DT0053
Design tip

6-point tumble sensor calibration

By Andrea Vitali

Main components	
H3LIS331DL	MEMS motion sensor: low power high g 3-axis digital accelerometer
LSM6DS3	iNEMO inertial module: 3D accelerometer and 3D gyroscope

Purpose and benefits

This design tip explains how to compute offsets, gains, and cross-axis gains for a 3-axis sensor (usually an accelerometer) by performing a 6-point tumble calibration. A generalization of the algorithm is also described.

Benefits:

- Added functionality with respect to calibration provided by the osxMotionFX library which only provides Magnetometer and Gyroscope calibration and not Accelerometer calibration.
- Short and 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).

Specification

Algorithm specification:

- Input from 3-axis sensor: [x,y,z] data triplets for each position (minimum 4, usually 6, N using the generalized algorithm)
- Output: offsets for each axis (Xofs, Yofs, Zofs), gains for each axis (Xgain, Ygain, Zgain) and cross-gains (YtoX, ZtoX, XtoY, ZtoY, XtoZ, YtoZ)

Algorithm description

The algorithm is described for the particular case of an accelerometer. However, it can also be used with other sensors, e.g. a magnetometer. See notes at the end of this document.

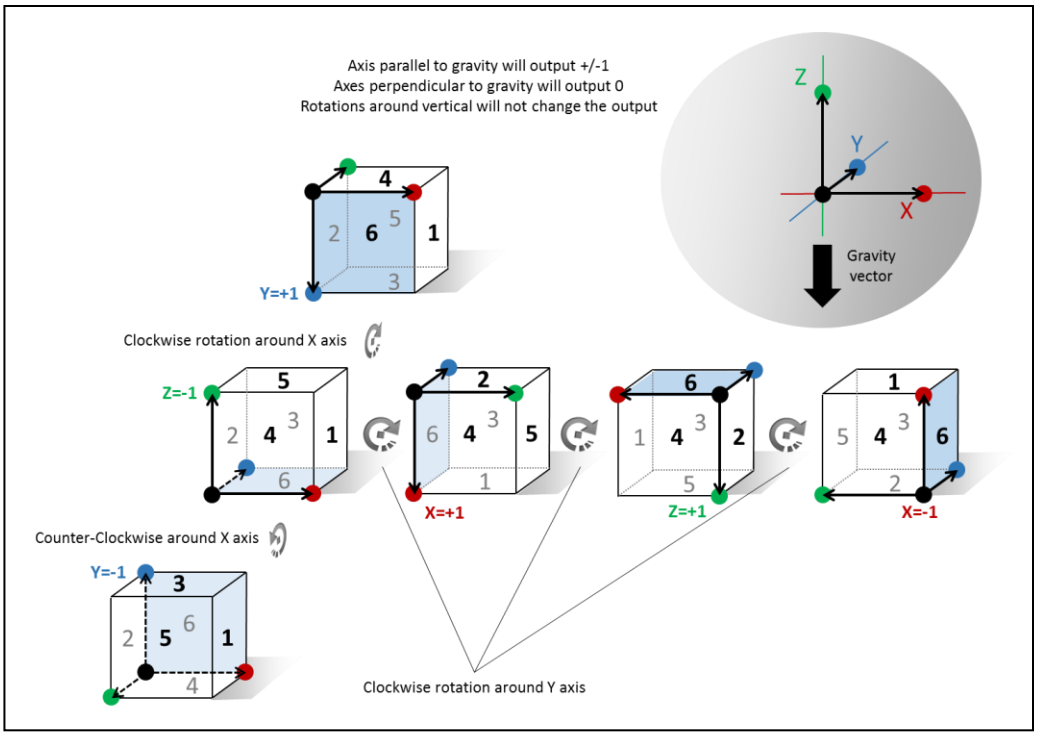
How true acceleration is related to measured acceleration:

$$\begin{bmatrix} \mathrm{AccX} \\ \mathrm{AccY} \\ \mathrm{AccZ} \end{bmatrix} = \begin{bmatrix} \mathrm{Xgain} & \mathrm{YtoX} & \mathrm{ZtoX} \\ \mathrm{XtoY} & \mathrm{Ygain} & \mathrm{ZtoY} \\ \mathrm{XtoZ} & \mathrm{YtoZ} & \mathrm{Zgain} \end{bmatrix} \begin{bmatrix} \mathrm{trueAccX} \\ \mathrm{trueAccY} \\ \mathrm{trueAccZ} \end{bmatrix} + \begin{bmatrix} \mathrm{Xofs} \\ \mathrm{Yofs} \\ \mathrm{Zofs} \end{bmatrix}.$$

