Analytical Mechanics of Space Systems (Hnspeter Schoub John L. Junkins)- textbook

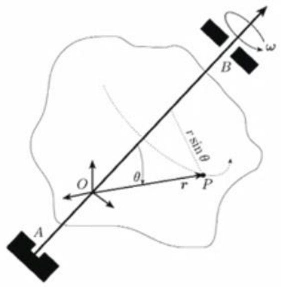
KINEMATICS

Particle kinematics:

Kinematics is the description of motion. Center of mass motion of the object. Velocity and acceleration related to rotating frames

Outline

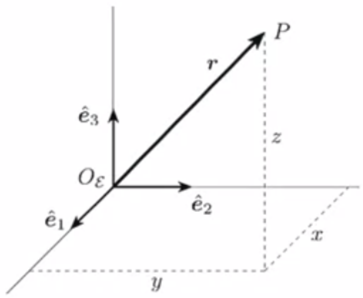
- Vector Notation
- Vector Differentiation
- Lots of brushing up on this material on your own!



What is a vector?

- Something with a direction and magnitude.
- A vector can be written as

$$\begin{aligned} \boldsymbol{r} &= x\hat{\boldsymbol{e}}_1 + y\hat{\boldsymbol{e}}_2 + z\hat{\boldsymbol{e}}_3 \\ &= r\hat{\boldsymbol{e}}_r \\ &= {}^{\mathcal{E}}\!\begin{pmatrix} x \\ y \\ z \end{pmatrix} \end{aligned}$$



Vector Addition

$$\boldsymbol{q} = \boldsymbol{r} + \boldsymbol{p} \quad \Rightarrow \quad \text{True}$$

$$ {}^{\mathcal{E}}\!\begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = {}^{\mathcal{E}}\!\begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} + {}^{\mathcal{B}}\!\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \Rightarrow \begin{matrix} q_1 = r_1 + p_1 \\ q_2 = r_2 + p_2 \\ q_3 = r_3 + p_3 \end{matrix} \quad \Rightarrow \quad \text{False}$$

$${}^{\mathcal{E}}\boldsymbol{q} = {}^{\mathcal{E}}\boldsymbol{r} + {}^{\mathcal{B}}\boldsymbol{p} \quad \Rightarrow \quad \text{False}$$

Coordinate frame

- Let a coordinate frame B be defined through the three unit orthogonal vectors:

$$\hat{\boldsymbol{b}}_1 \qquad \hat{\boldsymbol{b}}_2 \qquad \hat{\boldsymbol{b}}_3$$

- Let the origin of this frame be given by

$$\mathcal{O}_{\mathcal{B}}$$

- The frame is then defined through

$$\mathcal{B} : \{\mathcal{O}_{\mathcal{B}}, \hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\}$$

- If we can ignore the frame origin, then we often use the shorthand notation

$$\mathcal{B} : \{\hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\}$$

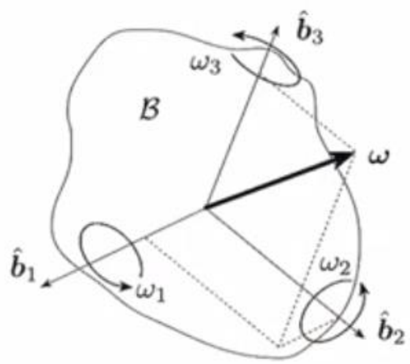
Angular Velocity Vector

- Angular velocity vector can be expressed as

$$\boldsymbol{\omega} = \omega_1 \hat{\boldsymbol{b}}_1 + \omega_2 \hat{\boldsymbol{b}}_2 + \omega_3 \hat{\boldsymbol{b}}_3$$

$${}^{\mathcal{B}}\boldsymbol{\omega} = {}^{\mathcal{B}}\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix}$$

- ω_i are instantaneous body rates about the orthogonal $\hat{\boldsymbol{b}}_i$ axes.



Fixed Axis Rotation

- The rigid body is rotating about a fixed axis.
- The speed of P is given by

$$|\dot{\boldsymbol{r}}| = (r \sin\theta)\, \omega$$

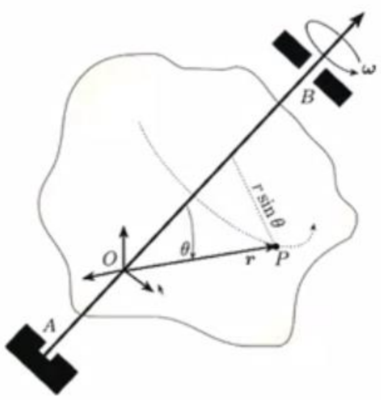
- note that

$$\dot{\boldsymbol{r}} = (r \sin\theta)\, \omega \left(\frac{\boldsymbol{\omega} \times \boldsymbol{r}}{|\boldsymbol{\omega} \times \boldsymbol{r}|} \right)$$

