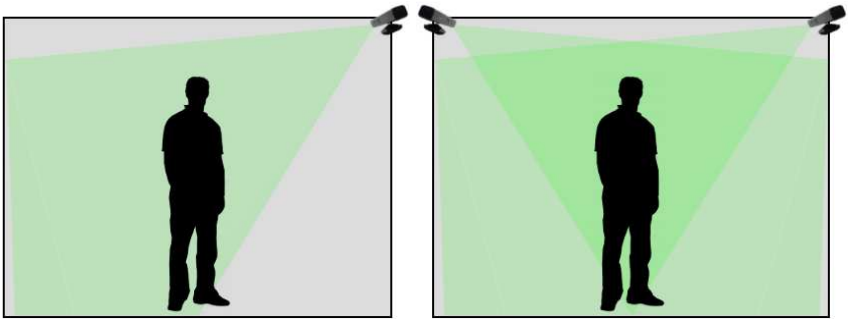

Abb. 1: Interaktionsraum innerhalb der CAVE, links mit einer Kinect-Kamera, rechts mit einer zweiten Kamera

Durch geeignete Platzierung einer zweiten Kinect lässt sich der Interaktionsraum nahezu vollständig abdecken (siehe Abbildung 1 rechts), wobei zu berücksichtigen ist, dass die Kinect-Kameras nur in einem Bereich von 80cm bis 4m Daten liefern.
Die Infrarotmuster, die von den Kameras ausgesendet werden, können sich gegenseitig stören, so dass die Erkennung insgesamt beeinträchtigt wird [4].
Deshalb wurde darauf geachtet, dass die beiden Kameras getrennte Bereiche abdecken. Beide Kameras sind jeweils zu gegenüberliegenden Wänden hin ausgerichtet und dabei lediglich um ca. 30 Grad nach unten gekippt, so dass nur nach oben hin ausgerichtete Flächen von beiden Kameras gesehen werden können. Dies betrifft vor allem die Bodenfläche der CAVE, darüber hinaus aber

auch einzelne Körperregionen des Benutzers wie Schultern und Scheitel. Beim Boden wurde darauf geachtet, dass die Kameras jeweils nur eine Hälfte des Bodens sehen, inwiefern Störungen im Schulter- und Kopfbereich auftreten werden, war zunächst unklar.

4 Kamerakalibrierung

Bisher wurde die Kinect-Kamera manuell kalibriert, wobei die Kamera horizontal ausgerichtet wurde und Höhe bzw. Winkel nach unten vermessen wurden. Für Interaktionsszenarien mit einer Kamera erwies sich diese Vorgehensweise als ausreichend. Bei Verwendung einer zweiten Kamera und der dadurch erforderlichen Registrierung der Skelettdaten beider Kameras ist eine genauere Kalibrierung der Kameras unumgänglich. Im Folgenden wird beschrieben, wie die extrinsischen Kameraparameter automatisch bestimmt werden.

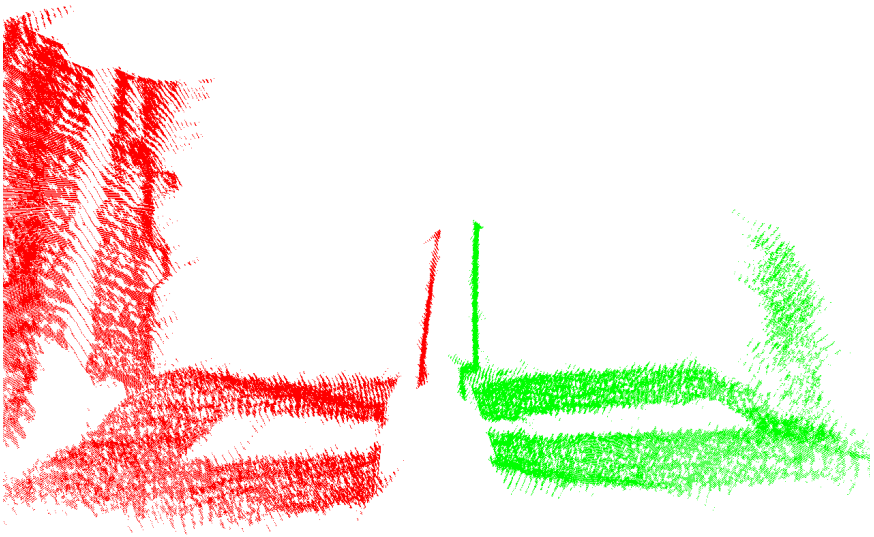

Abb. 2: Visualisierung der Punktwolken von Kinect1 (rot) und Kinect2 (grün) mit einer Holzplatte als Referenzobjekt in der Mitte der CAVE

