Hinderniserkennung mit Microsoft Kinect

NICLAS ZELLER[1], FRANZ QUINT[2] & LING GUAN[3]

Dieser Artikel beschreibt ein System zur 3D-Rekonstruktion der Umgebung und zur Hinderniserkennung für blinde Personen. Basierend auf einer Microsoft Kinect wurden Algorithmen zur 3D-Rekonstruktion entwickelt. Diese Algorithmen realisieren eine kombinierte Gradienten- und RANSAC-basierte Ebenensegmentierung. Die resultierenden Ebenensegmente werden anhand ihrer Schnittkanten zu 3D-Objekten zusammengefasst. Hierbei wird eine generalisierende und zuverlässige Rekonstruktion der Umgebung angestrebt, welche z.B. über modulierte Audiosignale an die blinde Person weitergegeben werden kann. In dem hier vorgestellten System werden Objekte daher ausschließlich durch Quader modelliert. Beispielszenen demonstrieren die Vor- und Nachteile des Systems.

1 Einleitung

Blinde Personen sind zur sicheren Navigation auf unterstützende Hilfsmittel angewiesen. Heutzutage sind der weiße Langstock (Blindenstock) und der Blindenführhund die beiden am weitesten verbreiteten navigationsunterstützenden Hilfen für Sehbehinderte. Trotz ihrer sehr großen Popularität zeigen beide Hilfsmittel auch Nachteile. Eine blinde Person ist mit beiden Hilfen z.B. nicht in der Lage, herabhängende oder seitlich hereinragende Hindernisse auf Kopfhöhe zuverlässig zu erkennen. Hinzu kommt, dass vor allem der Langstock eine sehr eingeschränkte Reichweite aufweist und Blindenführhunde oft nicht in Innenräumen eingesetzt werden können.

Eine Alternative zu diesen konventionellen Hilfen bieten elektronische Navigationsassistenten (Electronic Travel Aids (ETAs)). ETAs sind elektronische Navigationshilfen für Blinde, welche keine zusätzliche Infrastruktur zur Orientierung verwenden. Daher können sie in beliebiger Umgebung eingesetzt werden. Das Hauptziel eines ETAs ist es, Sehbehinderte vor nahenden Hindernissen zu warnen und darüber hinaus einen Eindruck der Umgebung zu vermitteln. Trotz der Vielzahl bereits existierender ETAs ist die Akzeptanz dieser Geräte relativ gering. DARKOPOULOS & BOURBAKIS (2010), sowie MANDUCHI & COUGHLAN (2012) geben einen Überblick über bereits existierende ETAs. Viele Systeme warnen den Benutzer lediglich vor direkt bevorstehenden Hindernissen und bieten somit keine Verbesserung gegenüber einem Blindenstock. Andere kamerabasierte Systeme, wie beispielsweise VOICE (1996) oder das System von GONZALEZ-MORA ET AL. (2006), versuchen zwar eine visuelle Wahrnehmung der Umgebung zu vermitteln, neigen aber in der Regel dazu, die sehr gut ausgeprägten Sinne einer blinden Person mit viel redundanter Information zu überfluten. Das Ziel des hier vorgestellten Systems ist es daher, die aufgenommene Umgebung auf wenige signifikante Informationen zu reduzieren und diese an den Benutzer zu übermitteln.

1) Niclas Zeller, Fakultät für Elektro- und Informationstechnik, Hochschule Karlsruhe Technik und Wirtschaft, Moltkestraße 30, 76133 Karlsruhe; E-Mail: niclas.zeller@hs-karlsruhe.de
2) Franz Quint, Fakultät für Elektro- und Informationstechnik, Hochschule Karlsruhe Technik und Wirtschaft, Moltkestraße 30, 76133 Karlsruhe; E-Mail: franz.quint@hs-karlsruhe.de
3) Ling Guan, Ryerson University, Department of Electrical and Computer Engineering, Ryerson University, 350 Victoria Street, Toronto, Ontario, Canada, M5B 2K3; E-Mail: lguan@ee.ryerson.ca

Zur Umgebungserfassung wurde im hier vorgestellten System der Sensor Microsoft Kinect verwendet. Ein großer Vorteil der Kinect gegenüber anderen tiefenerfassenden Kamerasystemen ist der geringe Preis sowie ein sehr umfangreiches Software Development Kit (SDK). Kinect bietet daher die Möglichkeit, in sehr kurzer Zeit einen preisgünstigen Prototypen zu entwickeln.