- thus the transport velocity is

$$|\boldsymbol{\omega} \times \boldsymbol{r}| = \omega r \sin\theta$$

$$\boxed{\dot{\boldsymbol{r}} = \boldsymbol{\omega} \times \boldsymbol{r}}$$



Transport Theorem

- Let a position vector be written as

$$\boldsymbol{r} = r_1\hat{\boldsymbol{b}}_1 + r_2\hat{\boldsymbol{b}}_2 + r_3\hat{\boldsymbol{b}}_3$$

while the angular velocity vector is written as

$$\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} = \omega_1\hat{\boldsymbol{b}}_1 + \omega_2\hat{\boldsymbol{b}}_2 + \omega_3\hat{\boldsymbol{b}}_3$$

Transport Theorem

- The inertial derivative of the position vector is

$$\frac{{}^{\mathcal{N}}\mathrm{d}}{\mathrm{d}t}(\boldsymbol{r}) = \dot{r}_1\hat{\boldsymbol{b}}_1 + \dot{r}_2\hat{\boldsymbol{b}}_2 + \dot{r}_3\hat{\boldsymbol{b}}_3 + r_1\frac{{}^{\mathcal{N}}\mathrm{d}}{\mathrm{d}t}\left(\hat{\boldsymbol{b}}_1\right) + r_2\frac{{}^{\mathcal{N}}\mathrm{d}}{\mathrm{d}t}\left(\hat{\boldsymbol{b}}_2\right) + r_3\frac{{}^{\mathcal{N}}\mathrm{d}}{\mathrm{d}t}\left(\hat{\boldsymbol{b}}_3\right)$$

- Note that $\hat{\boldsymbol{b}}_i$ are body fixed vectors, thus we find

$$\frac{{}^{\mathcal{N}}\mathrm{d}}{\mathrm{d}t}\left(\hat{\boldsymbol{b}}_i\right) = \boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} \times \hat{\boldsymbol{b}}_i$$

- This allows us to write the inertial derivative of the position vector as

$$\frac{{}^{\mathcal{N}}\mathrm{d}}{\mathrm{d}t}(\boldsymbol{r}) = \frac{{}^{\mathcal{B}}\mathrm{d}}{\mathrm{d}t}(\boldsymbol{r}) + \boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} \times \boldsymbol{r}$$

Transport Theorem

$$\boxed{\frac{{}^{\mathcal{N}}\mathrm{d}}{\mathrm{d}t}(\boldsymbol{r}) = \frac{{}^{\mathcal{B}}\mathrm{d}}{\mathrm{d}t}(\boldsymbol{r}) + \boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} \times \boldsymbol{r}}$$

Learn to be one with this equation, and three-dimensional rotations will never haunt you again!

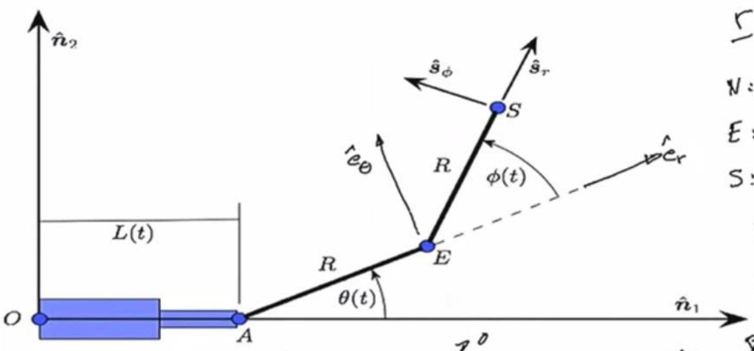
Comments

- Another noted otherwise, the following short-hand notation is used to denote inertial vector derivatives:

$$\frac{{}^{\mathcal{N}}\mathrm{d}}{\mathrm{d}t}(\boldsymbol{x}) \equiv \dot{\boldsymbol{x}}$$

- Note that we can analytically differentiate vectors, without first assigning specific coordinate frame. In fact, it is typically easier to wait until the very last steps before specifying a vectors through the vector components.

2-D EXAMPLE:



$$\underline{r} = L\hat{n}_1 + R\hat{e}_r + R\hat{s}_r = \underline{r}_{S/O}$$

$N: \{\hat{n}_1, \hat{n}_2, \hat{n}_3\}$

$E: \{\hat{e}_r, \hat{e}_\theta, \hat{n}_3\}$

$S: \{\hat{s}_r, \hat{s}_\phi, \hat{n}_3\}$

$\underline{\omega}_{E/N} = \dot{\theta}\,\hat{n}_3 \qquad \underline{\omega}_{S/N} = \underline{\omega}_{S/E} + \underline{\omega}_{E/N} = (\dot{\theta}+\dot{\phi})\hat{n}_3$

$\underline{\omega}_{S/E} = \dot{\phi}\,\hat{n}_3$

$$\dot{\underline{r}} = \frac{{}^N d}{dt}(\underline{r}) = \frac{{}^N d}{dt}(L\hat{n}_1) + \cancelto{0}{\frac{{}^E d}{dt}(R\hat{e}_r)} + \underline{\omega}_{E/N}\times(R\hat{e}_r) + \cancelto{0}{\frac{{}^S d}{dt}(R\hat{s}_r)} + \underline{\omega}_{S/N}\times(R\hat{s}_r)$$

