Orientation representation with quaternions

Robotics and Computer vision (RoVi)

Aljaz Kramberger

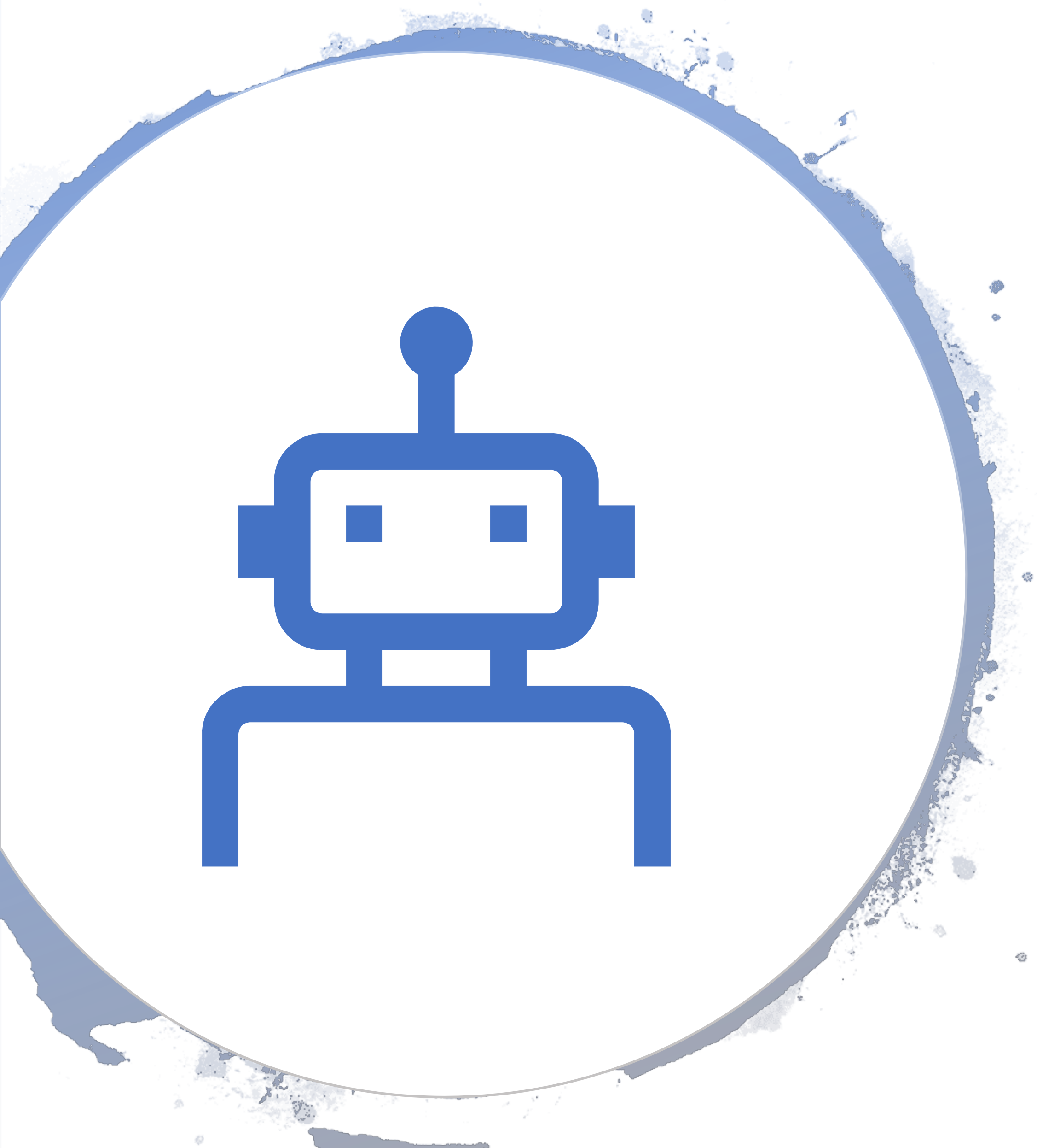
alk@mmmi.sdu.dk

SDU Robotics

The Maersk Mc-Kinney Moller Institute

University of Southern Denmark

Based on M. Mihelj, T. Bajd. et al. Robtics – second edition, Springer, 2018



Agenda

- Quaternion definition
  - Example
- Quaternion operations
  - Example
- Rotation conversions
  - Example