Es wird zunächst davon ausgegangen, dass beide Kameras in einer Höhe von 2,34m und einem Neigungswinkel von 30 Grad nach unten montiert sind, und beide sich in einer Entfernung von 3m gegenüberstehen. Basierend auf diesen Daten wird eine initiale Transformationsmatrix konstruiert, die 3D-Punkte aus dem Koordinatensystem der einen Kamera in das der anderen transformiert.
Als Referenzobjekt für die Kalibrierung könnte grundsätzlich auch das Skelett selbst verwendet werden. Da die Gelenkpunkte jedoch erst durch einen komplexen Klassifizierungsalgorithmus [5] bestimmt werden, erscheinen die Daten für den Zweck der Kalibrierung als zu ungenau. Stattdessen wird ein Referenzobjekt verwendet, das die Bestimmung der Tiefendaten mit Hilfe der

Laserprojektionsmuster nicht stört. Es wird eine vertikal ausgerichtete dünne Holzplatte in den Maßen 600mm x 800mm x 5mm verwendet, deren beide Seiten jeweils nur von einer Kamera gesehen werden können. Als Referenzpunkte sollen die von den Kinect-Kameras gelieferten Punkte in den Tiefenbildern verwendet werden.
Die Bestimmung der Transformation der beiden Kamerakoordinatensysteme in einander entspricht dann der Registrierung der Oberflächenpunkte des Referenzobjektes in beiden Punktwolken. In Abbildung 2 wird die initiale Transformation auf die Punktwolke der zweiten Kinect angewendet.
Für die Registrierung der Punkte des Referenzobjekts müssen zunächst die anderen Punkte der Tiefenbilder herausgefiltert werden. Die ungefähre Lage der Platte wird als bekannt vorausgesetzt. Es werden nur Punkte verwendet, deren x-Werte im CAVE-Koordinatensystem im Bereich von -0.75 m bis 0.75 m und deren z-Werte im Bereich von 1.75 m bis 3.25 m liegen.
Nach diesem Schritt bleiben bei der aktuellen Auflösung der Tiefenbilder von 640x480 Pixeln noch mehrere zehntausend 3D-Punkte übrig. Über den VoxelGrid-Filter der PointCloud-Bibliothek [6] wird die Anzahl der Punkte anschließend auf 8 Punkte pro cm^3 reduziert. In Abbildung 3 links ist zu erkennen, dass die Punktwolken relativ stark verrauscht sind, deshalb müssen die Daten zunächst geglättet werden (siehe Abbildung 3 rechts).

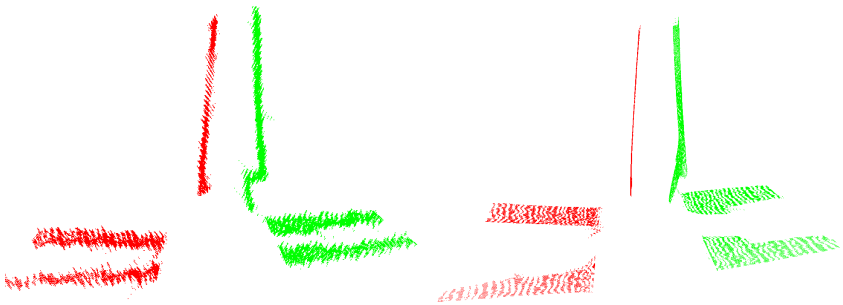

Abb. 3: links Visualisierung der gefilterten Punktwolken von Kinect1 (rot) und Kinect2 (grün), rechts nach der Glättung