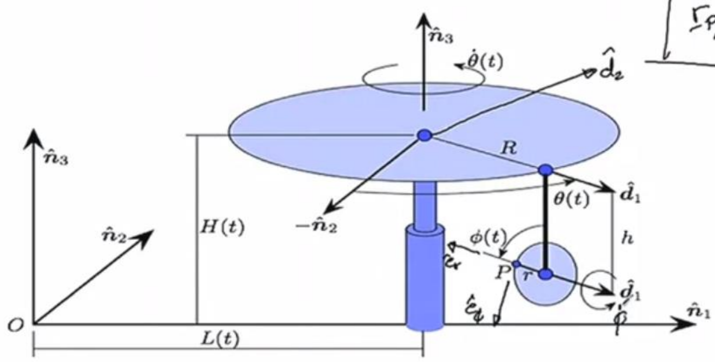
$$= \dot{L}\hat{n}_1 + (\dot{\theta}\hat{n}_3)\times(R\hat{e}_r) + (\dot{\theta}+\dot{\phi})\hat{n}_3\times R\hat{s}_r$$

$$\boxed{\dot{\underline{r}} = \dot{L}\hat{n}_1 + R\dot{\theta}\hat{e}_\theta + R(\dot{\theta}+\dot{\phi})\hat{s}_\phi}$$

$$\dot{\underline{r}} = \dot{L}\hat{n}_1 + R\dot{\theta}\hat{e}_\theta + R(\dot{\theta}+\dot{\phi})\hat{s}_\phi$$

$$\boxed{\ddot{\underline{r}} = \frac{{}^N d}{dt}(\dot{\underline{r}}) = \ddot{L}\hat{n}_1 + R\ddot{\theta}\hat{e}_\theta - R\dot{\theta}^2\hat{e}_r + R(\ddot{\theta}+\ddot{\phi})\hat{s}_\phi - R(\dot{\theta}+\dot{\phi})^2\hat{s}_r}$$

3-D EXAMPLE:



$$\boxed{\underline{r}_{P/O} = L\hat{n}_1 + H\hat{n}_3 + R\hat{d}_1 - h\hat{n}_3 + r\hat{e}_r}$$

$N: \{\hat{n}_1, \hat{n}_2, \hat{n}_3\}$

$D: \{\hat{d}_1, \hat{d}_2, \hat{n}_3\}$

$E: \{\hat{e}_r, \hat{e}_\phi, \hat{d}_1\}$

$\underline{\omega}_{D/N} = \dot{\theta}\,\hat{n}_3$

$\underline{\omega}_{E/D} = \dot{\phi}\,\hat{d}_1$

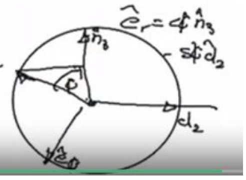
$\underline{\omega}_{E/N} = \underline{\omega}_{E/D} + \underline{\omega}_{D/N} = \dot{\phi}\hat{d}_1 + \dot{\theta}\hat{n}_3$

$$\dot{\underline{r}}_{P/O} = \dot{L}\hat{n}_1 + \dot{H}\hat{n}_3 + \cancelto{0}{\frac{{}^D d}{dt}(R\hat{d}_1)} + \underline{\omega}_{D/N}\times(R\hat{d}_1) - \cancelto{0}{\frac{{}^N d}{dt}(h\hat{n}_3)} + \cancelto{0}{\frac{{}^E d}{dt}(r\hat{e}_r)} + \underline{\omega}_{E/N}\times(r\hat{e}_r)$$

$$= \dot{L}\hat{n}_1 + \dot{H}\hat{n}_3 + R\dot{\theta}\hat{d}_2 + (\dot{\phi}\hat{d}_1 + \dot{\theta}\hat{n}_3)\times(r\hat{e}_r)$$

$$= \dot{L}\hat{n}_1 + \dot{H}\hat{n}_3 + R\dot{\theta}\hat{d}_2 + r\dot{\phi}\hat{e}_\phi + \left(r\dot{\theta}\,\hat{n}_3\times(\cancelto{0}{c\phi\hat{n}_3} - s\phi\hat{d}_2)\right) + r\dot{\theta}s\phi\hat{d}_1$$

$\hat{e}_r = c\phi\,\hat{n}_3 - s\phi\,\hat{d}_2$



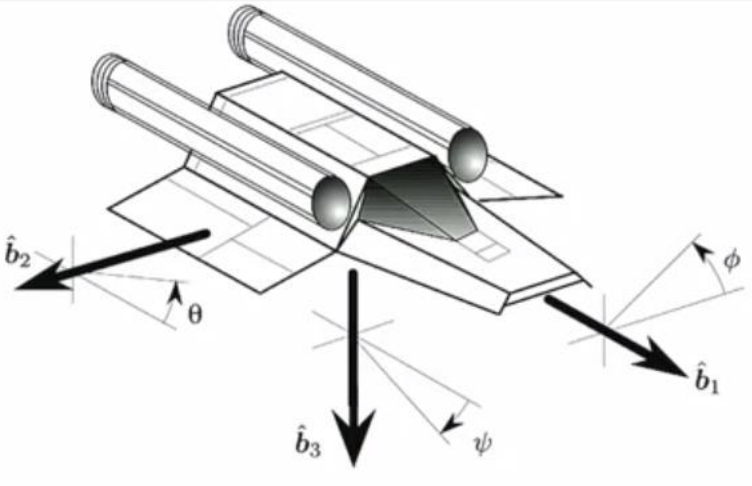
Euler Angles

The 101 of attitude coordinates...

