Computing Euler Angles from Direction Cosines

William Premerlani

Axis conventions

To describe the motion of an airplane it is necessary to define a suitable coordinate system. For most problems dealing with aircraft motion, two coordinate systems are used. One coordinate system is fixed to the earth and may be considered for the purpose of aircraft motion analysis to be an inertial coordinate system. The other coordinate system is fixed to the airplane and is referred to as a body coordinate system. Figure 1 shows the two right-handed coordinate systems.

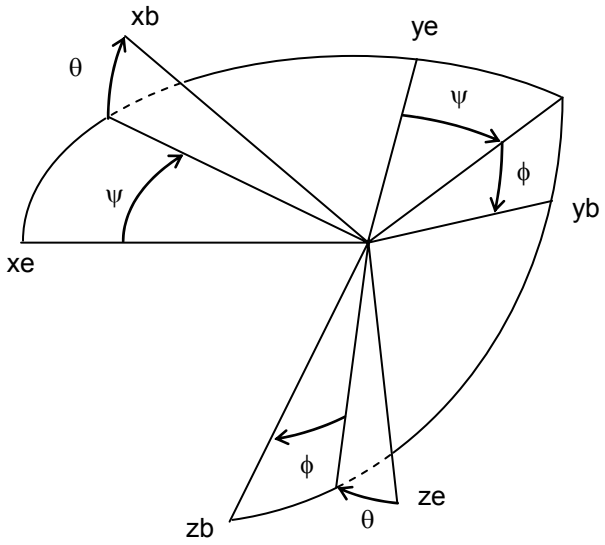

Figure 1 Body fixed frame and earth fixed frame

The orientation of the airplane is often described by three consecutive rotations, whose order is important. The angular rotations are called the Euler angles. The orientation of the body frame with respect to the fixed earth frame can be determined in the following manner. Imagine the airplane to be positioned so that the body axis system is parallel to the fixed frame and then apply the following rotations:

1. Rotate the body about its zb axis through the yaw angle ψ
2. Rotate the body about its yb axis through the pitch angle θ
3. Rotate the body about its xb axis through the roll angle ϕ

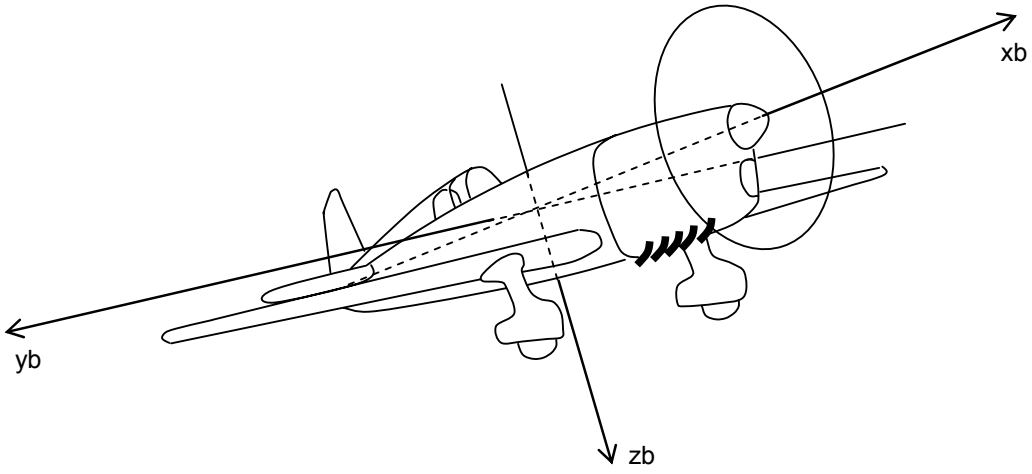

Figure 2 Body axes coordinate system

Direction cosine matrices

Certain types of vectors, such as directions, velocities, accelerations, and translations, (movements) can be transformed between rotated reference frames with a 3X3 matrix. We are interested in the plane frame of reference and the ground frame of reference. It is possible to rotate vectors by multiplying them by a matrix of direction cosines:

$$\mathbf{Q} = \begin{bmatrix} Q_x \\ Q_y \\ Q_z \end{bmatrix} = \text{a vector, such as a direction, velocity or acceleration}$$

$$\mathbf{Q_P} = \text{a vector } \mathbf{Q} \text{ measured in the frame of reference of the plane}$$

$$\mathbf{Q_G} = \text{a vector } \mathbf{Q} \text{ measured in the frame of reference of the ground} \qquad \text{Eqn. 1}$$

$$\mathbf{R} = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix} = \text{rotation matrix}$$

$$\mathbf{Q_G} = \mathbf{R}\mathbf{Q_P}$$

The relation between the direction cosine matrix and Euler angles is:

$$\mathbf{R} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \qquad \text{Eqn. 2}$$

Equation 1 and equation 2 express how to rotate a vector measured in the frame of reference of the plane to the frame of reference of the ground. Equation 1 is expressed in terms of direction cosines. Equation 2 is expressed in terms of Euler angles.

