



Embedded systems with drones – Hands-on lecture 4

Project-based Learning Center | ETH Zurich

Tommaso Polonelli tommaso.polonelli@pbl.ee.ethz.ch

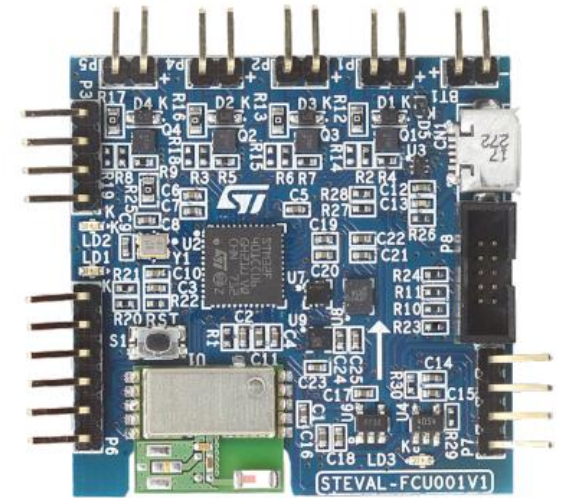
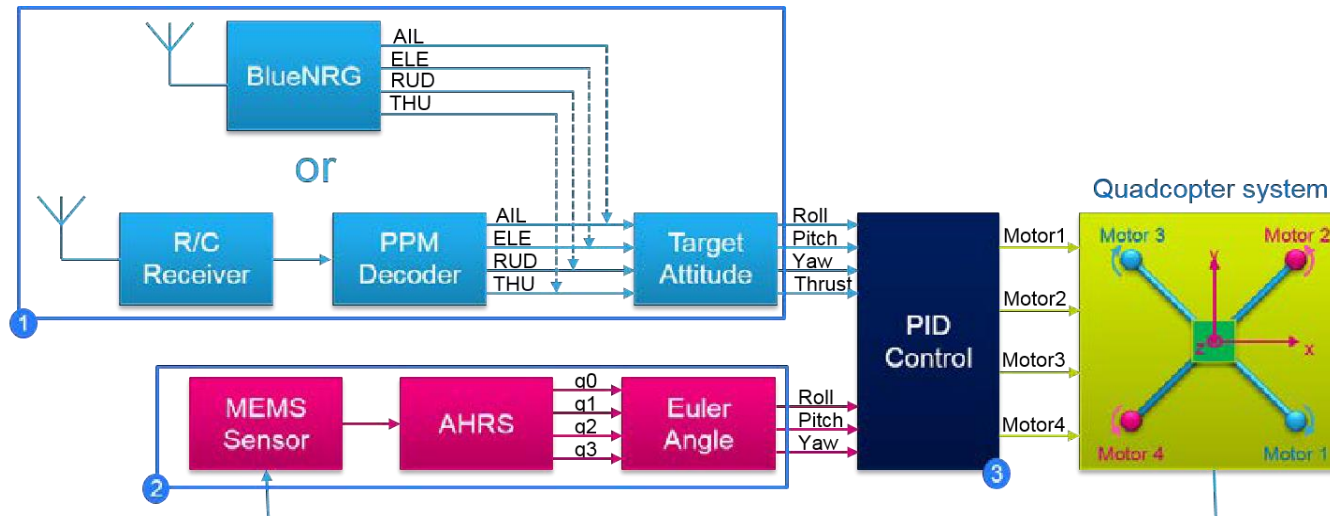
Hanna Müller hanmuell@iis.ee.ethz.ch

