ORIENTIERUNG VON LASERSCANNER-PUNKTWOLKEN

Michael Hofer und Helmut Pottmann
Institut für Geometrie, Technische Universität Wien, Wiedner Hauptstr. 8–10, A–1040 Wien
hofer@geometrie.tuwien.ac.at, pottmann@geometrie.tuwien.ac.at

ZUSAMMENFASSUNG

Die Orientierung von Laserscanner-Punktwolken ist einer der ersten Schritte bei der 3D-Digitalisierung realer Objekte. Unter Orientierung versteht man die Aufgabe, zwei oder mehrere Teilaufnahmen ein und desselben Objektes aus unterschiedlichen Positionen so zusammenzuführen, dass das gesamte aufgenommene Objekt dargestellt wird. Anstatt "Orientierung" werden in Fachgebieten außerhalb der Photogrammetrie und Fernerkundung auch oft die Begriffe "Registrierung" bzw. "Matching" verwendet. Dieser Artikel ist die schriftliche Ausarbeitung eines Vortrags, welcher vom ersten Autor im Rahmen des Universitätslehrgangs "Laserscanning" des Instituts für Photogrammetrie und Fernerkundung der Technischen Universität Wien am 24. September 2003 gehalten wurde. Es werden die grundlegenden Orientierungsverfahren von Laserscanner-Punktwolken besprochen.

ABSTRACT

The orientation of laser scanner point clouds is one of the fundamental steps in the process of creating three-dimensional digital representations of existing objects. Orientation is the task of finding the correct positions of two or more point clouds (that only partially represent the object) obtained from different viewing positions, such that the full object in consideration is correctly described. In communities outside of Photogrammetry and Remote Sensing, instead of 'orientation' the terms 'registration' or 'matching' are used to describe the same task. This article is an exposition of a talk given by the first author at the continuing education 'Laserscanning' of the Institute of Photogrammetry and Remote Sensing of the Vienna University of Technology on September 24th, 2003. The article gives an overview of fundamental orientation methods for laser scanner point clouds.

1 EINLEITUNG

Die vollständige Digitalisierung eines dreidimensionalen Objektes erfordert im Allgemeinen mehrere Aufnahmen von verschiedenen Seiten. Um dies zu bewerkstelligen muss entweder das Aufnahmegerät oder das Objekt selbst in verschiedene räumliche Lagen gebracht werden. Ist das Objekt klein genug, verwendet man zur Datenaufnahme üblicherweise einen Drehteller, welcher automatisch gesteuert wird. Aber auch in diesem Fall muss zumindest die Standfläche des Objektes extra gescannt werden, um ein vollständiges digitales Modell zu erhalten.

In den meisten praktischen Fällen stößt man also auf die Fragestellung, einzelne Aufnahmen ein und desselben Objektes, welche in verschiedenen Koordinatensystemen liegen, so in ein einziges Koordinatensystem zu transformieren, dass man eine vollständige Beschreibung des Objektes erhält. Diese Aufgabenstellung wird in der Photogrammetrie als 'Orientierung' bezeichnet[1]. In anderen Fachgebieten hat sich im letzten Jahrzehnt dafür der Begriff 'Registrierung' eingebürgert[2].

Verwendet man zur Datenaufnahme einen 3D-Laserscanner, dann liegen die Aufnahmen als dreidimensionale (geordnete) Punktwolken vor. Abb. 1 zeigt ein Beispiel von vier Laserscanner-Punktwolken vor und nach der Orientierung. Das Objekt ist ein Bronzering-Schmuckstück aus der Hallstattzeit, welches in Salzburg gefunden wurde und mit einem Minolta VI-900 3D-Laserscanner der TU Wien (und teilweiser Verwendung eines Drehtellers) digitalisiert worden ist.

(**a**) (**b**)

Abbildung 1: (**a**) Ausgangslage von vier Punktwolken. (**b**) Orientierte und vereinigte Punktwolken mit der Textur des gescannten Objektes. Abbildungen mit freundlicher Genehmigung der Landesarchäologie Salzburg.

Da es noch keinen zufriedenstellenden vollautomatischen Algorithmus zur Orientierung der einzelnen Aufnahmen gibt, ist die Orientierung der einzelnen Laserscanner-Punktwolken momentan auch in professionellen Softwarepaketen ein semiautomatischer Vorgang: Der Benutzer bringt zuerst die einzelnen Punktwolken interaktiv (z.B. durch Auswählen dreier Paare korrespondierender Punkte, siehe Abb. 2) in eine "gute" gegenseitige Startlage, und erst dann werden automatische Algorithmen verwendet, welche die Orientierung optimieren. Die Grundlagen der wichtigsten dieser Algorithmen werden in diesem Artikel beschrieben:

- Orientierung zweier Punktwolken mit bekannten Korrespondenzen

[1] Genau genommen handelt es sich hier um die relative Orientierung der Punktwolken, da die absolute Orientierung, d.h. Position und Stellung des Objektes in Bezug zu einem globalen System nicht von Interesse ist.

[2] Als Übersetzung des in der Englischen Fachliteratur verwendeten Begriffs 'registration', was soviel bedeutet wie 'genaues Ausrichten'.

  - Explizite Lösung
  - Iterativer Algorithmus

- Orientierung von mehreren Punktwolken mit bekannten Korrespondenzen: Iterativer Algorithmus
- Orientierung ohne feste Korrespondenzen
  - ICP-Algorithmus I: Punkt-zu-Punkt Abstand
  - ICP-Algorithmus II: Punkt-zu-Ebene Abstand

Im Anhang fassen wir die im Artikel verwendeten Grundlagen aus der Kinematik, insbesondere Quaternionen, zusammen.

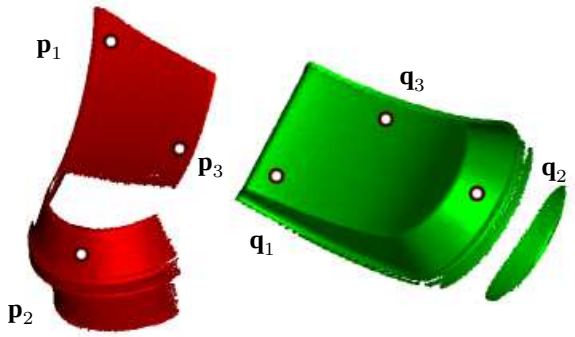


Abbildung 2: Manuelle Orientierung zweier Punktwolken durch interaktive Auswahl dreier Paare $(\mathbf{p}_1, \mathbf{q}_1)$, $(\mathbf{p}_2, \mathbf{q}_2)$, $(\mathbf{p}_3, \mathbf{q}_3)$ korrespondierender Punkte auf der festgehaltenen (links) und der zu bewegenden Punktwolke (rechts).

2 ORIENTIERUNG ZWEIER PUNKTWOLKEN MIT BEKANNTEN KORRESPONDENZEN

Die einfachste Aufgabenstellung im Bereich der Orientierung von Punktwolken ist die Orientierung von zwei Punktwolken mit bekannten Korrespondenzen. Gegeben sind zwei Punktwolken $X = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, $Y = (\mathbf{y}_1, \ldots, \mathbf{y}_N)$ und Punkte mit demselben Index seien zueinander korrespondierend. Das Ziel ist nun eine Euklidische Bewegung m (repräsentiert durch Verschiebungsvektor $\mathbf{r}$ und Drehmatrix R),

$$m(\mathbf{x}) := \mathbf{x}' = \mathbf{r} + R \cdot \mathbf{x},$$

so zu bestimmen, dass die Punktwolke X 'möglichst nahe' an die Punktwolke Y kommt. Formuliert man dies im Sinne der Methode der kleinsten Quadrate, so erhalten wir folgende Funktion, welche zu minimieren ist:

$$\sum_{i=1}^{N} \|\mathbf{x}'_i - \mathbf{y}_i\|^2 \to \min. \tag{1}$$

Wir fassen zuerst eine von [6] angegebene explizite Lösung zusammen und präsentieren dann einen alternativen Algorithmus von [11].

