Mensch und Computer 2014, 31. August - 03. September, München: Oldenbourg Verlag

Preprint

Intuitive Erzeugung von 3D-Modellen mit handgehaltenen Sensoren

Maximilian Sand, Dominik Henrich

Lehrstuhl für Robotik und Eingebettete Systeme, Universität Bayreuth

Zusammenfassung

Aufgrund der weit verbreiteten Verwendung von dreidimensionalen Computermodellen ist es wünschenswert, dass auch ungeübte Personen leicht ein digitales Modell eines realen Objektes oder einer Szene mit einem handgehaltenen Sensor erstellen können. Dazu wird eine Benutzerschnittstelle vorgestellt, die dem Anwender während der Aufnahme eine dreidimensionale Live-Rekonstruktion anzeigt, in die Hinweise zur Unterstützung des Aufnahmevorgangs eingeblendet werden. Unter Berücksichtigung von Verdeckungen wird der Nutzer auf noch nicht erfasste Bereiche hingewiesen. Dadurch kann der Aufnahmevorgang beschleunigt werden.

1 Einleitung und Motivation

Detaillierte dreidimensionale Computermodelle sind heutzutage allgegenwärtig. Mit den Ursprüngen in den Ingenieurswissenschaften zur Unterstützung der Konstruktion und Simulation, sind heute 3D-Modelle den meisten Personen durch Computerspiele oder Filme bekannt, in denen komplette virtuelle Welten erzeugt werden. Brillen für virtuelle Realität simulieren eine komplette Umgebung, in der die Nutzer zu unterschiedlichen Anwendungszwecken navigieren können. Die Erstellung eines dreidimensionalen Modells einer Umgebung oder einer realen Person ist hingegen meist nur mit Fachkenntnis zu bewerkstelligen. Durch die Entwicklung immer ausgereifterer Sensoren ist es möglich, 3D-Modelle mit einfacher Hardware zu erzeugen, indem ein Sensor um ein Objekt oder um eine Person oder durch eine Szene bewegt wird und dabei automatisch ein Computermodell erzeugt wird. Damit sind zukünftig viele Anwendungen vorstellbar: Die eigene Person könnte nach der Digitalisierung als dreidimensionaler Avatar in Spielen oder Kommunikationsplattformen verwendet werden. Planung und Einrichtung von Wohnungen könnte vereinfacht werden, indem ein reales Zimmer mit einem Smartphone aufgenommen wird und Möbel virtuell platziert werden.

Die Erzeugung von detailreichen, vollständigen Computermodellen von realen Objekten oder Umgebungen stellt den Anwender vor einige Herausforderungen: Es muss gewährleistet werden, dass jeder Teil eines Objekts mit der nötigen Auflösung erfasst wird. Zudem muss

das Objekt vollständig erfasst werden, sodass das resultierende Modell kein Lücken aufweist. Insbesondere wenn durch die Geometrie viele Verdeckungen auftreten, sind unter Umständen viele unterschiedliche Aufnahmepositionen nötig. Während automatisierte Systeme (beispielsweise unter Verwendung eines Roboterarms) dies berücksichtigen (Krainin et al. 2011), sind Rekonstruktionen durch handgehaltene Sensoren oft unvollständig, sodass erneute Aufnahmen oder eine Nachbearbeitung nötig sind.

In dieser Arbeit wird deshalb eine Schnittstelle zwischen Mensch und Computer vorgestellt und untersucht, die dem Anwender während eines Aufnahmeprozesses mit einer RGBD-Kamera Rückmeldungen über die Qualität und Vollständigkeit des bis dahin erzeugten Modells (kolorierte Punktwolke) liefert. Damit ist der Anwender in der Lage, direkt zu reagieren und weitere Aufnahmezyklen zu vermeiden. Diese Rückmeldungen des Systems werden in eine rekonstruierte 3D-Ansicht der Szene eingeblendet, sodass die Orientierung im Raum auch ungeübten Personen leicht fällt.

Dieser Beitrag gliedert sich wie folgt: Zuerst wird in Kapitel 2 der Stand der Forschung vorgestellt, in Kapitel 3 wird anschließend der eigene Ansatz zur Erzeugung der Rückmeldungen erläutert. In Kapitel 4 werden die durchgeführten Experimente mit Nutzern beschrieben.

