DT0058
Design tip

Computing tilt measurement and tilt-compensated e-compass

By Andrea Vitali

Main components	
LSM303AGR	Ultra compact high-performance e-compass: ultra-low-power 3D accelerometer and 3D magnetometer module
LSM303C	Ultra compact high-performance e-compass: 3D accelerometer and 3D magnetometer module
LSM303D	Ultra compact high-performance e-compass: 3D accelerometer and 3D magnetometer module

Purpose and benefits

This design tip explains how to compute tilt (Roll and Pitch angles) from accelerometer data. It also explains how to compute e-compass (Yaw angle), from tilt-compensated magnetometer data. The conversion from Euler angles to Quaternions is also shown.

Benefits:

- Added functionality with respect to data fusion provided by osxMotionFX library which provides 9-axis Acc+Mag+Gyro and 6-axis Acc+Gyro fusion but not 6-axis Acc+Mag.
- Reduction of firmware footprint with respect to using the full-blown data fusion provided by osxMotionFX library - see Open.MEMS in design Support Material paragraph.
- Short essential implementation, which enables easy customization and enhancement by the end-user (osxMotionFX is available only in binary format, not as source code)
- Easy to use on every microcontroller (osxMotionFX can only be run on STM32 and only when the proper license has been issued by Open.MEMS license server).

Description

Step 1: Computation of Phi (roll angle, also known as bank; see figure 1 for reference) based on accelerometer data (Gx, Gy and Gz; unit of measure does not matter)

Roll: Phi = Atan2(Gy, Gz)

Note: if Theta = +/-90 deg, then Gy and Gz are near-zero and Phi is unstable; if stability is desired, then Gz can be substituted by Gz+Gx*alpha, with alpha = 0.01-0.05.

Step 2: Computation of Theta (pitch angle, also known as attitude; see figure 1 for reference) based on accelerometer data (Gx, Gy and Gz; unit of measure does not matter)

Gz2 = Gy * Sin(Phi) + Gz * Cos(Phi)

Pitch: Theta = Atan(-Gx / Gz2)

Note: if Theta = +/-90 deg, then Gy and Gz are near-zero and Gz2 is also near-zero; division by zero should not be performed, Theta = -90 deg if Gx>0, and Theta = +90 deg if Gx<0; Gx cannot be zero

Step 3: Computation of Psi (yaw angle, also known as heading; see figure 1 for reference) based on magnetometer data (Bx, By and Bz; unit of measure does not matter)

By2 = Bz * Sin(Phi) – By * Cos(Phi)

Bz2 = By * Sin(Phi) + Bz * Cos(Phi)

Bx3 = Bx * Cos(Theta) + Bz2 * Sin(Theta)

Yaw: Psi = Atan2(By2 , Bx3)

Note: if Theta = +/-90 deg and if Phi is unstable, Psi will also be unstable; if Phi is made stable as mentioned above, then Psi will also be stable. Regardless of stability, the sum Phi+Psi will always be stable. See paragraph on singularities and gimbal-lock.

Step 4: Conversion from Euler angles to Quaternions (optional step)

Qw = Cos(Phi/2)*Cos(Theta/2)*Cos(Psi/2) + Sin(Phi/2)*Sin(Theta/2)* Sin(Psi/2)

Qx = Sin(Phi/2)*Cos(Theta/2)*Cos(Psi/2) – Cos(Phi/2)*Sin(Theta/2)*Sin(Psi/2)

Qy = Cos(Phi/2)*Sin(Theta/2)*Cos(Psi/2) + Sin(Phi/2)*Cos(Theta/2)*Sin(Psi/2)

Qz = Cos(Phi/2)*Cos(Theta/2)*Sin(Psi/2) – Sin(Phi/2)*Sin(Theta/2)*Cos(Psi/2)

Singularities and gimbal-lock

If Theta = +/-90 deg, Phi and Psi will describe a rotation around the same vertical axis (one degree of freedom is lost, this is also known as gimbal lock); the sum Psi+Phi will be correct when Theta=-90 deg, or Psi-Phi when Theta=+90 deg.