Zur Glättung wird der Moving Least Squares Algorithmus [7] verwendet, der in der Lage ist, aus Punktwolken Oberflächen zu rekonstruieren, wobei angenommen wird, dass Punkte in einem definierten Umkreis jeweils zu einer kontinuierlichen Fläche gehören. Die Glättung der Punktwolken erfolgt dabei implizit. Der Radius wird hier auf 200mm festgesetzt.
Über einen Random-Consensus-Algorithmus [8] der PointCloud-Bibliothek („pcl :: SACSegmentation“) werden in den geglätteten Daten Ebenen gesucht. Dazu werden die Normalen der zu findenden Ebenen benötigt. Die Normale des Bodens der CAVE entspricht dabei der y-Achse des CAVE-Koordinatensystems, die Normale der Holzplatte der z-Achse. Beide Normalen müssen in das für die Registrierung verwendete lokale Kinect-Koordinatensystem transformiert werden. Bei der Suche der Ebenen wird eine maximale Abweichung von 20 Grad

zugelassen. Im ersten Schritt werden dann alle Punkte, die als zum Boden zugehörig klassifiziert werden aus den jeweiligen Punktwolken entfernt, im zweiten Schritt werden die Ebenen der Holzplatte in den Punktwolken beider Kinects bestimmt.

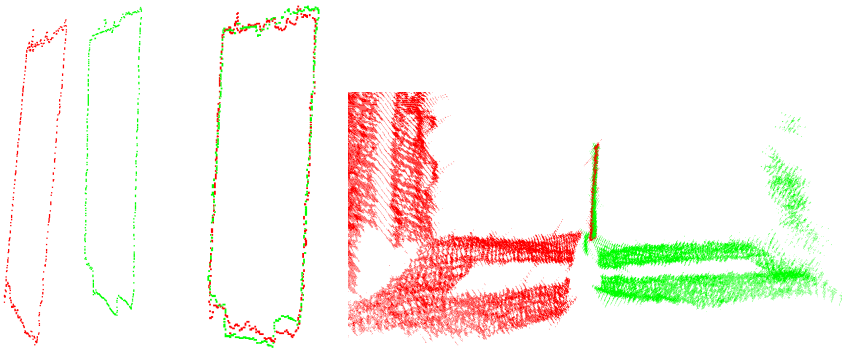

Abb. 4: links: konkave Hülle der Punkte der Holzplatte, Mitte: nach der Registrierung und Korrektur der Transformationsmatrix, rechts: gesamte Punktmenge

Für die Punkte, die zur Ebene der Holzplatte gehören, wird in beiden Punktmengen jeweils die konkave Hülle bestimmt. Zuletzt werden die Punkte der beiden konkaven Hüllen über den Iterative Closest Point-Algorithmus [9] registriert (siehe Abbildung 4). Die hierbei bestimmte Transformationsmatrix wird zur Korrektur auf die initiale Transformationsmatrix aufmultipliziert. Die resultierende Matrix ist nun geeignet, Punkte aus dem lokalen Koordinatensystem der einen Kinect in das der anderen zu transformieren.

5 Tracking der Skelette

Das Verwenden zweier Kinect-Kameras an einem PC ist zurzeit nur unter der Bedingung möglich, dass zwei unterschiedliche USB-Controller verwendet werden und die beiden Kinects von unterschiedlichen Prozessen angesprochen werden[10]. Das Zusammenspiel der beiden Prozesse lässt sich wie folgt beschreiben:

Die erste Kinect (Kinect-Master)

- baut die Verbindung zum Kinect-Client auf
- startet den internen VRPN-Server
- empfängt das Skelett vom Kinect-Client
- liest das eigene Skelett von der Kinect aus
- fusioniert beide Skelette
- glättet das fusionierte Skelett
- versendet das geglättete Skelett an die CAVE Anwendung mithilfe von VRPN

Die zweite Kinect (Kinect-Client)

- baut die Verbindung zum Kinect-Master auf
- liest das Skelett von der Kinect aus
- versendet das Skelett an den Kinect-Master