Basierend auf der von der Kinect gelieferten Tiefeninformation wurden Algorithmen entwickelt, welche Objekte in einer Szene erkennen und diese als Quader im dreidimensionalen Raum darstellen. Durch diese einfache Darstellung der Objekte ist es möglich, die Szeneninformation, z.B. mittels modulierten Audiosignalen, an den Benutzer zu übermitteln. Prototypisch wurden die Algorithmen für Kinect-Aufnahmen entwickelt, doch für den Einsatz in einem ETA sollen sie auf Tiefenbilder einer kleinen und handlichen plenoptischen Kamera angewandt werden.

Die vorgestellten Algorithmen führen zunächst eine Ebenensegmentierung durch. Diese ist eine Kombination aus einer gradientenbasierten Segmentierung des 2D Tiefenbilds und einer RANSAC-Ebenensegmentierung (FISCHLER & BOLLES (1981)). Hierbei wird der RANSAC-Algorithmus auf eine aus dem Tiefenbild berechnete 3D-Punktwolke angewendet. Im folgenden Abschnitt 2 wird diese Segmentierung näher beschrieben. Anschließend werden, wie in Abschnitt 3 dargestellt, benachbarte Ebenensegmente anhand ihrer Schnittkanten zu Objekten zusammengefasst und als Quader modelliert. In Abschnitt 4 werden zwei verschiedene Anwendungsbeispiele vorgestellt. Anschließend folgt in Abschnitt 5 ein kurzes Fazit zum entwickelten System sowie ein Ausblick für Erweiterungen und Verbesserungen.

2 Tiefenbildbasierte Ebenensegmentierung

Ein Pixel des Tiefenbilds ist durch seine Bildkoordinaten x_I und y_I definiert. Im Folgenden wird zur Benennung der Koordinaten eines Pixels ebenso die Vektorschreibweise $X_I = (x_I, y_I)^T$ verwendet. Der Tiefensensor der Kinect liefert für jedes Pixel einen Tiefenwert. Dieses Tiefenbild wird im Folgenden als zweidimensionale Funktion $d(X_I) = d(x_I, y_I)$ beschrieben.

Zusätzlich zu den Bildkoordinaten X_I wird ein 3D-Weltkoordinatensystem definiert, welches auf die Position und Orientierung der Kinect bezogen ist. Im Weltkoordinatensystem wird ein Punkt durch die Koordinaten x_W, y_W und z_W beschrieben. Im Folgenden wir äquivalent die Vektorschreibweise $X_W = (x_W, y_W, z_W)^T$ verwendet. Der Zusammenhang zwischen X_W und X_I wird durch die Transformationsmatrix A beschrieben, welche im Abschnitt 2.2 definiert ist.

2.1 Gradientenbasierte Segmentierung des Tiefenbilds

Durch die inhärenten Mittelungen des RANSAC-Algorithmus neigt dieser dazu, kleine Stufen in den Tiefenbildern zu übergehen und ähnliche Ebenen zu einer Ebene zusammenzufassen. Deshalb wird zunächst eine gradientenbasierte Segmentierung auf das 2D-Teifenbild angewendet.

Zur Segmentierung wird der Gradientenvektor $\vec{g}(X_I)$ über das Tiefenbild $d(X_I)$ berechnet. Hierfür kann prinzipiell ein beliebiges Gradientenfilter (z.B. Sobel-Operator oder Ableitung der Gauß-Funktion wie beim Canny-Operator) verwendet werden. Zur Unterdrückung von Interpolations- und Quantisierungsartefakten wird der resultierende Gradientenvektor anschließend Median-gefiltert. Anhand des Gradientenvektors $\vec{g}(X_I)$, sowie der Tiefeninformation $d(X_I)$, werden benachbarte Pixel im Tiefenbild zu Segmenten zusammengefasst. Zwei benachbarte Pixel X_I^i und X_I^j werden demselben Segment zugeordnet,

wenn gemäß Gl. (1) und (2) die Änderung des Gradienten sowie die Abweichung der gemessenen Tiefeninformation, von einem prädizierten Wert, die experimentell bestimmten Schwellen T_g und T_d nicht überschreiten:

$$\Delta g < T_g \quad \text{mit} \quad \Delta g = \left\| \vec{g}\left(X_I^i\right) - \vec{g}\left(X_I^j\right) \right\| \tag{1}$$

$$\Delta \mathrm{d} < T_d \quad \text{mit} \quad \Delta d = \left| d\left(X_I^i\right) + \vec{g}\left(X_I^i\right)^T \cdot \begin{pmatrix} x_I^j - x_I^i \\ y_I^j - y_I^i \end{pmatrix} - d\left(X_I^j\right) \right| \tag{2}$$

