Ein Kinect™ basiertes Überwachungssystem für Workflowerkennung und Gestensteuerung im Operationssaal

T.Beyl[1], P.Nicolai[1], J.Raczkowsky[1], H.Wörn[1]

[1] Karlsruher Institut für Technologie,
Institut für Prozessrechentechnik,
Automation und Robotik,
Karlsruhe,
Deutschland

Kontakt: tim.beyl@kit.edu

Abstract:

Mit der Einführung von Robotern und Navigationssystemen in den chirurgischen Workflow während minimalinvasiver bzw. offener Interventionen wird eine zusätzliche Anzahl an Gerätschaften im Operationssaal notwendig. Die Bedienung von Robotersystemen, Kamerasystemen, Trackingsystemen und der notwendigen Infrastruktur bedeuten, wenngleich der Nutzen erheblich sein kann, eine zusätzliche kognitive Last für den Chirurgen. In diesem Paper stellen wir einen Ansatz zur Überwachung des Operationsfeldes mit mehreren Kinect-Kameras und einen Ansatz zur Validierung der Echtzeitbilddaten mit Time of Flight-Kameras vor.

Schlüsselworte: Kinect, Chirurgischer Workflow, Robotik, Time of Flight

1 Problem

Mit der Einführung des Da Vinci Systems [1] erlebte die chirurgische Robotik einen deutlichen Aufschwung. Systeme wie Da Vinci sind für Interventionen konzipiert, bei denen ein Wechsel zwischen konventioneller Chirurgie und robotergestützter Chirurgie nicht, oder selten, auftritt. Im Rahmen der EU-Projekte SAFROS und ACTIVE wird der Einsatz und die Bedienung von Leichtbaurobotern (LBR4) der Firma KUKA [2] für diese Anwendungen untersucht. Ziel ist der flexible Einsatz von Robotern in der Chirurgie. Der Chirurg soll ohne großen Konfigurations- und Bedienungsaufwand wie Registrierung oder Positionierung in allen Stadien der Operation entscheiden können, ob der Robotereinsatz sinnvoll bzw. gewünscht oder im aktuellen Stadium nicht erforderlich ist. Der LBR4 stellt hierbei einen vollaktiven Roboter mit sieben Achsen dar, wobei sich Probleme bezüglich der Sicherheit des geteilten Arbeitsplatzes zwischen Chirurg und Roboter ergeben, wie z.B. etwaige Kollisionen. Ein Ansatz zur Lösung dieser Probleme ist in [3] und [4] zu finden. Die Schwierigkeit der Bedienbarkeit des Systems besteht weiterhin. Um zusätzliches Personal zur Konfiguration und damit Bedienungsoverhead im Operationssaal zu vermeiden, muss eine Lösung gefunden werden, um dem Chirurgen eine intuitive Möglichkeit zur Interaktion mit dem System zu geben. Eine Möglichkeit hierzu ist die Verwendung vordefinierter Workflows, eines Sensornetzwerkes zur Erkennung der Arbeitsschritte und geeigneter Verarbeitungsmethoden. Der vorgestellte Ansatz stützt sich hierbei vor allem auf die Observierung des Operationsfeldes mittels 3D Kameras zur Erkennung der Objekte bzw. Personen, sowie Bedienungsgesten innerhalb desselben.

2 Methoden

2.1 Versuchsaufbau

Zur Observation des Operationsfeldes im Setup kommen aktuell sieben Time-of-Flight-Kameras (ToF) der Firma PMDTec zum Einsatz. Diese bieten geringe Latenzzeiten und sind für industrielle Anwendungen konzipiert. ToF-Kameras erreichen Prinzip bedingt allerdings nur geringe Auflösungen und liefern keine Farbinformationen. Um diesen Nachteil auszugleichen und damit Methoden zur Personenerkennung sowie Workflowerkennung anwenden zu können wurde das vorhandene ToF-System um vier hochauflösende Microsoft Kinect-Kameras (640x480 Pixel) erweitert. Abbildung 1 zeigt die Anordnung der Kameras in der Draufsicht mit angedeutetem Gesichtsfeld. Die PMD CamCube 2.0 (204x204 Pixel) blickt hierbei senkrecht nach unten auf den Körper des Patienten um in dieser Projektionsebene eine vergleichsweise hohe Auflösung zu erreichen, während in den Ecken des Arbeitsfeldes die PMD S3 (64x48 Pixel) mit jeweils einer Kinect ergänzt wurden. Das komplette Setup umfasst damit vier Kinect-Kameras und sieben Time-of-

