Freescale Semiconductor
APPLICATION NOTE

Document Number: AN5021
Rev. 1, 9/2015

Calculation of Orientation Matrices from Sensor Data

Mark Pedley and Michael Stanley

Contents

1 Introduction

1.1 Summary

This application note introduces the rotation or orientation matrix and documents the functions in file *orientation.c* used by the *Freescale Sensor Fusion Library* to compute an orientation matrix from sensor measurements.

1.2 Terminology

Symbol	Definition
B	Geomagnetic field strength
$\boldsymbol{B}_c$	Calibrated magnetometer measurements in the sensor PCB frame
B_{cx}, B_{cy}, B_{cz}	x, y and z components of the calibrated magnetometer reading in the sensor frame
ENU	x=East, y=North, z=Up coordinate system (Android and Windows 8)
$\boldsymbol{G}_s$	Accelerometer reading in the sensor frame
G_{sx}, G_{sy}, G_{sz}	x, y and z components of the accelerometer reading in the sensor frame
NED	x=North, y=East, z=Down coordinate system (Aerospace)
R_{ij}	Element in row i and column j of orientation matrix $\boldsymbol{R}$
$\boldsymbol{R}_{Android}$	Orientation matrix in the Android coordinate system
$\boldsymbol{R}_{NED}$	Orientation matrix in the NED/Aerospace coordinate system
$\boldsymbol{R}_{Win8}$	Orientation matrix in the Windows 8 coordinate system
R_{xy}	Element in row x and column y of orientation matrix $\boldsymbol{R}$
δ	Geomagnetic inclination angle
θ	Pitch angle
ϕ	Roll angle
ψ	Yaw angle

1.3 Software Functions

Table 1. Sensor Fusion software functions

Function	Description	Section

`void f3DOFTiltNED (float fR[][3], float fGs[]);`
 	
`void f3DOFTiltAndroid (float fR[][3], float fGs[]);`
 	
`void f3DOFTiltWin8 (float fR[][3], float fGs[]);`
	Computes the tilt orientation matrix from accelerometer measurements in the Aerospace/NED, Android and windows 8 coordinate systems.	4.2
		4.3
		4.4

Function	Description	Section

`void f3DOFMagnetometerMatrixNED (float fR[][3], float fBc[]);`
 	
`void f3DOFMagnetometerMatrixAndroid (float fR[][3], float fBc[]);`
 	
`void f3DOFMagnetometerMatrixWin8 (float fR[][3], float fBc[]);`
	Computes the 2D eCompass yaw orientation matrix from calibrated magnetometer measurements in the Aerospace/NED, Android and Windows 8 coordinate systems.	5.2
		5.3
		5.4

`void feCompassNED (float fR[][3], float *pfDelta, float fBc[], float fGs[]);`
 	
`void feCompassAndroid (float fR[][3], float *pfDelta, float fBc[], float fGs[]);`
 	
`void feCompassWin8 (float fR[][3], float *pfDelta, float fBc[], float fGs[]);`
	Computes the full orientation matrix from accelerometer and calibrated magnetometer measurements in the Aerospace/NED, Android and Windows 8 coordinate systems and also returns the geomagnetic inclination angle.	6.2
		6.3
		6.4

2 General Rotation Matrix

2.1 Introduction

The general rotation matrix $\boldsymbol{R}$ which transforms a vector as a result of a rotation of the coordinate system around the axis $\hat{\boldsymbol{n}}$ by angle η is:

$$\boldsymbol{R} = \begin{pmatrix} R_{xx} & R_{xy} & R_{xz} \\ R_{yx} & R_{yy} & R_{yz} \\ R_{zx} & R_{zy} & R_{zz} \end{pmatrix} = \begin{pmatrix} \hat{n}_x{}^2 + (1 - \hat{n}_x{}^2)cos\eta & \hat{n}_x\hat{n}_y(1 - cos\eta) + \hat{n}_z sin\eta & \hat{n}_x\hat{n}_z(1 - cos\eta) - \hat{n}_y sin\eta \\ \hat{n}_x\hat{n}_y(1 - cos\eta) - \hat{n}_z sin\eta & \hat{n}_y{}^2 + (1 - \hat{n}_y{}^2)cos\eta & \hat{n}_y\hat{n}_z(1 - cos\eta) + \hat{n}_x sin\eta \\ \hat{n}_x\hat{n}_z(1 - cos\eta) + \hat{n}_y sin\eta & \hat{n}_y\hat{n}_z(1 - cos\eta) - \hat{n}_x sin\eta & \hat{n}_z{}^2 + (1 - \hat{n}_z{}^2)cos\eta \end{pmatrix} \quad \textbf{(1)}$$

Equation (1) is known as the Rodrigues rotation matrix. The product of the rotation axis $\hat{\boldsymbol{n}}$ and rotation angle η is termed the rotation vector $\hat{\boldsymbol{n}}\eta$.

If the coordinate system rotates by angle η about the *z* axis, then a vector rotates by angle $-\eta$ about the *z* axis, relative to the newly rotated coordinate system. When we refers to a rotation vector defining a rotation matrix, it always refers to a coordinate system rotation.

In addition, when a rotation matrix $\boldsymbol{R}$ models the orientation of the sensors relative to the earth (global) reference frame, the *Freescale Sensor Fusion Library* standard is that the product $\boldsymbol{Rv}$ transforms the vector $\boldsymbol{v}$ from the global to the sensor frame. The inverse rotation $\boldsymbol{R}^{-1}\boldsymbol{v} = \boldsymbol{R}^T\boldsymbol{v}$ transforms the vector $\boldsymbol{v}$ from the sensor to the global frame. The orientation matrix is always defined relative to an initial orientation in the global frame with the product flat and pointed to magnetic north.

