# Magnetometer and Accelerometer Calibration Algorithm*

CK-Explorer

## 1 Introduction

This document will explain briefly the mathematical procedures behind the calibration algorithm implemented inside both MATLAB script and C++ library.

In general, the algorithm comprises three main objectives, i.e. minimizing the problems of

i non-orthoganality between three axes and different axes' scale factors of each individual sensor

ii centre biases of each individual sensor

iii misalignment of triads between both sensors

Problems (i) and (ii) will be explained in section 2, of which the solution is based on the ellipsoid model that fits the sensors' data collected. Meanwhile, in section 3, problem (iii) will be reduced by finding the most optimum rotation to align both magnetometer and accelerometer triads through least square fitting.

## 2 Ellipsoid modelling

The method in this section is suitable for calibrating accelerometer and magnetometer separately.

Suppose the reference frame in this document be NED frame[1] without loss of generality, and this frame must be consistent throughout the calibration and also application processes. When the body frame aligns with the reference frame, the accelerometer senses gravity vector under stationary condition, while the magnetometer senses Earth's magnetic field vector when it is placed in an area without any magnetic interference. Hence, if we rotate both sensors in all possible orientations, a perfect sphere with zero origin as its centre should be traced out, since both vectors retain the same strength under the above conditions. However, due to possible manufacturing defects for both sensors and magnetic interferences are impossible to avoid, which introduce soft and hard iron distortions to the magnetometer, an ellipsoid with shifted centre will be formed instead.

*https://github.com/CK-Explorer/Magnetometer-and-Accelerometer-Calibration

[1] Any convention can be used, e.g. NED, NWU, ENU.

From [1], such ellipsoid could be considered as a consequence of a combination of errors from non-orthogonality between axes, different scale factors and non-null offsets, which can be represented in Eq. 1 [2].

$$\begin{pmatrix} x_s \\ y_s \\ z_s \end{pmatrix} = \begin{pmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ \sin\rho & \cos\rho & 0 \\ \cos\phi\sin\lambda & \sin\phi\sin\lambda & \cos\lambda \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} + \begin{pmatrix} x_o \\ y_o \\ z_o \end{pmatrix} \tag{1}$$

in which the non-orthogonality matrix can be depicted in figure 1.

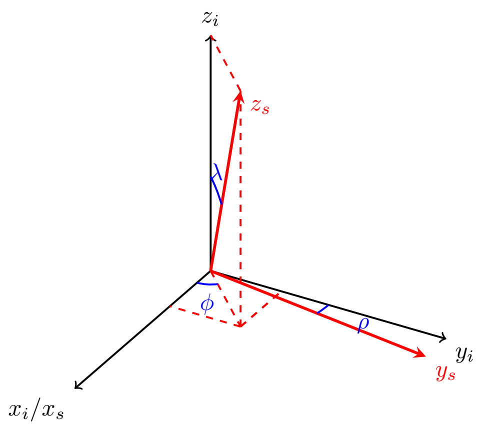


Figure 1: Non-orthoganality causes deviation of the sensor's axes (red) from the exact axes (black). As a result, these sensor's axes read the projected values from the exact orthonormal axes.

Rearranging Eq. 1 will form Eq. 2,

$$\begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} k_x & 0 & 0 \\ k_x\sin\rho & k_y\cos\rho & 0 \\ k_x\cos\phi\sin\lambda & k_y\sin\phi\sin\lambda & k_z\cos\lambda \end{pmatrix}^{-1} \begin{pmatrix} x_s - x_o \\ y_s - y_o \\ z_s - z_o \end{pmatrix} \tag{2}$$

or for simplicity,

$$X_i = LX_s \tag{3}$$

of which $L$ is also a lower triangular matrix.

A sphere equation of radius one can be easily formed from ideal sensor values, hence we have the following,

$$X_i^T X_i = (LX_s)^T(LX_s) = X_s^T L^T L X_s = 1 \tag{4}$$

which $L^T L$ will form a symmetrical matrix due to the property of lower triangular matrix.

[2] $x_i$, $y_i$, $z_i$ = ideal or true sensor values from x, y and z axes,
$x_s$, $y_s$, $z_s$ = observed sensor value from x, y and z axes,
$k_x$, $k_y$, $k_z$ = scale factors of observed sensor for x, y and z axes,
$x_o$, $y_o$, $z_o$ = offset from origin for x, y and z axes,
$\rho$, $\phi$, $\lambda$ = angles from figure 1.