Als vorläufiges Segmentierungsergebnis erhält man ein Markenbild, in dem alle zu einem Segment gehörenden Pixel dieselbe Marke, nämlich die Nummer des Segments, tragen.

2.2 Transformation vom Tiefenbild zur 3D Punktwolke

Um die genaue Größe und Position von Objekten im Raum bestimmen zu können, wird das projektiv abgebildete Tiefenbild $d(X_I)$ in Weltkoordinaten X_W, in Form einer 3D Punktwolke, umgerechnet. Anschließend wird der RANSAC-Algorithmus auf die Punktwolke angewendet und diese dadurch in Ebenensegmente aufgeteilt.

Die Transformation von Weltkoordinaten X_W in Bildkoordinaten X_I wird durch die Matrix A gemäß Gl. (3) in homogenen Koordinaten beschrieben. Diese Matrix umfasst sowohl die affine Transformation von Welt- in Kamerakoordinaten, als auch die Zentralprojektion von Kamera- in Bildkoordinaten. Aufgrund der Konstruktion des Kinect-Sensors kann angenommen werden, dass der Einheitsvektor der Tiefeninformation $\vec{e}_d$ orthogonal zur Bildebene steht und somit der z-Komponente des Kamerakoordinatensystems entspricht. Dadurch kann die vom Abstandssensor der Kinect gelieferte Tiefeninformation d unmittelbar in Gl. (3) eingesetzt werden.

$$\begin{pmatrix} k \cdot x_I \\ k \cdot y_I \\ d \\ k \end{pmatrix} = A \cdot \begin{pmatrix} x_W \\ y_W \\ z_W \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & 1 \end{pmatrix} \cdot \begin{pmatrix} x_W \\ y_W \\ z_W \\ 1 \end{pmatrix} \tag{3}$$

Durch einsetzen der vierten Zeile des Gleichungssystems (3) in die erste und zweite Zeile und anschließendes Umformen können die Gleichungen (4) und (5) aufgestellt werden. Unabhängig von Gl. (4) und (5) erhält man aus der dritten Zeile von Gl. (3) Gl. (6). Da Gl. (4) bis (6) linear in den Koeffizienten von A sind und sowohl Bild- als auch Weltkoordinaten durch die Kalibrieraufnahmen bekannt sind, können alle 15 Koeffizienten der Matrix A durch lineare Regression geschätzt werden. Da sich für jeden Referenzpunkt drei Gleichungen ergeben, werden mindestens 5 Punkte für die Lösung benötigt, zur robusten Schätzung jedoch deutlich mehr verwendet. In der Realisierung wurden ca. 50 Punkte verwendet.

$$x_I = a_{11} \cdot x_W + a_{12} \cdot y_W + a_{13} \cdot z_W + a_{14} - a_{41} \cdot x_W x_I - a_{42} \cdot y_W x_I - a_{43} \cdot z_W x_I \tag{4}$$

$$y_I = a_{21} \cdot x_W + a_{22} \cdot y_W + a_{23} \cdot z_W + a_{24} - a_{41} \cdot x_W y_I - a_{42} \cdot y_W y_I - a_{43} \cdot z_W y_I \tag{5}$$

$$d = a_{31} \cdot x_W + a_{32} \cdot y_W + a_{33} \cdot z_W + a_{34} \tag{6}$$

Nach Bestimmung der Transformationsmatrix A erhält man die zu einem Bildpunkt X_I gehörenden Weltkoordinaten X_W indem man Gl. (3) von links mit A^{-1} multipliziert.

Es scheint zunächst sinnvoller, die Koeffizienten von A^{-1} direkt zu schätzen, da so die geschätzte Matrix nicht invertiert werden müsste. Allerdings ist A^{-1} nicht in ein Gleichungssystem überführbar, welches linear in den Koeffizienten von A^{-1} ist.