2.2 Eigenvector and Eigenvalue

The product $\boldsymbol{R}\hat{\boldsymbol{n}}$ evaluates to $\hat{\boldsymbol{n}}$ confirming that $\hat{\boldsymbol{n}}$ is the eigenvector of $\boldsymbol{R}$ with eigenvalue 1. Because $\boldsymbol{R}$ is a rotation matrix, $\hat{\boldsymbol{n}}$ is, therefore, the rotation axis.

$$\boldsymbol{R}\hat{\boldsymbol{n}} = \begin{pmatrix} \hat{n}_x{}^2 + (1-\hat{n}_x{}^2)cos\eta & \hat{n}_x\hat{n}_y(1-cos\eta) + \hat{n}_z sin\eta & \hat{n}_x\hat{n}_z(1-cos\eta) - \hat{n}_y sin\eta \\ \hat{n}_x\hat{n}_y(1-cos\eta) - \hat{n}_z sin\eta & \hat{n}_y{}^2 + (1-\hat{n}_y{}^2)cos\eta & \hat{n}_y\hat{n}_z(1-cos\eta) + \hat{n}_x sin\eta \\ \hat{n}_x\hat{n}_z(1-cos\eta) + \hat{n}_y sin\eta & \hat{n}_y\hat{n}_z(1-cos\eta) - \hat{n}_x sin\eta & \hat{n}_z{}^2 + (1-\hat{n}_z{}^2)cos\eta \end{pmatrix} \begin{pmatrix} \hat{n}_x \\ \hat{n}_y \\ \hat{n}_z \end{pmatrix} \quad \textbf{(2)}$$

We can take advantage that the norm of $\hat{\boldsymbol{n}}$ has value 1 to rearrange and simplify (2) to the forms shown in (3):

$$\boldsymbol{R}\hat{\boldsymbol{n}} = \begin{pmatrix} \hat{n}_x(\hat{n}_x{}^2 + \hat{n}_y{}^2 + \hat{n}_z{}^2) - \hat{n}_x cos\eta(\hat{n}_x{}^2 + \hat{n}_y{}^2 + \hat{n}_z{}^2 - 1) \\ \hat{n}_y(\hat{n}_x{}^2 + \hat{n}_y{}^2 + \hat{n}_z{}^2) - \hat{n}_y cos\eta(\hat{n}_x{}^2 + \hat{n}_y{}^2 + \hat{n}_z{}^2 - 1) \\ \hat{n}_z(\hat{n}_x{}^2 + \hat{n}_y{}^2 + \hat{n}_z{}^2) - \hat{n}_z cos\eta(\hat{n}_x{}^2 + \hat{n}_y{}^2 + \hat{n}_z{}^2 - 1) \end{pmatrix} = \begin{pmatrix} \hat{n}_x \\ \hat{n}_y \\ \hat{n}_z \end{pmatrix} \quad \textbf{(3)}$$

3 Converting Between Rotation Matrix and Rotation Vector

3.1 Rotation Vector from Rotation Matrix

Referring back to (1), the rotation angle η can be determined from the trace of the rotation matrix $\boldsymbol{R}$:

$$tr(\boldsymbol{R}) = R_{xx} + R_{yy} + R_{zz} = \hat{n}_x{}^2 + (1-\hat{n}_x{}^2)cos\eta + \hat{n}_y{}^2 + (1-\hat{n}_y{}^2)cos\eta + \hat{n}_z{}^2 + (1-\hat{n}_z{}^2)cos\eta \quad \textbf{(4)}$$

$$= (\hat{n}_x{}^2 + \hat{n}_y{}^2 + \hat{n}_z{}^2) + 3cos\eta - (\hat{n}_x{}^2 + \hat{n}_y{}^2 + \hat{n}_z{}^2)cos\eta \quad \textbf{(5)}$$

$$\Rightarrow tr(\boldsymbol{R}) = 1 + 2cos\eta \quad \textbf{(6)}$$

$$\Rightarrow \eta = cos^{-1}\left(\frac{tr(\boldsymbol{R}) - 1}{2}\right) \quad \textbf{(7)}$$

There is no angle ambiguity in equation (7) because the inverse cosine returns an angle between 0° and 180° and the rotation angle η also varies between 0° and 180°. Rotation angles between –180° and 0° are equivalent to rotation by the negated angle $-\eta$ about the negated rotation axis $-\hat{\boldsymbol{n}}$. The rotation vector $\eta\hat{\boldsymbol{n}}$ is unaltered by a double sign change.

The trace of the matrix varies between –1 and +3. A trace of –1 corresponds to a rotation of 180° and a trace of +3 corresponds to a rotation of 0°.

In the general case, the rotation axis $\hat{\boldsymbol{n}}$ and rotation vector $\eta\hat{\boldsymbol{n}}$ are determined by differencing matrix elements across the leading diagonal in (1):

$$\hat{\boldsymbol{n}} = \frac{1}{2sin\eta}\begin{pmatrix} R_{yz} - R_{zy} \\ R_{zx} - R_{xz} \\ R_{xy} - R_{yx} \end{pmatrix} \quad \textbf{(8)}$$

$$2sin\eta = \sqrt{\left(R_{yz} - R_{zy}\right)^2 + (R_{zx} - R_{xz})^2 + \left(R_{xy} - R_{yx}\right)^2} \quad \textbf{(9)}$$

The positive square root can be taken in equation (9) because the sine of a rotation angle between 0° and 180° is always nonnegative. Substitution of equation (9) into (8) gives the rotation vector as:

$$\eta\hat{\boldsymbol{n}} = \frac{\eta}{\sqrt{\left(R_{yz} - R_{zy}\right)^2 + (R_{zx} - R_{xz})^2 + \left(R_{xy} - R_{yx}\right)^2}} \begin{pmatrix} R_{yz} - R_{zy} \\ R_{zx} - R_{xz} \\ R_{xy} - R_{yx} \end{pmatrix} \quad \textbf{(10)}$$

Equations (9) and (10) fail when $sin\eta$ approaches zero which occurs near $\eta = 0°$ and $\eta = 180°$ when the rotation matrix $\boldsymbol{R}$ becomes symmetric. These two special cases can be detected using:

- equation (9) to determine $sin\eta$ near zero
- equation (6) and the matrix trace to distinguish the two cases of $0°$ rotation angle (with trace = 3) and 180° rotation angle (with trace = –1) both of which give $sin\eta = 0$.