Vlad Niculescu vladn@iis.ee.ethz.ch

STEVAL – State Estimator

Exercise goals

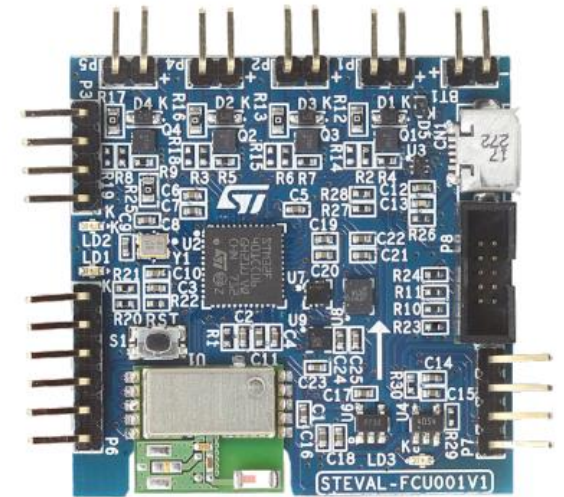
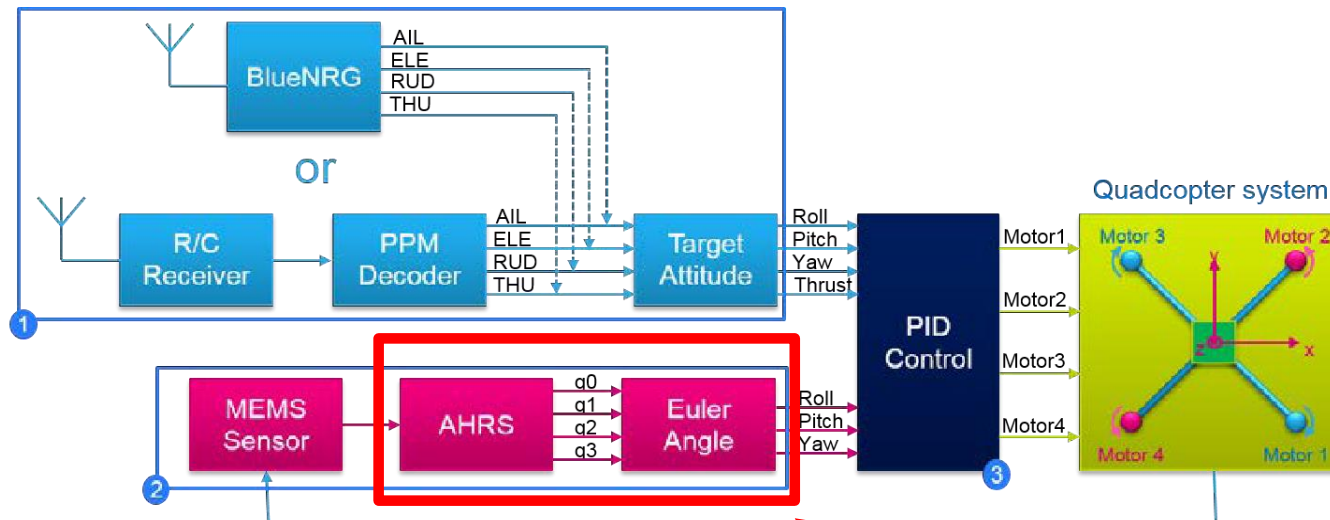
- Understand how the estimator (Mahony filter) of the STEVAL's firmware works
- Understanding how the orientation of an object can be mathematically represented and calculated using accelerometer and gyroscope data
- Understand the differences between this estimator and the EKF



STEVAL – State Estimator

Exercise goals

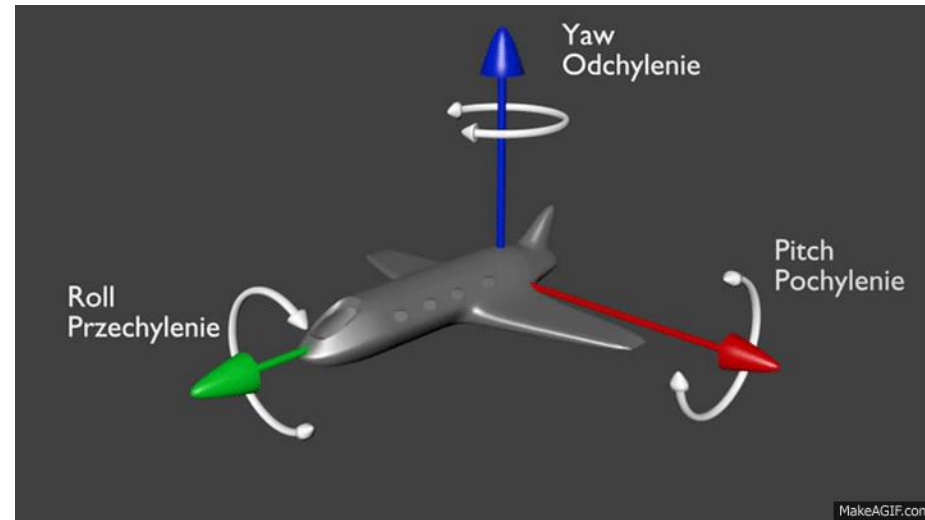
- Understand how the estimator (Mahony filter) of the STEVAL's firmware works
- Understanding how the orientation of an object can be mathematically represented and calculated using accelerometer and gyroscope data
- Understand the differences between this estimator and the EKF