Quaternions

Ref: https://www.3dgep.com/understanding-quaternions/,

https://www.euclideanspace.com/maths/algebra/realNormedAlgebra/quaternions/

Origin

- Quaternions were discovered on 16 October 1843 by William Rowan Hamilton.
- He spent years trying to find a three dimensional number systems, but with no success, when he looked in 4 dimensions instead of 3 it worked

Source: en.wikipedia.org

Quaternions

- Quaternions allow stable and constant interpolation of orientations
- Compact(-ish) representation
- Also easy to concatenate
- Faster multiplication algorithms to combine successive rotations than using rotation matrices
- Easier to normalize than rotation matrices
- Mathematically stable – suitable for statistics
- Given an angle and axis, easy to convert to and from quaternion

Applications

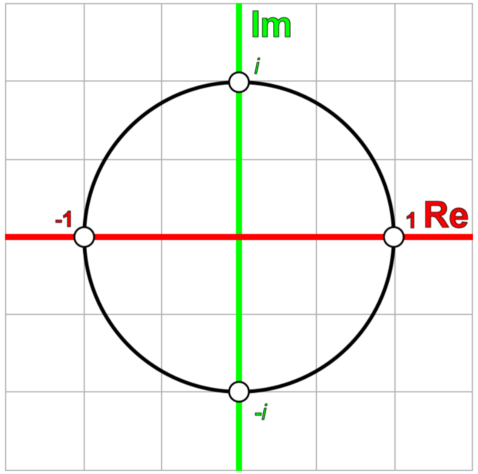
- Quaternions are used to represent rotations and orientations of objects in three-dimensional space in the following fields:
  - Computer graphics
  - Control theory
  - Signal processing
  - Attitude controls
  - Physics
  - Orbital mechanics
  - Quantum Computing

Complex numbers - recap

- Before we can fully understand quaterions, we must first understand where they came from. The root of quaternions is based on the concept of the complex number system.
- The set of complex numbers $\mathbb{C}$ is the sum of a real number and an imaginary number and has the form:

$$z = a + bi \quad a, b \in \mathbb{R}, i^2 = -1$$

- Operations in space of complex numbers can be useful



Source: www.3dgep.com

Definition

- We can extend complex numbers to 3-dimensional space by adding two imaginary numbers to our number system in addition to $\boldsymbol{i}$
- Quaternions have 4 dimensions (each quaternion consists of 4 scalar numbers), one real dimension and 3 imaginary dimensions.
- Each of these imaginary dimensions has a unit value of the square root of -1, but they are different square roots of -1 all mutually perpendicular to each other, known as $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$.
- We can represent 3D rotations as 3 numbers (e.g. Euler angles) but such a representation is non-linear and difficult to work with.
- An analogy of a two dimensional map of the earth

Definition

- A quaternion can be represented as follows:

$$q = w + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} \quad s, x, y, z \in \mathbb{R}$$

- Various annotation can be used:

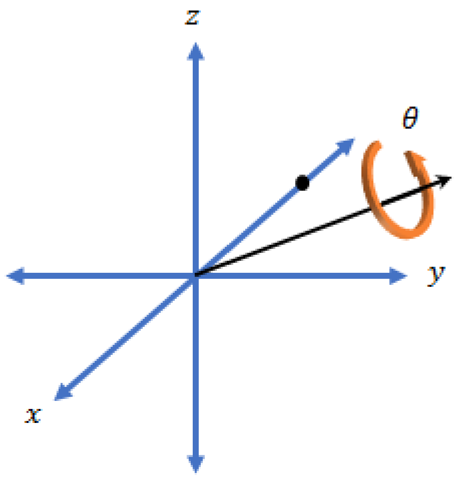
$$\begin{aligned} q &= [w, \boldsymbol{v}] \\ &= [w, x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}] \\ &= w + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} \end{aligned}$$

- In literature scalar and vector part are sometimes flipped $q = [\boldsymbol{v}, w]$

Definition

- A quaternion defines a rotation around a vector in 3d space
- Basic equation for rotation θ around a vector $\boldsymbol{v}$:

$$
\begin{aligned}
q &= [\cos(\theta/2), \sin(\theta/2)\,\boldsymbol{v}] \\
&= [\cos(\theta/2), \sin(\theta/2)\,\boldsymbol{i} + \sin(\theta/2)\,\boldsymbol{j} + \sin(\theta/2)\,\boldsymbol{k}] \\
&= \cos(\theta/2) + \sin(\theta/2)\,\boldsymbol{i} + \sin(\theta/2)\,\boldsymbol{j} + \sin(\theta/2)\,\boldsymbol{k}
\end{aligned}
$$