2.1 Explizite Lösung

Die explizite Lösung verwendet Quaternionen (siehe Anhang A) und kann in zwei Schritten beschrieben werden.

Lemma 1. *Die beste Euklidische Bewegung bringt den Schwerpunkt* $\mathbf{s}_x := \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$ *der Punktwolke* X *in den Schwerpunkt* $\mathbf{s}_y := \frac{1}{N}\sum_{i=1}^{N} \mathbf{y}_i$ *der Punktwolke* Y.

Beweis. Wir suchen ein Minimum der quadratischen Funktion

$$F(\mathbf{r}, R) = \sum_i (\mathbf{x}'_i - \mathbf{y}_i)^2 = \sum_i (\mathbf{r} + R \cdot \mathbf{x}_i - \mathbf{y}_i)^2. \tag{2}$$

Unter den notwendigen Bedingungen für ein Minimum von (2) sind die drei skalaren Gleichungen

$$\frac{\partial F}{\partial \mathbf{r}} = 2 \sum_i (\mathbf{r} + R \cdot \mathbf{x}_i - \mathbf{y}_i) = 0. \tag{3}$$

Falls die gesuchte Euklidische Bewegung $\mathbf{s}_x$ auf $\mathbf{s}_y$ abbildet, gilt $\mathbf{s}_y = \mathbf{r} + R \cdot \mathbf{s}_x$, d.h.,

$$\mathbf{r} = \mathbf{s}_y - R \cdot \mathbf{s}_x. \tag{4}$$

Setzen wir (4) in (3) ein, so erhalten wir

$$\begin{aligned} & \sum_i (\mathbf{s}_y - R \cdot \mathbf{s}_x + R \cdot \mathbf{x}_i - \mathbf{y}_i) \\ = \quad & \sum_i (\mathbf{s}_y - \mathbf{y}_i) + R \sum_i (\mathbf{x}_i - \mathbf{s}_x) = 0. \end{aligned}$$

Wir sehen also, dass die Gültigkeit von (3) bewirkt, dass der Schwerpunkt $\mathbf{s}_x$ von X in den Schwerpunkt $\mathbf{s}_y$ von Y abgebildet wird. □

Lemma 2. *Die optimale Drehung* R *ist gegeben durch den Eigenvektor* $\mathbf{q} \in \mathbb{R}^4$ *zum größten Eigenwert der Matrix* $M \in \mathbb{R}^{4\times4}$*, welche in Gleichung (9) definiert ist mit Bezeichnungen aus (7),(8) und Anhang A.1. Aus* $\mathbf{q} = (q_0, q_1, q_2, q_3)^T$ *kann die Drehmatrix* R *mit Satz 1 aus Anhang A.1 berechnet werden.*

Beweis. Auf die Punktwolke X wenden wir die Schiebung $\mathbf{s}_x \mapsto \mathbf{s}_y$ an, siehe Lemma 1. Dann wählen wir $\mathbf{s}_y$ als neuen Ursprung und bezeichnen die so verschobene Punktwolke wieder mit X. Nun berechnen wir eine Drehung um den Ursprung O, $\mathbf{x}' = R\mathbf{x}$ (mit einer orthogonalen Matrix R, d.h. $R^T R = RR^T = E$ mit der 3×3 Einheitsmatrix E), so dass

$$\begin{aligned} F &= \sum_i \|\mathbf{x}'_i - \mathbf{y}_i\|^2 = \sum_i (R\mathbf{x}_i - \mathbf{y}_i)^2 \\ &= \sum_i (\mathbf{x}_i^T R^T - \mathbf{y}_i^T)(R\mathbf{x}_i - \mathbf{y}_i) \\ &= \sum_i (\mathbf{x}_i^T R^T R\mathbf{x}_i - \mathbf{y}_i^T R\mathbf{x}_i - \mathbf{x}_i^T R^T \mathbf{y}_i + \mathbf{y}_i^T \mathbf{y}_i) \\ &= \sum_i (\mathbf{x}_i^T \mathbf{x}_i + \mathbf{y}_i^T \mathbf{y}_i - 2\mathbf{y}_i^T R\mathbf{x}_i) \to \min. \end{aligned} \tag{5}$$

Um (5) zu minimieren müssen wir R so bestimmen, dass

$$\sum_i \mathbf{y}_i^T R\mathbf{x}_i \to \max. \tag{6}$$

Wir schreiben die Drehung $\mathbf{x}_i \mapsto R\mathbf{x}_i$ mit einer Einheitsquaternion $\mathbf{q}$, und verwenden einige Eigenschaften des Rechnens mit Quaternionen (siehe Anhang A.1),

$$\begin{aligned}\mathbf{y}_i^T R\mathbf{x}_i &= \langle \mathbf{y}_i, R\mathbf{x}_i\rangle = \langle \mathbf{y}_i, \mathbf{q}\circ\mathbf{x}_i\circ\bar{\mathbf{q}}\rangle = \langle \mathbf{y}_i\circ\mathbf{q}, \mathbf{q}\circ\mathbf{x}_i\rangle \\ &= \langle Y_i\mathbf{q}, \tilde{X}_i\mathbf{q}\rangle = \mathbf{q}^T Y_i^T \tilde{X}_i\mathbf{q}. \end{aligned}\tag{7}$$

Daraus folgt, dass wir die folgende quadratische Funktion maximieren müssen

$$\sum_i \mathbf{y}_i^T R\mathbf{x}_i = \sum_i \mathbf{q}^T Y_i^T \tilde{X}_i\mathbf{q} = \mathbf{q}^T M\mathbf{q} \to \max. \tag{8}$$

Dabei ist

$$M := \sum_i Y_i^T \tilde{X}_i, \tag{9}$$

und die Nebenbedingung für die Maximierung von (8) lautet

$$\|\mathbf{q}\|^2 = \mathbf{q}^T\mathbf{q} = 1. \tag{10}$$

Dies führt bekanntlich auf die Lösung eines Eigenwertproblems. Der gemäß (10) normierte Eigenvektor $\mathbf{q} \in \mathbb{R}^4$ zum größten Eigenwert der Matrix M liefert die gewünschte Lösung. □

2.2 Iterativer Algorithmus

Wir diskutieren nun einen iterativen Algorithmus für die Orientierung von zwei Punktwolken mit bekannten Korrespondenzen. Dieser iterative Algorithmus hat im Vergleich zur expliziten Lösung im Abschnitt 2.1 den Nachteil, dass die gegenseitige Orientierung der beiden Punktwolken näherungsweise bekannt sein muss. Jedoch bietet sie im Gegensatz zur expliziten Lösung den Vorteil, dass sie auf beliebig viele simultan einzupassende Punktwolken erweitert werden kann; siehe Abschnitt 3.

