Student Satellite Project
Indian Institute of Technology, Bombay
Powai, Mumbai - 400076, INDIA

Website: www.aero.iitb.ac.in/satlab

Readme file for frames.py

Attitude Determination and Control Subsystem

latlon (x)

Author: Anant Joshi
Date:
To obtain the latitude and longitude in degrees given the position in earth centered earth fixed (ECEF) frame.
Input: position in ECEF frame (ndarray)
Output: latitude and longitude in degrees (tuple)
In these calculation it is assumed that the shape of earth is perfectly spherical.
Latitude is a geographic coordinate that specifies the north–south position of a point on the Earth's surface. These form circles parallel to equator on earth's surface. Equator is 0^0 latitude. It range from 0^0 to 90^0 from equator to north pole and 0^0 to -90^0 from equator to south pole.

$$latitude = \begin{cases} \cos^{-1}(\sqrt{\frac{x^2+y^2}{x^2+y^2+z^2}}) & z \geq 0 \\ -\cos^{-1}(\sqrt{\frac{x^2+y^2}{x^2+y^2+z^2}}) & z < 0 \end{cases}$$

Longitude is given as an angular measurement ranging from 0^0 at the Prime Meridian to $+180^0$ eastward and -180^0 westward.
For $y = 0$ prime meridian $(x > 0)$ is zero longitude and anti-prime-meridian is 180^0 longitude.Thus

$$longitude = \begin{cases} 0 & y = 0, x \geq 0 \\ 180 & y = 0, x < 0 \\ \cos^{-1}\frac{x}{\sqrt{x^2+y^2}} & y > 0 \\ -\cos^{-1}\frac{x}{\sqrt{x^2+y^2}} & y < 0 \end{cases}$$

sgn (x)

Author: Anant Joshi
Date:
This is signum function.

$$sgn(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases}$$

ecif2ecef(v_x_i,t)

Author: Anant Joshi
Date:
Input: vector with components in eci frame and time since epoch in seconds
Output: vector with components in ecef frame

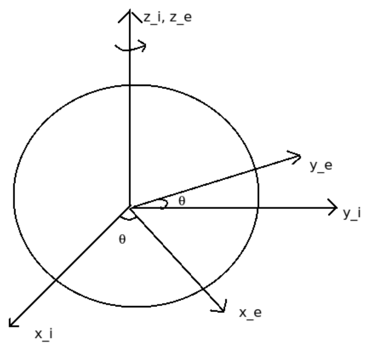

Figure 1: ECIF and ECEF

From this figure the co-ordinate transformation can be written as

$$v_e = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} v_I$$

ecif2ecefR(t)

Author: Anant Joshi
Date:
Input: time since epoch in seconds
Output: returns co-ordinate transformation matrix
This function is same as ecif2ecef. Only difference is that the time used is sidereal time. To obtain sidereal time multiply the time by STEPRUT (1.002738).

ecef2ecif(v_x_e,t)

Author: Anant Joshi
Date:
Input : vector components in ecef, time since epoch in seconds Output : vector components in eci frame

From figure 1, the co-ordinate transformation is

$$v_I = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} v_e$$

ecif2orbit(v_pos_i,v_vel_i,v_x_i)

Author: Sanket Chirame
Date:
Input : vector components in ecif, position and velocity in eci frame
Output : vector components in orbit frame

$$\vec{z} = -\vec{r_I}/|\vec{r_I}|$$
$$\vec{y} = \frac{\vec{v_I} \times \vec{r_I}}{|\vec{v_I} \times \vec{r_I}|}$$
$$\vec{x} = \frac{y \times z}{|y \times z|}$$
$$\vec{v}_o = [\,\vec{x}\;\vec{y}\;\vec{z}\,]^T\;\vec{v}_i$$

qBI2qBO(v_q_BI,v_pos_i,v_vel_i)

Author: Sanket Chirame
Date:
Input : Unit quaternion which rotates ecif vector to body frame, position and velocity in ecif
Output : Unit quaternion which rotates orbit frame vector to body frame From the definition of orbit frame

$$\vec{z} = -\vec{r_I}/|\vec{r_I}|$$
$$\vec{y} = \frac{\vec{v_I} \times \vec{r_I}}{|\vec{v_I} \times \vec{r_I}|}$$
$$\vec{x} = \frac{y \times z}{|y \times z|}$$

Where $\vec{x},\ \vec{y},\ \vec{z}$ are column vector in contrast to numpy arrays in code of dimension 1×3.

$$DCM_{IO} = [\,\vec{x}\;\vec{y}\;\vec{z}\,]$$

q_{IO} is calculated using this DCM using function $rotmat2quat$ function in this module.
The composition of quaternion is calculated using formula given in reference [1], (Equation 3.97),

$$\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} = \begin{bmatrix} \beta_0'' & -\beta_1'' & -\beta_2'' & -\beta_3'' \\ \beta_1'' & \beta_0'' & \beta_3'' & -\beta_2'' \\ \beta_2'' & -\beta_3'' & \beta_0'' & \beta_1'' \\ \beta_3'' & \beta_2'' & -\beta_1'' & \beta_0'' \end{bmatrix} \begin{bmatrix} \beta_0' \\ \beta_1' \\ \beta_2' \\ \beta_3' \end{bmatrix} \tag{1}$$

where $\vec{\beta} = q_{BO}$, $\vec{\beta}'' = q_{BI}$, $\vec{\beta}' = q_{IO}$. The sign of quaternion is flipped so that scalar part has positive sign. This is to ensure that quaternion corresponds to smallest rotation. Then it is normalized again to return unit quaternion.

qBO2qBI(v_q_BO,v_pos_i,v_vel_i)

Author: Sanket Chirame
Date:
Input: Unit quaternion which rotates orbit frame vector to body frame, position and velocity in

ecif
Output: Unit quaternion which rotates ecif vector to body frame From the definition of orbit frame

$$\vec{z} = -\vec{r_I}/|\vec{r_I}|$$

$$\vec{y} = \frac{\vec{v_I} \times \vec{r_I}}{|\vec{v_I} \times \vec{r_I}|}$$

$$\vec{x} = \frac{y \times z}{|y \times z|}$$

Where $\vec{x},\ \vec{y},\ \vec{z}$ are column vector in contrast to numpy arrays in code of dimension 1×3.

$$DCM_{OI} = [\,\vec{x}\,\vec{y}\,\vec{z}\,]^T$$

Quaternion corresponding to this DCM is obtained using $rotmat2quat$ function. q_{BI} is calculated using Equation (1) with $\vec{\beta} = q_{BI},\ \vec{\beta}'' = q_{BO},\ \vec{\beta}' = q_{OI}$

wBIb2wBOb(v_w_BI_b,v_q_BO)

Author: Sanket Chirame
Date:
Input : angular velocity of body w.r.t. ecif in body frame, unit quaternion which rotates orbit vector to body frame
Output : angular velocity of body frame w.r.t. orbit frame in body frame

$$\vec{w}_{BO} = \vec{w}_{BI} + \vec{w}_{IO}$$

Since we want components of $\vec{w}_{BO}$, all angular velocities on RHS should be expressed in body frame for component-wise addition.

References

[1] Hanspeter Schaub and John L Junkins. *Analytical mechanics of space systems*. Aiaa, 2003.