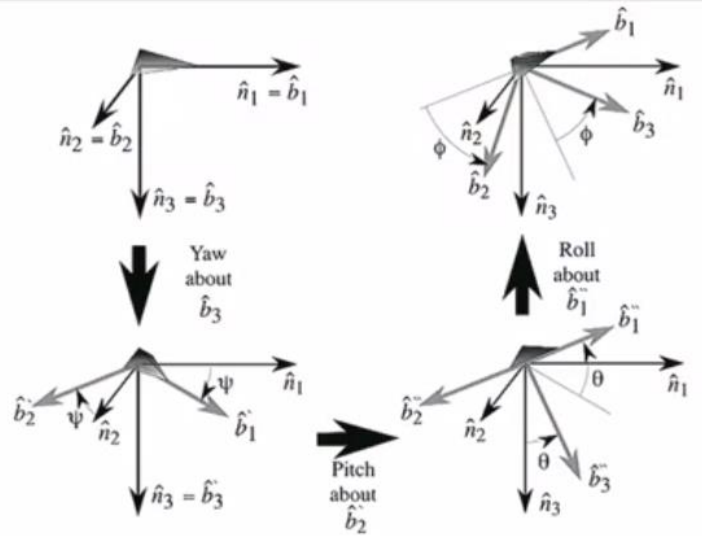
Description

- Most common set of attitude coordinates
- Describe the orientation between two frames using three *sequential* rotations
- Note that the order of rotation is important
- (*i*-*j*-*k*) Euler angles means we rotate first about the i^{th} axis, then about the j^{th} axis, and lastly about the k^{th} axis
- (3-2-1) Euler angles are the typical aircraft and spacecraft attitude angles
- Simple to visualize for small rotations

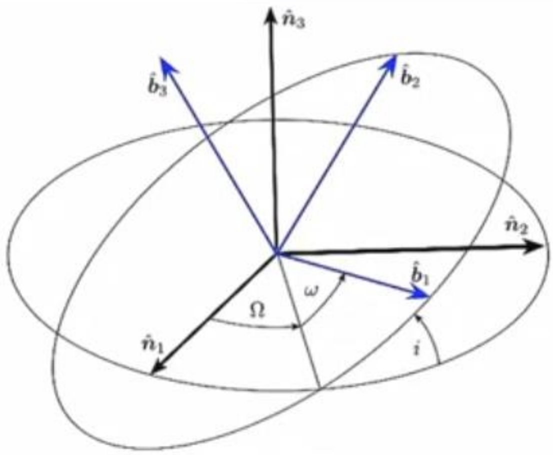
Aircraft/Spacecraft Orientation Angles



(3-2-1) Euler Angle Illustration

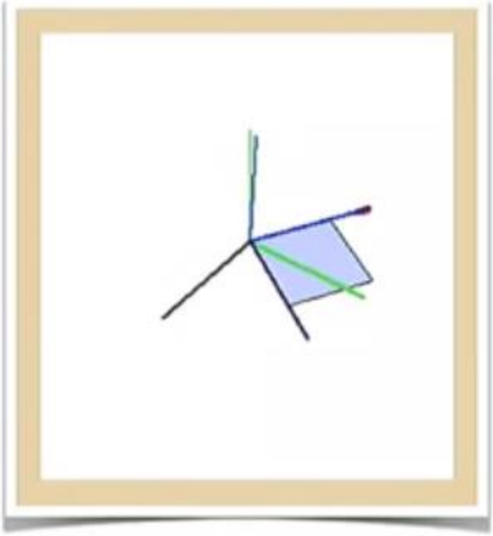


(3-1-3) Euler Angles

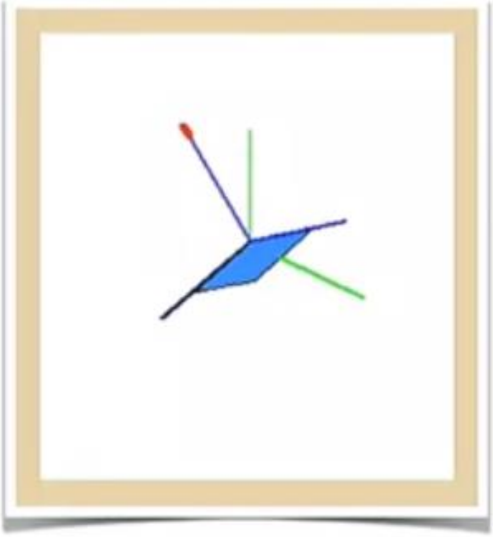


Commonly used to describe the orbit frame orientation relative to the inertial Frame *N*.