Beide Kinects können mehrere Skelette erkennen und können dabei naturgemäß auch verschiedene Kennzeichner für das Skelett ein und desselben Benutzers vergeben.
Liefern beide Kameras mehrere Personen, muss entschieden werden, welche Person aktiv getrackt werden soll. Jeweils eine Kinect (initial der Master) erhält die Kontrolle, zu entscheiden welches Skelett an die Anwendung übermittelt wird. Verlässt dieses Skelett die durch die Kinect abgedeckte Hälfte der CAVE, wird die Kontrolle an die jeweils andere Kinect abgegeben. Werden mehrere Personen getrackt, kann ein Benutzer erreichen, dass er aktiv getrackt wird, indem er einen Bereich in der Mitte der CAVE betritt („Benutzerwechsel“).
Wenn das Skelett eines Benutzers von beiden Kinects getrackt wird, müssen die Skelettdaten fusioniert werden. Der Klassifikationsalgorithmus des Microsoft Kinect SDKs geht immer davon aus, dass der Benutzer der Kinect zugewandt ist. Diese Annahme stimmt in dem hier beschriebenen Szenario jedoch nicht, so dass unter Umständen die Skelette einer Kamera zumindest hinsichtlich der Bezeichnung der Hälften der Benutzer (z. B. linke Hand / rechte Hand) vor der Fusion gespiegelt werden müssen.
Die Entscheidung, ob eines der beiden Skelette gespiegelt werden muss, wird anhand der Abstände der von beiden Kinects gelieferten Positionen für das jeweils gleiche Gelenk eines getrackten Skeletts getroffen. Zurzeit werden dabei nur die vier Gelenkpunkte der beiden Schultern und der Hüfte betrachtet. Sind also rechte Schulter und Hüfte des Skeletts der einen Kinect näher an der linken Schulter und Hüfte des Skeletts der anderen Kinect als an den jeweils rechten Gelenkpunkten, müssen die Bezeichner eines Skeletts getauscht werden. Es wird im Zweifel davon ausgegangen, dass der Benutzer in Richtung der vorderen Projektionsfläche schaut, da auf der gegenüberliegenden Seite keine Bilder projiziert werden.
Der Kinect-Algorithmus gibt für jeden Gelenkpunkt eines Skeletts an, ob er getrackt, geschlussfolgert oder nicht bestimmt wurde. Im Fall, dass die jeweiligen Gelenkpunkte in beiden Skeletten jeweils getrackt oder geschlussfolgert wurden, wird das arithmetische Mittel gebildet, im Fall, dass ein Gelenkpunkt getrackt, der andere geschlussfolgert wurde, wird ein Mittelwert mit 2/3 zu 1/3 zu Gunsten des getrackten Gelenkpunkts gebildet. Zuletzt werden die Gelenkpunkte über mehrere Frames geglättet, um das Zittern zu verringern.
Durch die beschriebenen Methoden konnte der Erfassungsbereich fast auf die komplette CAVE erweitert werden. Eine Ausnahme bildet hier der tote Winkel auf dem Boden der CAVE entlang der x-Achse. Dieser tote Winkel macht sich jedoch nur durch ein Zittern der Fußpositionen bemerkbar. Durch Erhöhung des Neigungswinkels beider Kinects ließe sich dieser Winkel verkleinern. Jedoch ist dann mit zunehmender Distanz zur jeweiligen Kamera der Kopf größerer Personen nicht mehr komplett sichtbar, wodurch die Kinect instabile Kopfpositionen liefern könnte. Eine ungenaue Kopfposition würde jedoch das Headtracking beeinträchtigen, so dass die gesamte Darstellung leiden würde. Daher ist die Verringerung des toten Winkels nur eingeschränkt möglich.

Unabhängig davon kann es zur Erzeugung von toten Winkeln durch die Benutzer in der CAVE selbst kommen. Dies ist z. B. der Fall, wenn der Benutzer nur noch von einer Kamera erfasst werden kann. In diesem Fall erzeugen Benutzer durch ihre Körper auf der zur Kamera abgewandten Seite tote Winkel (Abbildung 5). Wenn die Benutzer dann versuchen, in diesem Bereich die Arme zu bewegen, kann dies von keiner der beiden Kinects erfasst werden.
Unser Eindruck ist, dass sich die Erkennungsgenauigkeit in Bereichen, die bisher von der einen Kinect nur ungenau durch Schließen erkannt wurden, durch die Verwendung der zweiten Kinect verbessert hat. Eine quantitative Beurteilung ist jedoch sehr schwierig.