If we now let $L = \begin{pmatrix} a & 0 & 0 \\ b & c & 0 \\ d & e & f \end{pmatrix}$ without reducing the number of unknown variables as $k_x$, $k_y$, $k_z$, $\rho$, $\phi$, $\lambda$ (6 unknowns), we have

$$L^T L = \begin{pmatrix} a^2+b^2+g^2 & bc+eg & fg \\ bc+eg & c^2+e^2 & ef \\ fg & ef & f^2 \end{pmatrix} = \begin{pmatrix} A & D & E \\ D & B & F \\ E & F & C \end{pmatrix} \tag{5}$$

By plugging $L^T L$ back into Eq. 4, it will form an ellipsoid equation in matrix form [3],

$$\begin{pmatrix} x-x_o & y-y_o & z-z_o \end{pmatrix} \begin{pmatrix} A & D & E \\ D & B & F \\ E & F & C \end{pmatrix} \begin{pmatrix} x-x_o \\ y-y_o \\ z-z_o \end{pmatrix} = 1 \tag{6}$$

or in the algebraic form,

$$\begin{array}{rrrrrr} Ax^2 & -2Ax_ox & & & + & Ax_o^2 \\ +\ By^2 & & -2By_oy & & + & By_o^2 \\ +\ Cz^2 & & & -2Cz_oz & + & Cz_o^2 \\ +2Dxy & -2Dy_ox & -2Dx_oy & & & +2Dx_oy_o \\ +2Exz & -2Ez_ox & & -2Ex_oz & & +2Ex_oz_o \\ +2Fyz & & -2Fz_oy & -2Fy_oz & & +2Fy_oz_o \\ & & & & - & 1 \quad = 0 \end{array} \tag{7}$$

All the elements in $L^T L$ and centre biases, $x_o$, $y_o$ and $z_o$ can be easily determined through ellipsoid fitting process. This is performed through ellipsoid_fit_new function[4] and ellipsoid::fit function[5] in MATLAB script and C++ code respectively. Both functions will output all the centre biases and the coefficients of the ellipsoid function in the algebraic form as in Eq. 8.

$$v_0x^2 + v_1y^2 + v_2z^2 + 2v_3xy + 2v_4xz + 2v_5yz + 2v_6x + 2v_7y + 2v_8z + v_9 = 0 \tag{8}$$

The relationship between those coefficents in Eq. 8 and elements in Eq. 5 can be observed through Eq. 7. We can easily convert Eq. 8 to Eq. 7 by multiplying a certain factor of $m$ to the whole Eq. 8. By focusing only the most left hand side of Eq. 7, we will have

$$A = mv_0, \quad B = mv_1, \quad C = mv_2, \quad D = mv_3, \quad E = mv_4, \quad F = mv_5 \tag{9}$$

and the most right hand side of Eq. 7,

$$Ax_o^2 + By_o^2 + Cz_o^2 + 2Dx_oy_o + 2Ex_oz_o + 2Fy_oz_o - 1 = mv_9 \tag{10}$$

Since those three centre biases have been known, Eq. 10 can be rearranged and with the substitutions of those equations from 9, we will get

$$m\left\{ \begin{pmatrix} x_o & y_o & z_o \end{pmatrix} \begin{pmatrix} v_0 & v_3 & v_4 \\ v_3 & v_1 & v_5 \\ v_4 & v_5 & v_2 \end{pmatrix} \begin{pmatrix} x_o \\ y_o \\ z_o \end{pmatrix} - v_9 \right\} = m\kappa = 1 \tag{11}$$

and the factor of $m$ can be determined from the reciprocal of the value of $\kappa$, i.e. $m = \frac{1}{\kappa}$.

---

[3] $\begin{pmatrix} x & y & z \end{pmatrix} = \begin{pmatrix} x_s & y_s & z_s \end{pmatrix}$ for simplicity.