Da die Punktwolke X also bereits näherungsweise zur festen Punktwolke Y orientiert ist, brauchen wir nur mehr eine kleine Verlagerung $\mathbf{v}(\mathbf{x})$ von X zu betrachten. Im Anhang A.3 wird gezeigt, dass eine solche - streng genommen differentielle - Verlagerung $\mathbf{x} \to \mathbf{x}+\mathbf{v}(\mathbf{x})$ mit Hilfe des Geschwindigkeitsvektorfelds $\mathbf{v}(\mathbf{x})$ beschrieben werden kann:

$$\mathbf{v}(\mathbf{x}) = \bar{\mathbf{c}} + \mathbf{c}\times\mathbf{x}. \tag{11}$$

Die Verlagerung $\mathbf{v}(\mathbf{x})$ der Punktwolke X soll so bestimmt werden, dass der Abstand zwischen $\mathbf{x}_i + \mathbf{v}(\mathbf{x}_i)$ und $\mathbf{y}_i$ minimiert wird:

$$F(\mathbf{c},\bar{\mathbf{c}}) = \sum_{i=1}^N \|\mathbf{x}_i+\mathbf{v}(\mathbf{x}_i)-\mathbf{y}_i\|^2 = \sum_{i=1}^N \|\mathbf{x}_i+\bar{\mathbf{c}}+\mathbf{c}\times\mathbf{x}_i-\mathbf{y}_i\|^2.$$

Die Minimierungsfunktion $F(\mathbf{c},\bar{\mathbf{c}})$ ist quadratisch in den Unbekannten $\mathbf{c},\bar{\mathbf{c}}$ ist. Es seien $\mathbf{d}_i := \mathbf{x}_i-\mathbf{y}_i$, $\mathbf{u} := (\mathbf{c}^T,\bar{\mathbf{c}}^T)^T$, und

$$B(\mathbf{x}) := \begin{pmatrix} 0 & x_3 & -x_2 & 1 & 0 & 0 \\ -x_3 & 0 & x_1 & 0 & 1 & 0 \\ x_2 & -x_1 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{12}$$

Dann gilt $\bar{\mathbf{c}}+\mathbf{c}\times\mathbf{x} = B\cdot\mathbf{u}$. Ein Minimum von F ist durch die Lösung des folgenden linearen Gleichungssystems gegeben:

$$\sum_{i=1}^N B_i^T B_i\cdot\mathbf{u} = \sum_{i=1}^N B_i^T\mathbf{d}_i, \tag{13}$$

mit $B_i := B(\mathbf{x}_i)$. Wendet man das so berechnete Geschwindigkeitsvektorfeld auf die Punktwolke an, so wird diese nicht wie gewünscht 'starr' verlagert, sondern affin verzerrt. Diese Affinität resultiert daher, dass der Ansatz (11) nur im Differentiellen eine starre, d.h. Euklidische, Transformation beschreibt. Deshalb verwenden wir nicht die berechnete affine, sondern eine Euklidische Transformation, welche die Punkte $\mathbf{x}_i$ in eine Position $\mathbf{x}_i'$ nahe an $\mathbf{x}_i+\mathbf{v}(\mathbf{x}_i)$ bringt, siehe Abb. 3. Die Wahl dieser abgeleiteten Euklidischen Transformation hängt dabei vom berechneten Vektor $\mathbf{u}$ ab:

- $\mathbf{c} = \mathbf{o}$: wende die *Schiebung* mit Schiebvektor $\bar{\mathbf{c}}$ an.
- $\mathbf{c} \neq \mathbf{o}$: berechne aus $(\mathbf{c},\bar{\mathbf{c}})$ mit (47) die Achse G (mit Richtungsvektor $\mathbf{g}$ und Momentenvektor $\bar{\mathbf{g}}$) und den Schraubparameter p, sowie berechne den Drehwinkel als $\varphi = \arctan\|\mathbf{c}\|$;
  - falls $\mathbf{c}\cdot\bar{\mathbf{c}} = 0$: wende die *Drehung* um G mit Drehwinkel φ an
  - falls $\mathbf{c}\cdot\bar{\mathbf{c}} \neq 0$: wende die *Schraubung* mit Achse G, Drehwinkel φ und Schiebanteil parallel zu G mit Länge $p\cdot\varphi$ an.

Im allgemeinen Fall läßt sich die Zusammensetzung der Drehung um die Achse G mit Drehwinkel φ, und der Schiebung parallel zu G mit der Schieblänge $p\cdot\varphi$ wie folgt berechnen:

$$\mathbf{x}' = R(\mathbf{x}-\mathbf{p}) + (p\cdot\varphi)\mathbf{g} + \mathbf{p}. \tag{14}$$

Dabei ist R eine Drehmatrix (welche mit Bemerkung 7 und Satz 1 berechnet wird), und $\mathbf{p}$ ist ein beliebiger Punkt auf G (z.B. $\mathbf{p} = \mathbf{g}\times\bar{\mathbf{g}}$).

Nun iterieren wir die Berechnung des Geschwindigkeitsvektorfeldes und der daraus abgeleiteten Euklidischen Transformation bis wir ein Minimum der Funktion F gefunden haben.

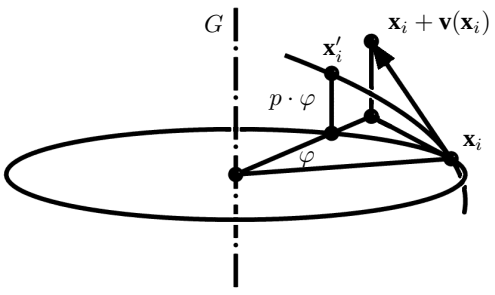


Abbildung 3: Neue Euklidische Position $\mathbf{x}_i'$ eines Punktes $\mathbf{x}_i$, im Gegensatz zur affinen Position $\mathbf{x}_i + \mathbf{v}(\mathbf{x}_i)$.

Bemerkung 1. Falls man mit Ausreißern in den Daten rechnen muss, wird man eine gewichtete Fehlerquadratsumme minimieren und in einer Gewichtsiteration den Einfluss der Ausreißer schrittweise zurückdrängen. Dies gilt auch für die folgenden Algorithmen, wird dort aber nicht mehr gesondert erwähnt.

3 SIMULTANE ORIENTIERUNG VON MEHREREN PUNKTWOLKEN MIT BEKANNTEN KORRESPONDENZEN

Wir fassen in diesem Absatz einen in [11] beschriebenen Algorithmus zusammen. Gegeben seien K Punktwolken ein und desselben Objektes, welche einander teilweise überlappen. Des weiteren seien bereits Korrespondenzen (mit einem gewissen Konfidenzwert) zwischen Punkten der einzelnen Punktwolken bekannt. Diese können zum Beispiel mit Hilfe von zusätzlich aufgenommenen digitalen Fotos gefunden werden. Die K Punktwolken werden als starre Systeme angenommen und mit $\Sigma_i, i = 1, \ldots, K$ bezeichnet. Eine beliebige Anzahl (≥ 1) der gegebenen Punktwolken wird festgehalten. Die anderen Punktwolken sollen so bewegt werden, dass nach der Anwendung der berechneten Verlagerungen die Summe der quadrierten Distanzen zwischen korrespondierenden Punkten, gewichtet mit den Konfidenzwerten, minimal wird. Für $K > 2$ ist nur mehr ein iterativer Algorithmus möglich.

Nur jene Punkte jeder Punktwolke werden im Algorithmus verwendet, welche im Überlappungsbereich mit einer oder mehreren anderen Punktwolken liegen. Die N Paare korrespondierender Punkte sind durch $(\mathbf{x}_i, \mathbf{y}_i)$ mit $\mathbf{x}_i \in \Sigma_j$ und $\mathbf{y}_i \in \Sigma_k$ gegeben. Jedes korrespondierende Punktepaar habe einen Konfidenzwert $w_i \in (0, 1]$.