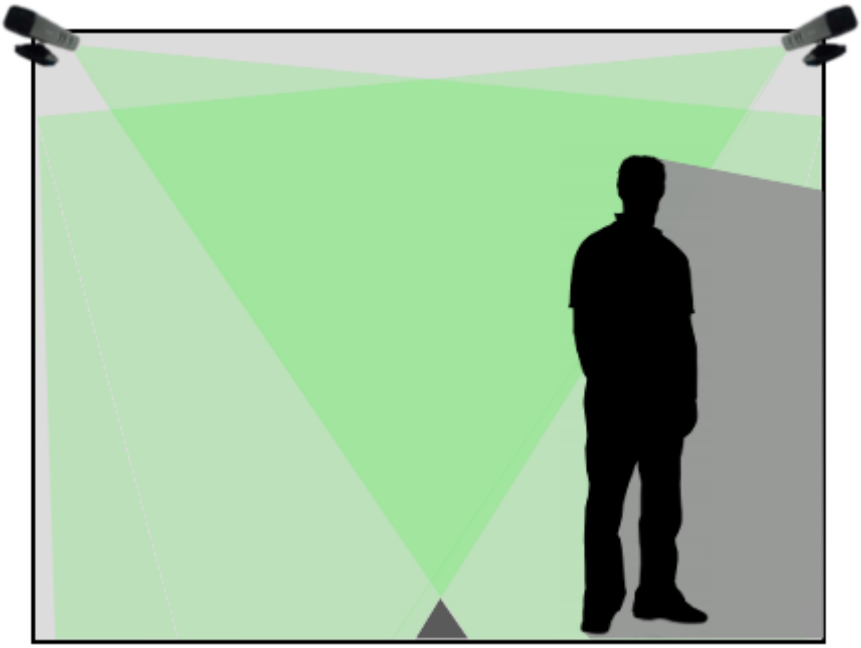

Abb. 5: Verringerung des erfassten Raumes der CAVE durch tote Winkel entlang der x-Achse (dunkelgrau) und erzeugt durch Benutzer (grau)

Es stellt sich die Frage, ob der Einsatz einer weiteren Kinect sinnvoll wäre. Der tote Winkel im Bereich der x-Achse könnte damit eliminiert werden, möglicherweise könnten durch Datenfusion die Gelenkdaten genauer bestimmt werden. Die Bedingung, dass mehrere Kinects nicht die gleichen Flächen sehen dürfen, wäre jedoch nicht mehr einzuhalten, wodurch der Betrieb der Geräte maßgeblich erschwert würde. Darüber hinaus würde die Problematik der durch Benutzer erzeugten toten Winkel nicht behoben werden.

6 Handgestenerkennung

Das Greifen von Objekten in virtuellen Umgebungen ist eine häufige Anforderung. Um diese Aufgabe ohne weitere Controller erledigen zu können, müssen

mindestens zwei unterschiedliche Handgesten eines Benutzers unterschieden werden können („Hand offen“ bzw. „Hand geschlossen“).
Das Farbbild der Kinect kann dazu nicht verwendet werden, da die Beleuchtungsverhältnisse in der CAVE stark variieren können und von der Handgestenerkennung nicht kontrolliert werden können.
Deshalb müssen die Gesten alleine mit Hilfe der Tiefenbilder erkannt werden. Die Erkennung von Handgesten in Tiefenbildern der Kinect gilt aufgrund der relativ niedrigen Auflösung von 640 x 480 Pixeln jedoch als schwierig [11][12]. Im Folgenden wird dennoch ein einfacher robuster Algorithmus zur Handgestenerkennung skizziert, der in der CAVE der HTW Berlin getestet wird.