[4] `https://www.mathworks.com/matlabcentral/fileexchange/24693-ellipsoid-fit`

[5] `https://github.com/BenjaminNavarro/ellipsoid-fit`

Now, the elements in $L^T L$ can be determined,

$$\begin{pmatrix} A \\ B \\ C \\ D \\ E \\ F \end{pmatrix} = \frac{1}{\kappa} \begin{pmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{pmatrix} \tag{12}$$

and consequently, using Eq. 5, all the elements in $L$ can be solved in the following set of equations (from left to right manner) [6],

$$f = \sqrt{C}, \quad e = \frac{F}{f}, \quad g = \frac{E}{f}, \quad c = \sqrt{B - e^2}, \quad b = \frac{D - eg}{c}, \quad a = \sqrt{A - b^2 - g^2} \tag{13}$$

Lastly, the non-calibrated sensor [7] outputs can be corrected and normalised by applying Eq. 14 [8],

$$\begin{pmatrix} \hat{x_s} \\ \hat{y_s} \\ \hat{z_s} \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ b & c & 0 \\ d & e & f \end{pmatrix} \begin{pmatrix} x_s - x_o \\ y_s - y_o \\ z_s - z_o \end{pmatrix} \tag{14}$$

The steps of using this section are summarised in the following pseudocode 1.

---

**Algorithm 1** Procedures of applying Section 2

---

**Input:** $x_s$, $y_s$, $z_s$ ▷ Sensor data
**Output:** $L$, $x_o$, $y_o$, $z_o$ ▷ Apply to sensor outputs via Eq. 14
  Perform ellipsoid fitting $\leftarrow x_s$, $y_s$, $z_s$ ▷ Produce $x_o$, $y_o$, $z_o$, Eq. 8
  Find $\kappa$ or $m$ ▷ Using Eq. 11
  Find the elements in $L^T L$ ▷ Using Eq. 12
  Find the elements in $L$ ▷ Using Eqs. 13

---

# 3 Misalignment of triads between two sensors

The method in this section requires both accelerometer and magnetometer data captured concurrently in the same orientations.

Suppose that we have both well calibrated accelerometer and magnetometer. Nonetheless, there is a possiblity that the triads of both sensors are not perfectly aligned to each other, which can be conceived as in figure 2, especially when these two sensors are not mounted on the same IMU, and we have to align them manually without any assistance from high precision machine. To reduce such error, dot product invariance between gravity and Earth's magnetic field vectors could be used as described in [2].

[6] $a, c, f \geq 0$ since $\rho, \lambda$ in Eq. 2 are assumed to be small values, which fulfill $-\frac{\pi}{2} < \rho, \lambda < \frac{\pi}{2}$.

[7] Can be used for accelerometer and magnetometer separately.

[8] $\hat{x_s}$, $\hat{y_s}$, $\hat{z_s}$ = calibrated sensor outputs in x, y and z axes, which are still estimations for ideal sensor outputs.

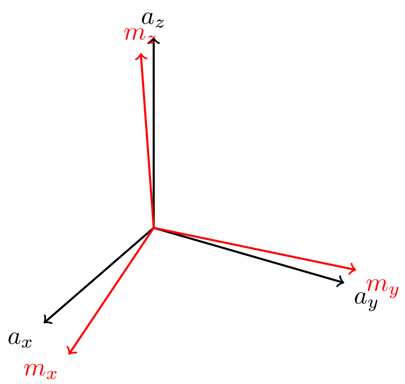


Figure 2: Misalignment between axes of the accelerometer and magnetometer.

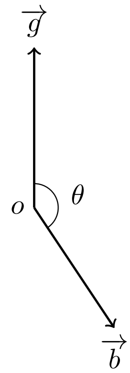


Figure 3: The gravity and Earth's magnetic field vectors lie on the same plane, and the value of $\theta$ depends on Earth's magnetic inclination.

Let us start with a situation that no misalignment occurs between accelerometer and magnetometer. By refering to figure 3, when the body frame is aligned with the NED reference frame, the accelerometer will sense $\overrightarrow{g} = \begin{pmatrix} 0 & 0 & -G \end{pmatrix}^T$, where $G$ is the gravitational acceleration under stationary condition, while the magnetometer senses a constant Earth's magnetic field vector, $\overrightarrow{b} = \begin{pmatrix} b_x & b_y & b_z \end{pmatrix}^T$, under no external magnetic interferences around. If we now rotate the body of IMU, of which its orientation is represented by rotation matrix, $R$, then the dot product of both ideal accelerometer reading, $\overrightarrow{a_i}$ and magnetometer reading, $\overrightarrow{m_i}$ [9] will produce a constant value regardless of any value of $R$, which can be seen as follows

$$\overrightarrow{a_i} \cdot \overrightarrow{m_i} = a_i^T m_i = g^T R^T R b = g^T b = |\overrightarrow{g}||\overrightarrow{b}| \cos\theta \tag{15}$$

This property allows us to find the most optimum misaligment matrix, $R_{mis}$, to rotate the magnetometer's axes back to the accelerometer's axes and $\cos\theta$ by minimizing the cost function in Eq. 16 [10],

$$\min \sum_{all\ a,m} \left(a^T R_{mis} m - |\overrightarrow{a}||\overrightarrow{m}|\beta\right)^2 \tag{16}$$

of which $\beta = \cos\theta$, while both $|\overrightarrow{a}|$ and $|\overrightarrow{m}|$ should have constant magnitudes in all orientations.