Attitude Representation

STEVAL's firmware only implements an attitude estimator
System state:

$$\mathbf{x} = \begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} \quad \begin{array}{l} \text{Yaw} \\ \text{Pitch} \\ \text{Roll} \end{array}$$

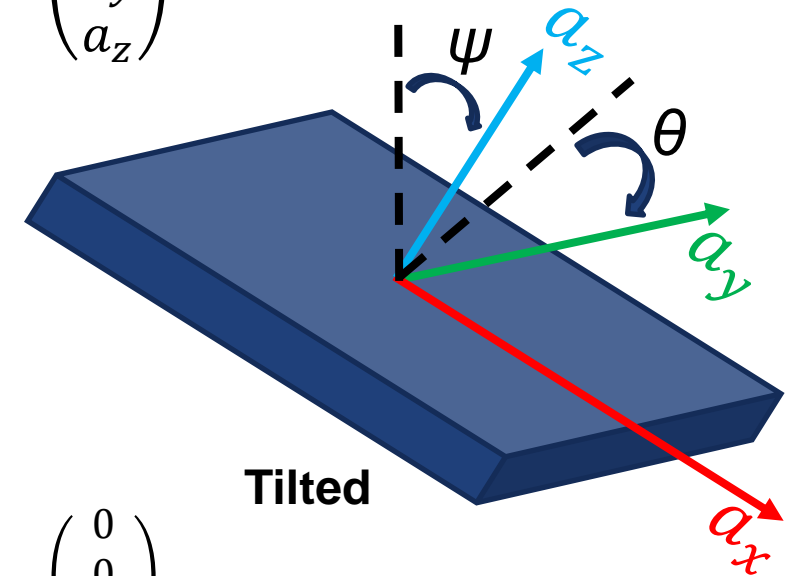
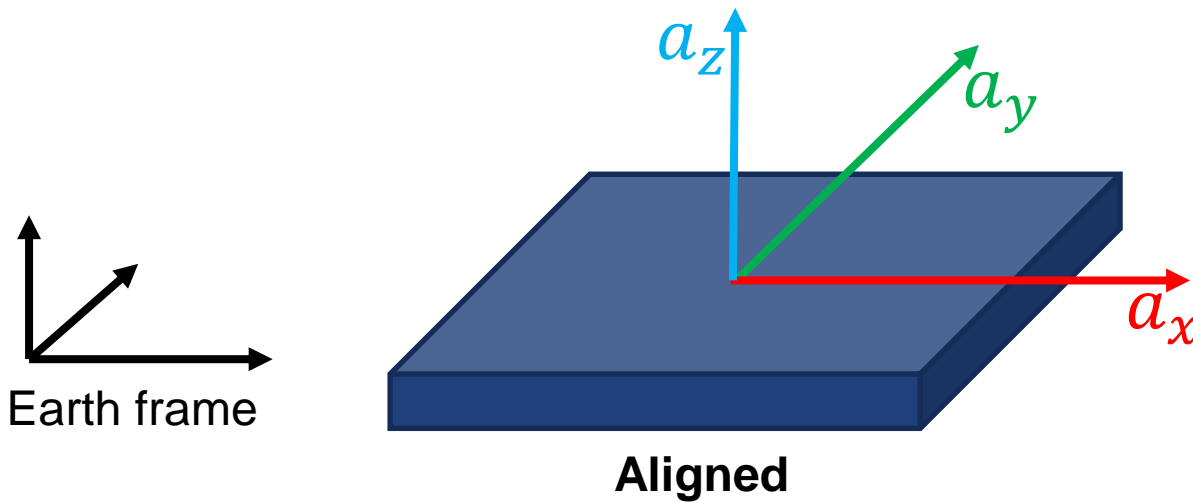


- The representation used by the AHRS module is the Euler angle system.
- They are a series of three angles (ψ, θ, ϕ) representing rotation around three axes, in sequence.
- A variety of Euler angle systems exist, the difference being the rotations order
- Our system rotates first about the z axis by the angle ψ , about the y axis by θ , and about the x axis by ϕ .