Source: se.mathworks.com

Example

Basic rotations

Basic rotations in python

1. Rotation around x-axis for 90°
2. Rotation around y-axis for -120°

- Basic equation for rotation around a vector:

$$q = [\cos(\theta/2), \sin(\theta/2)\,\boldsymbol{v}]$$

- Can check results with: rotx and roty function in *spatialmath.base*

Basic rotations in Python - solutions

1. Rotation around x-axis for 90°

   -> $[0.7071,\ 0.7071,\ 0,\ 0]$

2. Rotation around y-axis for -120°

   -> $[0.5,\ 0,\ -0.866,\ 0]$

Quaternion operations

Quaternion operations

- Addition
- Subtraction
- Multiplication
- Conjugation
- Norm (magnitude)
- Normalization (unit-norm)
- Inverse (division)

Example quaternions

- Two example quaternions can be given as:

$$q_a = [w_a, \boldsymbol{v}_a] = [w_a, x_a\boldsymbol{i} + y_a\boldsymbol{j} + z_a\boldsymbol{k}] = w_a + x_a\boldsymbol{i} + y_a\boldsymbol{j} + z_a\boldsymbol{k}$$

$$q_b = [w_b, \boldsymbol{v}_b] = [w_b, x_b\boldsymbol{i} + y_b\boldsymbol{j} + z_b\boldsymbol{k}] = w_b + x_b\boldsymbol{i} + y_b\boldsymbol{j} + z_b\boldsymbol{k}$$

Addition

- Adding two quaternions:

$$q_a + q_b =$$

$$[w_a + w_b, \boldsymbol{v}_a + \boldsymbol{v}_b] =$$

$$[w_a + w_b, (x_a + x_b)\boldsymbol{i} + (y_a + y_b)\boldsymbol{j} + (z_a + z_b)\boldsymbol{k}]$$

Subtraction

- Subtracting two quaternions:

$$q_a - q_b =$$

$$[w_a - w_b, \boldsymbol{v}_a - \boldsymbol{v}_b] =$$

$$[w_a - w_b, (x_a - x_b)\boldsymbol{i} + (y_a - y_b)\boldsymbol{j} + (z_a - z_b)\boldsymbol{k}]$$

Multiplication

- Two unit quaternions multiplied together result in another unit quaternion:

$$q_a q_b = [w_a \boldsymbol{v}_a][w_b \boldsymbol{v}_b]$$

Multiplication

- Two unit quaternions multiplied together result in another unit quaternion:

$$
\begin{aligned}
q_a q_b &= [w_a \boldsymbol{v}_a][w_b \boldsymbol{v}_b] \\
&= (w_a + x_a \boldsymbol{i} + y_a \boldsymbol{j} + z_a \boldsymbol{k})(w_b + x_b \boldsymbol{i} + y_b \boldsymbol{j} + z_b \boldsymbol{k})
\end{aligned}
$$

Multiplication

- Two unit quaternions multiplied together result in another unit quaternion:

$$\begin{aligned} q_a q_b &= [w_a \boldsymbol{v}_a][w_b \boldsymbol{v}_b] \\ &= (w_a + x_a \boldsymbol{i} + y_a \boldsymbol{j} + z_a \boldsymbol{k})(w_b + x_b \boldsymbol{i} + y_b \boldsymbol{j} + z_b \boldsymbol{k}) \\ &= (w_a w_b - x_a x_b - y_a y_b - z_a z_b) \\ &\quad + (w_a x_b + w_b x_a + y_a z_b - y_b z_a)\boldsymbol{i} \\ &\quad + (w_a y_b + w_b y_a + z_a x_b - z_b x_a)\boldsymbol{j} \\ &\quad + (w_a z_b + w_b z_a + x_a y_b - x_b y_a)\boldsymbol{k} \end{aligned}$$

Multiplication

- Two unit quaternions multiplied together result in another unit quaternion:

$$
\begin{aligned}
q_a q_b &= [w_a \boldsymbol{v}_a][w_b \boldsymbol{v}_b] \\
&= (w_a + x_a \boldsymbol{i} + y_a \boldsymbol{j} + z_a \boldsymbol{k})(w_b + x_b \boldsymbol{i} + y_b \boldsymbol{j} + z_b \boldsymbol{k}) \\
&= (w_a w_b - x_a x_b - y_a y_b - z_a z_b) \\
&\quad +(w_a x_b + w_b x_a + y_a z_b - y_b z_a)\boldsymbol{i} \\
&\quad +(w_a y_b + w_b y_a + z_a x_b - z_b x_a)\boldsymbol{j} \\
&\quad +(w_a z_b + w_b z_a + x_a y_b - x_b y_a)\boldsymbol{k} \\
&= [w_a w_b - \boldsymbol{v}_a \cdot \boldsymbol{v}_b, w_a \boldsymbol{v}_b + w_b \boldsymbol{v}_a + \boldsymbol{v}_a \times \boldsymbol{v}_b]
\end{aligned}
$$

