Un algoritmo per il calcolo dell' 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

Programmazione Sistemi Robotici

Introduzione

- Il controllo dei **sistemi avionici** richiede la determinazione dell'**assetto** di un velivolo in termini dei suoi **angoli di Eulero**:
  - **Roll**
  - **Pitch**
  - **Yaw**
- I sensori inerziali normalmente usati in questi sistemi **non sono in grado** di acquisire direttamente i dati su tali angoli
- Vengono pertanto adottati dei **filtri complementari** o **filtri di Kalman** in grado di effettuare una *"sensor fusion"* tra i dati dei sensori inerziali e stimare l'assetto del velivolo

Gyroscopi

- Misurano le **velocità angolari** lungo gli assi di roll, pitch e yaw
- Attraverso l' **integrazione numerica** è possibile effettuare un calcolo approssimato di roll, pitch e yaw

Accelerometri

- Misurano le **accelerazioni lineari** lungo gli assi di roll, pitch e yaw
- Essi misurano le sollecitazioni meccaniche ma anche il **vettore dell'accelerazione di gravità**
- Calcolando l'**inclinazione** del vettore *g* misurato è possibile stimare **roll** e **pitch**

Magnetometri

- Misurano il vettore del **campo magnetico terrestre** lungo gli assi di roll, pitch e yaw
- Calcolando l'**inclinazione** del vettore misurato rispetto al nord magnetico è possibile stimare l'angolo di **yaw**

Algoritmo di Sensor Fusion

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

while *True* **do**

On each ΔT;

