Attitude and Heading Reference System

Corrado Santoro

ARSLAB - Autonomous and Robotic Systems Laboratory
Dipartimento di Matematica e Informatica - Università di Catania, Italy
santoro@dmi.unict.it

Robotic Systems

Introduction

- Control of **avionic systems** implies to determine the **attitude** of an aircraft in terms of its **Euler angles**:
  - **Roll**
  - **Pitch**
  - **Yaw**
- Inertial sensors normally used **are not able** to directly sense these angles
- The solution is to adopt special **complementary filters** or **Kalman filters** that are able to perform a *"sensor fusion"* among the various intertial data in order to esimate the attitude of the aircraft

Sensors

Gyroscopes

- They measure the **angular speeds (rates)** along the roll, pitch and yaw axis
- By means of **numeric integration** we can approximately compute the roll, pitch and yaw angles

Accelerometers

- They measure the **linear acceleration** along the roll, pitch and yaw axis
- They measure mechanical solicitations but also the **gravity vector**
- If we compute the **inclination** of the g vector measured, we can estimate **roll** and **pitch**

Magnetometers

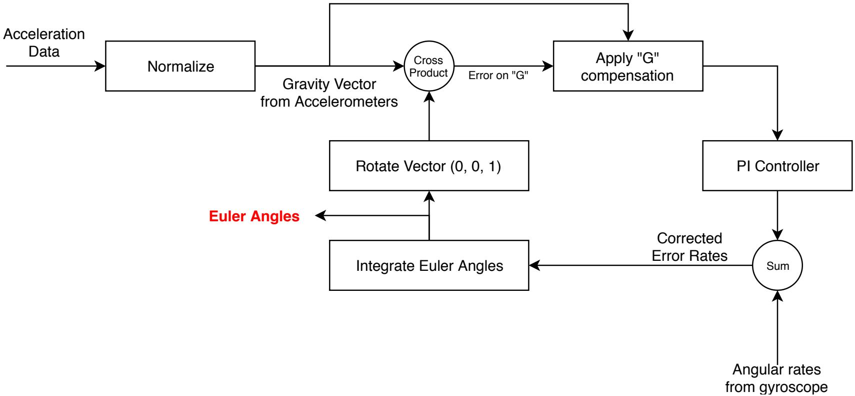
- They measure the **Earth magnetic field** along roll, pitch and yaw axis
- By computing the **inclination** of the measured vector w.r.t. the magnetic north, we can estimate the **yaw**

AHRS and Filters

Approaches used for AHRS

- **PI Controller:** Complementary Filter
- **Kalman Gain:** Kalman Filter

AHRS: the Complementary Filter



Algoritmo di Sensor Fusion

AHRS: Attitude and Heading Reference System (caso Roll e Pitch)

while *True* **do**

On each ΔT;

$\{ax, ay, az\} \leftarrow read\_accelerometers();$

$\{gx, gy, gz\} \leftarrow read\_gyroscopes();$

$/ * Rotate\ Gravity\ Vector\ using\ \phi, \theta, \psi * /$

$\{gravX, gravY, gravZ\} \leftarrow rotate\_vector(\{0, 0, 1\});$

$/ * Normalize\ Acceleration\ Data * /$

$\{ax, ay, az\} \leftarrow \frac{\{ax, ay, az\}}{||\{ax, ay, az\}||};$

$/ * Compute\ Error\ using\ Cross\ Product * /$

$\{ex, ey, ez\} \leftarrow \{ax, ay, az\} \times \{gravX, gravY, gravZ\};$

$/ * Apply\ PI\ Controller\ to\ each\ element\ of\ the\ error\ vector * /$

$\{corrX, corrY, corrZ\} \leftarrow PI\_control(\{ex, ey, ez\});$

$/ * Correct\ Gyro\ Measures * /;$

$\{gx, gy, gz\} \leftarrow \{gx, gy, gz\} + \{corrX, corrY, corrZ\};$

$/ * Update\ angles\ using\ integration * /$

$\{\phi, \theta, \psi\} \leftarrow \{\phi, \theta, \psi\} + \{gx, gy, gz\}\Delta T;$

end

Dynamic Compensation of the Accelerations

- When a rigid body is subject to mechanical solicitations, accelerometers will measures both the g vector and such solicitations
- But they are "noise" with respect to the measurement algorithm
- We note that:
  - In **static conditions**, we have $\sqrt{ax^2 + ay^2 + az^2} = g$
  - In case of **solicitations**, we have $\sqrt{ax^2 + ay^2 + az^2} \neq g$
- The g-compensation is applied by weighting the gyroscope correction on the basis of the difference:

$$\sqrt{ax^2 + ay^2 + az^2} - g$$

Computational Aspects of the Sensor Fusion Algorithm

while *True* **do**

  On each ΔT;

  ...