Da wir nur kleine Verlagerungen erwarten, verwenden wir wieder Geschwindigkeitsvektorfelder. Die Verlagerung des Systems Σ_l gegenüber einem fixen System Σ_0 ist durch zwei Vektoren $\mathbf{c}_l, \bar{\mathbf{c}}_l \in \mathbb{R}^3$ gegeben. Der Geschwindigkeitsvektor $\mathbf{v}_{l0}(\mathbf{x}_i)$ eines Punktes $\mathbf{x}_i \in \Sigma_l$ ist dann gegeben als

$$\mathbf{v}_{l0}(\mathbf{x}_i) = \bar{\mathbf{c}}_l + \mathbf{c}_l \times \mathbf{x}_i. \tag{15}$$

Für ein Paar korrespondierender Punkte $(\mathbf{x}_i, \mathbf{y}_i)$ möchten wir den Abstand nach Verlagerung der Systeme Σ_j und Σ_k abschätzen. Für eine Näherung erster Ordnung verwenden wir die Geschwindigkeitsvektorfelder. Damit berechnet sich der quadrierte Abstand zweier verlagerter Punkte $\mathbf{x}_i$ und $\mathbf{y}_i$ als

$$\begin{aligned} &Q_1(\mathbf{x}_i, \mathbf{y}_i) = (\mathbf{x}_i + \mathbf{v}_{j0}(\mathbf{x}_i) - \mathbf{y}_i - \mathbf{v}_{k0}(\mathbf{y}_i))^2 = \\ &(\mathbf{x}_i - \mathbf{y}_i + (\bar{\mathbf{c}}_j + \mathbf{c}_j \times \mathbf{x}_i) - (\bar{\mathbf{c}}_k + \mathbf{c}_k \times \mathbf{y}_i))^2. \end{aligned} \tag{16}$$

Der Ausdruck $Q_1(\mathbf{x}_i, \mathbf{y}_i)$ ist eine *quadratische* Funktion in den Unbekannten $\mathbf{c}_j, \bar{\mathbf{c}}_j, \mathbf{c}_k, \bar{\mathbf{c}}_k$.

Eine Alternative zu (16) ist die folgende: Anstatt die Bewegung von Σ_j gegenüber Σ_0, und die Bewegung von Σ_k gegenüber Σ_0 zu linearisieren, kann man die Relativbewegung von Σ_j gegenüber Σ_k linearisieren. Der Geschwindigkeitsvektor $\mathbf{v}_{jk}$ eines Punktes $\mathbf{x}_i \in \Sigma_j$ für diese Relativbewegung ist gegeben durch

$$\mathbf{v}_{jk}(\mathbf{x}_i) = \mathbf{v}_{j0}(\mathbf{x}_i) - \mathbf{v}_{k0}(\mathbf{x}_i). \tag{17}$$

Der zu minimierende Abstand ist nun zwischen den Punkten $\mathbf{x}_i + \mathbf{v}_{jk}(\mathbf{x}_i)$ und $\mathbf{y}_i$ (d.h., $\mathbf{x}_i$ wird mit dem System Σ_j relativ zum Punkt $\mathbf{y}_i$ des Systems Σ_k verlagert). Der quadrierte Abstand dieser beiden Punkte ist gegeben als

$$\begin{aligned} &Q_2(\mathbf{x}_i, \mathbf{y}_i) = (\mathbf{x}_i + \mathbf{v}_{jk}(\mathbf{x}_i) - \mathbf{y}_i)^2 = \\ &(\mathbf{x}_i - \mathbf{y}_i + (\bar{\mathbf{c}}_j + \mathbf{c}_j \times \mathbf{x}_i) - (\bar{\mathbf{c}}_k + \mathbf{c}_k \times \mathbf{x}_i))^2. \end{aligned} \tag{18}$$

Der Ausdruck $Q_2(\mathbf{x}_i, \mathbf{y}_i)$ ist wieder eine quadratische Funktion in den Unbekannten $\mathbf{c}_j, \bar{\mathbf{c}}_j, \mathbf{c}_k, \bar{\mathbf{c}}_k$. Jedes Paar korrespondierender Punkte trägt einen entsprechenden Ausdruck Q_1 oder Q_2 in den noch unbekannten Bewegungsparametern bei. Um die Summe der quadrierten Abstände zwischen allen Paaren korrespondierender Punkte zu minimieren, minimieren wir die folgenden gewichtete Summe:

$$F = \sum_i w_i Q_2(\mathbf{x}_i, \mathbf{y}_i). \tag{19}$$

Das Gewicht w_i ist der bekannte Konfidenzwert des Punktepaares $(\mathbf{x}_i, \mathbf{y}_i)$. Da die Korrespondenzen bekannt sind, kann entweder Q_1 oder Q_2 verwendet werden. Die Minimierung der Funktion (19) führt auf die Lösung eines linearen Gleichungssystems in den $6K$ Unbekannten $\mathbf{c}_1, \bar{\mathbf{c}}_1, \mathbf{c}_2, \bar{\mathbf{c}}_2, \ldots, \mathbf{c}_K, \bar{\mathbf{c}}_K$. Dieser Zugang erlaubt das Festhalten von mehr als einem System ganz einfach: Soll das System Σ_l fest bleiben, dann braucht man bloß $\mathbf{c}_l$ und $\bar{\mathbf{c}}_l$ gleich Null setzen.

Die eigentlichen Verlagerungen werden dann, wie in Absatz 2.2 beschrieben, für jedes System berechnet und angewendet. Die beschriebene Prozedur wird dann solange iteriert, bis ein gegebenes Abbruchkriterium erfüllt wird.

Abb. 4 zeigt ein Beispiel zur simultanen Orientierung mehrerer Punktwolken mit bekannten Korrespondenzen, deren Orientierungen mit obigem Algorithmus optimiert wurden.

4 ORIENTIERUNG OHNE FESTE KORRESPONDENZEN

Wir kommen nun zur automatischen Feinorientierung, welche ohne gegebene feste Korrespondenzen durchgeführt wird. Im folgenden sei P das bewegte System und M das festgehaltene. Ziel ist es nun P iterativ so zu transformieren, dass M und P im Überlappungsbereich möglichst gut zur Deckung kommen. Zur Lösung dieser Aufgabenstellung präsentieren wir zwei Varianten des ICP (Iterative Closest Point) Algorithmus, welche (I) die Summe der quadrierten Punkt-zu-Punkt Abstände, und (II) die Summe der quadrierten Punkt-zu-Ebene Abstände minimieren. Eine Vielzahl von Varianten des ICP-Algorithmus wurden im letzten Jahrzehnt publiziert; einen guten Überblick bietet [12].

4.1 ICP-Algorithmus I: Punkt-zu-Punkt Abstand

Der *ICP (Iterative Closest Point) Algorithmus* wurde von Chen und Medioni [5] sowie Besl und McKay [1] eingeführt, und durchläuft die folgenden Schritte:

1. Im ersten Schritt jeder Iteration wird für eine bestimmte Anzahl von Datenpunkten aus der Punktwolke P

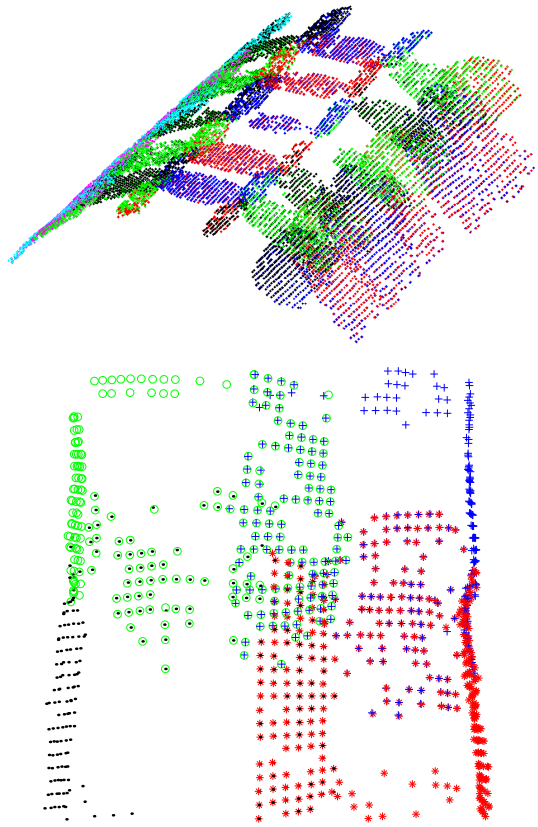


Abbildung 4: Simultane Orientierung von 30 gegebenen Punktwolken (oben) mit bekannten Korrespondenzen. Ausschnitt mit 4 Punktwolken welche die überlappenden Bereiche zeigen (unten). Daten mit freundlicher Genehmigung von Gerhard Paar, Joanneum Research Graz.

der näheste Punkt auf der Fläche M bestimmt. Das ist der zeitintensivste Schritt des Algorithmus und sollte daher effizient implementiert werden. Wir erhalten so eine Folge von Punkten $Y = (\mathbf{y}_1, \mathbf{y}_2, \ldots) \in M$ welche zu Punkten $X = (\mathbf{x}_1, \mathbf{x}_2, \ldots) \subset P$ am nähesten liegen. Jeder Punkt $\mathbf{x}_i$ korrespondiert dabei zu einem Punkt $\mathbf{y}_i$ mit demselben Index i.