2 Stand der Forschung

Der Großteil der Verfahren zur Rekonstruktion von realen Objekten sind vollautomatisierte Systeme, in denen der Sensor oder das aufzunehmende Objekt auf einem Gerät (beispielsweise auf einem Drehteller oder einem Roboterarm) montiert ist, und so die Aufnahmeposition angepasst werden kann. In einem ständigen Zyklus aus Planen, Aufnehmen, Registrieren und Integrieren (Scott et al. 2003) wird für jede Aufnahme die bestmögliche nächste Sensorposition ermittelt, sodass möglichst wenige Aufnahmen durchzuführen sind. Diese Problemstellung wird als *Next-Best-View*-Problem bezeichnet. Um auch bei einer hohen Anzahl an Freiheitsgraden einer Pose im Raum (drei für die Position plus drei für die Orientierung) brauchbare Laufzeiten zu erzielen, wird der Suchraum um das Objekt eingeschränkt, beispielsweise auf einen Zylinder (Pito 1999) oder eine Kugel (Banta et al. 2000). Keines dieser Verfahren kann mit mehreren Objekten und den daraus entstehenden Verdeckungen umgehen. Besser geeignet ist hier ein Verfahren auf Basis eines Voxelraumes (Potthast & Sukhatme 2013). Dabei wird der Raum in Voxel aufgeteilt, in denen jeweils ein Zustand (belegt, frei und unbekannt) gespeichert wird. Mit einem probabilistischen Ansatz wird anschließend eine Kamerapose hinsichtlich ihres Informationszugewinns bewertet.

Bestehende Systeme, in denen ein Mensch den Sensor bewegt, leisten zwar die Erstellung eines Computermodells, interagieren mit der Person aber wenig bis gar nicht. Systeme, die dem Anwender nur eine Live-Rekonstruktion in Form einer Punktwolke bieten (Izadi et al. 2011; Endres et al. 2012; Whelan et al. 2013; Stückler & Behnke 2014), haben den Nachteil, dass unnötige Wege zurückgelegt werden müssen, da fehlende oder schlechte erfasste Bereiche besonders in stark unstrukturierten Umgebungen mit vielen Verdeckungen nur schwer

erkennbar sind. Nach einer Inspektion des Modells ist es dann nötig, erneut Szenenbilder aufzunehmen, um die Rekonstruktion zu verbessern.

Systeme, die eine Interaktion mit dem Nutzer unterstützen, teilen sich in zwei Gruppen auf: Bei der einen Gruppe unterstützen Benutzereingaben das Verfahren mit zusätzlichem Wissen, um ein besseres Ergebnis zu erhalten. Beispiele hierfür sind eine durch Handskizzen unterstützte Segmentierung (Sedlacek & Zara 2012) oder eine Korrektur von falschen Korrespondenzen (Schneider & Eisert 2012). Bei der anderen Gruppe werden die Nutzer direkt durch Rückmeldungen unterstützt. Das Verfahren in (Dellepiane et al. 2013) schlägt gute Aufnahmeposen vor, ist allerdings nur für einzelne Objekte anwendbar. Für unstrukturierte Umgebungen mit Verdeckungen zeigt die Arbeit von (Du et al. 2011) dem Nutzer eine Übersicht über noch nicht erfasste Bereiche als horizontaler 2D-Schnitt an. Damit kann der Nutzer jedoch nie die ganze Szene auf einmal überwachen und erhält keine Hinweise auf mögliche Aufnahmepositionen.

Insgesamt lässt sich feststellen, dass kein Verfahren in unstrukturierten Umgebungen anwendbar ist und gleichzeitig Rückmeldungen zu möglichen Aufnahmepositionen liefert, um so den Nutzer auf noch nicht erfasste Bereiche hinzuweisen.

3 Ansatz

In diesem Kapitel wird das entwickelte Verfahren zur Erzeugung von intuitiven Rückmeldungen an den Nutzer beschrieben. In Abschnitt 3.1 wird untersucht, welche Unterschiede zu automatisierten Systemen bestehen und welche Konsequenzen sich daraus ergeben. Abschnitt 3.2 beschreibt anschließend das Vorgehen zur Erzeugung von Rückmeldungen.