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

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

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

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

$/* \; Normalize\ Acceleration\ Data \; */$

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

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

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

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

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

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

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

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

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

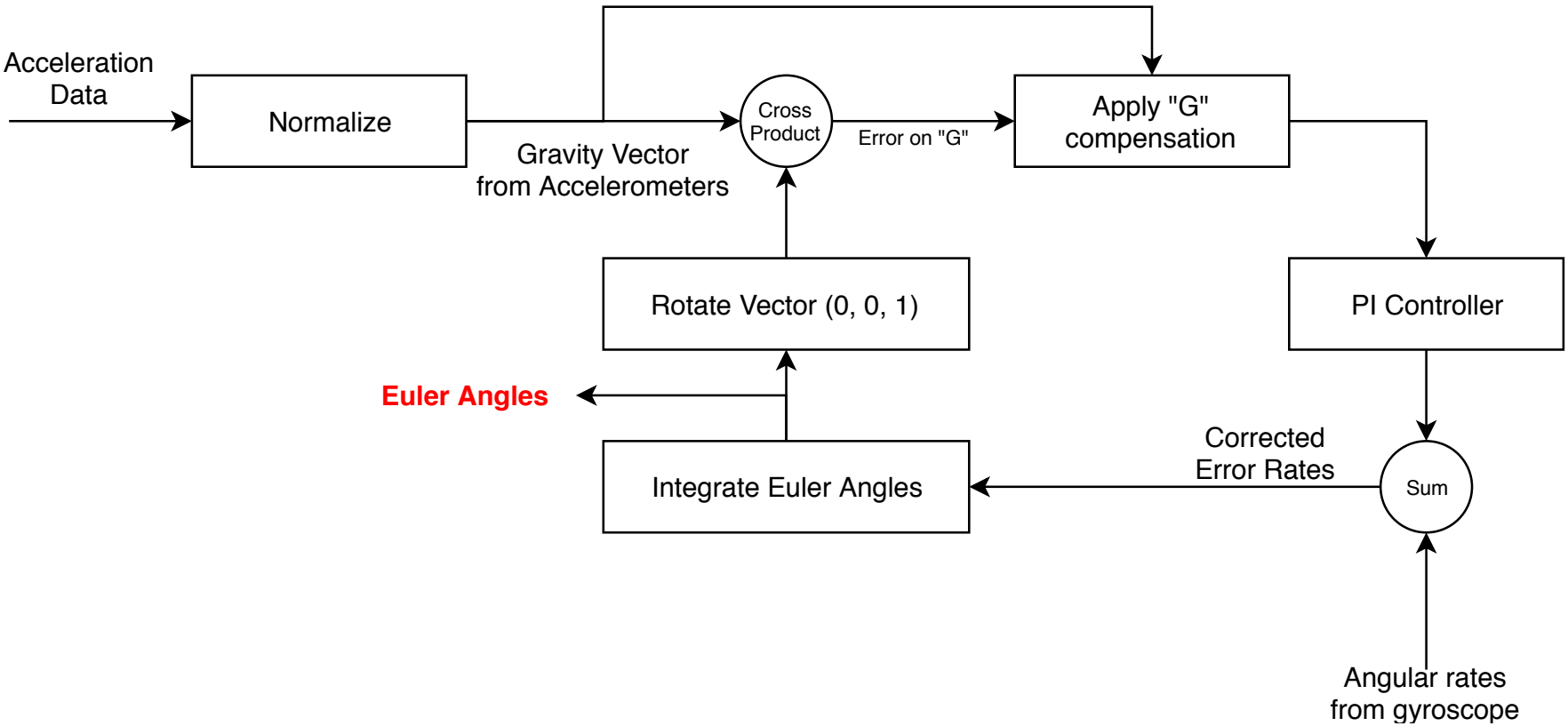
end

Compensazione Dinamica delle Accelerazioni Indotte

- Qualora il corpo sia soggetto a sollecitazioni meccaniche, gli accelerometri, resistuiranno oltre al vettore g, anche le misure relative a tali sollecitazioni
- Esse sono però "disturbi" che inficiano l'algoritmo di misura
- Tuttavia notiamo che:
  - In condizioni di **staticità**, si ha $\sqrt{ax^2 + ay^2 + az^2} = g$
  - In condizioni di **sollecitazioni**, si ha $\sqrt{ax^2 + ay^2 + az^2} \neq g$
- La compensazione su g si applica dunque "pesando meno" la correzione sui giroscopi sulla base della differenza:

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

AHRS: Attitude and Heading Reference System



Approcci usati per l'algoritmo di AHRS

- **Controllore PI:** Filtro Complementare
- **Kalman Gain:** Filtro di Kalman

Aspetti Computazionali dell'algoritmo di Sensor Fusion

while *True* **do**
    On each ΔT;
    ...
    $\{gravX, gravY, gravZ\} \leftarrow rotate\_vector(\{0, 0, 1\})$;
    ...
end

La parte più critica computazionalmente
è la rotazione del vettore $\{0, 0, 1\}$

Rotazione di un Vettore in R^3

Rotazione intorno asse x, Angolo di **roll**

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

Rotazione di un Vettore in R^3

Rotazione intorno asse y, Angolo di **pitch**

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

Rotazione di un Vettore in R^3

Rotazione intorno asse z, Angolo di **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

La DCM è la **matrice di rotazione** di un corpo rigido il cui assetto è espresso tramte gli angolo di Eulero θ, ϕ, ψ

Direction Cosine Matrix

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

Ad ogni step dell'algoritmo, occorre calcolare la DCM

Definizioni

Un **quaternione** è un **numero complesso** a quattro componenti, dotato di una parte reale e tre parti immaginarie:

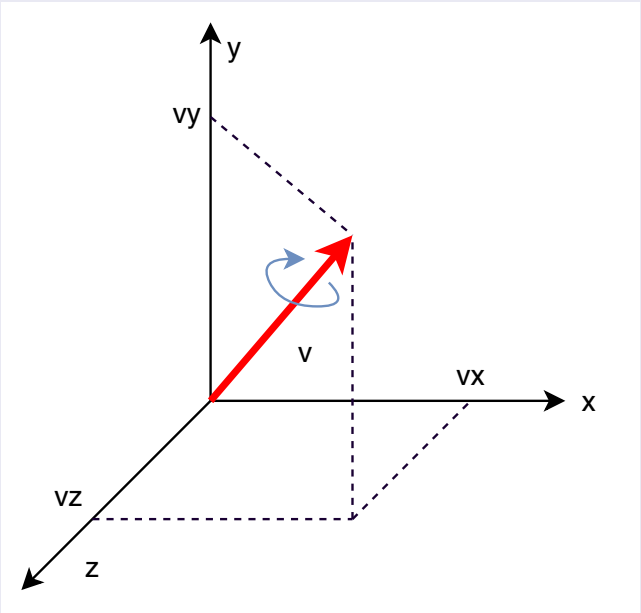
$$q = \{q_0, q_1, q_2, q_3\} = q_0 + q_1 i + q_2 j + q_3 k$$

i, j e k sono unità immaginarie e sono caratterizzate dalle seguenti proprietà:

$$i^2 = -1 \qquad j^2 = -1 \qquad k^2 = -1$$
$$-ij = ij = k \quad -jk = kj = i \quad -ki = ik = j$$