Sensor Output

STEVAL uses Accelerometer and Gyroscope for attitude estimation.

Accelerometer: provides the 3D acceleration vector *in the body frame* $\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$



- When stationary and aligned to the earth frame, the accelerometer's output is $\begin{pmatrix} 0 \\ 0 \\ -g \end{pmatrix}$
- Knowing the acceleration at $\phi=\theta=0$, by geometry we calculate roll (ϕ) and pitch (θ) from the misalignment with the gravity

$$\theta = \tan^{-1} \frac{-a_x}{\sqrt{a_y^2 + a_z^2}} \quad \psi = \tan^{-1} \frac{a_y}{a_z}$$

Sensor Output

STEVAL uses Accelerometer and Gyroscope for attitude estimation.

Gyroscope: provides the 3D angular velocity vector *in the body frame* $\begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}$

Euler angles are obtained by integration

$$\begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} \leftarrow \begin{pmatrix} \phi + \Delta_t \cdot \omega_x \\ \theta + \Delta_t \cdot \omega_y \\ \psi + \Delta_t \cdot \omega_z \end{pmatrix}$$

Accelerometer { Roll
Pitch

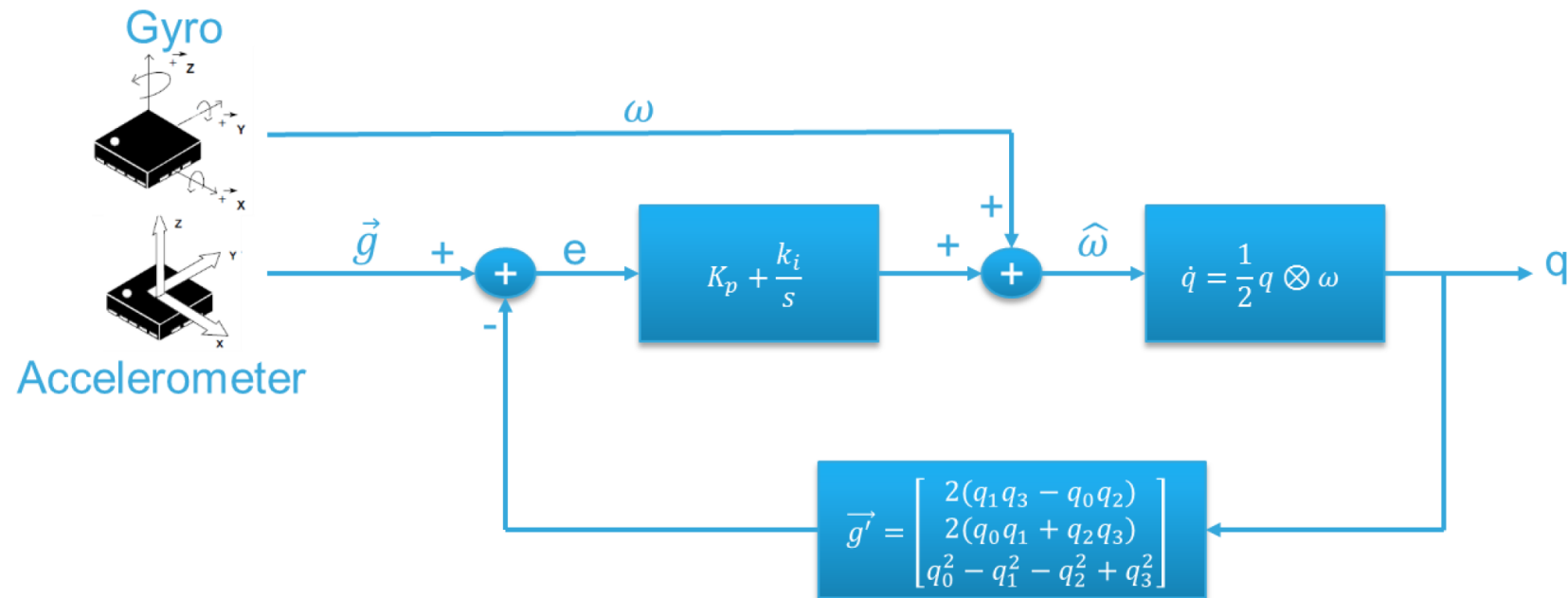
Gyroscope { Roll (derivative)
Pitch (derivative)
Yaw (derivative)