3.1 Problemanalyse

Eine mögliche Herangehensweise für ein interaktives System, das den Nutzer während der Aufnahme unterstützt, ist die Übertragung von Verfahren von vollautomatisierten Systemen. Die Anwendung von Next-Best-View-Algorithmen lassen sich jedoch nicht direkt übertragen, denn in automatisierten Systemen gelten folgende Eigenschaften: Erstens kann eine Sensorpose mit sehr hoher Genauigkeit sowie sehr schnell eingenommen werden, zweitens besitzt die ausführende Einheit kein Berechnungsvermögen oder Intelligenz. Für den Fall eines handgehaltenen Sensors gelten jedoch andere Voraussetzungen: Ein Mensch, der eine Kamera durch einen Raum bewegt, kennt weder das verwendete Koordinatensystem, noch könnte er im Raum eine vorgegebene Position ohne Markierungen oder andere Hilfssysteme zur Positionsbestimmung exakt einnehmen. Des Weiteren besitzt der Mensch ein räumliches Vorstellungsvermögen, sowie gute visuelle Wahrnehmung. Teilaufgaben, die für einen Computer aufwändig zu berechnen sind, aber für einen Menschen sofort ersichtlich sind, können dem Menschen überlassen werden. Daraus ergeben sich folgende Schlussfolgerungen für ein System mit handgehaltenen Sensoren: Das System muss unbekannte oder schlecht erfasste Bereiche feststellen und erkennen, aus welcher groben Richtung eine unverdeckte Erfassung möglich ist. Die Berechnung einer exakten Aufnahmepose ist nicht

nötig, da der Nutzer diese nicht exakt einnehmen kann. Das Problem weist nun erheblich weniger Komplexität auf, da keine Pose mit sechs Freiheitsgraden gefunden werden muss. Es bleibt allerdings die Aufgabenstellung, eine passende Darstellung für diese Mensch-Maschine-Interaktion zu finden, die eindeutig und klar wiedergibt, welche Bereiche im Raum gemeint sind, sodass der Nutzer leicht darauf reagieren kann.

3.2 Algorithmus

Dieses Kapitel stellt den entwickelten Algorithmus zum Erstellen von Nutzerhinweisen vor. Gegeben sei eine Sequenz von m Aufnahmen $I = (I_0, I_1, \dots, I_{m-1})$. Jede Aufnahme I_i besteht dabei aus der Position $p_i \epsilon\, \mathbb{R}^3$ der Kamera, der Orientierung $R_i \epsilon\, \mathbb{R}^{3\times3}$ und den Sensorwerten als Punktwolke in Weltkoordinaten C_i, also $I_i = (p_i, R_i, C_i)$. Gesucht ist eine Menge von Nutzerhinweisen, die dem Benutzer schlecht erfasste Stellen aufzeigen soll. Ein Hinweis bestehe aus einer Position und einer Richtung mit folgender Bedeutung: Der Nutzer soll diese Position aus ungefähr dieser Richtung erfassen. Abbildung 1 zeigt ein Beispiel in 2D.

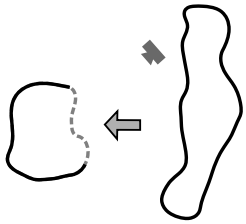


Abbildung 1: 2D-Illustration mit Objekten, von denen das linke noch nicht erfasste Bereiche enthält (gestrichelte Linie). Ein möglicher Nutzerhinweis (Pfeil) weist den Benutzer auf die Fehlstelle hin. Die fehlenden Stellen sollen dann durch den Benutzer von einer Position seiner Wahl (Kamerasymbol) aus aufgenommen werden.

Grundlegend kann man das Verfahren in sechs Schritte aufteilen: Zuerst wird ein Voxelgitter für den zu untersuchenden Bereich erstellt. Jedes Voxel enthält für jede seiner sechs Seitenflächen einen Zustand mit den möglichen Werten *unbekannt*, *belegt* und *frei*, sowie die Anzahl der Punkte im Voxel. Kommt eine neue Aufnahme hinzu, so werden die Zustände und der Punktzähler mittels eines Raycasting-Ansatzes aktualisiert. Anschließend werden sogenannte Fehlstrahlen aus dem Voxelraum extrahiert. Fehlstrahlen sind Kandidaten für spätere Nutzerhinweise und enthalten mehrere benachbarte Voxel mit unbekannten Seitenflächen in derselben Richtung. Alle Fehlstrahlen werden nun mithilfe eines Bewertungssystems hinsichtlich ihres Nutzens für den Anwender beurteilt. Fehlstrahlen mit hoher Bewertung werden anschließend in Nutzerhinweise umgewandelt, die als visuelle Repräsentation für den Nutzer dienen. Um die Zahl der Hinweise noch weiter zu reduzieren, können in einem letzten Schritt mehrere benachbarte Nutzerhinweise noch zusammengefasst werden. Im Folgenden werden nun die einzelnen Schritte näher erläutert.