Multiplication

- Two unit quaternions multiplied together result in another unit quaternion:

$$
\begin{aligned}
q_a q_b &= [w_a \boldsymbol{v}_a][w_b \boldsymbol{v}_b] \\
&= (w_a + x_a \boldsymbol{i} + y_a \boldsymbol{j} + z_a \boldsymbol{k})(w_b + x_b \boldsymbol{i} + y_b \boldsymbol{j} + z_b \boldsymbol{k}) \\
&= (w_a w_b - x_a x_b - y_a y_b - z_a z_b) \\
&\quad + (w_a x_b + w_b x_a + y_a z_b - y_b z_a)\boldsymbol{i} \\
&\quad + (w_a y_b + w_b y_a + z_a x_b - z_b x_a)\boldsymbol{j} \\
&\quad + (w_a z_b + w_b z_a + x_a y_b - x_b y_a)\boldsymbol{k} \\
&= [w_a w_b - \boldsymbol{v}_a \cdot \boldsymbol{v}_b, w_a \boldsymbol{v}_b + w_b \boldsymbol{v}_a + \boxed{\boldsymbol{v}_a \times \boldsymbol{v}_b}]
\end{aligned}
$$

- Non-commutative:

$$q_a q_b \neq q_b q_a$$

Conjugation

- The quaternion conjugate can be computed by negating the vector part of the quaternion:

$$
\begin{aligned}
q \;\; &= [w, v] = [w, x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}] = w + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} \\
q^* &= [w, -v] = [w, -x\boldsymbol{i} - y\boldsymbol{j} - z\boldsymbol{k}] = w - x\boldsymbol{i} - y\boldsymbol{j} - z\boldsymbol{k}
\end{aligned}
$$

Conjugation

- The quaternion conjugate can be computed by negating the vector part of the quaternion:

$$q \ \ = [w, v] = [w, x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}] = w + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k}$$
$$q^* = [w, -v] = [w, -x\boldsymbol{i} - y\boldsymbol{j} - z\boldsymbol{k}] = w - x\boldsymbol{i} - y\boldsymbol{j} - z\boldsymbol{k}$$

- The product of a quaternion and its conjugate gives:

$$\begin{aligned} qq^* &= [w, \boldsymbol{v}][w, -\boldsymbol{v}] \\ &= [s^2 - \boldsymbol{v} \ \cdot -\boldsymbol{v}, -w\boldsymbol{v} + w\boldsymbol{v} + \boldsymbol{v} \times -\boldsymbol{v}] \\ &= [s^2\boldsymbol{v} \cdot \boldsymbol{v}, \boldsymbol{0}] \\ &= [s^2 + v^2, \boldsymbol{0}] \end{aligned}$$

Norm

- Norm is the length from the origin
- Can also be denoted as magnitude

Norm

- The definition of the norm of a complex number:

$$|z| = \sqrt{a^2 + b^2}$$
$$zz^* = |z|^2$$

Norm

- The definition of the norm of a complex number:

$$|z| = \sqrt{a^2 + b^2}$$
$$zz^* = |z|^2$$

- Similarly, the norm of a quaternion is defined as:

$$q = [w, \boldsymbol{v}]$$
$$|q| = \sqrt{w^2 + v^2}$$

Norm

- The definition of the norm of a complex number:

$$|z| = \sqrt{a^2 + b^2}$$
$$zz^* = |z|^2$$

- Similarly, the norm of a quaternion is defined as:

$$q = [w, \boldsymbol{v}]$$
$$|q| = \sqrt{w^2 + v^2}$$

- Which leads to the norm of a quaternion expressed as:

$$qq^* = |q|^2$$

Normalization

- A quaternion is normalized by dividing it by its norm:

$$q' = \frac{q}{|q|} = \frac{q}{\sqrt{w^2 + v^2}}$$

- A normalized quaternion is denoted as a unit quaternion and has a length of 1

Inverse (division)

- Used for division
- Similar usage as inverse of a rotational matrix
- Denoted as q^{-1}