Magnetometer { Yaw



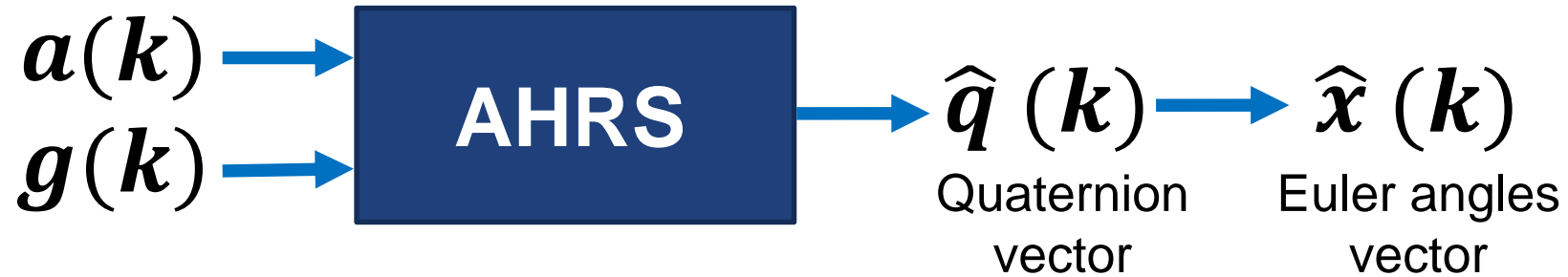
Sensor Output

- STEVAL does not use EKF because it has higher complexity
- It uses a more lightweight state estimator, called Mahony filter



- Accelerometer is always influenced by high-frequency noise, and gyroscope has a low-frequency offset.
- Gyroscope results are accurate in short-term, but generate an offset over time.
- Accelerometer is not accurate in short-term because of noise, but is reliable over time.

Evaluation board – STEVAL-FCU001V1



Raw data from accelerometer and gyro

$$a(k) = \begin{pmatrix} a_x(k) \\ a_y(k) \\ a_z(k) \end{pmatrix} \quad g(k) = \begin{pmatrix} g_x(k) \\ g_y(k) \\ g_z(k) \end{pmatrix}$$

