Yaw, Pitch, Roll and Omega, Phi, Kappa angles and conversion

Pix4D product documentation

January 2021

Content reflects behaviour of Pix4D products, it is in correspondence with [Pix4D Support Pages]. Notations are used as in [Baumker].

1 YPR - Yaw, Pitch and Roll

In inertial navigation Yaw (ψ), Pitch (θ) and Roll (ϕ) are used to transform a vector from a body coordinate system b to the navigation system n via the matrix C_b^n defined by the counter-clockwise rotation angles ψ , θ and ϕ:

$$
\begin{aligned}
C_b^n &= R_z(\psi)R_y(\theta)R_x(\phi) \qquad (1)\\
&= \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{pmatrix} \\
&= \begin{pmatrix} \cos\psi\,\cos\theta & \cos\psi\,\sin\theta\,\sin\phi - \sin\psi\,\cos\phi & \cos\psi\,\sin\theta\,\cos\phi + \sin\psi\,\sin\phi \\ \sin\psi\,\cos\theta & \sin\psi\,\sin\theta\,\sin\phi + \cos\psi\,\cos\phi & \sin\psi\,\sin\theta\,\cos\phi - \cos\psi\,\sin\phi \\ -\sin\theta & \cos\theta\,\sin\phi & \cos\theta\,\cos\phi \end{pmatrix}
\end{aligned}
$$

Both systems are cartesian. The **navigation system** n has the same origin as the body coordinate system, and it changes depending on the position on earth. It is determined with respect to a centered earth fixed coordinate (ECEF) system. It has the NED axes convention:

$$
\begin{aligned}
x^n &= \text{pointing north} \qquad (2)\\
y^n &= \text{pointing east} \\
z^n &= \text{pointing down (vertical in direction of the plumb line)}
\end{aligned}
$$

The **body system** (imagine an aircraft) is defined by the axes:

$$
\begin{aligned}
x^b &= \text{body front} \qquad (3)\\
y^b &= \text{body right} \\
z^b &= \text{body bottom}
\end{aligned}
$$

For $\psi = \theta = \phi = 0$ both systems coincide, i.e. the plane heading north, with its right wing pointing east.

2 OPK - Omega, Phi and Kappa

In the context of photogrammetry Omega (ω) , Phi (φ) and Kappa (κ) are used to transform a vector from an image coordinate system B to an object system E.[1] The Pix4D rotation transformation matrix C_B^E is defined by counter-clockwise rotation angles ω , φ and κ:

$$\begin{aligned} C_B^E &:= R_x(\omega)R_y(\varphi)R_z(\kappa) \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\omega & -\sin\omega \\ 0 & \sin\omega & \cos\omega \end{pmatrix} \begin{pmatrix} \cos\varphi & 0 & \sin\varphi \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\varphi \end{pmatrix} \begin{pmatrix} \cos\kappa & -\sin\kappa & 0 \\ \sin\kappa & \cos\kappa & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} \cos\varphi\,\cos\kappa & -\cos\varphi\,\sin\kappa & \sin\varphi \\ \cos\omega\,\sin\kappa + \sin\omega\,\sin\varphi\,\cos\kappa & \cos\omega\,\cos\kappa - \sin\omega\,\sin\varphi\,\sin\kappa & -\sin\omega\,\cos\varphi \\ \sin\omega\,\sin\kappa - \cos\omega\,\sin\varphi\,\cos\kappa & \sin\omega\,\cos\kappa + \cos\omega\,\sin\varphi\,\sin\kappa & \cos\omega\,\cos\varphi \end{pmatrix} \end{aligned} \tag{4}$$

Both systems are cartesian. The **object system** E can have a different origin from the image coordinate system, i.e. the matrix describes the transform of rotations from one system to another, but not of positions. In contrast to the navigation system, the object system does **not** change with its position on earth. Generally the object system is used locally to describe many camera orientations and chosen such that the origin lies in the center of the cameras. It has the ENU axes convention:

$$\begin{aligned} x^E &= \text{pointing east (not north!)} \\ y^E &= \text{pointing north (not east!)} \\ z^E &= \text{pointing up (towards the zenith, not down!)} \end{aligned} \tag{5}$$

Note that the x and y axes only correspond to true local east and north axes at the tangent point, but not at all points in the cartesian coordinate system.