For the case of a rotation angle η near zero radians, $sin\eta \approx \eta$ and the rotation vector becomes:

$$\eta\hat{\boldsymbol{n}} = \frac{1}{2}\begin{pmatrix} R_{yz} - R_{zy} \\ R_{zx} - R_{xz} \\ R_{xy} - R_{yx} \end{pmatrix} \quad \textbf{(11)}$$

For the case of the rotation angle η near 180°, the rotation matrix $\boldsymbol{R}$ approximates the form below with a trace of –1:

$$\boldsymbol{R} = \begin{pmatrix} 2\hat{n}_x{}^2 - 1 & \hat{n}_x\hat{n}_y(1 - cos\eta) + \hat{n}_z sin\eta & \hat{n}_x\hat{n}_z(1 - cos\eta) - \hat{n}_y sin\eta \\ \hat{n}_x\hat{n}_y(1 - cos\eta) - \hat{n}_z sin\eta & 2\hat{n}_y{}^2 - 1 & \hat{n}_y\hat{n}_z(1 - cos\eta) + \hat{n}_x sin\eta \\ \hat{n}_x\hat{n}_z(1 - cos\eta) + \hat{n}_y sin\eta & \hat{n}_y\hat{n}_z(1 - cos\eta) - \hat{n}_x sin\eta & 2\hat{n}_z{}^2 - 1 \end{pmatrix} \quad \textbf{(12)}$$

$$\Rightarrow \begin{pmatrix} \hat{n}_x \\ \hat{n}_y \\ \hat{n}_z \end{pmatrix} = \begin{pmatrix} \pm\sqrt{\frac{R_{xx} + 1}{2}} \\ \pm\sqrt{\frac{R_{yy} + 1}{2}} \\ \pm\sqrt{\frac{R_{zz} + 1}{2}} \end{pmatrix} \quad \textbf{(13)}$$

The sign ambiguity can be resolved by differencing across the leading diagonal and using the result that $sin\eta$ is nonnegative.

$$R_{yz} - R_{zy} = 2\hat{n}_x sin\eta \Rightarrow sign(\hat{n}_x) = sign\left(R_{yz} - R_{zy}\right) \quad \textbf{(14)}$$

$$R_{zx} - R_{xz} = 2\hat{n}_y sin\eta \Rightarrow sign(\hat{n}_y) = sign(R_{zx} - R_{xz}) \quad \textbf{(15)}$$

$$R_{xy} - R_{yx} = 2\hat{n}_z sin\eta \Rightarrow sign(\hat{n}_z) = sign(R_{xy} - R_{yx}) \quad \textbf{(16)}$$

Any rotation can be represented by its normalized rotation axis $\hat{\boldsymbol{n}}$ and the rotation angle η about that axis, giving a total of three degrees of freedom. The rotation vector $\eta\hat{\boldsymbol{n}}$ is the most efficient encoding of a rotation but has no useful mathematical properties. The rotation quaternion and rotation matrix are, therefore, used in preference.

The output of the gyroscope sensor (in degrees/second) multiplied by the sampling interval is, however, a rotation vector and is typically immediately converted to a rotation matrix or quaternion for orientation integration.

3.2 Rotation Matrix from Rotation Vector

Equation (1) defines the Rodrigues rotation matrix directly in terms of the rotation axis $\hat{\boldsymbol{n}}$ and rotation angle η. These can easily be determined for the case of non-zero rotation vector $\eta\hat{\boldsymbol{n}}$ using:

$$\eta = |\eta\hat{\boldsymbol{n}}| \quad \textbf{(17)}$$

$$\hat{\boldsymbol{n}} = \frac{\eta\hat{\boldsymbol{n}}}{|\eta\hat{\boldsymbol{n}}|} \quad \textbf{(18)}$$

If the rotation vector is zero then the rotation matrix is the identity matrix.

4 Accelerometer Tilt Orientation Matrix

4.1 Introduction

Accelerometer sensors are sensitive to both linear acceleration and gravity and are insensitive to rotation about the vertical gravity axis. An accelerometer operating without other sensors can, therefore, detect roll, pitch and tilt angles from horizontal, ***in the absence of linear acceleration***. An accelerometer cannot detect compass heading.

The mathematics below assumes that the yaw or compass angle is always zero degrees. One consequence of this assumption is that the orientation undergoes a twist of 180° at ±90° pitch (Aerospace / NED coordinate system) and at ±180° roll (in Android and Windows 8 coordinate systems) to ensure that the orientation remains pointing northward.

In the Aerospace / NED coordinate system, another way of describing the same phenomenon is to apply a 180° roll (resulting in the circuit board pointed northward but inverted) followed by a 180° yaw rotation (which has no effect because the accelerometer is insensitive to rotation about the vertical gravity axis). The same orientation can also be reached by a 180° pitch rotation. Without the 180° orientation twist at 90° pitch, the same final accelerometer reading would lead to two different orientations depending upon the path taken.

4.2 Aerospace / NED

This section documents the mathematics underlying the function `f3DOFTiltNED` in file *orientation.c*.