Second angle is the trouble-maker and +-90



(3-2-1) Euler Angles
(60,50,70) Degrees



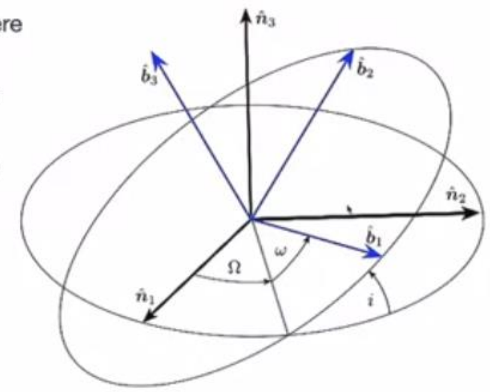
(1-3-2) Euler Angles
(37.2,-3.7,71.2) Degrees

Types of Euler Angles

- There are two types of Euler angles
  - Symmetric Set: Here the first and last rotation axis number is repeated. For example: 3-1-3 set used in astrodynamics to describe the orbit plane
  - Asymmetric Set: Here no axis rotation number is repeated. For example, the 3-2-1 (yaw-pitch-roll) angles used to describe many vehicles.
- Each type of Euler angles will have common mathematical properties and singularities.

Singularities

- Each set of Euler angles has a geometric singularity where two angles are not uniquely defined.
- It is always the second angle which defines this singular orientation.
  - Symmetric Set: 2nd angle is 0 or 180 degrees. For example, the 3-1-3 orbit angles with zero inclination.
  - Asymmetric Set: 2nd angle is +/- 90 degrees. For example, the 3-2-1 angle of an aircraft with 90 degrees pitch.



Single-Axis DCM

- The rotation matrix $[M_i]$ for a single axis rotation about the i^{th} body axis is given by

$$[M_1(\theta)] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}$$

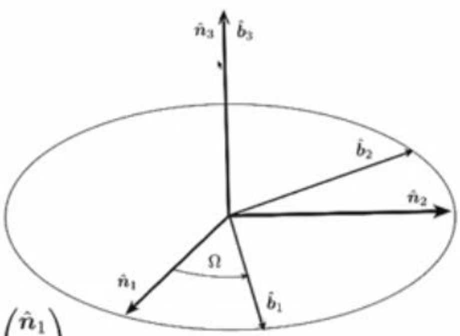
$$[M_2(\theta)] = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$[M_3(\theta)] = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Example

- Consider the 3-axis rotation using Ω
- The B and N frame axis are related through

$$\hat{\boldsymbol{b}}_1 = \cos\Omega\hat{\boldsymbol{n}}_1 + \sin\Omega\hat{\boldsymbol{n}}_2$$
$$\hat{\boldsymbol{b}}_2 = -\sin\Omega\hat{\boldsymbol{n}}_1 + \cos\Omega\hat{\boldsymbol{n}}_2$$
$$\hat{\boldsymbol{b}}_3 = \hat{\boldsymbol{n}}_3$$



This allows us to write

$$\begin{pmatrix} \hat{\boldsymbol{b}}_1 \\ \hat{\boldsymbol{b}}_2 \\ \hat{\boldsymbol{b}}_3 \end{pmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \hat{\boldsymbol{n}}_1 \\ \hat{\boldsymbol{n}}_2 \\ \hat{\boldsymbol{n}}_3 \end{pmatrix}$$

Mapping Euler Angles to Rotation Matrix

- Let the (α, β, γ) Euler angle sequence be $(\theta_1, \theta_2, \theta_3)$ To obtain the final rotation matrix $[BN]$ which maps inertial frame vector components to body frame vector components, we make use of the composite rotation matrix property $[RN]=[RB][BN]$.

$$\boxed{[C(\theta_1, \theta_2, \theta_3)] = [M_\gamma(\theta_3)][M_\beta(\theta_2)][M_\alpha(\theta_1)]}$$

- Carrying out this matrix algebra, we can find formulas which will map any Euler angle set to the corresponding rotation matrix.

Single-Axis DCM

- The rotation matrix $[M_i]$ for a single axis rotation about the i^{th} body axis is given by

$$[M_1(\theta)] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}$$

$$[M_2(\theta)] = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$[M_3(\theta)] = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3-2-1 Euler Angles

- Given the yaw, pitch and roll angles, we can compute the DCM using the three elemental rotation matrices:

$$[BN] = [M_1(\theta_3)][M_2(\theta_2)][M_3(\theta_1)] = [M_1(\phi)][M_2(\theta)][M_3(\psi)]$$

$$[BN] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3-2-1 Euler Angles

- Forward mapping is given by:

$$[BN] = \begin{bmatrix} c\theta_2 c\theta_1 & c\theta_2 s\theta_1 & -s\theta_2 \\ s\theta_3 s\theta_2 c\theta_1 - c\theta_3 s\theta_1 & s\theta_3 s\theta_2 s\theta_1 + c\theta_3 c\theta_1 & s\theta_3 c\theta_2 \\ c\theta_3 s\theta_2 c\theta_1 + s\theta_3 s\theta_1 & c\theta_3 s\theta_2 s\theta_1 - s\theta_3 c\theta_1 & c\theta_3 c\theta_2 \end{bmatrix}$$

- Inverse mapping back to Euler angles is found by examining the matrix element entries.

$$\psi = \theta_1 = \tan^{-1}\left(\frac{C_{12}}{C_{11}}\right)$$

$$\theta = \theta_2 = -\sin^{-1}(C_{13})$$

$$\phi = \theta_3 = \tan^{-1}\left(\frac{C_{23}}{C_{33}}\right)$$

Note that the quadrants must be checked with the inverse tangent function!

