

# Quaternionen mit Java

Christian Basler

## Inhaltsverzeichnis

<b>1</b>	<b>Zusammenfassung</b>	<b>1</b>
<b>2</b>	<b>Grundlagen</b>	<b>1</b>
2.1	Polardarstellung . . . . .	2
2.2	Rotation . . . . .	2
<b>3</b>	<b>Java-Bibliothek</b>	<b>3</b>
<b>4</b>	<b>Beispielanwendungen</b>	<b>4</b>
4.1	Wo ist unten? . . . . .	4
4.2	Künstlicher Horizont . . . . .	4
<b>5</b>	<b>Diskussion</b>	<b>4</b>
<b>6</b>	<b>Literatur</b>	<b>4</b>
<b>7</b>	<b>Anhang</b>	<b>4</b>

## 1 Zusammenfassung

## 2 Grundlagen

Quaternionen  $\mathbb{H}$  erweitern die Komplexen Zahlen  $\mathbb{C}$  um die Komponenten  $j$  und  $k$ .

$$q = q_0 + q_1i + q_2j + q_3k$$

Dabei gilt  $i^2 = j^2 = k^2 = ijk = -1$  und daher auch z.B.  $ij = k$  und  $jk = i$ .

Euklidische Vektoren können dabei wie folgt in eine Quaternion abgebildet werden:

$$q_{\vec{v}} = 0 + v_x \mathbf{i} + v_y \mathbf{j} + v_z \mathbf{k}$$

Daher wird der Imaginärteil einer Quaternion auch Vektorteil genannt. Eine solche Quaternion, welche nur aus Vektorteil besteht, wird auch als *reine Quaternion* bezeichnet.

## 2.1 Polardarstellung

Quaternionen  $\notin \mathbb{R}$  lassen sich eindeutig in der Form

$$q = |q|(\cos \phi + \epsilon \sin \phi)$$

darstellen mit dem Betrag

$$|q| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$$

dem Polarwinkel

$$\phi := \arccos q = \arccos \operatorname{Re} q$$

und der reinen Einheitsquaternion

$$\epsilon = \frac{\operatorname{Im} q}{|\operatorname{Im} q|}$$

## 2.2 Rotation

Quaternionen erlauben eine elegante Darstellung von Drehungen im dreidimensionalen Raum:

$$y = qxq^{-1} = qx\bar{q}$$

$$q = \cos \frac{\alpha}{2} + \epsilon \sin \frac{\alpha}{2}$$

$q$  ist dabei eine Einheitsquaternion<sup>1</sup> und stellt eine Drehung um Achse  $\epsilon$  mit Winkel  $\alpha$  dar.

---

<sup>1</sup> $|q| = 1$

### 3 Java-Bibliothek

Die Java-Bibliothek stellt ein Objekt "Quaternion" mit folgenden Methoden zur Verfügung:

- $q.add(r) = q + r$
- $q.subtract(r) = q - r$
- $q.multiply(r) = qr$
- $q.conjugate() = \bar{q}$
- $q.norm() = |q|$
- $q.normalize() = \frac{q}{|q|}$
- $q.reciprocal() = q^{-1}$
- $q.divide(r) = qr^{-1}$
- $q.rotate(\theta, x, y, z)$  = Rotation um Achse  $(x, y, z)$  mit Winkel  $\theta$
- $q.exp() = e^q$
- $q.ln() = \ln q$
- $q.dot(r) = q \cdot r = q_0r_0 + q_1r_1 + q_2r_2 + q_3r_3$
- $q.cross(r) = \vec{q} \times \vec{r}$  (d.h.  $q_0$  und  $r_0$  werden ignoriert)
- $q.getRe() = \mathbf{Re} \, q$
- $q.getIm() = \mathbf{Im} \, q$
- $q.getPhi()$
- $q.getEpsilon()$
- $q.equals(r, \delta) = |q - r|^2 < \delta$
- $q.equals(r) = q.equals(r, \text{Quaternion.DELTA})$

Zum Erstellen neuer Quaternionen besteht ausserdem die statische Methode `H` in folgenden Ausführungen:

- $H(q_0, q_1, q_2, q_3) = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$
- $H(x, y, z) = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$
- $H(w) = w$
- $H(\alpha, \vec{v}) = \cos \frac{\alpha}{2} + \mathbf{i} \sin \frac{\alpha}{2} v_x + \mathbf{j} \sin \frac{\alpha}{2} v_y + \mathbf{k} \sin \frac{\alpha}{2} v_z$
- $H([x, y, z]), H([w, x, y, z])$
- $\text{getRotation}(p, q) = r$ , so dass  $rp\bar{r} = q$

Für Fälle wo euklidische Vektoren benötigt werden, z.B. beim Konstruktor  $H(\alpha, \vec{v})$ , gibt es ausserdem die Klasse **Vector**, welche jedoch nur sehr eingeschränkte Funktionen bietet.

## 4 Beispielanwendungen

### 4.1 Wo ist unten?

### 4.2 Künstlicher Horizont

## 5 Diskussion

## 6 Literatur

## 7 Anhang