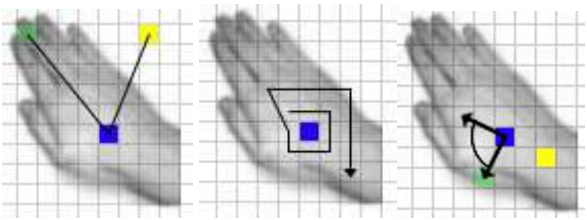

Abb. 6: von der Kinect gelieferter Handmittelpunkt (blau), Handpixel (grün), Hintergrundpixel bzw. Armpixel (gelb)

Zunächst werden ausgehend vom Handskelettpunkt alle in der Nähe liegenden Punkte klassifiziert. Tiefenpixel deren 3D-Positionen weniger als 20 cm vom Handmittelpunkt entfernt sind, werden zunächst als zur Hand gehörend klassifiziert, alle anderen Pixel gehören zum Hintergrund (siehe Abbildung 6 links). Werden beim kreisförmigen Traversieren (siehe Abbildung 6 Mitte) bei einer Umdrehung nur noch Hintergrundpixel bestimmt, wird die Traversierung abgebrochen.
Über das Skelett lässt sich ein Handrichtungsvektor als Verbindungslinie zwischen Handgelenkpunkt und Handmittelpunkt bestimmen. In einem weiteren Klassifizierungsschritt werden Punkte aussortiert, deren Richtungsvektor vom Handmittelpunkt aus mit dem Handrichtungsvektor einen Winkel bilden, der größer als 90 Grad ist, da diese zum Unterarm des Benutzers gehören werden (siehe Abbildung 6 rechts).

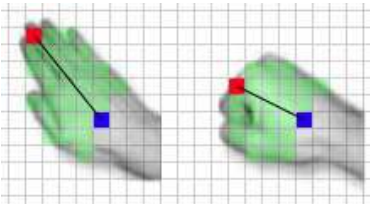

Abb. 7: Endgültig klassifizierte Handpixel (grün), von der Kinect gelieferter Handmittelpunkt (blau), Tiefenpixel mit größter Entfernung (rot).

Das Handtiefenpixel mit dem größten 3D-Abstand zum Handmittelpunkt wird zur Unterscheidung der beiden möglichen Gesten verwendet (siehe Abbildung 6).

Überschreitet der 3D-Abstand einen vorgegebenen Schwellwert, wird die Hand als „geöffnet“ sonst als „geschlossen“ klassifiziert (siehe Abbildung 7).
Für die nachfolgend beschriebenen Experimente werden zwei Schwellwerte von 10cm bzw. 11cm verwendet, um den Erkennungsalgorithmus zu stabilisieren. Unterschreitet der Abstand 10cm gilt die Hand als „geschlossen“, überschreitet der Abstand den Wert von 11cm gilt die Hand als „geöffnet“, bei Werten dazwischen bleibt der jeweils letzte Zustand erhalten.

	Hand als geöffnet erkannt	Hand als geschlossen erkannt
Hand ist offen	2811	145
Hand ist geschlossen	717	1697
Vorhersagewert	79,7%	92,1%

Tabelle 1: Positiver und negativer Vorhersagewert der Handgestenerkennung

Zum Test der Handgestenerkennung wurde ein Testdatensatz aufgezeichnet, bei dem ein Benutzer an unterschiedlichen Positionen der CAVE seine Hand insgesamt im Zeitraum von ca. 3 Minuten 412 mal öffnete oder schloss. (siehe Abbildung 8) Der Testdatensatz umfasste über 5000 Tiefenbilder mit zugehörigen Skelettdaten und Farbbildern. Für alle Bilder wurde manuell festgelegt, ob die Hand geöffnet oder geschlossen ist. Die hier beschriebene automatische Handgestenerkennung lieferte eine akzeptable Erkennungsrate (siehe Tabelle 1)