True $[x,y,z]$ acceleration for each position in a 6-point tumble calibration (see figure 1):

1. Gravity vector along +X axis: $\mathrm{trueAcc} = [+1,\ 0,\ 0]$
2. Gravity vector along -X axis: $\mathrm{trueAcc} = [-1,\ 0,\ 0]$
3. Gravity vector along +Y axis: $\mathrm{trueAcc} = [\ 0, +1,\ 0]$
4. Gravity vector along -Y axis: $\mathrm{trueAcc} = [\ 0, -1,\ 0\ ]$
5. Gravity vector along +Z axis: $\mathrm{trueAcc} = [\ 0,\ 0, +1]$
6. Gravity vector along -Z axis: $\mathrm{trueAcc} = [\ 0,\ 0, -1]$

Figure 1. Reference for sensor orientation while performing 6-point tumble calibration.



Axis parallel to gravity will output +/-1
Axes perpendicular to gravity will output 0
Rotations around vertical will not change the output
Z
Y
X
Gravity vector
Y=+1
Clockwise rotation around X axis
Z=-1
X=+1
Z=+1
X=-1
Counter-Clockwise around X axis
Y=-1
Clockwise rotation around Y axis

Measured acceleration for each position in a 6-point tumble calibration, derived from the equation shown at the beginning, by plugging-in the values listed above for true acceleration:

1. AccX1 = Xgain+Xofs, AccY1 = XtoY+Yofs, AccZ1 = XtoZ+Zofs
2. AccX2 = -Xgain+Xofs, AccY2 = -XtoY+Yofs, AccZ2 = -XtoZ+Zofs
3. AccX3 = YtoX+Xofs, AccY3 = Ygain+Yofs, AccZ3 = YtoZ+Zofs
4. AccX4 = -YtoX+Xofs, AccY4 = -Ygain+Yofs, AccZ4 = -YtoZ+Zofs
5. AccX5 = ZtoX+Xofs, AccY5 = ZtoY+Yofs, AccZ5 = Zgain+Zofs
6. AccX6 = -ZtoX+Xofs, AccY6 = -ZtoY+Yofs, AccZ6 = -Zgain+Zofs

Offsets can be computed in several ways by summing two out of six measures listed above, carefully chosen so that gains and cross-gains cancel out:

- AccX1+AccX2 = 2 Xofs, AccX3+AccX4 = 2 Xofs, AccX5+AccX6 = 2 Xofs
- AccY1+AccY2 = 2 Yofs, AccY3+AccY4 = 2 Yofs, AccY5+AccY6 = 2 Yofs
- AccZ1+AccZ2 = 2 Zofs, AccZ3+AccZ4 = 2 Zofs, AccZ5+AccZ6 = 2 Zofs

Depending on available measures, more than one estimate may be available for each offset. In this case, averaging can be used to improve the quality of the final estimate.

Once offsets are computed, gains and cross-axis gains can be computed as follows; again averaging can be used if more than one estimate is available:

- Xgain = AccX1-Xofs, Xgain = -AccX2+Xofs
- XtoY = AccY1-Yofs, XtoY = -AccY2+Yofs
- XtoZ = AccZ2-Zofs, XtoZ = -AccZ2+Zofs
- YtoX = AccX3-Xofs, YtoX = -AccX4+Xofs
- Ygain = AccY3-Yofs, Ygain = -AccY4+Yofs
- YtoZ = AccZ3-Zofs, YtoZ = -AccZ4+Zofs
- ZtoX = AccX5-Xofs, ZtoX = -AccX6+Xofs
- ZtoY = AccY5-Yofs, ZtoY = -AccY6+Yofs
- Zgain = AccZ5-Zofs, Zgain = -AccZ6+Zofs

Now offsets can be subtracted, and multiplication by the inverse matrix can be done to go from measured acceleration to true acceleration.