Äquivalent zu der Zuordnung der Bildpunkte X_I werden die 3D-Punkte X_W den Segmenten zugeordnet.

2.3 RANSAC-Ebenensegmentierung

Zur endgültigen Segmentierung wird auf die 3D-Punkte X_W der RANSAC-Algorithmus, wie von FISCHLER & BOLLES (1981) beschrieben, angewendet. Als Modell wird eine Ebene im 3D-Raum verwendet. Hierbei wird der Algorithmus auf die Punkte jedes Segments einzeln angewendet. Dadurch resultiert eine feinere Segmentierung, da jedes Segment aus der gradientenbasierten Segmentierung in mehrere neue Segmente unterteilt wird. Hierbei definiert der RANSAC-Algorithmus für jedes neue Segment eine Ebene.

3 Modellierung von Objekten

Die meisten Szenarien in denen sich eine blinde Person bewegt, sind von künstlichen Objekten dominiert, welche i.d.R. gut durch Quader modelliert werden können. Daher ist es in der Regel ausreichend, quaderförmige Objekte zu modellieren. Die Erweiterung auf andere Körper (z.B. Zylinder und Kugelausschnitte) kann durch Anpassung des RANSAC-Algorithmus zur Segmentierung quadratischer Formen ohne Konzeptänderung durchgeführt werden.

Die Objektmodellierung unterteilt sich in mehrere Schritte. Zunächst werden Schnittkanten zwischen benachbarten Ebenen berechnet. Daraufhin wird die Bodenebene detektiert und in einem dritten Schritt werden die Ebenen zu Objekten zusammengefasst.

3.1 Berechnen der Schnittkanten benachbarter Ebenen

Da eine Ebene laut Definition nicht begrenzt ist, kann theoretisch zwischen zwei beliebigen Ebenen, sofern diese nicht parallel sind, eine Schnittkante berechnet werden. Die Herausforderung besteht also darin, nur real existierende Schnittkanten zu finden. Zur Lösung dieses Problems wird zunächst, basierend auf den 2D-Bildkoordinaten X_I, ein Nachbarschaftsgraph erstellt. In diesem Graphen ist jedes Segment durch einen Knoten repräsentiert und benachbarte Segmente sind durch eine gerichtete Kante miteinander verbunden. Hierbei ist die Richtung der Kante zunächst frei wählbar. Im Folgenden werden die beiden benachbarten Segmente durch die Indizes i und j bezeichnet und sind durch die Kante e_{ij} verbunden. Die Reihenfolge der Indices gibt die Richtung der Kante (von i nach j) an. Aus allen 3D-Punkten, welche zu den aneinander angrenzenden Pixeln der benachbarten Segmente gehören wird zunächst eine Ausgleichsgerade $\hat{L}_c^{ij}$ berechnet. Zusätzlich wird die tatsächliche Schnittkante L_c^{ij} zwischen den beiden Ebenen Π_i und Π_j bestimmt. Anhand der maximalen Abweichung zwischen den beiden Geraden $\hat{L}_c^{ij}$ und L_c^{ij} wird entschieden, ob die Schnittkante real existiert oder nicht.

Jede Schnittkante L_c^{ij} wird als gerichtete Gerade definiert. Hierbei wird die Richtung der Gerade durch ihren Richtungsvektor $\vec{d}_c^{ij}$ angegeben. L_c^{ij} ist so definiert, dass sich das Segment i links und das Segment j rechts von der Projektion von L_c^{ij} auf die Bildebene befindet.

3.2 Rekonstruktion von Objekten

Bevor benachbarte Ebenen zu Objekten zusammengefasst werden, wird die Bodenebene detektiert, da diese kein Hindernis darstellt. Die Detektion geschieht anhand der Orientierung der Ebene im Weltkoordinatensystem, sowie deren Abstand zum Projektionszentrum. Beide Merkmale werden für die Bodenebene als nahezu konstant angenommen.