Flight-Kameras. Um gegenseitige Störungen zu vermeiden werden die ToF-Kameras durch eine speziell entwickelte Bibliothek angesteuert, die die Kameras im Zeit- und Frequenzmultiplexing-Verfahren triggert (siehe [3]).

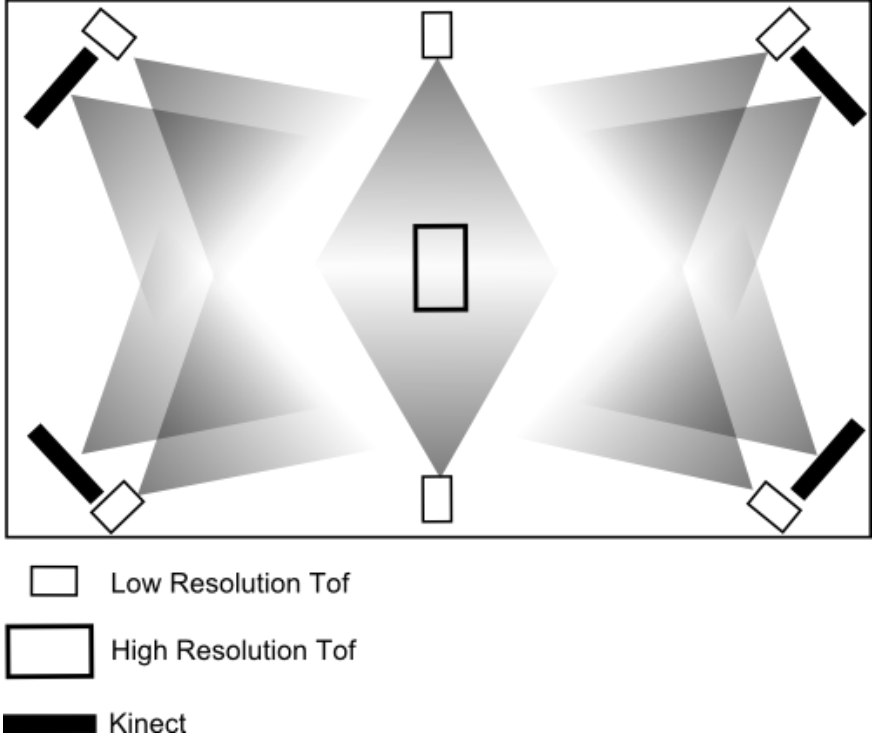


Abbildung 1: Anordnung der 3D Kameras

2.2 Technische Realisierung

Die Einführung der Kinect-Kameras stellt die technische Infrastruktur vor völlig neue Herausforderungen. Jede Kinect-Kamera liefert 11 Bit Tiefendaten mit einer Auflösung von 640x480 Pixel sowie ein zusätzliches 8bit RGB Bild mit einer Auflösung von 640x480 Pixeln sowie einer Framerate von 30 Bildern pro Sekunde (fps).
Damit ergibt sich ein theoretisches Datenvolumen für das RGB-Bild von