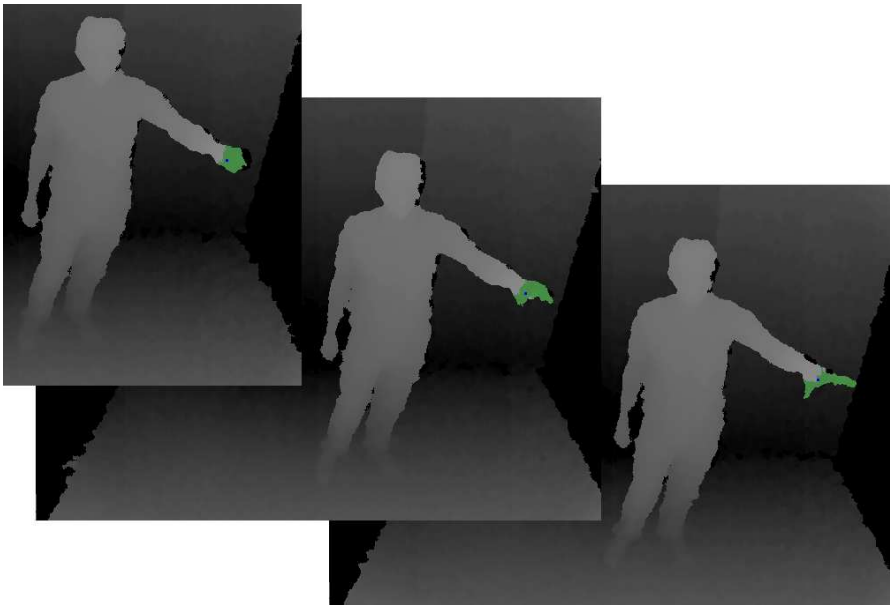

Abb. 8: Drei aufeinanderfolgende Tiefenbilder des Testdatensatzes.

Diese Werte sind zunächst recht vielversprechend. Weitere Experimente sollen Aufschluss darüber bringen, ob Interaktionskonzepte basierend auf der automatischen Handgestenerkennung mit hinreichender Qualität realisiert werden können. Eine Quelle für falsch positive Ergebnisse sind fehlerhafte Handpositionen im von der Kinect gelieferten Skelett. Bisher wurden die Experimente nur auf Basis der Skelettdaten einer Kinect-Kamera durchgeführt. Als nächstes soll untersucht werden, ob die Verwendung der zweiten Kinect auch die Handgestenerkennung verbessert.

Referenzen:

1. Cruz-Neira C., D. J. Sandin and T. A. DeFanti, "Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE", Computer Graphics, SIGGRAPH Annual Conference Proceedings, 1993.
2. Jung T., Krohn S., Schmidt P., "Ein Natural User Interface zur Interaktion in einem CAVE Automatic Virtual Environment basierend auf optischem Tracking", In Proc Workshop 3D-NordOst 2011, Berlin, Germany, December 2011, pp 93-102
3. Screentech: "ST-Silver-Screen-3D", http://www.screen-tech.de/Silver-Screen/ST-Silver-Screen-3D-D.htm, Abrufdatum: 18.5.2012
4. Yannic Schröder, Alexander Scholz, Kai Berger, Kai Ruhl, Stefan Guthe, and Marcus Magnor, "Multiple Kinect Studies", Technical Report no. 09-15, ICG, TU Braunschweig, October 2011
5. Shotton J, Fitzgibbon A, Cook M, Sharp T, Finocchio M, Moore R, Kipman A, and Blake A, "Real-Time Human Pose Recognition in Parts from a Single Depth Image", in IEEE Conference on Computer Vision and Pattern Recognition, June 2011
6. Rusu, Radu B. ; Cousins, Steve: "3D is here: Point Cloud Library (PCL)". In:IEEE International Conference on Robotics and Automation (ICRA). Shanghai, China, May 9-13 2011
7. Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin and Claudio T. Silva: "Computing and Rendering Point Set Surfaces" IEEE TVCG 9(1) Jan 2003
8. Martin A. Fischler and Robert C. Bolles (June 1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". Comm. of the ACM 24 (6): 381–395
9. Besl, P. and McKay, N. "A Method for Registration of 3-D Shapes," Trans. PAMI, Vol. 14, No. 2, 1992.
10. Microsoft-Corporation: "How to run two parallel applications with one Kinect for each application on the same computer?" 2012. http://social.msdn.microsoft.com/Forums/en-US/kinectsdk/thread/da8b44e4-07cd-49bd-b2c3-b9f095bbc7d7. Abrufdatum: 20.09.2012
11. Zhou Ren, Jingjing Meng, Junsong Yuan, Zhengyou Zhang, "Robust hand gesture recognition with kinect sensor" Proceedings of the 19th ACM international conference on Multimedia Pages 759-760 ACM New York, NY, USA 2011
12. I. Oikonomidis, N. Kyriazis and A.A. Argyros, "Efficient model-based 3D tracking of hand articulations using Kinect", in Proceedings of the 22nd British Machine Vision Conference, BMVC'2011, University of Dundee, UK, Aug. 29-Sep. 1, 2011.