The Aerospace / NED orientation matrix for zero yaw angle ψ after rotation from the horizontal by pitch angle θ followed by roll angle ϕ is:

$$\boldsymbol{R}_{NED}(\psi = 0) = \begin{pmatrix} cos\,\theta & 0 & -sin\theta \\ sin\theta sin\phi & cos\,\phi & cos\,\theta\, sin\phi \\ cos\phi sin\theta & -sin\phi & cos\,\theta\, cos\phi \end{pmatrix} \tag{19}$$

The Aerospace / NED accelerometer reading $\boldsymbol{G}_s$ (where the subscript s indicates the measurement is in the rotated sensor frame) at this orientation can be computed by multiplying the reference gravity vector by $\boldsymbol{R}_{NED}(\psi = 0)$:

$$\begin{pmatrix} G_{sx} \\ G_{sy} \\ G_{sz} \end{pmatrix} = \begin{pmatrix} cos\,\theta & 0 & -sin\theta \\ sin\theta sin\phi & cos\,\phi & cos\,\theta\, sin\phi \\ cos\phi sin\theta & -sin\phi & cos\,\theta\, cos\phi \end{pmatrix} \begin{bmatrix} 0 \\ 0 \\ |\boldsymbol{G}_s| \end{bmatrix} \tag{20a}$$

$$= |\boldsymbol{G}_s| \begin{pmatrix} -sin\theta \\ sin\phi cos\theta \\ cos\phi cos\theta \end{pmatrix} = |\boldsymbol{G}_s| \begin{pmatrix} -sin\theta \\ sin\phi \sqrt{1 - \left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2} \\ cos\phi \sqrt{1 - \left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2} \end{pmatrix} \tag{20b}$$

$$sin\theta = \frac{-G_{sx}}{|\boldsymbol{G}_s|} \tag{21}$$

$$cos\theta = \sqrt{1 - \left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2} \tag{22}$$

$$sin\phi = \frac{G_{sy}}{|\boldsymbol{G}_s| \sqrt{1 - \left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2}} \tag{23}$$

$$cos\phi = \frac{G_{sz}}{|\boldsymbol{G}_s| \sqrt{1 - \left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2}} \tag{24}$$

The positive square root for $cos\theta$ can always be taken in equation (22) because the pitch angle θ in the Aerospace / NED coordinate system varies between –90° and +90° giving a nonnegative value of $cos\theta$.

Substituting back into equation (19) gives the Aerospace / NED 3DOF orientation matrix directly in terms of the accelerometer reading $\boldsymbol{G}_s$ as:

Accelerometer Tilt Orientation Matrix

$$\boldsymbol{R}_{NED}(\psi=0)=\begin{pmatrix} \frac{\sqrt{{G_{sy}}^2+{G_{sz}}^2}}{|\boldsymbol{G}_s|} & 0 & \frac{G_{sx}}{|\boldsymbol{G}_s|} \\ \frac{-G_{sx}G_{sy}}{|\boldsymbol{G}_s|\sqrt{{G_{sy}}^2+{G_{sz}}^2}} & \frac{G_{sz}}{\sqrt{{G_{sy}}^2+{G_{sz}}^2}} & \frac{G_{sy}}{|\boldsymbol{G}_s|} \\ \frac{-G_{sx}G_{sz}}{|\boldsymbol{G}_s|\sqrt{{G_{sy}}^2+{G_{sz}}^2}} & \frac{-G_{sy}}{\sqrt{{G_{sy}}^2+{G_{sz}}^2}} & \frac{G_{sz}}{|\boldsymbol{G}_s|} \end{pmatrix}=\begin{pmatrix} \frac{\sqrt{{G_{sy}}^2+{G_{sz}}^2}}{|\boldsymbol{G}_s|} & 0 & \frac{G_{sx}}{|\boldsymbol{G}_s|} \\ \frac{-R_{xz}R_{yz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2+{G_{sz}}^2}} & \frac{R_{zz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2+{G_{sz}}^2}} & \frac{G_{sy}}{|\boldsymbol{G}_s|} \\ \frac{-R_{xz}R_{zz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2+{G_{sz}}^2}} & \frac{-R_{yz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2+{G_{sz}}^2}} & \frac{G_{sz}}{|\boldsymbol{G}_s|} \end{pmatrix} \quad \textbf{(25)}$$

At the gimbal lock orientations with ±90° pitch rotation, where $G_{sy}=G_{sz}=0$ and $G_{sx}=|\boldsymbol{G}_s|$, equations (23) to (25) show that the roll angle ϕ and the rotation matrix $\boldsymbol{R}_{NED}$ are undefined. This simplest solution is to set the roll angle ϕ to zero giving:

$$\boldsymbol{R}_{NED}(\phi=\psi=0)=\begin{pmatrix} 0 & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & 0 \end{pmatrix}=\begin{pmatrix} 0 & 0 & \frac{G_{sx}}{|\boldsymbol{G}_s|} \\ 0 & 1 & 0 \\ \frac{-G_{sx}}{|\boldsymbol{G}_s|} & 0 & 0 \end{pmatrix}=\begin{pmatrix} 0 & 0 & sign(G_{sx}) \\ 0 & 1 & 0 \\ -sign(G_{sx}) & 0 & 0 \end{pmatrix} \quad \textbf{(26)}$$

4.3 Android

This section documents the mathematics underlying the function `f3DOFTiltAndroid` in file *orientation.c*.

The Android rotation matrix for zero yaw angle ψ after rotation from the horizontal by roll angle ϕ followed by pitch angle θ is:

$$\boldsymbol{R}_{Android}(\psi=0)=\begin{pmatrix} cos\,\phi & 0 & sin\phi \\ sin\phi sin\theta & cos\,\theta & -cos\,\phi\,sin\theta \\ -cos\,\theta\,sin\phi & sin\theta & cos\,\phi\,cos\,\theta \end{pmatrix} \quad \textbf{(27)}$$