  $\{gravX, gravY, gravZ\} \leftarrow rotate\_vector(\{0, 0, 1\})$;

  ...

end

The most critical part is the rotation of the vector $\{0, 0, 1\}$

Rotation of a Vector in R^3

Rotation along x axis, **roll**

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = R_\phi \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Rotation of a Vector in R^3

Rotation along y axis, **pitch**

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = R_\theta \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Rotation of a Vector in R^3

Rotation along z axis, **yaw**

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = R_\psi \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Rotazione di un Vettore in R^3

Rotazione intorno tutti gli assi usando **roll, pitch e yaw**

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R_\phi R_\theta R_\psi \begin{bmatrix} x \\ y \\ z \end{bmatrix} = DCM \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Direction Cosine Matrix

The DCM is the **rotation matrix** of a rigid body whose attitude is expressed with Euler angles θ, ϕ, ψ

Direction Cosine Matrix

$$\begin{bmatrix} cos\theta cos\psi & sin\phi sin\theta cos\psi - cos\phi sin\psi & cos\phi sin\theta cos\psi + sin\phi sin\psi \\ cos\theta sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & cos\phi sin\theta sin\psi - sin\phi cos\psi \\ -sin\theta & sin\phi cos\theta & cos\phi cos\theta \end{bmatrix}$$

For each step of the algorithm, we should compute the DCM

Definition

A **quaternion** is a **complex number** with four components, a real part and three imaginary parts:

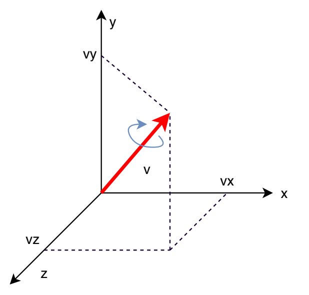
$$q = \{q_0, q_1, q_2, q_3\} = q_0 + q_1 i + q_2 j + q_3 k$$

i, j e k are imaginary units and are characterised by the following properties:

$$\begin{array}{ccc} i^2 = -1 & j^2 = -1 & k^2 = -1 \\ -ij = ij = k & -jk = kj = i & -ki = ik = j \end{array}$$

Quaternions obey to an algebra in with the classical operations are defined: (algebraic) sum, products, ration, norm, etc.

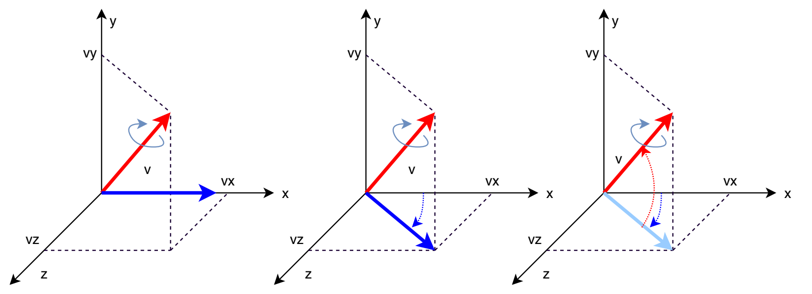
Quaternions and Rotations



Any vector $v = \{v_x, y_y, v_z\}$ in R^3, and **rotated** (on itself) of an angle α can be represented by a **quaternion**:

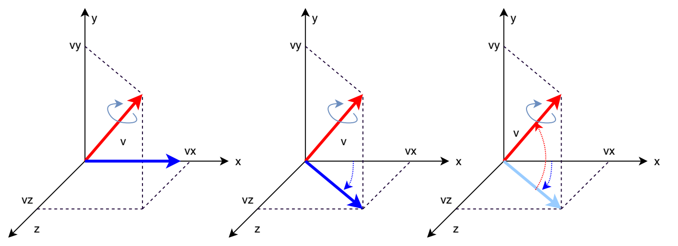
$$q = \{cos\frac{\alpha}{2}, v_x sin\frac{\alpha}{2}, v_y sin\frac{\alpha}{2}, v_z sin\frac{\alpha}{2}\}$$

Quaternions and Rotations



Any vector $v = \{v_x, y_y, v_z\}$ in R^3, and **rotated** (on itself) of an angle α can be represented as the vector $\{||v||, 0, 0\}$ to which **two rotations** (two angles) plus the angle α are applied, $\rightarrow$ **the Euler angles.**