3.2.1 Erstellung des Voxelraumes

Durch die Verwendung eines Voxelgitters wird der Raum in Raumwürfel unterteilt. Jedes Voxel $v \in V$ des Voxelraumes V mit $n = n_x \cdot n_y \cdot n_z$ Voxeln enthalte die Anzahl der Punkte

im Voxel, sowie pro Seite einen Zustand $s(v,d) \in \{unbekannt, belegt, frei\}$. Die sechs Seiten eines Voxels seien mit $D = \{X^+, X^-, Y^+, Y^-, Z^+, Z^-\}$ bezeichnet. Diese Aufteilung in sechs Richtungen wird für die Hinweise genutzt, indem nur Hinweise generiert werden, die parallel zu den Normalen der sechs Voxelseiten orientiert sind. Wie in der Problemanalyse festgestellt, genügt diese Einschränkung, da der Nutzer die tatsächliche Aufnahmerichtung anschließend selbst bestimmt. Durch die Betrachtung von sechs Richtungen pro Voxel genügt ein verhältnismäßig grober Voxelraum für das Erstellen von Nutzerhinweisen. Dies beschleunigt auch das Raycasting und das Finden von Hinweisen. Die Größe einzelner Voxel sollte so gewählt werden, dass im Freiraum zwischen zwei realen Objekten mindestens ein komplettes Voxel liegt. Es empfiehlt sich daher eine Voxelseitengröße von $\frac{d_{min}}{2}$, wobei d_{min} der minimale Abstand zwischen zwei Objekten in der aufzunehmenden Szene ist.

3.2.2 Raycasting

Initial besitzen alle Seiten aller Voxel den Zustand *unbekannt*. Kommt eine neue Aufnahme $I_i = (p_i, R_i, C_i)$ hinzu, so müssen die Zustände aktualisiert werden. Dazu wird ein Raycasting-Ansatz für Punkte der Punktwolke C_i verwendet. Für einen Punkt $c \in C_i$ der Wolke müssen alle Voxel entlang des Strahls von p_i bis c auf *frei*, sowie das Voxel, das c enthält, auf *belegt* gesetzt werden. Zur Traversierung der Voxel wird das Verfahren von (Amanatides & Woo 1987) verwendet, das für jeden Punkt der Punktwolke parallel ausgeführt werden kann. Von jedem besuchten Voxel werden entsprechend der Komponenten des Richtungsvektors des Strahls die Zustände der drei zugehörigen Seiten auf *belegt* gesetzt. Beim Raycasting der Punkte einer Wolke ist es durch die grobe Auflösung des Voxelgitters möglich, dass mehrere Strahlen durch die gleiche Seite eines Voxel gehen, ein Teil davon im Voxel aufgrund von Punkten endet, ein anderer Teil davon allerdings durch das Voxel hindurch verläuft. In letzeren Fall wird der Zustand der betreffenden Voxelseite auf *belegt* gesetzt.

3.2.3 Finden von Fehlstrahlen

Im nächsten Schritt werden nun sogenannte Fehlstrahlen gefunden. Ein Fehlstrahl *f* sei definiert als das Tupel $f = (v_{start}, v_{ende}, d)$ mit $v_{start}, v_{ende} \in V$ und $d \in D$, und beschreibt eine Sequenz von hintereinanderliegenden Voxeln beginnend bei v_{start} und endend bei v_{ende}, die alle in der Richtung d den Zustand *unbekannt* besitzen. Der Grund, diese Voxel in einem Fehlstrahl zusammenzufassen, ist die Tatsache, dass diese unbekannten Seiten mit einer einzigen Aufnahme erfasst werden können, da sie hintereinanderliegen.

Die Position eines Voxels im Voxelraum *V* kann eindeutig durch seine Koordinate $(x, y, z) \in \mathbb{N}_0^3$ und $0 \leq x \leq n_x, 0 \leq \mathrm{y} \leq n_y, 0 \leq \mathrm{z} \leq n_z$ beschrieben werden. Im Folgenden werde nun eine einzelne Voxelsequenz betrachtet, die V in Richtung d durchläuft. Dadurch sind zwei der drei Koordinaten der Voxel fest, nur eine Koordinate ändert sich. Sei o.B.d.A. y und z fest und $0 \leq x \leq n_x$. Betrachtet man die Zustände der Voxel in d-Richtung, so entsteht eine Folge von Zuständen, die mit $s(x)$ bezeichnet werde (Abbildung 2).

Das Ziel ist es nun, innerhalb einer Voxelsequenz alle Fehlstrahlen, also eine Sequenz von aufeinanderfolgenden Voxeln mit Zustand *unbekannt*, zu finden. Die Positionen, an denen ein Fehlstrahl beginnt (v_{start}) und endet (v_{ende}), können dabei leicht ermittelt werden, indem eine Fallunterscheidung für zwei aufeinanderfolgende Zustände durchgeführt wird.