The **image coordinate system** B is defined via the axes:

$$\begin{aligned} x^B &= \text{camera/image right (looking through the camera/image)} \\ y^B &= \text{camera/image top (looking through the camera/image)} \\ z^B &= \text{camera back (opposite to viewing direction through camera)} \end{aligned} \tag{6}$$

For $\omega = \varphi = \kappa = 0$ both systems coincide, i.e. the camera is looking down, with the image top pointing north and the right of the image east. Note that the x/y axes of this (3D) image coordinate system are not aligned with the standard (2D) image coordinate system that is used to describe (pixel) positions on an image: The y-axis is flipped.

[1] [Baumker] wrongly states the inverse.

3 Converting from YPR to OPK

Given a object coordinate system and its relationship to an ECEF system has been established and a matrix C_B^b relating image coordinate and body system, YPR data can be converted to OPK individually for each camera at position p (i.e. with navigation system n(p)) as follows:

1. YPR angles at a point p $\rightarrow$ YPR rotation matrix $C_b^{n(p)}$
2. YPR rotation matrix $C_b^{n(p)} \rightarrow$ OPK rotation matrix C_B^E
3. OPK rotation matrix $C_B^E \rightarrow$ OPK angles

Step 1 is straightforward following section 1. The matrix C_B^E in step 2 can be computed via

$$C_B^E = C_{n(p)}^E \cdot C_b^{n(p)} \cdot C_B^b \tag{7}$$

Then, $C_{n(p)}^E$ needs to be computed. Let $(lat, long)$ be the coordinates at point p and $\delta > 0$ sufficiently small. Using the function $pos : Lat \times Long \times Alt \rightarrow E$ relating object coordinate system and ECEF system, we can compute

$$x^{n(p)} = \frac{p_1 - p_2}{||p_1 - p_2||}, \quad p_1 = pos(lat + \delta, long, alt), \quad p_2 = pos(lat - \delta, long, alt) \tag{8}$$

a unit vector in the North direction. Following the definitions in sections 1 and 2, $z^{n(p)} = (0, 0, -1)^T$, and $y^{n(p)} = z^{n(p)} \times x^{n(p)}$. This gives the transform

$$C_{n(p)}^E = \left(x^{n(p)} | y^{n(p)} | z^{n(p)} \right) \tag{9}$$

After computation of the rotation matrix $C_B^E = \begin{pmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{pmatrix}$, OPK angles can be extracted using the formulas:

$$\omega = \arctan2\left(\frac{-c_{23}}{c_{33}}\right), \qquad \varphi = \arcsin\left(c_{13}\right), \qquad \kappa = \arctan2\left(\frac{-c_{12}}{c_{11}}\right) \tag{10}$$

In Pix4D we assume by default (`perpCamera=true`) that the camera is mounted in a way, that the image top y^B points in flying direction x^b, and the camera is looking down, i.e. $-z_B$ and z_b have the same direction. This means we use

$$C_B^b = C_b^B = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{pmatrix} \tag{11}$$

to describe the relation between image and body coordinate system. We also provide a YPR mode (`perpCamera=false`) for a camera mounted in a way that the image top y^B points right y^b and the camera is looking up, i.e. z_B and z_b have the same direction.

References

[Baumker] Baumker, Manfred and Heimes, F., New Calibration and Computing Method for Direct Georeferencing of Image and Scanner Data Using the Position and Angular Data of an Hybrid Inertial Navigation System, Proceedings of OEEPE Workshop on Integrated Sensor Orientation, 2002.

[Pix4D Support Pages] Pix4D Support Team, Yaw, Pitch, Roll and Omega, Phi, Kappa angles,
How are the Internal and External Camera Parameters defined?