*640*480*3*8 Bit*30 fps = 221.184.000 Bits pro Sekunde = 27.648.000 Bytes pro Sekunde*

sowie

*640*480*11 Bit *30 fps = 101.376.000 Bits pro Sekunde = 12.672.000 Bytes pro Sekunde*

für das Tiefenbild. Es resultiert eine Gesamtdatenrate von

12.672.000 + 27.648.000 = 40.320.000 Bytes pro Sekunde = 39,375 MB pro Sekunde

Die Kinect verfügt hierzu zur Übertragung über eine USB 2.0 Hi-Speed Schnittstelle, die in der Lage ist bis zu 480Mbit/s = 60MB/s zu übertragen [5]. Ein einzelner USB Controller eines Computers ist somit in der Lage ausschließlich eine Kinect zu betreiben. Um das beschriebene Setup zu betreiben wäre somit bereits in der ersten Ausbaustufe ein Computer mit vier USB Host Controllern notwendig geworden, der sich in der Nähe der Kinect Kameras befinden müsste und somit den Hardwareaufwand im OP massiv erhöhen würde. Deshalb fiel die Wahl auf einen modularen Ansatz unter Zuhilfenahme des Robot Operating System (ROS) von Willow Garage [6]. Die Anbindung der Kinect Kameras wurde über zwei Zotac Nano AD10 Computer ausgeführt, die mit 4GB Ram und einem AMD e-350 Prozessor ausgestattet sind sowie über je zwei USB 2.0 Hi-Speed fähige Host Controller und eine Gigabit Ethernet Schnittstelle verfügen. Zum Einsatz kommen Ubuntu 12.04 und ROS Fuerte zusammen mit dem OpenNI Kinect Treiber um sowohl die 11 Bit Tiefenkarte als auch die RGB Daten der Kinect-Kameras über ein dediziertes Gigabit Netzwerk an einen zentralen Verarbeitungsknoten auf leistungsfähigere Hardware zu streamen. Dabei betreibt jeder AD10 zwei Kinect-Kameras, führt dabei allerdings keinerlei Vorverarbeitung der Daten durch.

Als zentraler Knoten kommt eine Workstation ausgestattet mit einer Geforce GTX480, 4Gb Ram und einem AMD Phenom II X6 1090T sowie Ubuntu 12.04 und ROS Fuerte zum Einsatz.
Die empfangenen Daten aller vier Kinect-Kameras werden hier unter Zuhilfenahme der bekannten Tiefenverteilung der 11 Bit Tiefenkarte zu einer dreidimensionalen Punktwolke mit Farbinformationen des RGB Sensors je Kinect kombiniert.

2.3 Registrierung

Um ein gemeinsames Referenzkoordinatensystem für die vier Punktwolken der Kinect-Kameras zu erhalten wurde in ROS Fuerte, OpenCV und der PointCloudLibrary (PCL) [7] ein Registrierungsalgorithmus unter Zuhilfenahme eines Schachbretts entworfen. Dabei fungiert eine der vier Kinect-Kameras als Referenzkoordinatensystem. Diese muss jederzeit freie Sicht auf das Schachbrett haben. Zur Registrierung jeweils einer weiteren Kamera positioniert der Benutzer das Schachbrett so, dass es im Sichtfeld der Referenzkamera sowie der der lokal zu registrierenden Kinect-Kamera liegt.

Der Algorithmus läuft in folgenden Schritten ab:

1. Das Schachbrett wird mittels der RGB Kamera erkannt: Mithilfe OpenCV werden auf den RGB Daten beider Kinect die Ecken des Schachbretts gesucht und ihre Koordinaten lokal gespeichert. Ein auf dem Schachbrett positionierter Kreis, der mittels Hough-Transformation gefunden wird, stellt sicher, dass die Orientierung des Schachbrettes festgestellt werden kann ist.
2. Zu jedem RGB Wert existiert ein korrespondierender Tiefenwert des Tiefensensors der Kinect. Dieser berech- net sich über die intrinsische Kalibrierung von RGB- zu Tiefenkamera sowie bekanntem Tiefenprofil des Sen- sors. Dennoch kann nicht gewährleistet werden, dass die Tiefenkamera an der entsprechenden Position des Schachbrettes einen Wert misst. Fehlende Werte resultieren aus mangelnder Beleuchtung oder unzureichendem Reflektionsgrad des eingesetzten Materials. Außerdem unterliegt die Tiefenmessung des Sensors stochasti- schem Rauschen, das sich durch Mittelwertfilterung weitestgehend eliminieren lässt. Um mit diesem Problem umzugehen wird eine frei wählbare Menge an Frames gesammelt und für jede Schachbrettecke die Anzahl der Tiefenwerte erfasst, die über alle Frames gemessen werden konnten. Anschließend wird für jeden Punkt der Mittelwert aus den erfassten 3D-Koordinaten berechnet. Mit 300 Frames konnte in Versuchen eine ausreichend große Menge an rauscharmen Tiefenwerten gesammelt werden.
3. Zu jeder Ecke des Schachbretts wird ein Korrespondenzpaar zwischen der Ecke im Koordinatensystem der zu registrierenden Kinect und der Ecke im Koordinatensystem der Referenz hergestellt. Damit ergibt sich ein Satz von Korrespondenzen, der in Schritt 5 weiterverwendet wird.
4. Schritt 2+3 müssen für verschiedene Positionen des Schachbrettes wiederholt werden. Um die Rotation zwi- schen den Kinect-Kameras akkurat zu bestimmen ist es hierbei notwendig, die Schachbrettpositionen so zu wählen, dass die Ebenen des Schachbretts an unterschiedlichen Positionen nicht parallel zueinander liegen. Mit fünf verschiedenen Positionen konnte in unseren Versuchen bereits eine zufriedenstellende Registrierung er- reicht werden.
5. Aus den erfassten Korrespondenzen wird mittels Hauptkomponenten-Analyse (PCA) die Rotation und Transla- tion der beiden Kinects zueinander berechnet.

Abbildung 2 zeigt die gemeinsam visualisierten Punktwolken aus den Daten von vier deckenmontierten Kinect (siehe Abbildung 1), die mit dem hier vorgestellten Algorithmus registriert wurden. An schwarzen oder leeren Flächen liegen keine Messpunkte vor, z.B. aufgrund von Verdeckungen durch den OP-Tisch oder schlecht reflektierenden Materialen.

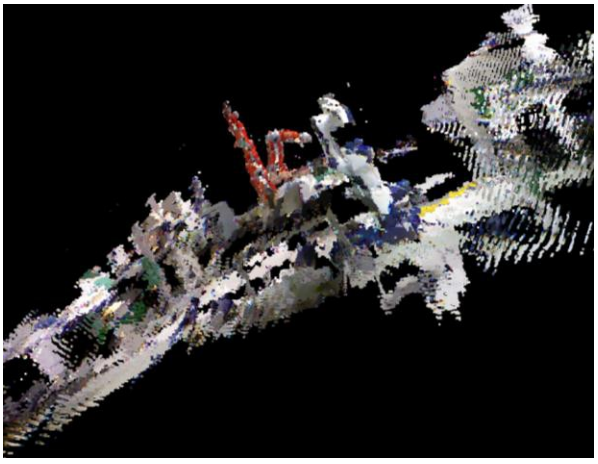
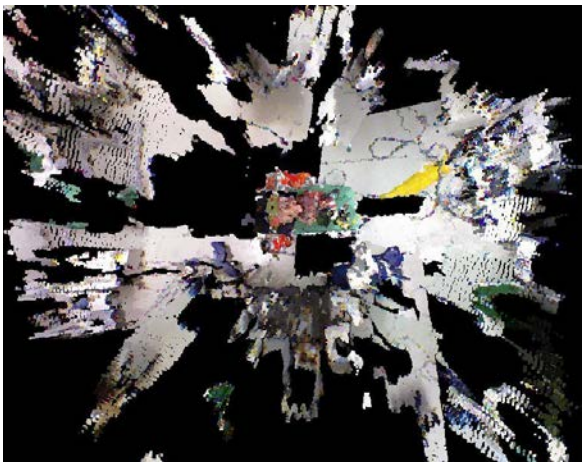


Abbildung 2: Gemeinsam visualisierte Punktwolken aus den Ansichten von vier registrierten Kinect-Kameras; links: Seitenansicht; rechts: Aufsicht