High-G motion

Accelerometer data cannot be used for tilt measurement (Phi and Theta) if high-g is ongoing. Accelerometer data can only be used when the modulus is near g: $Gx^2 + Gy^2 + Gz^2 = 1$, when Gx / Gy / Gz are expressed in g.

Gyroscope data can be used to update the tilt output when the accelerometer data cannot be used.

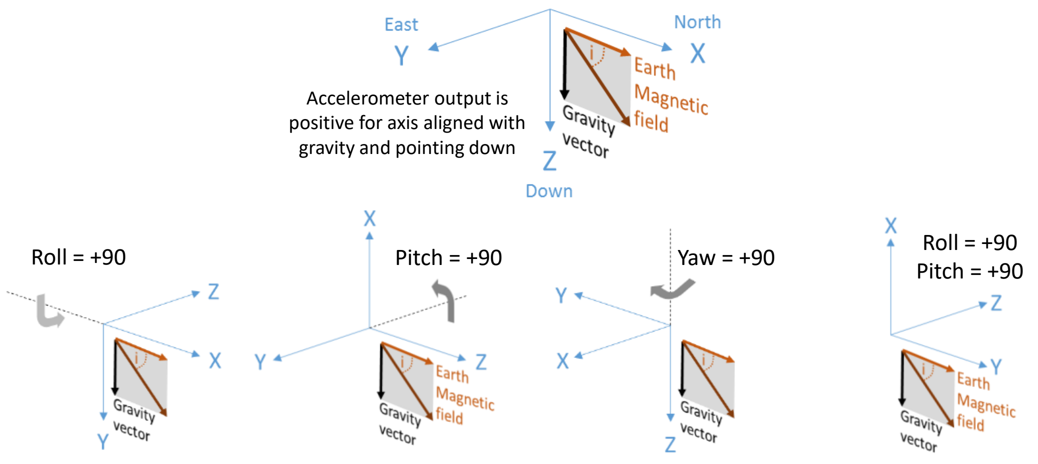
Hard/Soft iron effects and Magnetic anomalies

Magnetometer data cannot be used for the e-compass (Psi) if hard/soft iron effects are not compensated and/or if magnetic anomalies are present. Hard iron effects are offsets that must be subtracted from Bx / By / Bz. Soft iron effects are non-unity axis sensitivities and non-zero cross-axis sensitivities that must be compensated by matrix multiplication.

Magnetometer data can only be used when the modulus is of near nominal earth field value, which happens when there are no magnetic anomalies and hard/soft iron effects are compensated: $Bx^2 + By^2 + Bz^2 = B$.

Gyroscope data can be used to update the e-compass output when the magnetometer data cannot be used.

Figure 1. Reference orientation for input data from accelerometer and magnetometer, and reference orientation for output data: roll, pitch and yaw angles.



Roll, Pitch, Yaw degrees	Acc (Gravity vector) G = 1g = 1000mg	Mag (Earth mag field) B = 10..90uT = 100..900 mG
0, 0, 0	0, 0, **+G**	**+B Cos(i),** 0, +B Sin(i)
+90, 0, 0	0, **+G**, 0	**+B Cos(i),** +B Sin(i), 0
0, **+90**, 0	**-G**, 0, 0	-B Sin(i), 0, **+B Cos(i)**
0, 0, **+90**	0, 0, **+G**	0, **-B Cos(i),** +B Sin(i)
+90, **+90**, 0	**-G**, 0, 0	-B Sin(i), **+B Cos(i),** 0

Generic Tilt angle with respect to horizontal plane

The generic Tilt angle, with respect to the horizontal plane, can be derived from the formula of dot-product or vector-product of current acceleration vector and gravity vector pointing downward. The gravity vector has components x=0, y=0, z=1.

The dot-product of two vectors A and B is **A.B = |A| |B| cos(angle)** = Ax Bx + Ay By + Az Bz. When A is the current acceleration (Gx, Gy, Gz) and B is the gravity pointing downward (x=0, y=0, z=1): |G| cos(angle) = Gz, therefore tilt angle = acos (Gz / |G|).