The Android accelerometer reading at this orientation is:

$$\begin{pmatrix} G_{sx} \\ G_{sy} \\ G_{sz} \end{pmatrix}=|\boldsymbol{G}_s|\begin{pmatrix} sin\phi \\ -cos\phi\,sin\,\theta \\ cos\phi\,cos\,\theta \end{pmatrix}=|\boldsymbol{G}_s|\begin{pmatrix} sin\phi \\ -sin\,\theta\sqrt{1-\left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2} \\ cos\,\theta\sqrt{1-\left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2} \end{pmatrix} \quad \textbf{(28)}$$

$$sin\phi=\frac{G_{sx}}{|\boldsymbol{G}_s|} \quad \textbf{(29)}$$

$$cos\phi = \sqrt{1 - \left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2} \tag{30}$$

$$sin\,\theta = \frac{-G_{sy}}{|\boldsymbol{G}_s|\sqrt{1 - \left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2}} \tag{31}$$

$$cos\,\theta = \frac{G_{sz}}{|\boldsymbol{G}_s|\sqrt{1 - \left(\frac{G_{sx}}{|\boldsymbol{G}_s|}\right)^2}} \tag{32}$$

The positive square root for $cos\phi$ can always be taken in equation (30) because the roll angle ϕ in the Android coordinate system varies between –90° and +90° giving a non-negative value of $cos\phi$.

Substituting into equation (27) gives the 3DOF Android orientation matrix directly in terms of the accelerometer reading as:

$$\boldsymbol{R}_{Android}(\psi = 0) = \frac{1}{|\boldsymbol{G}_s|}\begin{pmatrix} \sqrt{{G_{sy}}^2 + {G_{sz}}^2} & 0 & G_{sx} \\ \frac{-G_{sx}G_{sy}}{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}} & \frac{G_{sz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}} & G_{sy} \\ \frac{-G_{sx}G_{sz}}{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}} & \frac{-G_{sy}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}} & G_{sz} \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}}{|\boldsymbol{G}_s|} & 0 & \frac{G_{sx}}{|\boldsymbol{G}_s|} \\ \frac{-R_{xz}R_{yz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}} & \frac{R_{zz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}} & \frac{G_{sy}}{|\boldsymbol{G}_s|} \\ \frac{-R_{xz}R_{zz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}} & \frac{-R_{yz}|\boldsymbol{G}_s|}{\sqrt{{G_{sy}}^2 + {G_{sz}}^2}} & \frac{G_{sz}}{|\boldsymbol{G}_s|} \end{pmatrix} \tag{33}$$

In the case of gimbal lock after ±90° roll rotation, when $G_{sy} = G_{sz} = 0$, the pitch angle θ is undefined and can be set to zero:

$$\boldsymbol{R}_{Android}(\theta = \psi = 0) = \begin{pmatrix} 0 & 0 & sin\phi \\ 0 & 1 & 0 \\ -sin\phi & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & \frac{G_{sx}}{|\boldsymbol{G}_s|} \\ 0 & 1 & 0 \\ \frac{-G_{sx}}{|\boldsymbol{G}_s|} & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & sign(G_{sx}) \\ 0 & 1 & 0 \\ -sign(G_{sx}) & 0 & 0 \end{pmatrix} \tag{34}$$

NOTE: These equations are identical to the equations for the Aerospace / NED coordinate systems.

4.4 Windows 8

This section documents the mathematics underlying the function `f3DOFTiltWin8` in file *orientation.c*.

The Windows 8 rotation matrix for zero yaw angle ψ, after rotation from the horizontal by pitch angle θ followed by roll angle ϕ, is:

Accelerometer Tilt Orientation Matrix

$$\boldsymbol{R}_{Win8}(\psi = 0) = \begin{pmatrix} cos\,\phi & sin\phi sin\theta & -sin\phi\,cos\,\theta \\ 0 & cos\,\theta & sin\theta \\ sin\phi & -cos\,\phi\,sin\theta & cos\,\phi\,cos\,\theta \end{pmatrix} \quad \textbf{(35)}$$

The Windows 8 accelerometer reading at this orientation is:

$$\begin{pmatrix} G_{sx} \\ G_{sy} \\ G_{sz} \end{pmatrix} = |\boldsymbol{G}_s| \begin{pmatrix} cos\,\theta\,sin\phi \\ -sin\,\theta \\ -cos\,\theta\,cos\phi \end{pmatrix} \quad \textbf{(36)}$$

$$tan\phi = \frac{-G_{sx}}{G_{sz}} \Rightarrow cos\phi = \frac{1}{\sqrt{1 + \left(\frac{G_{sx}}{G_{sz}}\right)^2}} \quad \textbf{(37)}$$

$$\Rightarrow sin\phi = tan\phi cos\phi = \frac{-G_{sx}}{G_{sz}\sqrt{1 + \left(\frac{G_{sx}}{G_{sz}}\right)^2}} \quad \textbf{(38)}$$

$$sin\,\theta = \frac{-G_{sy}}{|\boldsymbol{G}_s|} \quad \textbf{(39)}$$

$$cos\,\theta = \frac{-G_{sz}}{|\boldsymbol{G}_s| cos\phi} = \frac{-G_{sz}}{|\boldsymbol{G}_s|}\sqrt{1 + \left(\frac{G_{sx}}{G_{sz}}\right)^2} \quad \textbf{(40)}$$

The positive square root for $cos\phi$ can always be taken in equation (37) because the roll angle ϕ in the Windows 8 coordinate system varies between –90° and +90° resulting in a nonnegative value of $cos\phi$.