2. Im zweiten Schritt jeder Iteration wird jene Euklidische Bewegung m berechnet, welche die Funktion (1) minimiert. Die Lösung wird wie in Absatz 2.1 angegeben berechnet.

Nach dem zweiten Schritt wird die neue Position der Punktwolke P via $P_{\text{neu}} = m(P_{\text{alt}})$ berechnet. Anschließend werden die Schritte 1 und 2 wiederholt, und die Lage der Punktwolke P wird jeweils aktualisiert. Die Prozedur wird beendet, falls der Funktionswert einen gewissen Schwellwert unterschreitet, oder eine maximale Anzahl von Iterationen erreicht wird.

Bemerkung 2. Hinsichtlich der Auswahl von Datenpunkten verweisen wir auf die Literatur [12], es sei jedoch hervorgehoben, dass für die Feinorientierung von Teilaufnahmen desselben Objektes ausschließlich Punkte aus Überlappungsbereichen der Punktwolken zu verwenden sind. Die Überlappungsbereiche folgen im ersten Schritt aus der manuell vorgenommenen Groborientierung.

4.2 ICP-Algorithmus II: Punkt-zu-Ebene Abstand

Im Gegensatz zur ICP-Variante I in Absatz 4.1 wird hier nicht der quadrierte Abstand zwischen korrespondierenden Punktpaaren minimiert, sondern der quadrierte Abstand zwischen Punkten der einen Punktwolke und den Tangentialebenen in den korrespondierenden Lotfußpunkten der anderen Punktwolke. Diese Idee findet sich bereits in Chen und Medioni [5]. Wir präsentieren hier eine Lösung, welche Überlegungen aus der Momentankinematik und Liniengeometrie verwendet und von Pottmann et al. [10] beschrieben wurde. Der erste Schritt besteht wie beim ICP-Algorithmus I im Bestimmen von Lotfußpunkten. Wir erhalten so wieder eine Folge von Punkten $Y = (\mathbf{y}_1, \mathbf{y}_2, \ldots) \in M$, welche zu den Punkten $X = (\mathbf{x}_1, \mathbf{x}_2, \ldots) \subset P$ am nächsten liegen. Dann linearisieren wir die gesuchte Euklidische Bewegung durch ein Geschwindigkeitsvektorfeld

$$\mathbf{v}(\mathbf{x}) = \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}.$$

Eine ähnliche Idee, die Bewegung zu linearisieren, findet sich bereits bei [4].

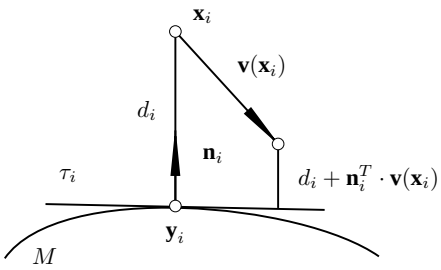


Abbildung 5: Abstand Punkt zu Ebene.

Der Abstand eines Punktes $\mathbf{x}_i + \mathbf{v}(\mathbf{x}_i)$ zur Tangentialebene τ_i im Punkt $\mathbf{y}_i$ mit Einheitsnormalvektor $\mathbf{n}_i$ von M ist gegeben durch $d_i + \mathbf{n}_i^T \cdot \mathbf{v}(\mathbf{x}_i)$, siehe Abb. 5 . Dabei ist d_i der orientierte Abstand von $\mathbf{x}_i$ zu $\mathbf{y}_i$. Wir minimieren nun die Summe der quadrierten Abstände der Punkte $\mathbf{x}_i$ zur jeweiligen Tangentialebene im korrespondierenden Lotfußpunkt $\mathbf{y}_i$,

$$F(\mathbf{c}, \bar{\mathbf{c}}) = \sum_{i=1}^{N} (d_i + \mathbf{n}_i^T \cdot (\bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}_i))^2 \to \min . \quad (20)$$

Die Funktion (20) ist quadratisch in den Unbekannten $(\mathbf{c}, \bar{\mathbf{c}})$, die Lösung wird also durch Berechnen eines linearen Gleichungssystems gefunden. Wie in Absatz 2.2 wird die Punktwolke P dann nicht mit dem Geschwindigkeitsvektorfeld verlagert, sondern mit der daraus berechneten Euklidischen Bewegung m, welche eine Überlagerung einer Drehung um eine Achse G durch einen Winkel $\varphi = \arctan(\|\mathbf{c}\|)$ und einer Schiebung parallel zu G um die Länge $p \cdot \varphi$ ist. Für die Berechnung von G und p siehe Anhang A.3.

Wie in der Variante I des ICP-Algorithmus, siehe Absatz 4.1, wird die neue Lage der Punktwolke P via $P_{\text{neu}} =$

$m(P_{\text{alt}})$ berechnet und dann wird die Prozedur so lange wiederholt, bis der Funktionswert eine bestimmte Schranke unterschreitet bzw. eine maximale Anzahl von Iterationen erreicht wird.

Bemerkung 3. Flächen, die Teile von Zylinder-, Dreh- oder Schraubflächen darstellen, gestatten Bewegungen in sich. Daher ist eine rein auf Geometrie basierende Orientierung nicht eindeutig durchführbar. In der Praxis wird man dann auch Texturen oder andere Zusatzinformationen berücksichtigen.

4.3 Vergleich ICP I und ICP II

Um den Unterschied im Konvergenzverhalten zwischen ICP I und ICP II zu erklären, verwenden wir ein von Pottmann und Hofer [9] hergeleitetes Ergebnis über lokale quadratische Approximanten der quadrierten Distanzfunktion zu Kurven und Flächen. Wir fassen hier nur das Ergebnis für den Flächenfall zusammen. Eine nicht-negative quadratische Approximation F_d^+ der quadrierten Distanzfunktion zu einer Fläche in einem Raumpunkt $\mathbf{p}$ ist im Gaußschen Dreibein (bestimmt durch die beiden Hauptkrümmungsrichtungen und die Flächennormale) im Normalenfußpunkt $\mathbf{f}$ gegeben durch

$$F_d^+(x_1,x_2,x_3) = \frac{d}{d+\rho_1}x_1^2 + \frac{d}{d+\rho_2}x_2^2 + x_3^2. \qquad (21)$$

Dabei sind ρ_1, ρ_2 die beiden Hauptkrümmungsradien im Flächenpunkt und $d = \|\mathbf{p}-\mathbf{f}\|$. Zwei Spezialfälle von (21) sind

$$F_\infty = x_1^2 + x_2^2 + x_3^2, \quad \text{und} \quad F_0 = x_3^2. \qquad (22)$$

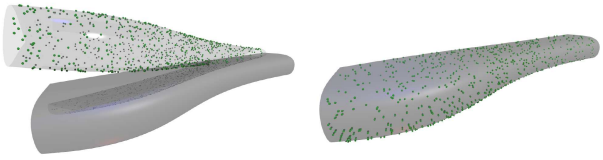
F_∞ ist die quadrierte Distanzfunktion zum Lotfußpunkt und F_0 ist die quadrierte Distanzfunktion zur Tangentialebene. In anderen Worten, F_∞ bzw. F_0 sind "gute" lokale quadratische Approximationen der quadrierten Distanzfunktion in weit von der Fläche entfernten Punkten bzw. in nahe bei der Fläche liegenden Punkten. Der ICP-Algorithmus I verwendet F_∞, während der ICP-Algorithmus II mit F_0 arbeitet. Damit wird das in Abb. 7 illustrierte unterschiedliche Konvergenzverhalten der beiden ICP-Varianten erklärt.