Inverse

- To compute the inverse of a quaternion, we take the conjugate of the quaternion and divide it by the square of the norm:

$$q^{-1} = \frac{q^*}{|q|^2}$$

Inverse

- Let's show how we get the definition
- By definition of the inverse:

$$qq^{-1} = [1, \mathbf{0}] = 1$$

- And multiply both sides by the conjugate of the quaternion gives:

$$q^* q q^{-1} = q^*$$

- Then by substitution (expression of norm):

$$|q|^2 q^{-1} = q^*$$
$$q^{-1} = \frac{q^*}{|q|^2}$$

Inverse

- To compute the inverse of a quaternion, we take the conjugate of the quaternion and divide it by the square of the norm:

$$q^{-1} = \frac{q^*}{|q|^2}$$

- A unit-norm vector has a norm of 1, so we can write:

$$q^{-1} = \frac{q^*}{1^2} = q^*$$

- An inverse of a unit quaternion is its conjugation

Inverse (division)

- Division done by multiplication with an inverse is non-communal (same as multiplication):

$$q_a^{-1} q_b \neq q_b q_a^{-1}$$

- Similarities to operations with rotational matrices

Example

Quaternion operations

Operations in python

Help (https://petercorke.github.io/spatialmath-python/3d_quaternion.html)

- Example quaternions:

$q_a = [1, [1,0,0]]$ and $q_b = \left[2, \left[0, -2 * \sqrt{3}, 0\right]\right]$ (the quaternions are not normalized)

- Operations:

1. Norm of both $|q|$
2. Unit-norm of both q'
3. Addition/subtraction $q_a +/- q_b$
4. Multiplication $q_a q_b$ and $q_b q_a$ (compare, use quaternion())
5. Division with inverse $q_a^{-1} q_b$ and $q_b q_a^{-1}$ (conjugation, compare, use quaternion(), ./, .\ to check)

Operations in python

- Calculate the relative orientation between the following quaternions

$q_a = [1, [1,0,0]]$ and $q_b = \left[2, \left[0, -2 * \sqrt{3}, 0\right]\right]$

- hint: try out division, inverse and conjugate.

Rotation conversions

Quaternion transformations

- Quaternion can be transformed to other representations:
  - Euler rotation angles $\leftrightarrow$ quaternion
  - Rotation matrix $\leftrightarrow$ quaternion
  - Axis angle $\leftrightarrow$ quaternion

Quaternion to Rotation matrix

- To convert a quaternion to a rotation matrix:

$$R = \begin{bmatrix} w^2 + x^2 - y^2 - z^2 & 2(xy - wz) & 2(xz + wy) \\ 2(xy + wz) & w^2 - x^2 + y^2 - z^2 & 2(yz - wx) \\ 2(xz - wy) & 2(yz + wx) & w^2 - x^2 - y^2 + z^2 \end{bmatrix}$$

Rotation matrix to Quaternion

- It can be performed in many ways
- Shepperd's method, thanks to a voting scheme between four possible solutions, always works far from formulation singularities
- Multiple equations give multiple (four) possible solutions

$$4w^2 = 1 + r_{11} + r_{22} + r_{33}$$
$$4x^2 = 1 + r_{11} - r_{22} - r_{33}$$
$$4y^2 = 1 - r_{11} + r_{22} - r_{33}$$
$$4z^2 = 1 - r_{11} - r_{22} + r_{33}$$
$$4yz = r_{23} + r_{32}$$
$$4xz = r_{31} + r_{13}$$
$$4xy = r_{12} + r_{21}$$
$$4wx = r_{32} - r_{23}$$
$$4wy = r_{13} - r_{31}$$
$$4wz = r_{21} - r_{12}$$

Other conversions

- Conversions to and from Euler angles and axis angles are simple with some restrictions and checks for stability
- Can be checked on: https://www.euclideanspace.com/maths/geometry/rotations/conversions/index.htm

Example

Python example

$$q = \left[2, \left[0, -2 * \sqrt{3}, 0\right]\right]$$

Transform the given quaternion to:

Help(https://petercorke.github.io/spatialmath-python/func_quat.html)

- Euler angles
- rotation matrix

Use Python functions: q2r, r2x (part of *spatialmath.base*)

Recap

Quaternions

- Pros and cons of quaternions w.r.t. other representations:
  - Stable, compact, interpolation, no singularities, fast, easy to convert
- Definition of a quaternion
  - A combination of a scaler and 3D complex vector – 4D sphere
  - Defined as an angle of rotation around a (unit) vector
- Quaternion operations
  - All needed with multiplication (and division) non-communal and inverse being the conjugated for unit quaternions
- Conversion to other representation fairly easy.