Anschließend wird jede Schnittkante L_c^{ij} als konvex oder konkav klassifiziert. Hierfür werden die Normalenvektoren der Ebenensegmente $\vec{n}$ so ausgerichtet, dass sie in Richtung der Abbildungsebene zeigen und nicht entgegengesetzt. Ein Normalenvektor welcher in die entgegengesetzte Richtung zeigt, wird mit -1 multipliziert (um 180° gedreht). Dies ändert nichts an der Ebene selbst, lediglich das Vorzeichen der Koeffizienten in der Definitionsgleichung. Eine Schnittkante L_c^{ij} kann dann durch die in Gl. (7) gegebene Vorschrift klassifiziert werden.

$$L_c^{ij} \widehat{=} \begin{cases} \text{konvex} & \text{wenn} \quad \frac{\vec{n}_i}{\|\vec{n}_i\|} \times \frac{\vec{n}_j}{\|\vec{n}_j\|} = \frac{\vec{d}_{ij}}{\|\vec{d}_{ij}\|}, \\ \text{konkav} & \text{sonst.} \end{cases} \tag{7}$$

Alle Ebenen, welche durch eine konvexe Schnittkante miteinander verbunden sind, werden zu einem Objekt zusammengefasst. Für jedes Objekt wird eine Quader als Einhüllende definiert.

4 Anwendung

Dieser Abschnitt zeigt anhand von zwei Beispielszenen die Ergebnisse der Algorithmen. Abb. 1 (links) zeigt das RGB-Bild der ersten Szene. In dieser Szene befinden sich viele unterschiedliche Objekte. Der Tisch im Hintergrund sowie der Papierkorb sind Objekte, welche unbedingt erkannt werden müssen. Außerdem stellt das Buch im Vordergrund eine Stolperfalle dar, welche für eine sichere Navigation ebenfalls erkannt werden muss. Die Objekte auf und unter dem Tisch sind eher von geringem Interesse. Abb. 1 (rechts) zeigt die bereits fertige automatische 3D-Rekonstuktion der Szene. Prinzipiell wurden alle wichtigen Objekte erkannt. Auch das Buch, welches lediglich eine Höhe von ca. 2 cm besitzt wurde als eigenständiges Objekt modelliert. Für die einzelnen Objekte auf dem Tisch gestaltet sich die Rekonstruktion schwierig, da diese sehr verwinkelt sind und sich teilweise gegenseitig im Tiefenbild abschatten. Außerdem ist hier das angenommene Modell von ebenen Flächen nicht immer erfüllt. Ebenso wurde die linke Seite des Tisches in viele kleine Segmente zerlegt. Dies liegt vor allem an dem für große Abstände stark quantisierten Tiefenbild der Kinect. Hierbei resultieren Ebenen, welche in einem sehr steilen Winkel zur Bildebene stehen, im Tiefenbild nicht als homogene Ebenen mit einem konstanten Gradienten, sondern eher als stufige Gebilde, welche bei der Segmentierung in einzelne Ebenen zerlegt werden. In Abb. 2 ist eine weitere Aufnahme zu sehen, welche einen Treppenaufgang zeigt. In der 3D-Rekonstruktion sind die ersten fünf Stufen korrekt modelliert. Alle weiteren Stufen sind in der Rekonstruktion nicht vorhanden, da diese außerhalb der Reichweite des Kinect Tiefensensors von ca. 5 m liegen. Trotzdem enthält die Rekonstruktion alle zur Hinderniswarnung erforderlichen Informationen.

5 Fazit und Ausblick

Die in Abschnitt 4 vorgestellten Ergebnisse zeigen, dass das in diesem Artikel vorgestellte System in der Lange ist, 3D-Rekonstruktion basierend auf primitiven geometrischen Körpern zu

erstellen. Einschränkungen ergeben sich aus den Einschränkungen des verwendeten Tiefensensors hinsichtlich Reichweite und Quantisierung. Wenn Objekte in ihrer Form zu stark vom angenommenen quaderförmigen Modell abweichen, leidet die Rekonstruktionsgenauigkeit ebenfalls, allerdings ist das für die Anwendung Hindernisdetektion nicht ausschlaggebend. Die laufenden Entwicklungen zielen darauf ab, auch Bereiche unterhalb der Bodenebene zu betrachten, um so auch abfallende Stufen und Abgründe zu erkennen. Da einerseits die Tiefeninformation der Kinect teilweise unzureichend ist und sie andererseits wegen ihrer Größe nur für prototypische Anwendungen ausreicht, werden die Algorithmen gegenwärtig auf Tiefenbilder plenoptischer Kameras angewandt.