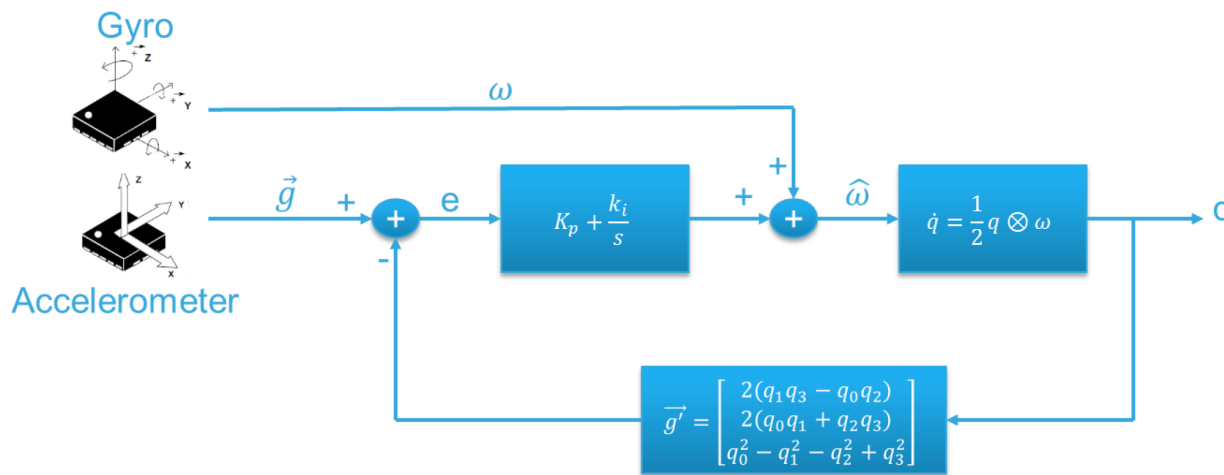
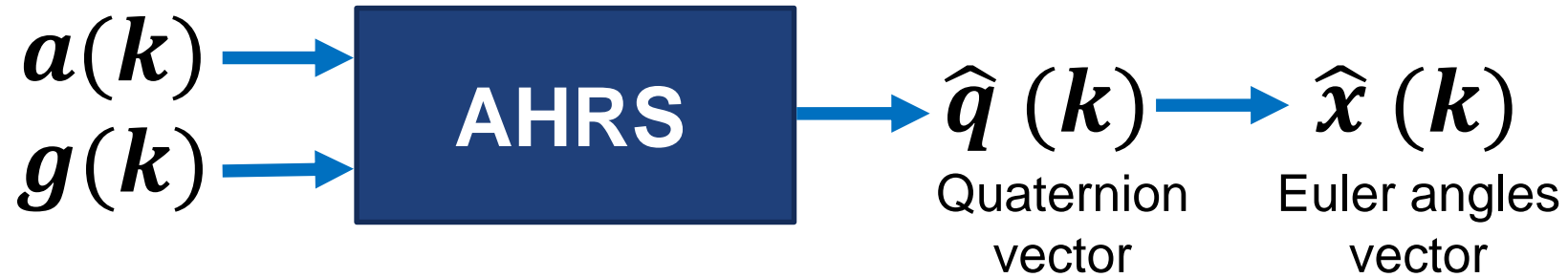
Quaternion estimate

$$\hat{q}(k) = \begin{pmatrix} q_x(k) \\ q_y(k) \\ q_z(k) \\ q_w(k) \end{pmatrix}$$

Algorithm steps (Mahony Filter):

1. Collect $a(k)$, $g(k)$
2. Estimate gravity vector G from $\hat{q}(k-1)$
3. Calculate error vector $e = G \times a(k)$
4. $e_{INT} = e_{INT} + k_I \cdot e \cdot \Delta_t$
5. $g^*(k) \leftarrow g(k) + k_P \cdot e + e_{INT}$
6. Update $\hat{q}(k)$ knowing $g^*(k) \cdot \Delta_t$

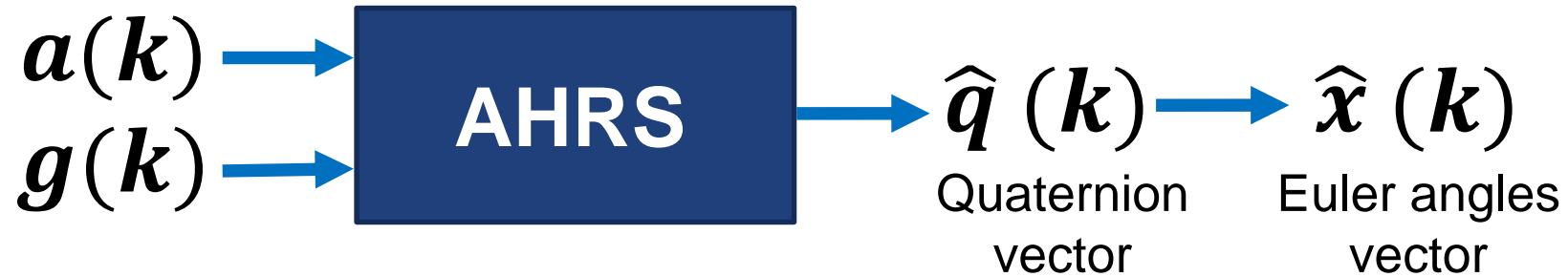
Evaluation board – STEVAL-FCU001V1



Algorithm steps (Mahony Filter):