I quaternioni obbediscono ad un'algebra in cui sono definite le tipiche operazioni di somma (algebrica), prodotto, rapporto, norma, etc.

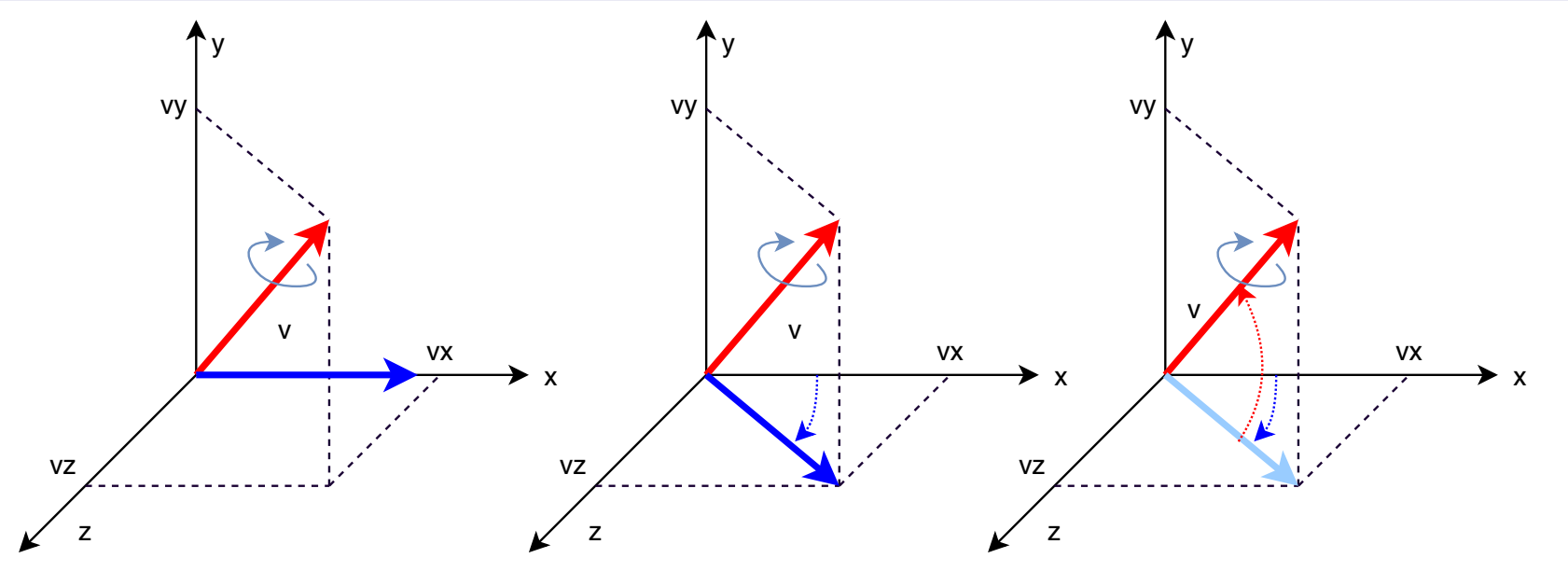
Quaternioni e Rotazioni



Un qualunque vettore $v = \{v_x, y_y, v_z\}$, definito nello spazio R^3, e **ruotato** (su se stesso) di un angolo α può essere rappresentato con un **quaternione**:

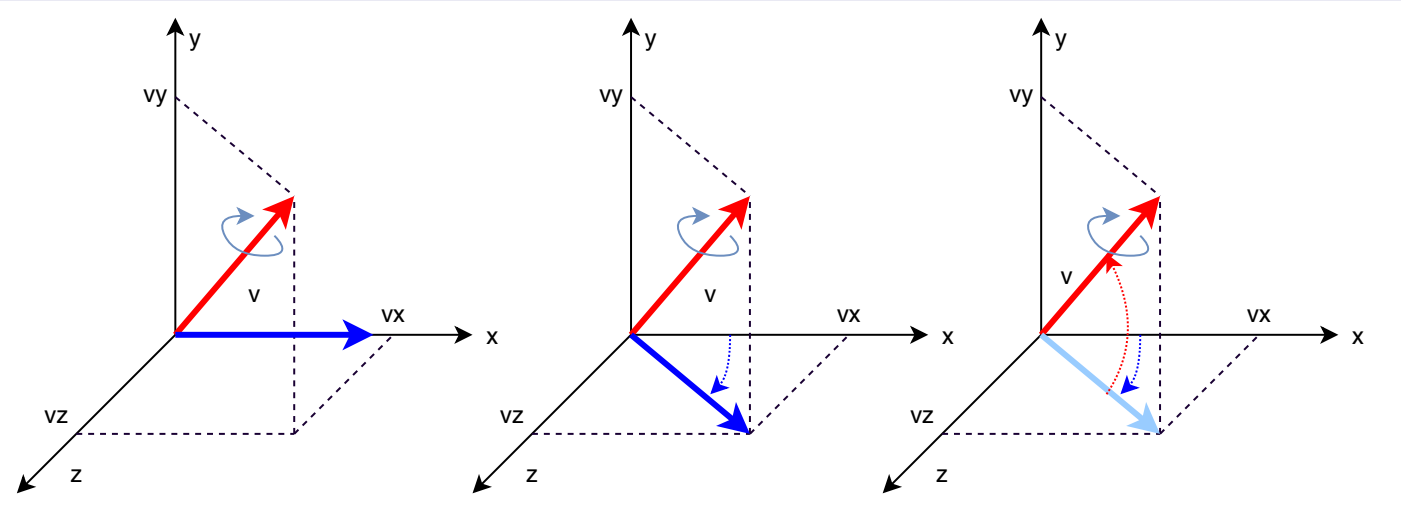
$$q = \{cos\frac{\alpha}{2}, v_x sin\frac{\alpha}{2}, v_y sin\frac{\alpha}{2}, v_z sin\frac{\alpha}{2}\}$$

Quaternioni e Rotazioni



Un qualunque vettore $v = \{v_x, y_y, v_z\}$, definito nello spazio R^3, e **ruotato** (su se stesso) di un angolo α può essere rappresentato come il vettore $\{||v||, 0, 0\}$ a cui applichiamo **due rotazioni** (due angoli) più l'angolo $\alpha \rightarrow$ **angoli di Eulero**.

Quaternioni e Rotazioni



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

Se consideriamo un vettore di modulo unitario, un quaternione può rappresentare il vettore ruotato secondo gli angoli di Eulero
$\rightarrow$**Un quaternione è una rappresentazione alternativa degli angoli di Eulero** ϕ, θ, ψ.

Quaternioni e DCM

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

Da un quaternione è possibile ricavare facimente la **matrice di rotazione equivalente**:

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

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

Rotazione del vettore di gravità

Il vettore di gravità $\{0, 0, 1\}$ può essere dunque ruotato moltiplicandolo per la matrice di rotazione.

Il vettore ruotato corrisponde alla **terza colonna** della matrice di rotazione:

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

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

Usando i quaternioni non è necessario ricorrere al calcolo di funzioni trigonometriche

Integrazione di un Quaternione

Se i nostri angolo di Eulero sono rappresentati da un quaternione q, occorre capire come aggiornarlo sulla base delle misure delle velocità angolari $\{\dot{\phi}, \dot{\theta}, \dot{\psi}\}$ fornite dai giroscopi.