To avoid numerical instability while using yaw, pitch and roll angles to find $R_{mis}$, a rotation matrix represented by an unit quaternion , $\boldsymbol{q} = \begin{pmatrix} q_0 & q_1 & q_2 & q_3 \end{pmatrix}$, is used instead, which forms Eq. 17.

$$R_{mis} = \begin{pmatrix} 1 - 2q_2^2 - 2q_3^2 & 2q_1q_2 - 2q_0q_3 & 2q_1q_3 + 2q_0q_2 \\ 2q_1q_2 + 2q_0q_3 & 1 - 2q_1^2 - 2q_3^2 & 2q_2q_3 - 2q_0q_1 \\ 2q_1q_3 - 2q_0q_2 & 2q_2q_3 + 2q_0q_1 & 1 - 2q_1^2 - 2q_2^2 \end{pmatrix} \tag{17}$$

[9] $\overrightarrow{a_i}$, $\overrightarrow{m_i}$ = readings of accelerometer and magnetometer of which axes are perfectly aligned.
[10] $\overrightarrow{a}$, $\overrightarrow{m}$ = readings of accelerometer and magnetometer of which axes are assumed to be not aligned.

To ensure that the magnitude of $\boldsymbol{q}$ always equals to one during numerical process, stereographic projection of $\boldsymbol{q}$ is performed. This method can be described as a process of projecting the surface of a hypersphere, represented by the unit quaternion's definition of $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$, to a hyperplane or a plane governed by 3 parameters.

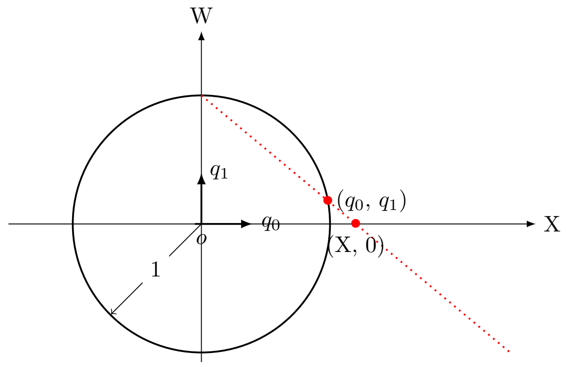


Figure 4: The convention of the stereographic projection used in this document, of which the origin of the quaternion coordinate system coincides with the centre of the unit hypersphere at $(0, 0, 0, 0)$. [11]

In figure 4, $(q_0, q_1, q_2, q_3)$ can be converted to $(X, Y, Z)$ by means of the property of similar triangles, forming Eqs. 18, 19 [12] and 20 [13] .

$$X = \frac{q_0}{1 - q_1} \tag{18}$$

$$Y = \frac{q_2}{1 - q_1} \tag{19}$$

$$Z = \frac{q_3}{1 - q_1} \tag{20}$$

It is important to note that $q_1$ is the axis which coincides with $W$ axis in the above derivations of Eqs. 18, 19 and 20. However, $q_2$ and $q_3$ can also be used, but not $q_0$ axis. Such case is avoided because singularities will be formed for $X$, $Y$ and $Z$ when $\boldsymbol{q} = \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix}$. This value of $\boldsymbol{q}$ is essential as it will produce an identity matrix which is the most possible solution for $R_{mis}$.