Bemerkung 4. Es ist leicht einzusehen, dass ICP I ein lineares Konvergenzverhalten aufweist. Hingegen lässt sich zeigen, dass ICP II quadratisch konvergiert, sofern die Abweichungen zwischen P und M im Überlappungsbereich gering sind.

4.4 Beispiele (Industrielle Inspektion)

Eine Aufgabenstellung aus der industriellen Praxis ist die folgende: Gegeben sind ein CAD (Computer Aided Design) Modell M eines Objektes und eine Laserscanner-Punktwolke P eines gefertigten Teils desselben Objektes. Man berechne und visualisiere die Abweichungen zwischen M und P zur Qualitätskontrolle von gefertigten Teilen.

Dies ist ein Orientierungsproblem ohne feste Korrespondenzen. Für das Beispiel in Abb. 6 haben wir synthetische Daten verwendet. Die Punktwolke wurde durch Auswählen bestimmter Punkte des CAD Modells generiert, und anschließend mit Gaußschem Rauschen versehen. Für die bessere Visualisierung ist die Punktwolke in Abb. 6 (**a**) zusammen mit einer transparenten Fläche abgebildet, welche allerdings für die Berechnung nicht verwendet wird. In jedem Iterationsschritt wird die oben beschriebene Schraubung berechnet und auf die Punktwolke angewendet. Nach vier Iterationsschritten von ICP II erhalten wir die in Abb. 6 (**b**) gezeigte Endlage.



(**a**) (**b**)

Abbildung 6: Orientierung einer Punktwolke zu einer Fläche. (**a**) Startlage und (**b**) Endlage nach 4 Iterationsschritten mit ICP II.

In Tabelle 7 vergleichen wir die Konvergenzgeschwindigkeit von ICP I und ICP II, welche in Absatz 4.3 begründet wird. Anschaulich betrachtet erlaubt ICP II tangentiale Bewegungen der Punktwolke, was besonders in den späteren Schritten des ICP-Algorithmus von Bedeutung ist.

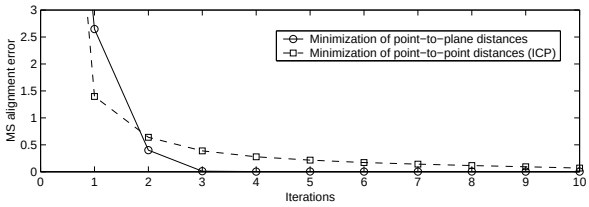
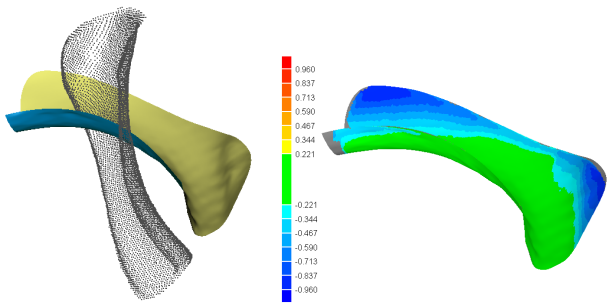


Abbildung 7: Vergleich der Konvergenzgeschwindigkeit von ICP I mit ICP II.

Ein zweites Beispiel mit realen Daten zeigt in Abb. 8 (**a**) die Ausgangslage der Punktwolke zum CAD Modell und in Abb. 8 (**b**) die Endlage mit farbcodierten Abweichungen zwischen Punktwolke und CAD Modell.



(**a**) (**b**)

Abbildung 8: Industrielle Inspektion. (**a**) Startlage und (**b**) Endlage mit farbcodierten Abweichungen. Abbildungen mit freundlicher Genehmigung der Firma Mediceram.

Bemerkung 5. Hat man die Aufgabenstellung, eine ganze Fertigungsserie desselben Objektes einer Qualitätskontrolle zu unterziehen, dann ist folgende Strategie sinnvoll, siehe [7]: In einem Vorverarbeitungsschritt wird der umgebende Raum des CAD-Modells in Zellen zerlegt, welche in einer geeigneten Datenstruktur abgespeichert werden. Für jede Zelle wird eine Funktion F_d (siehe Absatz 4.3) berechnet und abgespeichert. Im Inspektionsschritt werden dann die Zellen bestimmt, in denen die beteiligten Punkte der Punktwolke liegen und F_d wird in den jeweiligen Zellen ausgewertet und diese Terme werden zu einer in den Unbekannten $\mathbf{c}, \bar{\mathbf{c}}$ quadratischen Funktion aufsummiert. Die Minimierung erfolgt durch Lösung eines linearen Gleichungssystems. Da das zeitaufwendige Bestimmen von Korrespondenzen entfällt, ist damit eine Inspektion in Echtzeit durchführbar.

DANKSAGUNG

Die Autoren bedanken sich für die Unterstützung ihrer Arbeit durch das FWF Projekt P16002-N05 und durch das innovative Projekt "3D Technik" der Technischen Universität Wien. Ein besonderes Dankeschön gilt Camillo Ressel für die wertvollen Hinweise zu einer ersten Fassung dieser Arbeit.

A KINEMATIK

Eine umfassende Abhandlung über Kinematik findet man im Klassiker von Bottema und Roth [3].

A.1 Quaternionen

Eine *Quaternion* $\mathbf{q}$ kann als Erweiterung einer komplexen Zahl auf vier Komponenten angesehen werden,

$$\mathbf{q} = q_0 + iq_1 + jq_2 + kq_3, \tag{23}$$

mit $q_0, \ldots, q_3 \in \mathbb{R}$. Die Symbole i, j, k heißen *imaginäre Einheiten* und erfüllen die folgenden Rechenregeln

$$i \circ i = j \circ j = k \circ k = -1 \tag{24}$$

$$\begin{aligned} i \circ j &= k = -j \circ i && (25)\\ j \circ k &= i = -k \circ j && (26)\\ k \circ i &= j = -i \circ k. && (27) \end{aligned}$$

$\mathbb{H}$ sei die Menge aller Quaternionen. Die Addition $+ : \mathbb{H} \times \mathbb{H} \to \mathbb{H}$ von Quaternionen erfolgt komponentenweise. Die Multiplikation $\circ : \mathbb{H} \times \mathbb{H} \to \mathbb{H}$ von Quaternionen folgt aus den obigen Multiplikationsregeln für die imaginären Einheiten, und ist nicht kommutativ. Daher bildet $(\mathbb{H}, +, \circ)$ einen *Schiefkörper*. Zu jeder Quaternion $\mathbf{q}$ ist die *konjugierte Quaternion* $\bar{\mathbf{q}}$ definiert als

$$\bar{\mathbf{q}} = q_0 - iq_1 - jq_2 - kq_3. \tag{28}$$

Für je zwei Quaternionen $\mathbf{q}, \mathbf{p}$ gilt

$$\overline{\mathbf{q} \circ \mathbf{p}} = \bar{\mathbf{p}} \circ \bar{\mathbf{q}}. \tag{29}$$