If we have the full direction cosine matrix, we can convert to Euler angles from the last row and the first column of the matrix:

$$\begin{aligned} \theta &= -\arcsin\left(r_{zx}\right) \\ \phi &= \operatorname{atan2}\left(r_{zy}, r_{zz}\right) \\ \psi &= \operatorname{atan2}\left(r_{yx}, r_{xx}\right) \end{aligned} \qquad \text{Eqn. 3}$$

The pitch angle is between -90 degrees and +90 degrees. Note that we must use atan2 in order to get a four quadrant result. Finally, note that the atan2 function takes its arguments as (y,x), not (x,y). That often leads to confusion.

Some people have run into problems in trying to apply equation 3 to computing Euler angles using the direction cosines that are computed by the firmware, such as MatrixPilot, that runs on the UAV DevBoard.

The main problem is that the UAV DevBoard does not use the coordinate system shown in figures 1 and 2. Instead, the UAV DevBoard uses yb where xb is in Figure 2, and it uses –xb where yb is in Figure 2. The reason for that is historical: the axis labels were placed on the board to align with the axes of the three axis accelerometer chip. Later, we discovered that was not the convention, but by then it was too late. The relationship between the direction cosines used in the UAV DevBoard, and the ones given in Equation 1 and Equation 2 is:

$$\mathbf{R}' = \begin{bmatrix} rmat[0] & rmat[1] & rmat[2] \\ rmat[3] & rmat[4] & rmat[5] \\ rmat[6] & rmat[7] & rmat[8] \end{bmatrix} = \begin{bmatrix} r_{yy} & -r_{yx} & -r_{yz} \\ -r_{xy} & r_{xx} & r_{xz} \\ -r_{zy} & r_{zx} & r_{zz} \end{bmatrix} \qquad \text{Eqn. 4}$$

So, in terms of the elements of the array rmat[], the Euler angles are given by:

$$\begin{aligned} \theta &= -\arcsin\left(rmat[7]\right) \\ \phi &= \operatorname{atan2}\left(-rmat[6], rmat[8]\right) \\ \psi &= \operatorname{atan2}\left(-rmat[1], rmat[4]\right) \end{aligned} \qquad \text{Eqn. 5}$$

By the way, the latest versions of MatrixPilot offer the option of including Euler angles in telemetry output, and it uses a clever way of doing the computations of equation 5. First, a highly efficient CORDIC method is used to compute atan2. Furthermore, the arcsin computation is avoided all together, because one of the byproducts of the CORDIC atan2 is the production of the square root of the sum of the squares of the x and y arguments. So, the calculation proceeds as follows:

$$\left(\sqrt{rmat[6]^2 + rmat[8]^2}, \phi\right) = CORDIC\,\text{atan}\,2\left(-rmat[6], rmat[8]\right)$$
$$\theta = \text{atan}\,2\left(-rmat[7], \sqrt{rmat[6]^2 + rmat[8]^2}\right) \qquad \text{Eqn. 6}$$
$$\psi = \text{atan}\,2\left(-rmat[1], rmat[4]\right)$$

Also note in equation 6 that it is not necessary to take into account the scaling of the representation of the rmat[] elements. That is because atan2 gives you the same result if you scale both of its arguments by the same factor.

Here is the actual code:

```
matrix_accum.x = rmat[8] ;
matrix_accum.y = -rmat[6] ;
earth_roll = rect_to_polar(&matrix_accum) ;

matrix_accum.y = -rmat[7] ;
earth_pitch = rect_to_polar(&matrix_accum) ;

matrix_accum.x = rmat[4] ;
matrix_accum.y = -rmat[1] ;
earth_yaw = rect_to_polar(&matrix_accum) ;
```

The variable matrix_accum is a structure with a y component and an x component. rmat[1,4,6,7,8] are values of the direction cosines used in the firmware. The routine rect_to_polar converts from x and y coordinates into magnitude and angle. The structure is passed in by address so that it can be modified. The routine rect_to_polar returns the angle of x and y. In the process of the calculation, the y coordinate is driven to zero, and the x coordinate becomes the magnitude.

.