Substituting into equation (35) gives the Windows 8 3DOF orientation matrix directly in terms of the accelerometer reading as:

$$\boldsymbol{R}_{Win8}(\psi = 0) = \frac{1}{|\boldsymbol{G}_s|}\begin{pmatrix} \frac{|\boldsymbol{G}_s|}{\sqrt{1 + \left(\frac{G_{sx}}{G_{sz}}\right)^2}} & \frac{G_{sx}G_{sy}}{G_{sz}\sqrt{1 + \left(\frac{G_{sx}}{G_{sz}}\right)^2}} & -G_{sx} \\ 0 & -G_{sz}\sqrt{1 + \left(\frac{G_{sx}}{G_{sz}}\right)^2} & -G_{sy} \\ \frac{-G_{sx}|\boldsymbol{G}_s|}{G_{sz}\sqrt{1 + \left(\frac{G_{sx}}{G_{sz}}\right)^2}} & \frac{G_{sy}}{\sqrt{1 + \left(\frac{G_{sx}}{G_{sz}}\right)^2}} & -G_{sz} \end{pmatrix} \quad \textbf{(41)}$$

$$= \begin{pmatrix} \dfrac{G_{sz}}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}} & \dfrac{-G_{sx}R_{yz}}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}} & \dfrac{-G_{sx}}{|\boldsymbol{G}_s|} \\ 0 & \dfrac{-sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}}{|\boldsymbol{G}_s|} & \dfrac{-G_{sy}}{|\boldsymbol{G}_s|} \\ \dfrac{-G_{sx}}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}} & \dfrac{-G_{sy}R_{zz}}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}} & \dfrac{-G_{sz}}{|\boldsymbol{G}_s|} \end{pmatrix} \quad \textbf{(42)}$$

$$= \begin{pmatrix} \dfrac{-R_{zz}|\boldsymbol{G}_s|}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}} & \dfrac{R_{xz}R_{yz}|\boldsymbol{G}_s|}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}} & \dfrac{-G_{sx}}{|\boldsymbol{G}_s|} \\ 0 & \dfrac{-1}{\left\{\dfrac{|\boldsymbol{G}_s|}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}}\right\}} & \dfrac{-G_{sy}}{|\boldsymbol{G}_s|} \\ \dfrac{R_{xz}|\boldsymbol{G}_s|}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}} & \dfrac{R_{yz}R_{zz}|\boldsymbol{G}_s|}{sign(G_{sz})\sqrt{{G_{sx}}^2 + {G_{sz}}^2}} & \dfrac{-G_{sz}}{|\boldsymbol{G}_s|} \end{pmatrix} \quad \textbf{(43)}$$

In the case of gimbal lock after ±90° pitch rotation, when $G_{sx} = G_{sz} = 0$, the roll angle ϕ is undefined and can be set to zero:

$$\boldsymbol{R}_{Win8}(\phi = \psi = 0) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & sin\theta \\ 0 & -sin\theta & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & \dfrac{-G_{sy}}{|\boldsymbol{G}_s|} \\ 0 & \dfrac{G_{sy}}{|\boldsymbol{G}_s|} & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -sign(G_{sy}) \\ 0 & sign(G_{sy}) & 0 \end{pmatrix} \quad \textbf{(44)}$$

5 Magnetometer Compass Orientation Matrix

5.1 Introduction

This section documents the 2D, magnetometer-only, eCompass functions that calculate the orientation matrix from calibrated magnetometer measurements with the assumption that the magnetometer is always flat and has zero tilt away from the horizontal. The market application for this arrangement is an automotive compass where the normal accelerometer plus magnetometer tilt-compensated eCompass functions cannot be used because of vehicle acceleration, braking and cornering forces.

Use these functions with caution. In the automotive application, the 2D eCompass will be robust against the effects of acceleration, braking and cornering forces because the magnetometer sensor is unaffected by acceleration. However, the compass heading will change when the vehicle tilt angle,

whether roll or pitch, deviates from horizontal, violating the assumption that the magnetometer sensor is always flat.

5.2 Aerospace / NED

This section documents the mathematics underlying the function `f3DOFMagnetometerMatrixNED` in file *orientation.c*.

The Aerospace / NED rotation matrix for rotation in yaw angle ψ only while remaining horizontal is:

$$\boldsymbol{R}_{NED}(\psi) = \begin{pmatrix} cos\,\psi & sin\psi & 0 \\ -sin\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{45}$$

The Aerospace / NED calibrated magnetometer reading at this orientation is:

$$\begin{pmatrix} B_{cx} \\ B_{cy} \\ B_{cz} \end{pmatrix} = \begin{pmatrix} cos\,\psi & sin\psi & 0 \\ -sin\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} B \begin{pmatrix} cos\delta \\ 0 \\ sin\delta \end{pmatrix} = B \begin{pmatrix} cos\,\psi\, cos\delta \\ -sin\psi cos\delta \\ sin\delta \end{pmatrix} \tag{46}$$

The horizontal component of the geomagnetic field has the value:

$$Bcos\delta = \sqrt{{B_{cx}}^2 + {B_{cy}}^2} \tag{47}$$

The yaw ψ (and compass heading angle) is given by:

$$tan\psi = \frac{sin\psi}{cos\psi} = \frac{-B_{cy}}{B_{cx}} \tag{48}$$

$$sin\psi = \frac{-B_{cy}}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \tag{49}$$

$$cos\psi = \frac{B_{cx}}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \tag{50}$$

The orientation matrix is independent of the unknown geomagnetic inclination angle δ and has the value:

$$\boldsymbol{R}_{NED}(\psi) = \frac{1}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \begin{pmatrix} B_{cx} & -B_{cy} & 0 \\ B_{cy} & B_{cx} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{51}$$

5.3 Android

This section documents the mathematics underlying the function `f3DOFMagnetometerMatrixAndroid` in file *orientation.c*.

