

Lab Course :: 3D Computer Vision

Transformations in 3D

Ben Glocker, Tobias Sielhorst, Joerg Traub

Original Slides by Darko Zikic

22 April 2005

chair for computer aided medical procedures

department of computer science | technische universität münchen

Transformations in 3D :: Projective Space

- a general transformation matrix (4x4):
$$\begin{bmatrix} A & t \\ v^T & u \end{bmatrix}$$
- A: affine matrix (3x3). includes the rotation matrix R
- R: the rotation matrix(3x3)
- t: the translation vector (3x1)
- v: enables projective transformations (3x1)

Rotations in 3D

- Euler Angles
- Rotation Matrix (3x3)
- Quaternions

Rotation in 3D :: Rotation Matrix

- rotation around the X-axis:

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) \\ 0 & -\sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

- rotation around the Y-axis:

$$B = \begin{bmatrix} \cos(\beta) & 0 & -\sin(\beta) \\ 0 & 1 & 0 \\ \sin(\beta) & 0 & \cos(\beta) \end{bmatrix}$$

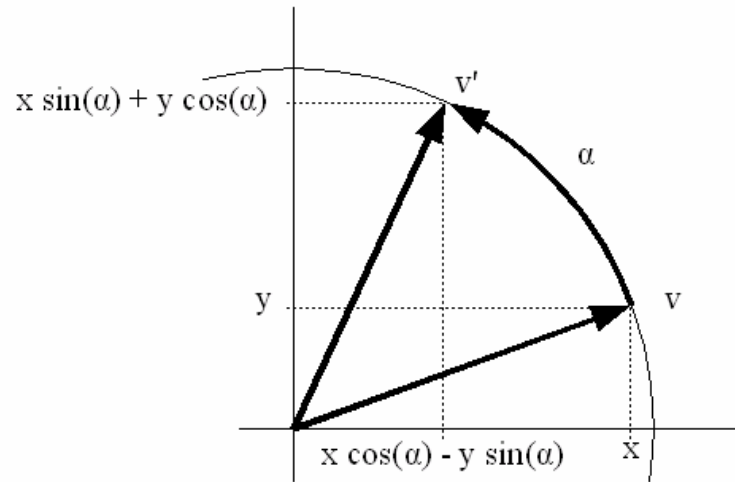
- rotation around the Z-axis:

$$C = \begin{bmatrix} \cos(\gamma) & \sin(\gamma) & 0 \\ -\sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Rotation in 3D space: $R = CBA$

Rotation in 3D :: Rotation Matrix Example

- rotation around the z axis



$$\begin{bmatrix} \cos(\phi) & \sin(\phi) & 0 \\ -\sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos(\phi) \cdot x + \sin(\phi) \cdot y \\ -\sin(\phi) \cdot x + \cos(\phi) \cdot y \\ z \end{bmatrix}$$

Rotation in 3D :: Quaternions

- quaternions intuitively: 3D vector extended by a real number



- quaternions rotate vectors by multiplication
 - r describes the rotation angle
 - a, b, c are the rotation axis

Transformation in 3D using Quaternions

- (v,q,s) : vector, quaternion, scalar
- transformation:
$$p' = s \cdot (q \cdot p \cdot q^{-1}) + v$$
- vector: translation
- quaternion: rotation
- scalar: scaling

Quaternions :: Definitions :: Hypercomplex Number

- quaternion Q :
$$Q = 1 \cdot q_1 + i \cdot q_2 + j \cdot q_3 + k \cdot q_4$$
- i, j, k : symbolic characters with:
$$i^2 = -1, j^2 = -1, k^2 = -1$$
- axiomatic properties:
$$ir = ri, jr = rj, kr = rk$$
 - it follows:
$$ij = k, ji = -k$$
$$jk = i, kj = -i$$
$$ki = j, ik = -j$$
- defined addition and multiplication

Properties of Quaternions

□ for a quaternion $Q = [q_r, q_v]$

□ conjugate: $\bar{Q} = [q_r, -q_v]$

□ inverse: $Q^{-1} = \frac{\bar{Q}}{|Q|^2}$

□ unit quaternions: $|Q| = 1$

□ inverse for unit quaternions: $Q^{-1} = \bar{Q}$

Quaternions :: Different Notations

$$Q = 1 \cdot q_1 + i \cdot q_2 + j \cdot q_3 + k \cdot q_4$$

$$= [q_1, [q_2, q_3, q_4]]$$

$$= [q_r, q_v]$$

$$= [q_1, q_2, q_3, q_4]$$

Rotation in 3D :: Quaternions

- for a quaternion $Q = [q_r, q_v]$
- QVQ^{-1} is a rotation around q_v by $q_r = \cos\left(\frac{1}{2}\theta\right)$
- rotation composition
 - successive rotation by Q and then by P: $PQVQ^{-1}P^{-1}$

Projective Space vs. Quaternions

projective space

+ non-uniform scaling, shearing and projective transformations possible

- not intuitive / user-unfriendly
- unnecessarily large (results in numerical problems)
- ambiguous
- rotation interpolation difficult
- gimbal lock

quaternions

- + intuitive / simple
- + small size
- + non-ambiguous
- + easier rotation interpolation
- + no gimbal lock

- only rotation, translation and scaling possible