3-1-3 Euler Angles

- Forward mapping is given by:

$$[BN] = \begin{bmatrix} c\theta_3 c\theta_1 - s\theta_3 c\theta_2 s\theta_1 & c\theta_3 s\theta_1 + s\theta_3 c\theta_2 c\theta_1 & s\theta_3 s\theta_2 \\ -s\theta_3 c\theta_1 - c\theta_3 c\theta_2 s\theta_1 & -s\theta_3 s\theta_1 + c\theta_3 c\theta_2 c\theta_1 & c\theta_3 s\theta_2 \\ s\theta_2 s\theta_1 & -s\theta_2 c\theta_1 & c\theta_2 \end{bmatrix}$$

- Inverse mapping back to Euler angles is found by examining the matrix element entries.

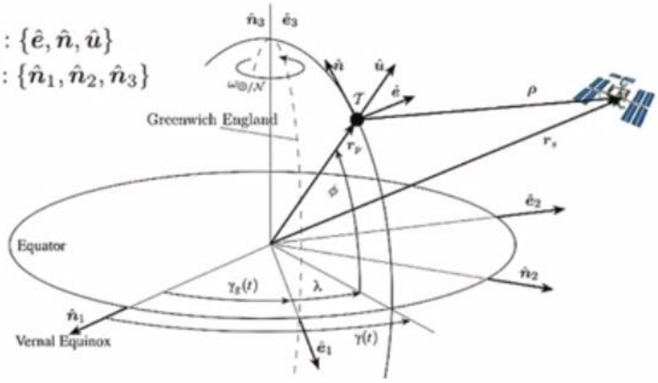
$$\Omega = \theta_1 = \tan^{-1}\left(\frac{C_{31}}{-C_{32}}\right)$$
$$i = \theta_2 = \cos^{-1}(C_{33})$$
$$\omega = \theta_3 = \tan^{-1}\left(\frac{C_{13}}{C_{23}}\right)$$

Note that the quadrants must be checked with the inverse tangent function!

Example

- Consider the astrodynamics problem, where the topographic frame (surface frame) *T* is defined as shown in the figure below.

$$\mathcal{T} : \{\hat{e}, \hat{n}, \hat{u}\}$$
$$\mathcal{N} : \{\hat{n}_1, \hat{n}_2, \hat{n}_3\}$$



- Here the rotation matrix *[TN]* was given as

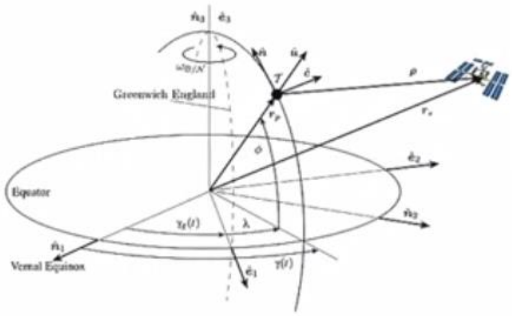
$$[TN] = \begin{bmatrix} -\sin\gamma(t) & \cos\gamma(t) & 0 \\ -\cos\gamma(t)\sin\phi & -\sin\gamma(t)\sin\phi & \cos\phi \\ \cos\gamma(t)\cos\phi & \sin\gamma(t)\cos\phi & \sin\phi \end{bmatrix}$$

- Let's derive this rotation matrix expression. To go from the *N* frame to the *T* frame, the first rotation is a 3-axis rotation by the angle Ω.

$$[M_3(\gamma(t))] = \begin{bmatrix} \cos\gamma(t) & \sin\gamma(t) & 0 \\ -\sin\gamma(t) & \cos\gamma(t) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- The next rotation is about the 2-axis with the angle Φ.

$$[M_2(-\phi)] = \begin{bmatrix} \cos(-\phi) & 0 & -\sin(-\phi) \\ 0 & 1 & 0 \\ \sin(-\phi) & 0 & \cos(-\phi) \end{bmatrix}$$

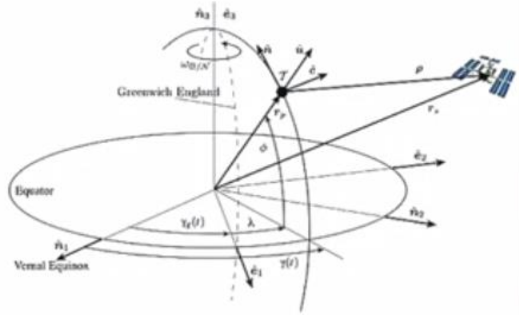


- However, we are not yet done. We still need to align the 1,2 and 3 axis of our current frame to that of the *T* frame. First we correct the 1-axis by doing a 90 degree rotation about our current 3-axis

$$[M_3(90^o)] = \begin{bmatrix} \cos 90^o & \sin 90^o & 0 \\ -\sin 90^o & \cos 90^o & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Next, we fix both the 2 and 3 axis orientation by doing 90 degree rotation about the current 1-axis.