The Android rotation matrix for rotation in yaw angle ψ only while remaining horizontal is:

$$\boldsymbol{R}_{Android}(\psi) = \begin{pmatrix} cos\,\psi & -sin\psi & 0 \\ sin\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{52}$$

The Android calibrated magnetometer reading at this orientation is:

$$\begin{pmatrix} B_{cx} \\ B_{cy} \\ B_{cz} \end{pmatrix} = \begin{pmatrix} cos\,\psi & -sin\psi & 0 \\ sin\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} B \begin{pmatrix} 0 \\ cos\delta \\ -sin\delta \end{pmatrix} = B \begin{pmatrix} -sin\psi cos\delta \\ cos\psi cos\delta \\ -sin\delta \end{pmatrix} \tag{53}$$

The horizontal component of the geomagnetic field has the value:

$$Bcos\delta = \sqrt{{B_{cx}}^2 + {B_{cy}}^2} \tag{54}$$

The yaw ψ and compass heading angles are given by:

$$tan\psi = \frac{sin\psi}{cos\psi} = \frac{-B_{cx}}{B_{cy}} \tag{55}$$

$$sin\psi = \frac{-B_{cx}}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \tag{56}$$

$$cos\psi = \frac{B_{cy}}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \tag{4}$$

The orientation matrix is independent of the unknown geomagnetic inclination angle δ and has the value:

$$\boldsymbol{R}_{Android}(\psi) = \frac{1}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \begin{pmatrix} B_{cy} & B_{cx} & 0 \\ -B_{cx} & B_{cy} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{58}$$

5.4 Windows 8

This section documents the mathematics underlying the function `f3DOFMagnetometerMatrixWin8` in file *orientation.c.*

The Windows 8 rotation matrix for rotation in yaw angle ψ only while remaining horizontal is:

$$\boldsymbol{R}_{Win8}(\psi) = \begin{pmatrix} cos\,\psi & sin\psi & 0 \\ -sin\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{59}$$

The Windows 8 calibrated magnetometer reading at this orientation is:

$$\begin{pmatrix} B_{cx} \\ B_{cy} \\ B_{cz} \end{pmatrix} = \begin{pmatrix} cos\,\psi & sin\psi & 0 \\ -sin\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} B \begin{pmatrix} 0 \\ cos\delta \\ -sin\delta \end{pmatrix} = B \begin{pmatrix} sin\psi cos\delta \\ cos\psi cos\delta \\ -sin\delta \end{pmatrix} \tag{60}$$

The horizontal component of the geomagnetic field has the value:

$$\boldsymbol{B cos\delta} = \sqrt{\boldsymbol{B_{cx}}^2 + \boldsymbol{B_{cy}}^2} \tag{61}$$

The yaw ψ angle, which is equal to minus the compass heading in the Windows 8 coordinate system, is given by:

$$tan\psi = \frac{sin\psi}{cos\psi} = \frac{B_{cx}}{B_{cy}} \tag{62}$$

$$sin\psi = \frac{B_{cx}}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \tag{63}$$

$$cos\psi = \frac{B_{cy}}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \tag{64}$$

The orientation matrix is independent of the unknown geomagnetic inclination angle δ and has the value:

$$\boldsymbol{R}_{Win8}(\psi) = \frac{1}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2}} \begin{pmatrix} B_{cy} & B_{cx} & 0 \\ -B_{cx} & B_{cy} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{65}$$

6 Accelerometer Magnetometer Tilt eCompass Orientation Matrix

6.1 Introduction

This section documents the calculation of the orientation matrix from accelerometer and magnetometer measurements. This is the classic tilt-compensated eCompass which provides a complete determination of orientation. The assumption is that there is no linear acceleration so that the accelerometer reading is a function of orientation in the earth's gravitational field only.

The orientation matrix can be calculated with an algorithm which is most easily understood from the trivial mathematical identity below:

$$\begin{pmatrix} R_{xx} & R_{xy} & R_{xz} \\ R_{yx} & R_{yy} & R_{yz} \\ R_{zx} & R_{zy} & R_{zz} \end{pmatrix} = \begin{pmatrix} R_{xx} & R_{xy} & R_{xz} \\ R_{yx} & R_{yy} & R_{yz} \\ R_{zx} & R_{zy} & R_{zz} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{66}$$

The three columns of the identity matrix on the right hand side are the three unit vectors in the *x*, *y* and *z* directions aligned with the initial orientation in the global reference frame. The accelerometer measures the gravity vector rotated into its coordinate system and, therefore, provides the right-hand column of the orientation matrix. The geomagnetic vector in the global frame points northward and downward (upward) in the northern (southern) hemisphere. The vector product between the rotated gravity and rotated geomagnetic vector is, therefore, the rotated easterly-pointing vector from the global frame which forms the *y* (Aerospace / NED coordinate system) or *x* (Android and Windows 8 coordinate systems) column vector in the orientation matrix. Because the orientation matrix is orthonormal, a final vector product between the two vectors determined so far gives the third column vector.

6.2 Aerospace / NED

This section documents the mathematics underlying the function `feCompassNED` in file *orientation.c*.

The *z* axis in the initial NED orientation is aligned downwards and parallel to gravity. The *z* axis column vector of the orientation matrix is the rotated gravity vector which, on the assumption of zero acceleration, is the normalized accelerometer measurement $\boldsymbol{G}_s$:

$$\begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} = \frac{\boldsymbol{G}_s}{|\boldsymbol{G}_s|} = \frac{1}{\sqrt{{G_{sx}}^2 + {G_{sy}}^2 + {G_{sz}}^2}} \begin{pmatrix} G_{sx} \\ G_{sy} \\ G_{sz} \end{pmatrix} \tag{67}$$

The geomagnetic vector, estimated from the calibrated magnetometer reading, points northwards in the initial orientation (in the *x* direction for the NED coordinate system) and downward (upward) in the northern (southern) hemisphere. The vector product of the rotation matrix *z* column vector and the calibrated magnetometer vector gives a vector orthogonal to both which is, therefore, aligned along the east pointing *y* axis:

$$\begin{pmatrix} R_{xy} \\ R_{yy} \\ R_{zy} \end{pmatrix} = \begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} \times \frac{\boldsymbol{B}_c}{|\boldsymbol{B}_c|} = \begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} \times \left\{ \frac{1}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2 + {B_{cz}}^2}} \begin{pmatrix} B_{cx} \\ B_{cy} \\ B_{cz} \end{pmatrix} \right\} \quad \textbf{(68)}$$