Reverting back from $(X, Y, Z)$ to $(q_0, q_1, q_2, q_3)$ can be achieved through Eqs. 21, 22, 23 and 24, which are derived from Eqs. 18, 19, 20.

$$q_0 = \frac{2X}{1 + X^2 + Y^2 + Z^2} \tag{21}$$

$$q_1 = \frac{X^2 + Y^2 + Z^2 - 1}{1 + X^2 + Y^2 + Z^2} \tag{22}$$

[11] The coordinates system representing the hypersphere is $(X, Y, Z, W)$.
[12] $q_2$ and $Y$ are swapped with $q_0$ and $X$ respectively in figure 4.
[13] $q_3$ and $Z$ are swapped with $q_0$ and $X$ respectively in figure 4.

$$q_2 = \frac{2Y}{1 + X^2 + Y^2 + Z^2} \tag{23}$$

$$q_3 = \frac{2Z}{1 + X^2 + Y^2 + Z^2} \tag{24}$$

Hence, there are only 4 parameters that need to be found from minimizing the cost function in Eq. 16, i.e. $X$, $Y$, $Z$ and $\beta$. Their initial guesses should be $\boldsymbol{q_{est}} = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix}$ for $X_{est}$, $Y_{est}$, $Z_{est}$ with the assumption that all the axes of both magnetometer and accelerometer are nearly aligned, while $\beta_{est} = \cos\theta_{est}$, of which $\theta_{est}$ should be $90°$ or $\frac{\pi}{2}$ rad plus with the local magnetic inclination [14].

This least square problem is solved using Levenberg–Marquardt algorithm, which is implemented using lsqcurvefit and alglib::lsfitfit [15] functions in Matlab and C++ respectively. Since both of these functions can perform minimization without the need of Jacobian matrix, then direct substitution of $(X, Y, Z)$ into $\boldsymbol{q}$ in Eq. 17 is not required, which simplifies the process of coding. The steps of solving this least square problem are summarised in the following pseudocode 2.

**Algorithm 2** Solving the least square problem in Eq. 16

**Input:** $X_{est}$, $Y_{est}$, $Z_{est}$, $\beta_{est}$ ▷ Initial estimates
$\overrightarrow{a}$, $\overrightarrow{m}$ ▷ Collected accelerometer and magnetometer data
**Output:** $\hat{X}$, $\hat{Y}$, $\hat{Z}$, $\hat{\beta}$ ▷ Apply to Eq. 17 to get $R_{mis}$
$X$, $Y$, $Z$, $\beta \leftarrow X_{est}$, $Y_{est}$, $Z_{est}$, $\beta_{est}$ ▷ Initialisation
**while** convergence condition is not fulfilled **do**
 Find $\boldsymbol{q} \leftarrow X$, $Y$, $Z$ ▷ Using Eqs. 21, 22, 23, 24
 Find $R_{mis} \leftarrow \boldsymbol{q}$ ▷ Using Eq. 17
 Find the value of cost function $\leftarrow R_{mis}$, $\beta$, $\overrightarrow{a}$, $\overrightarrow{m}$ ▷ Using Eq. 16
 Pass the cost function value into the solver ▷ lsqcurvefit, alglib::lsfitfit
 Update $X$, $Y$, $Z$, $\beta$
**end while**
$\hat{X}$, $\hat{Y}$, $\hat{Z}$, $\hat{\beta} \leftarrow X$, $Y$, $Z$, $\beta$

Finally, with $R_{mis}$ obtained using $\hat{X}$, $\hat{Y}$ and $\hat{Z}$ from Eq. 17, the non-calibrated magnetometer[16] outputs can be corrected using the following Eq. 25 [17].

$$\begin{pmatrix} \hat{m_x} \\ \hat{m_y} \\ \hat{m_z} \end{pmatrix} = R_{mis} \begin{pmatrix} a & 0 & 0 \\ b & c & 0 \\ d & e & f \end{pmatrix} \begin{pmatrix} m_x - m_x^o \\ m_y - m_y^o \\ m_z - m_z^o \end{pmatrix} \tag{25}$$

[14] For NED frame
[15] from ALGLIB.
[16] Only magnetometer, accelerometer is not required to be corrected.
[17] $a$, $b$, $c$, $d$, $e$, $f$ are elements from $L$ in Section 2, refer to Eq. 14.
$m_x^o$, $m_y^o$, $m_z^o$ are the centre biases obtained from Section 2, refer to Eq. 14.

# References

[1] C. C. Foster and G. H. Elkaim, "Extension of a two-step calibration methodology to include nonorthogonal sensor axes," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 3, pp. 1070–1078, 2008.

[2] M. Kok, J. D. Hol, T. B. Schön, F. Gustafsson, and H. Luinge, "Calibration of a magnetometer in combination with inertial sensors," in *2012 15th International Conference on Information Fusion*, 2012, pp. 787–793.

END