Folgt beispielsweise auf einen *freien* Voxel ein *unbekannter* Voxel, so beginnt ein Fehlstrahl (z.B. bei $x = 1$ in Abbildung 2). Nach *belegten* Voxeln beginnt ein Fehlstrahl erst beim zweiten *unbekannten* Voxel ($x = 5$ in Abbildung 2). Dadurch wird verhindert, dass Fehlstrahlen Voxelseiten enthalten, die nie erfasst werden könnten, weil das in Richtung *d* vorhergehende Voxel *belegt* ist und somit diese Seite immer verdecken würde. Eine besondere Betrachtung verdienen Voxel, die aus Richtung *d unbekannt* sind, aber mehr als eine vorgegebene Anzahl an Punkten enthalten. Ein solches Voxel muss dann mindestens eine andere Seite besitzen, die *belegt* ist, und wird dann auch mit hoher Wahrscheinlichkeit aus Richtung *d belegt* sein. Das Verwenden einer Schwelle dient dazu, Fehlmessungen und Rauschen, die sich im Vorhandensein von einzelnen Punkten innerhalb eines Voxels widerspiegeln, zu ignorieren. Zusammenfassend ergibt sich also folgende Berechnungsvorschrift für eine Voxelsequenz: Durchlaufe die Voxelsequenz von Index $x = 0$ bis n_x und starte beziehungsweise beende einen Fehlstrahl anhand der Fälle in Tabelle 1. Falls ein Strahl noch aktiv ist, wenn das Ende der Sequenz erreicht ist, so beende diesen Strahl mit $v_{ende} = (n_x, y, z)$.

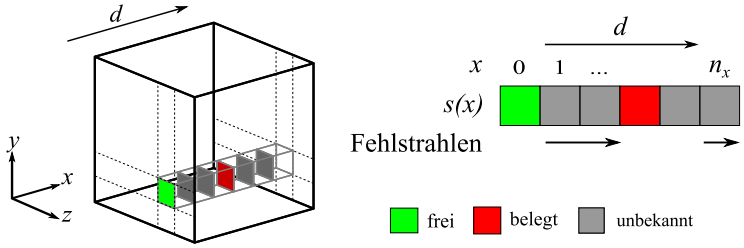


Abbildung 2: Eine Voxelsequenz in Richtung d innerhalb des Voxelraumes (links). Betrachtet man nur die Zustände in Richtung d, so entsteht eine Folge aus Zuständen s(x), die mit x parametrisiert ist (rechts). Daraus lassen sich Fehlstrahlen extrahieren.

Zustände $s(x-1)$,	$s(x)$	**Strahl ist nicht aktiv**	**Strahl ist aktiv**
frei, *unbekannt o.P.*,	*unbekannt o.P.*	Neuer Strahl mit $v_{start} = (x, y, z)$	-
frei,	*unbekannt m.P.*	Strahl der Länge 1 mit $v_{start} = v_{ende} = (x, y, z)$	-
unbekannt o.P.,	*belegt*	-	Beende Strahl bei $v_{ende} = (x-1, y, z)$
unbekannt o.P.,	*frei*		
unbekannt m.P.,	*belegt*		
unbekannt m.P. ,	*frei*		
unbekannt o.P.,	*unbekannt m.P.*	Strahl der Länge 1 mit $v_{start} = v_{ende} = (x, y, z)$	Beende Strahl bei $v_{ende} = (x-1, y, z)$
sonst		-	-

Tabelle 1: Fallunterscheidung für den Durchlauf von Voxelsequenzen zur Erkennung von Fehlstrahlen in Richtung d. $0 < x < n_x$; y, z *fest. Beim Zustand* unbekannt *wird unterschieden zwischen* unbekannt ohne Punkte *(o.P.) und* unbekannt mit Punkten *(m.P.).*

Bisher wurden beim Durchlauf in Richtung d auch nur die Zustände in Richtung d betrachtet. Durch Anpassung der Fallunterscheidungen können auch Fehlstrahlen in die zu d entgegengesetzte Richtung erkannt werden. Insgesamt erfolgt also die Berechnung aller Fehlstrahlen folgendermaßen: Durchlaufe alle Voxel und überprüfe für alle sechs Richtungen, ob ein Fehlstrahl im aktuellen Voxel begonnen oder beendet werden muss, indem der Zustand dieses Voxels sowie die drei Zustände der benachbarten Vorgängervoxel ausgewertet werden.

3.2.4 Bewertung von Fehlstrahlen

Die Gesamtzahl der Fehlstrahlen im Voxelraum kann sehr groß werden und ist für eine sinnvolle Rückmeldung an den Benutzer noch weiter zu reduzieren. Dazu wird für jeden Fehlstrahl eine Bewertung errechnet, die dessen Relevanz beschreibt. Die Weiterverarbeitung der Fehlstrahlen kann dann beispielsweise nur für Fehlstrahlen mit einer Bewertung über einer bestimmten Schwelle erfolgen oder nur für die besten Fehlstrahlen.