Algorithm generalization

A more general approach can be used: measures are taken in any number of positions (N) and combined to find the unknowns (offsets, gains and cross-axis gains). For each position, there is the need to know the true acceleration together with the measure taken by the sensor.

It is convenient to combine all the unknowns in one matrix:

$$\begin{bmatrix} \mathbf{AccX} \\ \mathbf{AccY} \\ \mathbf{AccZ} \end{bmatrix} = \begin{bmatrix} \mathbf{Xgain} & \mathbf{YtoX} & \mathbf{ZtoX} & \mathbf{Xofs} \\ \mathbf{XtoY} & \mathbf{Ygain} & \mathbf{ZtoY} & \mathbf{Yofs} \\ \mathbf{XtoZ} & \mathbf{YtoZ} & \mathbf{Zgain} & \mathbf{Zofs} \end{bmatrix} \begin{bmatrix} \mathbf{trueAccX} \\ \mathbf{trueAccY} \\ \mathbf{trueAccZ} \\ \mathbf{1} \end{bmatrix}$$

Then all acceleration triplets and corresponding true reference acceleration triplets are stacked up to form N-lines matrixes [rows x N-lines]:

$$\mathbf{Acc[3xN] = Unknowns[3x4]\ trueAcc[4xN]}$$

Now, the least-square error approximation can be computed for the unknowns, by using the pseudo-inverse of the non-square matrix **trueAcc[]**. First, both sides are multiplied by the transpose $\mathbf{trueAcc^T[]}$. Second, both sides are multiplied by the inverse of the 4x4 matrix.

$$\mathbf{Acc[3xN]\ trueAcc^T[Nx4] = Unknowns[3x4]\ trueAcc[4xN]\ trueAcc^T[Nx4]}$$

$$\mathbf{Acc[3xN]\ trueAcc^T[Nx4]\ inv(trueAcc[4xN]\ trueAcc^T[Nx4]) = Unknowns[3x4]}$$

Notes

Hints for a compact real-time implementation on a microcontroller:

- Only the product $\mathbf{Acc[3xN]\ trueAcc^T[Nx4]}$ needs to be maintained in memory, this is a 3x4 matrix made of 12 numbers.
- Only the product $\mathbf{trueAcc[4xN]\ trueAcc^T[Nx4]}$ needs to be maintained in memory, this is a 4x4 matrix made of 16 numbers.
- Gaussian elimination can be implemented to compute the inverse of the aforementioned 4x4 matrix when enough triplets have been collected.

Application to other sensors:

- While it may be easy to exploit the known gravity vector to impose the wanted true reference on the accelerometer, it may not be possible to achieve perfect accuracy when switching from one position to another during the 6-point tumble calibration
- An alternative way to perform the calibration is to use a gold reference sensor which has the same orientation of the sensor to be calibrated, so that the wanted true reference can be measured and fed to the generalized algorithm.

- Special equipment, arrangement or procedure may be needed to impose or measure the wanted true reference on other sensors such as magnetometers or gyroscopes
- For the case of the magnetometer: Helmholtz coils can be used to impose the wanted true reference; alternatively the earth magnetic field vector can be used together with a gold reference sensor.
- For the case of the gyroscope: a single-axis turn table or a step-motor spin table can be used to impose the wanted true reference; alternatively a gold reference sensor can be used as described above.

Other algorithms: *sphere-fitting* can be used to estimate offsets based on a set of triplets from the sensor, *ellipsoid-fitting* can be used to estimate offsets and gains, and *rotated ellipsoid fitting* can be used to estimate offsets, gains and cross-axis gains (in this latter case, cross-axis gains are usually symmetrical: $\mathbf{XtoY} = \mathbf{YtoX}$, $\mathbf{XtoZ} = \mathbf{ZtoX}$, $\mathbf{YtoZ} = \mathbf{ZtoY}$). See relevant document.

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, AN4508, Parameters and calibration of a low-g 3-axis accelerometer
Application note, AN4615, Fusion and compass calibration APIs for the STM32 Nucleo with the X-NUCLEO-IKS01A1 sensors expansion board

Revision history

Date	Version	Changes
25-Nov-2015	1	Initial release