Die *Norm* $N(\mathbf{q})$ und die *multiplikative Inverse* $\mathbf{q}^{-1}$ einer Quaternion $\mathbf{q}$ sind definiert bzw. gegeben als

$$\begin{aligned} N(\mathbf{q}) &:= q_0^2 + q_1^2 + q_2^2 + q_3^2 = \mathbf{q} \circ \bar{\mathbf{q}} = \bar{\mathbf{q}} \circ \mathbf{q}, && (30)\\ \mathbf{q}^{-1} &= \frac{\bar{\mathbf{q}}}{N(\mathbf{q})}. && (31) \end{aligned}$$

Das *Skalarprodukt* zweier Quaternionen $\mathbf{q}, \mathbf{p}$ ist gegeben durch das Skalarprodukt von zwei Vektoren des $\mathbb{R}^4$,

$$\langle \mathbf{q}, \mathbf{p} \rangle := q_0 p_0 + q_1 p_1 + q_2 p_2 + q_3 p_3, \tag{32}$$

und kann auch wie folgt berechnet werden,

$$\langle \mathbf{q}, \mathbf{p} \rangle = \frac{1}{2}(\mathbf{q} \circ \bar{\mathbf{p}} + \mathbf{p} \circ \bar{\mathbf{q}}) = \frac{1}{2}(\bar{\mathbf{q}} \circ \mathbf{p} + \bar{\mathbf{p}} \circ \mathbf{q}). \tag{33}$$

Eine weitere nützliche Eigenschaft des Skalarprodukts von Quaternionen ist

$$\langle \mathbf{q} \circ \mathbf{p}, \mathbf{r} \rangle = \langle \mathbf{p}, \bar{\mathbf{q}} \circ \mathbf{r} \rangle. \tag{34}$$

Quaternionen-Multiplikation einer Quaternion $\mathbf{x}$ mit einer Quaternion $\mathbf{q}$ von links kann in Matrizenschreibweise geschrieben werden als

$$\mathbf{q} \circ \mathbf{x} = \begin{pmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} = Q \cdot \mathbf{x}.$$

Falls $N(\mathbf{q}) = q_0^2 + \ldots + q_3^2 = 1$, dann ist die Matrix Q orthogonal mit $\det Q = 1$. Quaternionen-Multiplikation einer Quaternion $\mathbf{x}$ mit einer Quaternion $\mathbf{q}$ von rechts kann in Matrizenschreibweise geschrieben werden als

$$\mathbf{x} \circ \mathbf{q} = \begin{pmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & q_3 & -q_2 \\ q_2 & -q_3 & q_0 & q_1 \\ q_3 & q_2 & -q_1 & q_0 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} = \tilde{Q} \cdot \mathbf{x}.$$

Die Matrix $\tilde{Q}$ unterscheidet sich von Q nur in der rechten unteren 3×3 Untermatrix, welche die Transponierte der entsprechenden Untermatrix von Q ist. Wieder gilt, dass $\tilde{Q}$ orthogonal ist, falls $N(\mathbf{q}) = 1$.

Vektoren $\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$ werden in $\mathbb{H} = \mathbb{R}^4$ als Quaternionen $(0, x_1, x_2, x_3) = ix_1 + jx_2 + kx_3$ eingebettet. Wegen (28) gilt dann $\bar{\mathbf{x}} = -\mathbf{x}$. Für eine Quaternion $\mathbf{q} = q_0 + iq_1 + jq_2 + kq_3$ heißen

$$S(\mathbf{q}) := q_0, \quad V(\mathbf{q}) := iq_1 + jq_2 + kq_3 \tag{35}$$

Skalarteil $S(\mathbf{q})$ bzw. *Vektorteil* $V(\mathbf{q})$. Analog zum Realteil und Imaginärteil einer komplexen Zahl, welche aus der komplexen Zahl und ihrer Konjugierten berechnet werden können, lassen sich Skalar- und Vektorteil einer Quaternion aus der Quaternion und ihrer Konjugierten wie folgt berechnen:

$$S(\mathbf{q}) = \frac{1}{2}(\mathbf{q} + \bar{\mathbf{q}}), \quad V(\mathbf{q}) = \frac{1}{2}(\mathbf{q} - \bar{\mathbf{q}}). \tag{36}$$

Wir studieren nun für $\mathbf{x} \in \mathbb{R}^3$ (d.h., $S(\mathbf{x}) = 0$) und eine feste Quaternion $\mathbf{q} \in \mathbb{H}$ die Abbildung

$$\mathbf{x} \longmapsto \mathbf{x}' := \mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}}. \tag{37}$$

Zuerst berechnen wir den Skalarteil von $\mathbf{x}'$,

$$\begin{aligned} 2S(\mathbf{x}') = \mathbf{x}' + \bar{\mathbf{x}}' &= \mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}} + \overline{\mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}}} \\ &= \mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}} + \mathbf{q} \circ \bar{\mathbf{x}} \circ \bar{\mathbf{q}} \\ &= \mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}} - \mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}} = 0. \end{aligned} \quad (38)$$

Das Bild $\mathbf{x}'$ ist also wieder ein Vektor in $\mathbb{R}^3$. Falls $N(\mathbf{q}) = \bar{\mathbf{q}} \circ \mathbf{q} = \mathbf{q} \circ \bar{\mathbf{q}} = 1$, dann gilt

$$N(\mathbf{x}') = \mathbf{x}' \circ \bar{\mathbf{x}}' = \mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}} \circ \mathbf{q} \circ \bar{\mathbf{x}} \circ \bar{\mathbf{q}} = N(\mathbf{x}). \quad (39)$$

Also erhält die Abbildung die Länge von Vektoren. Somit ist für $N(\mathbf{q}) = 1$ die Abbildung (37) eine lineare Abbildung von $\mathbb{R}^3 \mapsto \mathbb{R}^3$ (mit $\det > 0$), welche Längen erhält. Die Abbildung beschreibt also eine Drehung (eine sphärische Bewegung).

Satz 1. *Die Abbildung $\mathbf{x}' = \mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}}$ mit einer festen Einheitsquaternion $\mathbf{q} \in \mathbb{H}$, $N(\mathbf{q}) = 1$, und $\mathbf{x} \in \mathbb{R}^3$ beschreibt eine Drehung $\mathbf{x}' = R \cdot \mathbf{x}$ mit der Drehmatrix*

$$R = \begin{pmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{pmatrix}. \quad (40)$$

Dies liefert eine explizite Parametrisierung der Gruppe der orthogonalen Matrizen mit den Parametern q_0, q_1, q_2, q_3.

Bemerkung 6 (Berechnung der Drehmatrix aus Drehachse und Drehwinkel). Aus dem normierten Richtungsvektor $\mathbf{g}$ der Drehachse und dem Drehwinkel φ berechnet man die Einheitsquaternion $\mathbf{q}$ von Satz 1 wie folgt:

$$\mathbf{q} = \cos\frac{\varphi}{2} + \mathbf{g}\sin\frac{\varphi}{2}. \quad (41)$$

Man beachte, dass $\mathbf{q}$ und $-\mathbf{q}$ dieselbe Drehung darstellen, und dass $\mathbf{q}$ unverändert bleibt, wenn man die Orientierung von $\mathbf{g}$ und das Vorzeichen von φ gleichzeitig umkehrt.

Bemerkung 7. Für die Quaternionen-Multiplikation haben wir die Matrix-Schreibweisen $\mathbf{q} \circ \mathbf{x} = Q \cdot \mathbf{x}$ und $\bar{\mathbf{q}} \circ \mathbf{x} = Q^T \cdot \mathbf{x}$ hergeleitet. Damit läßt sich für $\mathbf{x} \in \mathbb{R}^3$ die folgende Darstellung der Abbildung $\mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}}$ herleiten,

$$\mathbf{q} \circ \mathbf{x} \circ \bar{\mathbf{q}} = \mathbf{y} \circ \bar{\mathbf{q}} = \tilde{Q}^T \cdot \mathbf{y} = \tilde{Q}^T \cdot \mathbf{q} \circ \mathbf{x} = \tilde{Q}^T Q \cdot \mathbf{x}, \quad (42)$$

mit $\mathbf{y} := \mathbf{q} \circ \mathbf{x}$. Die Matrix

$$\tilde{Q}^T Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & & & \\ 0 & & R & \\ 0 & & & \end{pmatrix} \quad (43)$$

ist eine orthogonale Matrix in $\mathbb{R}^4$ welche die orthogonale Matrix R aus (40) in der rechten unteren 3×3 Untermatrix enthält.