Die Bewertung eines Fehlstrahls lässt sich nach folgenden Regeln berechnen: Enthält der letzte Voxel im Fehlstrahl oder der darauf folgende Voxel eine geringere Anzahl an Punkten, als eine Schwelle vorgibt, so ist die Bewertung Null. Dies bewirkt, dass nur Fehlstrahlen, die auf Voxeln mit Punkten, also mit bereits rekonstruierten Teilen der Szene, enden, eine hohe Bewertung bekommen und Punkte, die durch Sensorrauschen entstanden sind, ignoriert werden. Alle anderen Fehlstrahlen werden anhand ihrer Position im Voxelraum bewertet. Dies modelliert die Hypothese, dass Objekte im Inneren des Voxelraumes dem Nutzer wichtiger sind als Objekte direkt am Rand. Zur Berechnung wird für jeden Voxel im Fehlstrahl ein Wert errechnet, das Maximum davon ist die Bewertung des Fehlstrahls. Die Bewertung eines einzigen Voxel geschieht mittels einer Sigmoid-Funktion, die Voxel am Rand einen Wert von Null, und Voxel im Zentrum einen Wert von Eins zuweist.

3.2.5 Generierung von Nutzerhinweisen aus Fehlstrahlen

Um die Informationen der Fehlstrahlen an den Nutzer weiterzugeben, ist eine visuelle Darstellung, genannt Nutzerhinweis, nötig. Ein Nutzerhinweis besitzt eine Position und eine Richtung, die den Anwender anleiten soll, an dieser Position und aus ungefähr dieser Richtung eine weitere Erfassung der Szene durchzuführen. Eine einfache Visualisierung einer Position und einer Richtung ist ein Pfeil. Dieser kann innerhalb der 3D-Rekonstruktion eingeblendet werden, die Position im Raum sowie die Richtung sind damit leicht erkennbar. Für die Positionierung des Pfeils im Raum gibt es drei Möglichkeiten: Die visuelle Darstellung des Pfeil reicht von v_{start} bis v_{ende}, ein kurzer Pfeil wird bei v_{start} platziert, oder ein kurzer Pfeil wird bei v_{ende} platziert. Die erste Möglichkeit hat zwar den Vorteil, dass die Länge des zugrundeliegenden Fehlstrahls noch ersichtlich ist, doch die Visualisierung nimmt viel Platz in Anspruch. Ein Pfeil bei v_{start} besitzt den Vorteil, dass der komplette Fehlstrahl vor der Pfeilspitze liegt. Nimmt ein Nutzer die Position des Pfeils ein, so kann er vermutlich alle unbekannten Voxelseiten auf einmal erfassen. Platziert man den Pfeil bei v_{ende}, so deutet der Pfeil direkt auf einen Voxel, der Punkte und damit ein Objekt enthält. Dadurch ist klar, dass hier ein Objekt ist, das von dieser Seite aus noch nicht erfasst wurde.

3.2.6 Gruppierung von Nutzerhinweisen

Durch das Ignorieren von Fehlstrahlen mit einer niedrigen Bewertung kann man die Zahl der Nutzerhinweise einschränken. Nur die wichtigsten Stellen werden dadurch markiert. An flächigen Bereichen sind dennoch oftmals mehrere Nutzerhinweise direkt nebeneinander platziert. Diese können optional noch zu einem einzigen Nutzerhinweis zusammengefasst werden. Die Gruppierung ist dabei nur für Pfeile sinnvoll, die am Anfang oder am Ende des Fehlstrahls platziert sind. Alle Pfeile gleicher Richtung, die nebeneinander liegen, werden zusammengefasst. Es ist anzumerken, dass bei der Gruppierung von Pfeilen, die am Anfang eines Fehlstrahls platziert sind, ein anderes Ergebnis entsteht, als wenn Pfeile zusammengefasst werden, die am Ende eines Fehlstrahls platziert sind.