1. Collect $a(k)$, $g(k)$
2. Estimate gravity vector G from $\hat{q}(k-1)$
3. Calculate error vector $e = G \times a(k)$
4. $e_{INT} = e_{INT} + k_I \cdot e \cdot \Delta_t$
5. $g^*(k) \leftarrow g(k) + k_P \cdot e + e_{INT}$
6. Update $\hat{q}(k)$ knowing $g^*(k) \cdot \Delta_t$

Evaluation board – STEVAL-FCU001V1



Algorithm steps (Mahony Filter):

1. Collect $a(k)$, $g(k)$
2. Estimate gravity vector G from $\hat{q}(k-1)$
3. Calculate error vector $e = G \times a(k)$
4. $e_{INT} = e_{INT} + k_I \cdot e \cdot \Delta_t$
5. $g^*(k) \leftarrow g(k) + k_P \cdot e + e_{INT}$
6. Update $\hat{q}(k)$ knowing $g^*(k) \cdot \Delta_t$

Algorithm steps (EKF):

1. Collect $a(k)$, $g(k)$
2. Prediction step:
 $\hat{q}_p(k)$ from $\hat{q}_m(k-1)$ and $g(k) \cdot \Delta_t$
3. Update step:
 $\hat{q}_m(k)$ from $\hat{q}_p(k)$ and $a(k)$

Project proposal



Student Project:

Implement an alternative version for the drone based on the EKF:

- you will have to implement the 3-state attitude estimator
- compare the performance with the existing Mahoney filter



LAB4: Exercise overview

Template can be found in your Polybox folder LAB4

Exercise	Assignment	Concept
Exercise 1	In the file <i>main.c</i> , identify the functions that update the quaternion and convert it to Euler angles.	Understanding the code
Exercise 2	In the same file, print the Euler angles in the serial console right after they are updated.	AHRS, State estimator
Exercise 3	Analyze the implementation of the attitude estimator in <i>ahrs.c</i> and identify the code associated to each step of the algorithm in Slide 8.	Understanding the code
Exercise 4	Modify the variable AHRS_KI found in <i>ahrs.h</i> to 1.0f. What do you observe in the printed attitude?	AHRS, State estimator
Exercise 5	Comment out the correction steps (2-5 in Slide 8) of the gyroscope data. What do you observe? Hint: you can simply replace $\mathbf{g}^*(\mathbf{k})$ with $\mathbf{g}(\mathbf{k})$ in Step 6.	AHRS, State estimator

LAB4: Exercise overview

Template can be found in your Polybox folder LAB4

Exercise	Assignment	Concept
Exercise 6 (advanced)	Evaluate the computation time and the number of cycles necessary to run one iteration of the AHRS.	
Exercise 7 (advanced)	Use the print procedure learned in the LAB3 to log the output of the AHRS over time.	
Exercise 8 (advanced)	Log the roll and pitch as in Lab 3 and compare the results with the ones provided by the AHRS. What do you observe?	
Exercise 9 (advanced)	Using Python, plot the information acquired before (roll and pitch) on the same plot for a better visualization.	

Hint: Getting the elapsed number of cycles

```
// Defining the counter register address. Add this in the beginning of the main.c
```

```
#define ARM_CM_DEMCR      (*(uint32_t *)0xE000EDFC)
#define ARM_CM_DWT_CTRL  (*(uint32_t *)0xE0001000)
#define ARM_CM_DWT_CYCCNT (*(uint32_t *)0xE0001004)
```

```
// Enable the counter. Add this in the main() function before the “while” loop
```

```
if (ARM_CM_DWT_CTRL != 0) {    // See if DWT is available
    ARM_CM_DEMCR |= 1 << 24; // Set bit 24
    ARM_CM_DWT_CYCCNT = 0;
    ARM_CM_DWT_CTRL |= 1 << 0; // Set bit 0 }
```

```
// Count the number of cycles
```

```
ARM_CM_DWT_CYCCNT = 0;
t1 = ARM_CM_DWT_CYCCNT;
run_your_function();
t2 = ARM_CM_DWT_CYCCNT;
delta = t2 - t1;
```