Finally, the vector product of the rotation matrix *y* and *z* axes gives the remaining *x* column of the orientation matrix:

$$\begin{pmatrix} R_{xx} \\ R_{yx} \\ R_{zx} \end{pmatrix} = \begin{pmatrix} R_{xy} \\ R_{yy} \\ R_{zy} \end{pmatrix} \times \begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} \quad \textbf{(69)}$$

The geomagnetic inclination angle δ is the angle by which the geomagnetic field dips below horizontal in the global frame. Because the scalar product of two vectors is invariant under rotation, the inclination angle is determined from the scalar product of the normalized accelerometer and calibrated magnetic measurements as:

$$\boldsymbol{G}_s \cdot \boldsymbol{B}_c = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot B \begin{pmatrix} cos\delta \\ 0 \\ sin\delta \end{pmatrix} = Bsin\delta \Rightarrow sin\delta = \frac{G_{sx}B_{cx} + G_{sy}B_{cy} + G_{sz}B_{cz}}{|\boldsymbol{G}_s||\boldsymbol{B}_c|} \quad \textbf{(70)}$$

6.3 Android

This section documents the mathematics underlying the function `feCompassAndroid` in file *orientation.c*.

The accelerometer reading in the Android coordinate system in the initial orientation is also parallel to the *z* axis because the Android coordinate system is ENU with the *z* axis upward and is acceleration, not gravity, positive:

$$\begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} = \frac{\boldsymbol{G}_s}{|\boldsymbol{G}_s|} = \frac{1}{\sqrt{{G_{sx}}^2 + {G_{sy}}^2 + {G_{sz}}^2}} \begin{pmatrix} G_{sx} \\ G_{sy} \\ G_{sz} \end{pmatrix} \quad \textbf{(71)}$$

The geomagnetic vector, estimated from the calibrated magnetometer reading, points northwards in the initial orientation (in the *y* direction for the Android coordinate system) and downward (upward) in the northern (southern) hemisphere. The vector product of the rotation matrix *z* column vector and the calibrated magnetometer vector gives a vector orthogonal to both aligned along the north pointing *x* axis:

$$\begin{pmatrix} R_{xx} \\ R_{yx} \\ R_{zx} \end{pmatrix} = \frac{\boldsymbol{B}_c}{|\boldsymbol{B}_c|} \times \begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} = \left\{ \frac{1}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2 + {B_{cz}}^2}} \begin{pmatrix} B_{cx} \\ B_{cy} \\ B_{cz} \end{pmatrix} \right\} \times \begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} \quad \textbf{(72)}$$

Finally, the vector product of the rotation matrix *z* and *x* axes gives the remaining *y* component of the orientation matrix:

$$\begin{pmatrix} R_{xy} \\ R_{yy} \\ R_{zy} \end{pmatrix} = \begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} \times \begin{pmatrix} R_{xx} \\ R_{yx} \\ R_{zx} \end{pmatrix} \tag{73}$$

In the Android coordinate system, the inclination angle is related to the scalar product of the accelerometer and calibrated magnetic measurements by:

$$\boldsymbol{G}_s \cdot \boldsymbol{B}_c = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot B \begin{pmatrix} 0 \\ cos\delta \\ -sin\delta \end{pmatrix} = -Bsin\delta \Rightarrow sin\delta = \frac{-\left(G_{sx}B_{cx} + G_{sy}B_{cy} + G_{sz}B_{cz}\right)}{|\boldsymbol{G}_s||\boldsymbol{B}_c|} \tag{74}$$

6.4 Windows 8

This section documents the mathematics underlying the function `feCompassWin8` in file *orientation.c.*

The accelerometer reading in the Windows 8 coordinate system in the initial orientation is anti-parallel to the *z* axis because the Windows 8 coordinate system is ENU with the *z* axis upward and is gravity (not acceleration) positive:

$$\begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} = \frac{-1}{\sqrt{{G_{sx}}^2 + {G_{sy}}^2 + {G_{sz}}^2}} \begin{pmatrix} G_{sx} \\ G_{sy} \\ G_{sz} \end{pmatrix} \tag{75}$$

The geomagnetic vector, or the calibrated magnetometer reading, in the default orientation points northward (in the *y* direction for the Windows 8 coordinate system) and downward (upward) in the northern (southern) hemisphere. The vector product of the Windows 8 *y* and *z* vectors gives a vector orthogonal to both aligned along the *x* axis:

$$\begin{pmatrix} R_{xx} \\ R_{yx} \\ R_{zx} \end{pmatrix} = \left\{ \frac{1}{\sqrt{{B_{cx}}^2 + {B_{cy}}^2 + {B_{cz}}^2}} \begin{pmatrix} B_{cx} \\ B_{cy} \\ B_{cz} \end{pmatrix} \right\} \times \begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} \tag{76}$$

Finally, the vector product of the *z*- and *x* axes gives the remaining *y* component of the orientation matrix:

$$\begin{pmatrix} R_{xy} \\ R_{yy} \\ R_{zy} \end{pmatrix} = \begin{pmatrix} R_{xz} \\ R_{yz} \\ R_{zz} \end{pmatrix} \times \begin{pmatrix} R_{xx} \\ R_{yx} \\ R_{zx} \end{pmatrix} \tag{77}$$

In the Windows 8 coordinate system, the inclination angle is determined from the scalar product of the normalized accelerometer and calibrated magnetic measurements as:

$$\boldsymbol{G}_s \cdot \boldsymbol{B}_c = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} \cdot B \begin{pmatrix} 0 \\ cos\delta \\ -sin\delta \end{pmatrix} = Bsin\delta \Rightarrow sin\delta = \frac{G_{sx}B_{cx} + G_{sy}B_{cy} + G_{sz}B_{cz}}{|\boldsymbol{G}_s||\boldsymbol{B}_c|} \quad \textbf{(78)}$$

7 Revision History

Table 1. Revision history

Rev. No.	Date	Description
1	10/2015	Initial release

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support