A.2 Plücker-Koordinaten

Es sei G eine Gerade in $\mathbb{R}^3$ und $\mathbf{p}$ ein beliebiger Punkt auf G. Es seien weiters $\mathbf{g}$ ein *Richtungsvektor*, und $\bar{\mathbf{g}} := \mathbf{p} \times \mathbf{g}$ der sogenannte *Momentenvektor* der Gerade G. Die sechs Koordinaten $(\mathbf{g}, \bar{\mathbf{g}})$ heißen *Plücker Koordinaten* von G. Sie besitzen die folgenden Eigenschaften:

- Der Momentenvektor ist unabhängig von der Wahl von $\mathbf{p}$ auf G: Man wähle einen weiteren Punkt $\mathbf{p}' := \mathbf{p} + \lambda\mathbf{g}$ auf G, dann folgt $\mathbf{p}' \times \mathbf{g} = (\mathbf{p} + \lambda\mathbf{g}) \times \mathbf{g} = \mathbf{p} \times \mathbf{g}$.
- Die Plücker-Koordinaten sind *homogene Koordinaten*, da ein Richtungsvektor $\mathbf{g}' = \mu\mathbf{g}$ einen Momentenvektor $\bar{\mathbf{g}}' = \mathbf{p} \times \mathbf{g}' = \mu(\mathbf{p} \times \mathbf{g}) = \mu\bar{\mathbf{g}}$ mit demselben Faktor μ impliziert. Es gilt also, dass $\mu(\mathbf{g}, \bar{\mathbf{g}})$ dieselbe Gerade beschreibt wie $(\mathbf{g}, \bar{\mathbf{g}})$.
- Die Plücker-Bedingung $\mathbf{g}^T \cdot \bar{\mathbf{g}} = 0$ gilt.

Die Menge aller Geraden in $\mathbb{R}^3$ ist vierdimensional. Dies steht im Einklang damit, dass für die sechs Plücker-Koordinaten die Homogenitätseigenschaft und die Plücker-Bedingung gelten. Hinsichtlich der Zusammenhänge von Liniengeometrie und Kinematik verweisen wir auf [8].

A.3 Das Geschwindigkeitsvektorfeld eines einparametrigen Bewegungsvorgangs

Wir betrachten einen nach der Zeit t parametrisierten Bewegungsvorgang eines starren Systems Σ gegenüber einem ruhenden System Σ_0. In Σ und Σ_0 seien kartesische Koordinatensysteme ausgezeichnet. Dann lässt sich die Lage $\mathbf{x}_0(t)$ eines Punktes $\mathbf{x}$ aus Σ in Σ_0 darstellen als

$$\mathbf{x}_0(t) = \mathbf{u}(t) + A(t) \cdot \mathbf{x}. \quad (44)$$

Hierbei ist $\mathbf{u}(t)$ die Position des Ursprungs von Σ und A ist eine orthogonale Matrix mit $\det A = 1$. Gleichung (44) ist auch eine Parameterdarstellung der Bahnkurve von $\mathbf{x}$.

Durch Differentiation erhält man die Geschwindigkeitsvektoren $\mathbf{v}_0(\mathbf{x}_0)$ der Punkte $\mathbf{x}$ aus Σ. Es ist bekannt, dass sich die Geschwindigkeitsvektoren $\mathbf{v}_0(\mathbf{x}_0)$ aus den Punkten $\mathbf{x}_0$ linear nach dem folgenden Gesetz berechnen lassen:

$$\mathbf{v}_0(\mathbf{x}_0) = \bar{\mathbf{c}} + \mathbf{c} \times \mathbf{x}_0. \quad (45)$$

Offenbar ist $\bar{\mathbf{c}}$ Geschwindigkeitsvektor jenes Punktes des Gangsystems, welcher gerade mit dem Ursprung von Σ_0 zusammenfällt. Der Vektor $\mathbf{c} := (c_1, c_2, c_3)^T$ sammelt die drei wesentlichen Einträge der schiefsymmetrischen Matrix

$$C := \dot{A}A^T = \begin{pmatrix} 0 & -c_3 & c_2 \\ c_3 & 0 & -c_1 \\ -c_2 & c_1 & 0 \end{pmatrix}. \quad (46)$$

Man beachte $C \cdot \mathbf{x}_0 = \mathbf{c} \times \mathbf{x}_0$.

Es ist bekannt, dass das momentane Geschwindigkeitsvektorfeld eines einparametrigen Bewegungsvorgangs mit dem Geschwindigkeitsvektorfeld einer bestimmten Schraubung

(in Sonderfällen Drehung oder Schiebung) übereinstimmt. Aus $\mathbf{c}, \bar{\mathbf{c}}$ kann man die Achse G, mit Plücker-Koordinaten $(\mathbf{g}, \bar{\mathbf{g}})$, und den Schraubparameter p dieser Momentanschraubung wie folgt berechnen (siehe [8]):

$$\mathbf{g} = \frac{\mathbf{c}}{\|\mathbf{c}\|}, \quad \bar{\mathbf{g}} = \frac{\bar{\mathbf{c}} - p\mathbf{c}}{\|\mathbf{c}\|}, \quad p = \frac{\mathbf{c}^T \cdot \bar{\mathbf{c}}}{\mathbf{c}^2}. \qquad (47)$$

Literatur

[1] Besl, P.J., McKay, N.D. (1992) A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. and Machine Intell.* **14** (2), 239–256.

[2] Blaschke, W. (1960) *Kinematik und Quaternionen*. Deutscher Verlag der Wissenschaften, Berlin.

[3] Bottema, O., Roth, B. (1990) *Theoretical Kinematics*. Dover, New York.

[4] Bourdet, P., Clément, A. (1988) A study of optimal-criteria identification based on the small-displacement screw model. *Annals of the CIRP* **37**, 503–506.

[5] Chen, Y., Medioni, G. (1992) Object modelling by registration of multiple range images. *Image and Vision Computing* **10**, 145–155.

[6] Horn, B. K. P. (1987) Closed form solution of absolute orientation using unit quaternions. *Journal of the Optical Society A* **4**, 629–642.

[7] Leopoldseder, S., Pottmann, H., Zhao, H.-K. (2003) The d^2-tree: A hierarchical representation of the squared distance function. Technical Report No. 101, Institute of Geometry, Vienna University of Technology.

[8] Pottmann, H., Wallner, J. (2001) *Computational Line Geometry*. Springer-Verlag Berlin Heidelberg New York.

[9] Pottmann, H., Hofer, M. (2003) Geometry of the squared distance function to curves and surfaces. In: Hege, H.-C. and Polthier, K., eds., *Visualization and Mathematics III*, Springer, 221-242.

[10] Pottmann, H., Leopoldseder, S., Hofer, M. (2002a) Registration without ICP. Technical Report No. 91, Institute of Geometry, Vienna University of Technology.

[11] Pottmann, H., Leopoldseder, S., Hofer, M. (2002b) Simultaneous registration of multiple views of a 3D object. Proceedings PCV '02, *Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XXXIV, Part 3A, Commision III, 265-270.

[12] Rusinkiewicz, S., Levoy, M. (2001) Efficient variants of the ICP algorithm. In: *Proc. 3rd Int. Conf. on 3D Digital Imaging and Modeling*, Quebec.