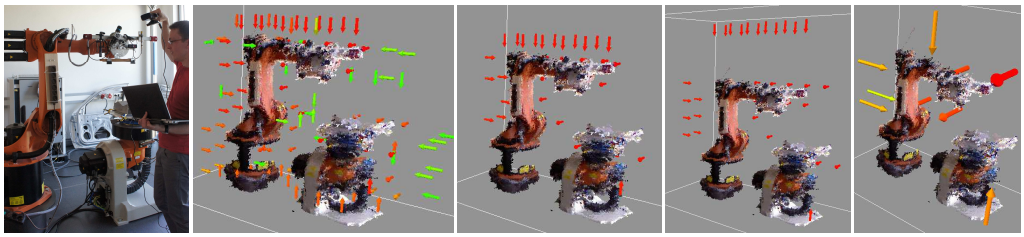
Um die Nutzerhinweise zu gruppieren, kann man wie folgt vorgehen: Alle Pfeile in Richtung d, die in Richtung d die gleiche Koordinate besitzen, liegen in einer Ebene und können potenziell zusammengefasst werden: Fasst man diese Ebene als Binärbild auf, in der entweder ein Pfeil vorhanden ist (Wert 1) oder nicht (Wert 0), so entspricht das Gruppierungsproblem einer Bildsegmentierung. Eine einfache Lösung liefert hier dann das *connected-components-labeling*, ein Standardverfahren aus der Bildverarbeitung (Stockman & Shapiro 2001).

4 Experimente

Die experimentelle Evaluation geschah mit folgendem Aufbau: Als Sensor wurde eine Microsoft Kinect XBox 360 verwendet, welche die erfassten Daten in Form einer kolorierten Punktwolke liefert. Die Bildrate wurde dabei auf 1 Hz festgelegt. Die Punktwolken wurden dann mittels Verfahren zur Berechnung der visuellen Odometrie auf Basis der Open-Source Software RTAB-Map (Labbè & Michaud 2013) registriert und fusioniert. Auf einem tragbaren Computer wird dem Anwender die Rekonstruktion als globale 3D-Punktwolke angezeigt. Die 3D-Ansicht kann dabei optional so eingestellt werden, dass die Szene stets aus der Perspektive der aktuellen Kameraposition angezeigt wird. Der Nutzer hat zudem die Möglichkeit, die Aufnahme zu starten, zu unterbrechen oder zu stoppen. In diese 3D-Ansicht werden während der Aufnahme die Nutzerhinweise als Pfeile eingeblendet, die mit jedem neuen Tiefenbild aktualisiert werden. Um die Menge der Pfeile zu reduzieren, werden nur Hinweise mit einer Bewertung von mindestens 0,9 angezeigt.

In einem ersten Versuch mit fünf Probanden wurde ermittelt, welche Art der Visualisierung (Position, Gruppierung und Anzahl der Nutzerhinweise) während der Aufnahme am intuitivsten für den Nutzer ist. Die Testpersonen rekonstruierten dabei mehrmals eine Roboterzelle und bewerteten anschließend die Rückmeldungen. In Abbildung 3 sind die Live-Rekonstruktionen mit unterschiedlichen Varianten der Visualisierung dargestellt. Durch Befragung ergab sich, dass die Probanden eine Positionierung am Fehlstrahlende klar bevorzugen: Es ergab sich eine Bewertung von +3,2 auf eines Skala von +4 (Strahlende) bis -4 (Strahlanfang). Ebenso ist eine Tendenz zu gruppierten Nutzerhinweisen zu erkennen (Bewertung von +1,8). Bei der Anzahl der Pfeile waren die meisten Probanden unentschieden, ob eine feste Anzahl oder eine feste Bewertungsschwelle verwendet wird (Bewertung von

-1.0). Da bei beiden Varianten der Unterschied in der Visualisierung nicht allzu groß ist, ist dieses Ergebnis nicht überraschend.



a) *b)* *c)* *d)* *e)*

Abbildung 3: Rekonstruktion eines Industrieroboters. Der Roboter wurde dabei noch nicht von hinten und von oben erfasst. Die generierten Nutzerhinweise (Pfeile) weisen den Anwender auf die fehlenden Bereiche hin. a) Aufnahmevorgang mit handgehaltenem Sensor. b)Ungruppierte Hinweise am Fehlstrahlende (rot = hohe Bewertung). c) Ungruppierte Hinweise am Fehlstrahlende mit Bewertung höher als 0,9. d) Ungruppierte Hinweisen am Fehlstrahlanfang mit Bewertung höher als 0,9. e) Gruppierte Hinweise am Fehlstrahlende mit Bewertung höher als 0,8.