Quaternions and Rotations



$$q = \{q_0, q_1, q_2, q_3\}$$

If we consider a **unit vector**, a quaternion can represent that vector rotated according the Euler angles
$\rightarrow$**a quaternion is an alternative representation of Euler angles** ϕ, θ, ψ

Quaternions e DCM

$$q = \{q_0, q_1, q_2, q_3\}$$

The DCM can be computed by a quaternion as::

$$\begin{bmatrix} cos\theta cos\psi & sin\phi sin\theta cos\psi - cos\phi sin\psi & cos\phi sin\theta cos\psi + sin\phi sin\psi \\ cos\theta sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & cos\phi sin\theta sin\psi - sin\phi cos\psi \\ -sin\theta & sin\phi cos\theta & cos\phi cos\theta \end{bmatrix}$$

$$\begin{bmatrix} q_0^2 - q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_3 q_2 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_2 - q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

Rotation of the gravity vector

The gravity vector $\{0, 0, 1\}$ can be rotated by means of a multiplication with the rotation matrix

The rotated vector is the **third column** of the rotation matrix:

$$\begin{bmatrix} cos\theta cos\psi & sin\phi sin\theta cos\psi - cos\phi sin\psi & cos\phi sin\theta cos\psi + sin\phi sin\psi \\ cos\theta sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & cos\phi sin\theta sin\psi - sin\phi cos\psi \\ -sin\theta & sin\phi cos\theta & cos\phi cos\theta \end{bmatrix}$$

$$\begin{bmatrix} q_0^2 - q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 + q_0 q_3) & 2(q_1 q_3 - q_0 q_2) \\ 2(q_1 q_2 - q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_3 q_2 + q_0 q_1) \\ 2(q_1 q_3 + q_0 q_2) & 2(q_2 q_2 - q_0 q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

By using a quaternion, trigonometric functions are not needed

Integration of a quaternion

When Euler angles are represented by a quaternion q, we must update it on the basis of the angular speeds $\{\dot{\phi}, \dot{\theta}, \dot{\psi}\}$ provided by gyroscopes

By using the **derivation rules** of quaternions, we have:

$$q = q + \frac{\Delta T}{2} \begin{bmatrix} 0 & -\dot{\phi} & -\dot{\theta} & -\dot{\psi} \\ \dot{\phi} & 0 & \dot{\psi} & -\dot{\theta} \\ \dot{\theta} & -\dot{\psi} & 0 & \dot{\phi} \\ \dot{\psi} & \dot{\theta} & -\dot{\phi} & 0 \end{bmatrix} q$$

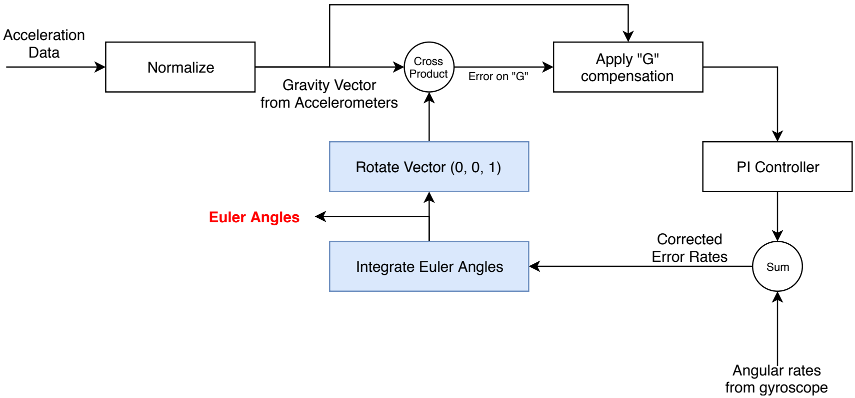
Quaternions

From quaternions to Euler angles

The equivalence quaternion/rotation matrix allows us to determine the formulas to compute Euler angles from a quaternion:

$$\begin{aligned}
\phi &= \tan^{-1} \frac{2(q_0 q_1 + q_2 q_3)}{q_0^2 - q_1^2 - q_2^2 + q_3^2} \\
\theta &= \sin^{-1} 2(q_0 q_2 + q_1 q_3) \\
\psi &= \tan^{-1} \frac{2(q_1 q_2 + q_0 q_3)}{1 - 2(q_2^2 + q_3^2)}
\end{aligned}$$

AHRS e Quaternioni



Quaternions are used in the cyan blocks

Attitude and Heading Reference System

Corrado Santoro

ARSLAB - Autonomous and Robotic Systems Laboratory
Dipartimento di Matematica e Informatica - Università di Catania, Italy
santoro@dmi.unict.it

Robotic Systems