Tilt = acos(Gz / sqrt(Gx^2+Gy^2+Gz^2)) from the dot-product formula

The vector-product of two vectors A and B is C = **AxB = |A| |B| N sin(angle),** where N is the unity vector orthogonal to both vectors A and B. Cx = Ay Bz – Az By, Cy = Az Bx – Ax Bz, Cz = Ax By – Ay Bx. When A is the current acceleration (Gx, Gy, Gz) and B is the gravity pointing downward (x=0, y=0, z=1): |C| = |G| cos(angle) = sqrt(Gx^2 + Gy^2), therefore tilt angle = acos (|C| / |G|).

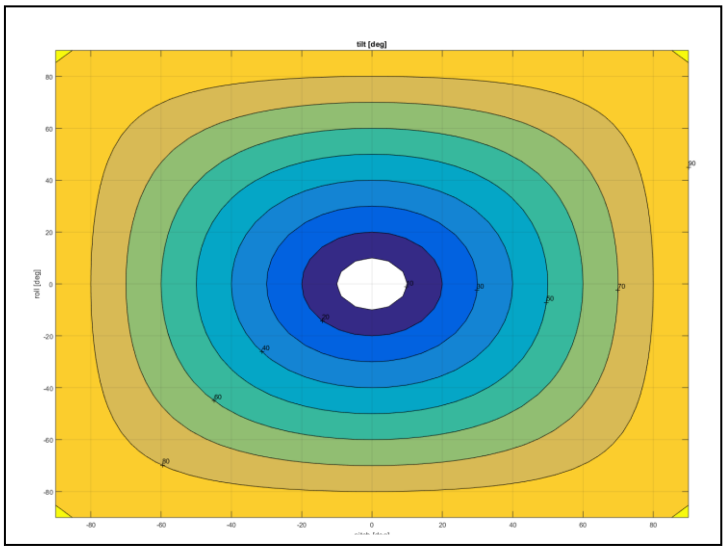
Tilt = asin(sqrt(Gx^2 + Gy^2) / sqrt(Gx^2+Gy^2+Gz^2)) from the vector-product

Taking into account measurement noise, asin function is more accurate if the angle is small, that is when -45<angle<+45 deg, while acos function is accurate when angle is large, that is angle<-45 or angle>+45. To get highest accuracy for all angles, use atan or atan2 function:

Tilt = atan(sqrt(Gx^2 + Gy^2) / Gz) if Gz=0, atan(+/-infinity)=+/-90 deg

Tilt = atan2(sqrt(Gx^2 + Gy^2) , Gz) no need to check if Gz=0

Figure 2. Roll and pitch angles compared to generic tilt angle with respect to horizontal plane.



Optimized implementation for trigonometric functions (sin, cos, atan2)

If a floating-point unit (FPU) is available on the microcontroller (e.g. on Cortex-M4 in STM32F4 or STM32L4), then the vectorized computation may be available to speed up computations. Data is collected in buffers, and then it is processed in batches.

If single-cycle multiplication is available on the microcontroller (e.g. on Cortex-M3 in STM32F1, STM32L1, STM32F3), then polynomial approximation with Horner evaluation order can be very effective.

Finally, the simplest microcontrollers may benefit from CORDIC implementation which requires only table look-up and shift-and-add operations. Roughly speaking, the CORDIC algorithm requires as many iterations as the number of precision bits; also, two functions are computed simultaneously: sin & cos, atan2 and modulus.

Support material

Related design support material
BlueMicrosystem1, Bluetooth low energy and sensors software expansion for STM32Cube
Open.MEMS, MotionFX, Real-time motion-sensor data fusion software expansion for STM32Cube
Documentation
Application note, AN3192, Using LSM303DLH for a tilt compensated electronic compass
Application note, AN4509, Tilt measurement using a low-g 3-axis accelerometer

Revision history

Date	Version	Changes
05-Nov-2015	1	Initial release
22-Nov-2018	2	Added paragraph on generic tilt; Fig.1 changed.