In einem weiteren Versuch wurde das unterschiedliche Aufnahmeverhalten mit und ohne Rückmeldungen untersucht. Dazu sollten die Probanden eine Szene einmal mit und einmal ohne Hinweise aufnehmen, anschließend wurden die Rekonstruktionen verglichen. Bei der Aufnahme ohne Hilfen fällt auf, dass alle Nutzer zwar Verdeckungen berücksichtigen und auch bewusst Zwischenräume aufnehmen. Vier von fünf Probanden vergaßen jedoch, Aufnahmen von einer sehr niedrigen Position nahe dem Boden zu erstellen, um Flächen zu erfassen, die nur von dort aus sichtbar sind. Unter Zuhilfenahme der Nutzerhinweise wurden diese Bereiche nicht vergessen, da das System die Anwender darauf hingewiesen hatte. Allen Probanden fiel es leicht, die vorgeschlagenen Posen auch einzunehmen, da diese direkt in die Rekonstruktion eingeblendet sind und so eine Orientierung im Raum erleichtern.

5 Fazit

Diese Arbeit präsentiert eine Methode, mit der sich während einer Rekonstruktion mit einem handgehaltenen Sensor intuitive Nutzerhinweise generieren lassen. Im Vergleich zu bestehenden Ansätzen werden dabei unstrukturierte Umgebungen mit Verdeckungen unterstützt, sowie noch nicht erfasste Bereiche erkannt und eine Aufnahmeposition vorgeschlagen. Diese Hinweise können direkt in eine 3D-Live-Rekonstruktion eingeblendet werden und vereinfachen so nachweislich den Aufnahmeprozess. Weitere Schritte in Richtung einer intuitiven Schnittstelle könnten zukünftig die Verwendung einer Datenbrille (Einblendung der Nutzerhinweise in das reale Bild) oder von Smartphones mit integrierten Stereo- oder Tiefenkameras sein.

Literaturverzeichnis

Amanatides, J., & Woo, A. (1987). A Fast Voxel Traversal Algorithm for Ray Tracing. *Eurographics '87*, S. 3–10.

Banta, J.E., Wong, L.R., Dumont, C. & Abidi, M.A. (2000). A next-best-view system for autonomous 3-D object reconstruction. Systems. *IEEE Transactions on Man and Cybernetics, Part A: Systems and Humans*, S. 589–598.

Dellepiane, M., Cavarretta, E., Cignoni, P. & Scopigno, R. (2013). Assisted Multi-view Stereo Reconstruction. *International Conference on 3DV-Conference 2013*, S. 318–325.

Du, H., Henry, P., Ren, X., Cheng, M., Goldman, D.B., Seitz, S.M. & Fox, D. (2011). Interactive 3D modeling of indoor environments with a consumer depth camera. *Proceedings of the 13th International Conference on Ubiquitous Computing*, S. 75–84.

Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D. & Burgard, W. (2012). An Evaluation of the RGB-D SLAM System. *IEEE International Conference on Robotics and Automation (ICRA)*, S. 1691-1696.

Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A. & Fitzgibbon, A. (2011). KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, S. 559–568.

Krainin, M., Curless, B. & Fox, D. (2011). Autonomous generation of complete 3D object models using next best view manipulation planning, *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. S. 5031–5037.

Labbé, M. & Michaud., F. (2013). Appearance-based loop closure detection in real-time for large-scale and long-term operation. *IEEE Transactions on Robotics*, 29(3), S. 734-745.

Pito, R. (1999). A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10), S. 1016–1030.

Potthast, C. & Sukhatme, G.S. (2013). A Probabilistic Framework for Next Best View Estimation in a Cluttered Environment. *Journal of Visual Communication and Image Representation.*

Schneider, D.C. & Eisert, P. (2012). On User-Interaction in 3D Reconstruction, *Eurographics 2012-Posters*, S. 13–14.

Scott, W.R., Roth, G., & Rivest, J.-F. (2003). View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, 35(1), S. 64–96.

Sedlacek, D. & Zara, J. (2012). User Driven 3D Reconstruction Environment. In Bebis, G., Boyle, R., Parvin, B., Koracin, D., Fowlkes, C., Wang, S., Choi, M.-H., Mantler, S., Schulze, J., Acevedo, D., Mueller, K., Papka, M. (Hrsg.): *Advances in Visual Computing*. Berlin, Heidelberg: Springer. S. 104–114.

Stockman, G. & Shapiro, L.G. (2001). *Computer Vision*. Upper Saddle River, NJ, USA: Prentice Hall.

Stückler, J. & Behnke, S. (2014). Multi-Resolution Surfel Maps for Efficient Dense 3D Modeling and Tracking. *Journal of Visual Communication and Image Representation*, 25(1), S. 137-147.

Whelan, T., Kaess, M., Leonard, J.J. & McDonald, J.B. (2013). Deformation-based Loop Closure for Large Scale Dense RGB-D SLAM, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, S. 548-555.

Kontaktinformationen

maximilian.sand@uni-bayreuth.de