3 Ergebnisse

Die Anordnung von Kinect- und Time-of-Flight-Kamers mit überlappenden Gesichtsfelder liefert zwei verschiede Ansichten derselben Szene. Das Setup ist damit in der Lage sowohl die Vorteile der ToF-Kameras zu nutzen und Kollisionen mit geringer Latenz und Verlässlichkeit zu detektieren, als auch die die hochauflösenden Daten der Kinect für Per-

sonentracking, Workflowdetektion und Gestensteuerung zu verwenden.
Die Auslegung des Systems ermöglicht bei einer Registrierung des Kinect-Kamera-Systems zum ToF-Kamera-System die Validierung der Kinect-Daten mittels der ToF-Daten. Hierzu könnten beispielsweise bekannte Objekte wie die Leichtbauroboter, die in beiden Datensätzen leicht erkennbar sind verwendet werden um eine online Validierung der Daten zu berechnen.
Dennoch muss beim Einsatz der Kinect, speziell in der vorgestellten Anwendung, die die Daten über das Netzwerk streamt, mit größeren Latenzen gerechnet werden. So ist im Video Stream eine deutliche Verzögerung zu sehen, die die 300ms Latenz der Kinect deutlich übersteigt und ebenfalls noch genauer quantifiziert werden muss.

Die Qualität des Registrierungsergebnisses hängt maßgeblich von der Beleuchtung sowie der Anzahl der Korrespondenzen ab. Im aktuellen Setup lassen sich Registrierungsfehler im Bereich weniger Zentimeter erzielen, der in weiteren Untersuchungen näher quantifiziert werden muss. Erste Versuche zeigen Registrierungsfehler bei 5 Schachbrettposen mit jeweils 300 Samples die Im Median deutlich unter 4cm liegen.
Die Einzelpunktwolken bleiben bei dem beschriebenen Verfahren erhalten und können jederzeit unter Nutzung der bestimmten Transformationsmatrizen zu einer Gesamtpunktwolke erweitert werden. Hierbei werden bisher keine Daten verworfen, sondern auf Wunsch alle Tiefen- und alle Farbinformationen der Kameras in einer gemeinsamen Punktwolke abgelegt, die zur Weiterverarbeitung verwendet werden kann.

Der Einsatz der Zotac Nano AD10-Rechner zeigt, dass auch verhältnismäßig langsame Rechner durchaus in der Lage sind Kinect-Daten zu streamen. In Experimenten wurde eine Geschwindigkeit von ca. 20-25 fps unter Verwendung aller vier Kinect Kameras und der Nachverarbeitung auf der Workstation erreicht. Seit Kurzem existiert weiterhin ein Kinect-Treiber für den Nvidia Tegra 3 Prozessor [8], der sich durch geringe Wärmeentwicklung auszeichnet und passiv gekühlt werden kann.
In medizinischen Applikationen kann damit kleine und leichte Hardware auch im Operationssaal eingesetzt werden, durch aufwendige Kühlmaßnahmen mittels Lüftern das Risiko der Einbringung von Keimen zu vergrößern. Auch der AMD E-350-Rechner wäre ohne großen Mehraufwand passiv zu kühlen. Durch Austausch der Kinect gegen eine lüfterlose Asus Xtion [9], die kompatibel zum Protokoll der Kinect ist und dieselbe Sensorik verwendet, kann eine komplett lüfterlose Lösung erzielt werden.