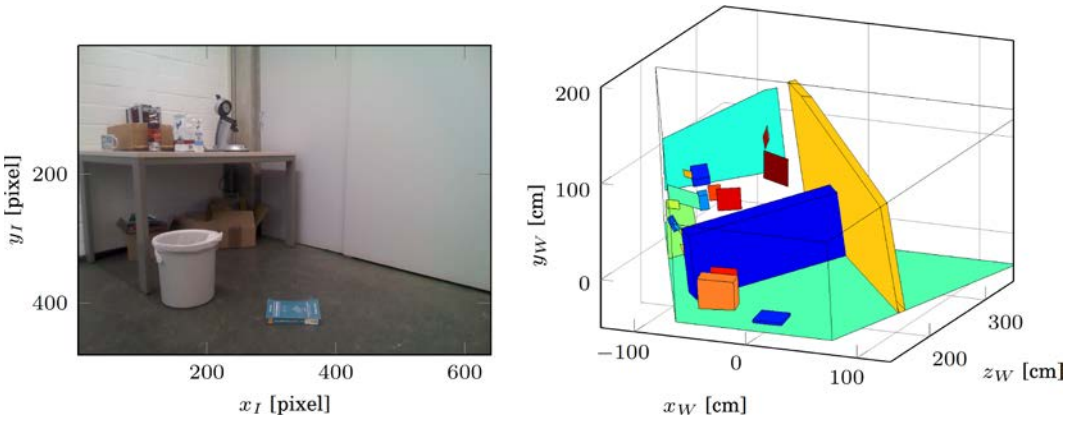

Abb. 1: Links: RGB-Bild einer Testszene, Szene mit verschiedenen zu detektierenden Objekten. Rechts: resultierende 3D-Rekonstruktion.

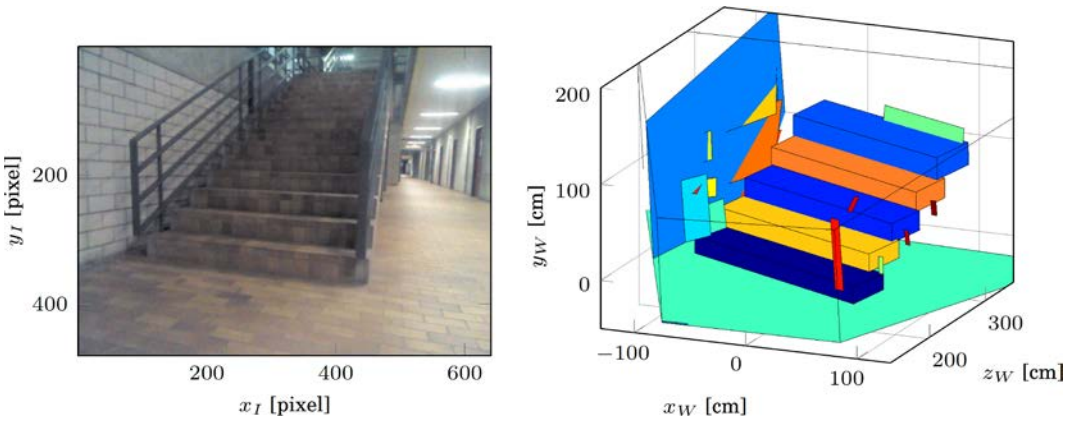

Abb. 2: Links: RGB-Bild einer Testszene, Treppenaufgang. Rechts: resultierende 3D-Rekonstruktion.

6 Literaturverzeichnis

DARKOPOULOS, D. & BOURBAKIS, N. G., 2010: Wearable obstacle avoidance electronic travel aids for blind: A survey. IEEE Transaction on Systems, Man, and Cybernetics, Part C: Applications and Reviews, **40** (1), S. 25-35.

FISCHLER, M. A. & BOLLES R. C., 1981: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, **24** (6), S. 381-395.

GONZALEZ-MORA, J. L.; RODRIGUEZ-HERNANDEZ, A. F.; BURUNAT, E.; MARTIN, F. & CASTELLANO, M. A., 2006: Seeing the world by hearing: Virtual Acoustic Space (VAS) a

new space perception system for blind people. ICTTA'06 2nd, Information and Communication Technologies, **1**, S. 837-842.

MANDUCHI, R. & COUGHLAN, J., 2012: (Computer) vision without sight. Communications of the ACM, **55** (1), S. 96-104.

VOICE, 1996: Augmented Reality for Totally Blind. Zuletzt abgerufen im Oktober 2013, www.seeingwithsound.com.