$$[M_1(90^o)] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos 90^o & \sin 90^o \\ 0 & -\sin 90^o & \cos 90^o \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$



- Finally, we add up all these rotation matrices to find the desired [*TN*] direction cosine matrix:

$$[TN] = [M_1(90^o)][M_3(90^o)][M_2(-\phi)][M_3(\gamma(t))]$$

$$[TN] = \begin{bmatrix} -\sin\gamma(t) & \cos\gamma(t) & 0 \\ -\cos\gamma(t)\sin\phi & -\sin\gamma(t)\sin\phi & \cos\phi \\ \cos\gamma(t)\cos\phi & \sin\gamma(t)\cos\phi & \sin\phi \end{bmatrix}$$

Rotation Addition

- Assume we have a yaw-pitch-roll rotation defined from the inertial frame *N* to the reference frame *R* through

$$\boldsymbol{\theta}_{RN} = \{\psi_{RN}, \theta_{RN}, \phi_{RN}\}$$

- Assume we also know the yaw-pitch-roll rotation defined from the reference frame *R* to the body frame *B* through

$$\boldsymbol{\theta}_{BR} = \{\psi_{BR}, \theta_{BR}, \phi_{BR}\}$$

- The question is, what are the yaw-pitch-roll angles that will take us directly from the inertial frame *N* to the body frame *B*.

$$\boldsymbol{\theta}_{BN} = \{\psi_{BN}, \theta_{BN}, \phi_{BN}\}$$

- Note that $\boldsymbol{\theta}_{BN} \neq \boldsymbol{\theta}_{BR} + \boldsymbol{\theta}_{RN}$

Rotation Addition

- To add two Euler angle rotations, we go back to the rotation matrix addition property. First, we find:

$$\boldsymbol{\theta}_{BR} \quad \Rightarrow \quad [BR(\boldsymbol{\theta}_{BR})] \qquad \boldsymbol{\theta}_{RN} \quad \Rightarrow \quad [RN(\boldsymbol{\theta}_{RN})]$$

- Then, we compute [*BN*] using:

$$[BN(\boldsymbol{\theta}_{BN})] = [BR(\boldsymbol{\theta}_{BR})][RN(\boldsymbol{\theta}_{RN})]$$

- Last, we find the desired 3-2-1 Euler angles using the inverse mapping:

$$[BN(\boldsymbol{\theta}_{BN})] \quad \Rightarrow \quad \boldsymbol{\theta}_{BN} = \{\psi_{BN}, \theta_{BN}, \phi_{BN}\}$$

Rotation Subtraction

- Similarly, assume that we are given:

$$\boldsymbol{\theta}_{BN} = \{\psi_{BN}, \theta_{BN}, \phi_{BN}\}$$
$$\boldsymbol{\theta}_{RN} = \{\psi_{RN}, \theta_{RN}, \phi_{RN}\}$$

- In this case we would like to find the attitude tracking error of body *B* relative to the reference orientation *R*.

$$\begin{aligned}
&\boldsymbol{\theta}_{BN} \quad \Rightarrow \quad [BN(\boldsymbol{\theta}_{BN})] \\
&\boldsymbol{\theta}_{RN} \quad \Rightarrow \quad [RN(\boldsymbol{\theta}_{RN})] \\
&[BR(\boldsymbol{\theta}_{BR})] = [BN(\boldsymbol{\theta}_{BN})][RN(\boldsymbol{\theta}_{RN})]^T \\
&[BR(\boldsymbol{\theta}_{BR})] \quad \Rightarrow \quad \boldsymbol{\theta}_{BR} = \{\psi_{BR}, \theta_{BR}, \phi_{BR}\}
\end{aligned}$$

Example 3.2

- Let the orientation of two spacecraft *B* and *F* relative to an inertial frame *N* be given through the (3-2-1) Euler angles:
- The orientation matrices of these Euler angles are found using Eq. (3.20):

$$\boldsymbol{\theta}_{\mathcal{B}} = (30^o, -45^o, 60^o)^T \qquad \boldsymbol{\theta}_{\mathcal{F}} = (10^o, 25^o, -15^o)^T$$

$$[BN] = \begin{bmatrix} 0.612372 & 0.353553 & 0.707107 \\ -0.78033 & 0.126826 & 0.612372 \\ 0.126826 & -0.926777 & 0.353553 \end{bmatrix}$$

$$[FN] = \begin{bmatrix} 0.892539 & 0.157379 & -0.422618 \\ -0.275451 & 0.932257 & -0.234570 \\ 0.357073 & 0.325773 & 0.875426 \end{bmatrix}$$