A tale scopo, utilizzando le regole di **derivazione** dei quaternioni, otteniamo:

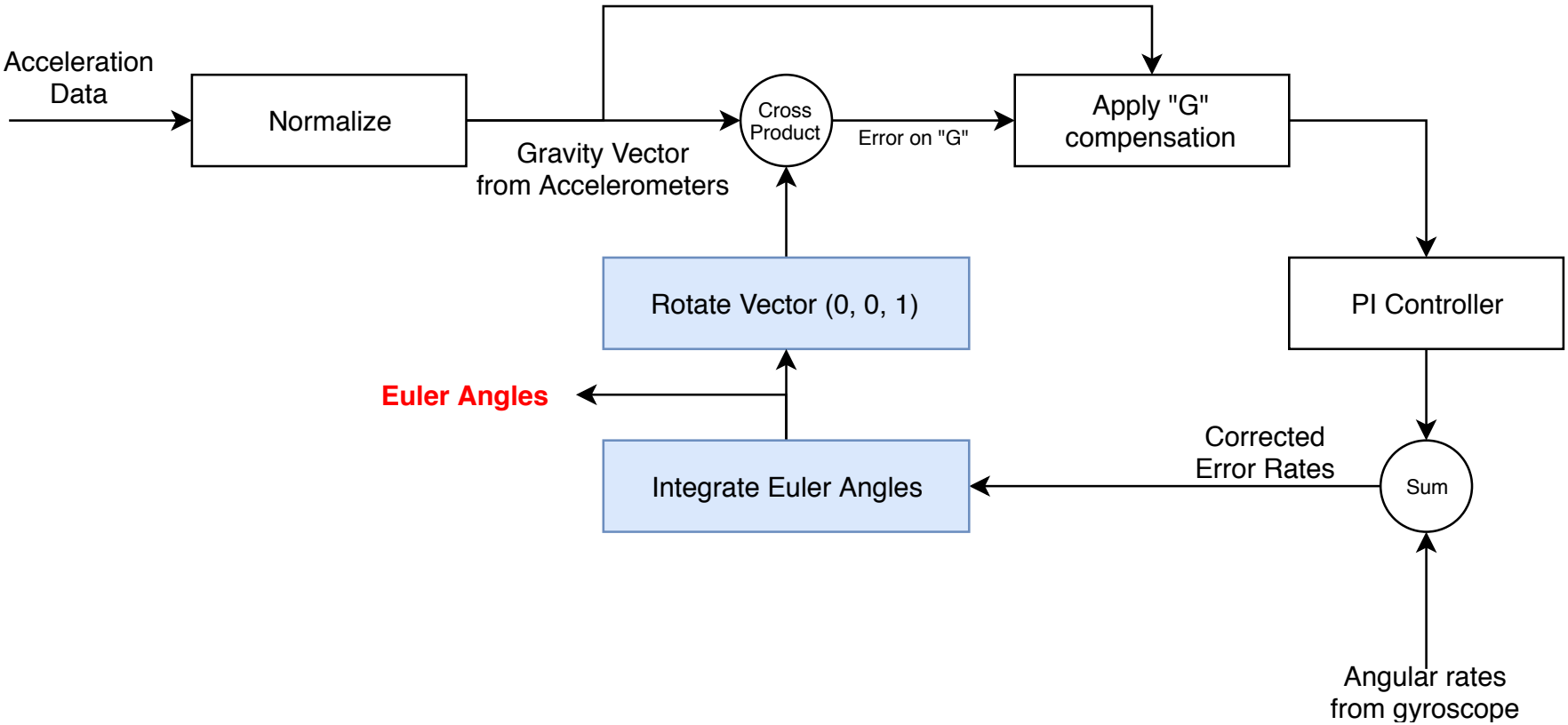
$$q = q + \frac{\Delta T}{2} \begin{bmatrix} 0 & -\dot{\phi} & -\dot{\theta} & -\dot{\psi} \\ \dot{\phi} & 0 & \dot{\psi} & -\dot{\theta} \\ \dot{\theta} & -\dot{\psi} & 0 & \dot{\phi} \\ \dot{\psi} & \dot{\theta} & -\dot{\phi} & 0 \end{bmatrix} q$$

Da quaternione a angoli di Eulero

L'equivalenza quaternione/matrice di rotazione ci permette di determinare le formule per calcolare gli angoli di Eulero da un quaternione:

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

AHRS e Quaternioni



I quaternioni sono utilizzati nei blocchi in azzurro

Un algoritmo per il calcolo dell' 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

Programmazione Sistemi Robotici