4 Diskussion

Es wurde ein modulares und skalierbares System zur Detektion von Personen im Operationssaal entwickelt. Durch den zusätzlichen Einsatz von ToF-Kameras soll der sichere Einsatz von Kinect-Kameras im Operationssaal ermöglicht werden. Diese bieten die Möglichkeit einen online Redundanzcheck der gesammelten 3D-Daten durchzuführen und damit die Entwicklungsumgebung OP:Sense [10] bestmöglich zu ergänzen. Es ergibt sich dadurch die Möglichkeit mithilfe der ToF-Kameras mit geringstmöglicher Latenzzeit auf Änderungen in der Umgebung zu reagieren, um Kollisionen zu vermeiden als auch zuverlässige hochauflösende ToF validierte Kinect-Daten zu sammeln, die z.B. bei der Workflowerkennung erforderlich sind. Aufgrund der Möglichkeit, die Daten unkomprimiert über ein Gigabit Netzwerk zu übertragen, kann jederzeit entschieden werden, ob für die spezifische Anwendung mit maximaler Auflösung gearbeitet wird oder ob auch ein auflösungsreduzierter Datensatz verwendet werden kann, um die Prozessorlast zu reduzieren.
Dennoch bleibt der relativ große Fehler der Kinect-Kameras (im Bereich mehrerer Zentimeter), was unter Anderem einer bisher nicht durchgeführten intrinsischen Kalibrierung des Tiefensensors geschuldet ist, die für jede Kinect gesondert durchgeführt werden muss. Die Durchführung dieser intrinsischen Kalibrierung, die mit hoher Wahrscheinlichkeit zu einer Verbesserung des Registrierungsergebnisses führen wird, ist für die Zukunft geplant. Der beschriebene und eingesetzte Registrierungsalgorithmus ermöglicht die Registrierung des Systems ohne zusätzliche externe Hardware und weitere Fehlerquellen und bietet damit die Möglichkeit das System schnell und einfach transportieren und vor Ort registrieren zu können. Wir arbeiten an weiteren Verfahren um nach der Registrierung mit einem modifizierten Algorithmus ebenfalls automatisch ohne externe Hardware den Fehler zu bestimmen, der später zur Konfidenz Schätzung verwendet werden soll.

Um den sinnvollen Einsatz im chirurgischen Kontext evaluieren zu können müssen weitere Versuche unter Verwendung von Robotern zusammen mit Chirurgen durchgeführt werden, um das System auf spezifische chirurgische Interventionen hin optimieren zu können.
Zu erwarten steht eine Verringerung der Arbeitslast der Chirurgen insbesondere bei Interventionen, bei denen der Roboter sowohl handgeführt als auch autonom oder telemanipuliert eingesetzt wird und bei denen häufig zwischen den Arbeitsmodi gewechselt wird. Hier kann eine intuitive Bedienung der Roboter und ein Wechsel zwischen den Betriebsmodi über ein 3D-Kamerasystem realisiert werden.

5 Danksagung

Diese Arbeit wurde durch das FP7 Framework Programm der Europäischen Union innerhalb der Projekte „Patient Safety in Robotic Surgery (SAFROS)" unter Grant Nr. 248960 und „Active Constraints Technologies for Ill-defined or Volatile Environments (ACTIVE)" unter Grant Nr. 270460 unterstützt und gefördert.

6 Referenzen

[1] http://www.intuitivesurgical.com Letzter Aufruf: 30.07.2012
[2] http://www.kuka-robotics.com/germany/de/products/addons/lwr/ Letzter Aufruf: 30.07.2012

[3] P. Nicolai, H. Mönnich, J. Raczkowksy, H. Wörn, J. Bernshausen. Überwachung eines Operationssaals für die kooperative robotergestützte Chirurgie mittels neuartiger Tiefenbildkameras. (2010) Tagungsband der 9. Jahres- tagung der Deutschen Gesellschaft für Computer- und Roboterassistierete Chirurgie.

[4] H. Mönnich, P. Nicolai, T. Beyl, J. Raczkowsky, H. Wörn. A Supervision System for the Intuitive Usage of a Telemanipulated Surgical Robotic Setup (2011) Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO 2011)

[5] Universal Serial Bus Specification Revision 2.0, 27.04.2000

[6] M.Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, A. Ng. ROS: an open- source Robot Operating System (2009), ICRA Workshop on Open Source Software

[7] R. Rusu, S. Cousins. 3D is here: Point Cloud Library (PCL) (2011). International Conference on Robotics and Automation.

[8] http://pointclouds.org/blog/nvcs/raymondlo84/index.php Letzter Aufruf: 30.07.2012, R. Lo, Nvidia Code Sprint

[9] http://www.asus.de/Multimedia/Motion_Sensor/Xtion_PRO/ Letzter Aufruf: 30.07.2012

[10] P. Nicolai, T. Beyl, H. Moennich, J.Raczkowsky, H. Wörn. OP:Sense – An integrated Rapid Development En- vironment in the Context of Robot Assisted Surgery and Operation Room Sensing. (2011) Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO 2011)