- The rotation matrix relating the B and F frames is found to be

$$[BF] = [BN][FN]^T = \begin{bmatrix} 0.303372 & -0.0049418 & 0.952859 \\ -0.935315 & 0.1895340 & 0.298769 \\ -0.182075 & -0.9818620 & 0.052877 \end{bmatrix}$$

- The rotation matrix relating the B and F frames is found to be

$$[BF] = [BN][FN]^T = \begin{bmatrix} 0.303372 & -0.0049418 & 0.952859 \\ -0.935315 & 0.1895340 & 0.298769 \\ -0.182075 & -0.9818620 & 0.052877 \end{bmatrix}$$

- Using the transformations in Eq. (3.34), the Euler angles are computed using

$$\begin{aligned} \psi &= \tan^{-1}\left(\frac{-0.0049418}{0.303372}\right) = -0.933242 \text{ deg} \\ \theta &= -\sin^{-1}(0.952859) = -72.3373 \text{ deg} \\ \phi &= \tan^{-1}\left(\frac{0.298769}{0.052877}\right) = 79.9636 \text{ deg} \end{aligned}$$

(3-2-1) Kinematic Differential Equation

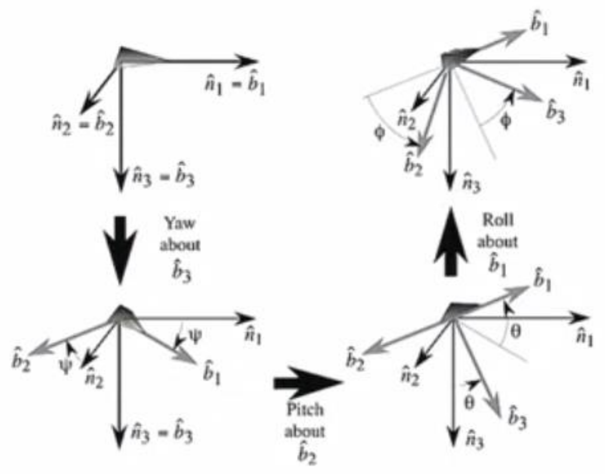
- We would like to find the differential equations of the Euler angles (i.e. yaw, pitch and roll angles).

$$\dot{\psi}(t) \qquad \dot{\theta}(t) \qquad \dot{\phi}(t)$$

- The angular rotation rate is not measured as yaw, pitch and roll rates, but rather through the body angular velocity vector

$$\boldsymbol{\omega} = \omega_1 \hat{\boldsymbol{b}}_1 + \omega_2 \hat{\boldsymbol{b}}_2 + \omega_3 \hat{\boldsymbol{b}}_3$$

- We need to find out how these Euler angle rates and the body angular velocity components are related.

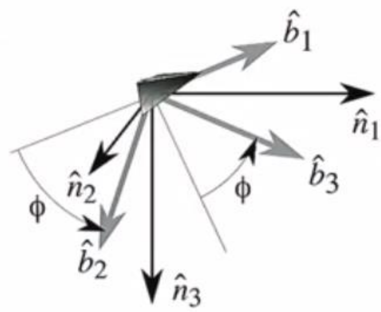


Using the above figure, it is evident that $\boldsymbol{\omega} = \dot{\psi}\hat{\boldsymbol{n}}_3 + \dot{\theta}\hat{\boldsymbol{b}}'_2 + \dot{\phi}\hat{\boldsymbol{b}}_1$

Recall that angular velocity vectors are truly vectors and can be simply added up.

- Next, we need to express the $\hat{\boldsymbol{b}}'_2$ in terms of $\{\hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\}$ vectors:

$$\hat{\boldsymbol{b}}'_2 = \cos\phi\hat{\boldsymbol{b}}_2 - \sin\phi\hat{\boldsymbol{b}}_3$$



- To write the $\hat{\boldsymbol{n}}_3$ in terms of $\{\hat{\boldsymbol{b}}_1, \hat{\boldsymbol{b}}_2, \hat{\boldsymbol{b}}_3\}$ vector, we use the mapping between the (3-2-1) Euler angles and [*BN*]:

$$\hat{\boldsymbol{n}}_3 = -\sin\theta\hat{\boldsymbol{b}}_1 + \sin\phi\cos\theta\hat{\boldsymbol{b}}_2 + \cos\phi\cos\theta\hat{\boldsymbol{b}}_3$$

- The last step is to equate the vector components by setting

$$\boldsymbol{\omega} = \omega_1\hat{\boldsymbol{b}}_1 + \omega_2\hat{\boldsymbol{b}}_2 + \omega_3\hat{\boldsymbol{b}}_3 = \dot{\psi}\hat{\boldsymbol{n}}_3 + \dot{\theta}\hat{\boldsymbol{b}}'_2 + \dot{\phi}\hat{\boldsymbol{b}}_1$$

- Finally, we can relate the Euler angle rates and the body angular velocity vector components through:

$${}^{\mathcal{B}}\boldsymbol{\omega} = {}^{\mathcal{B}}\begin{pmatrix}\omega_1\\ \omega_2\\ \omega_3\end{pmatrix} = \begin{bmatrix}-\sin\theta & 0 & 1\\ \sin\phi\cos\theta & \cos\phi & 0\\ \cos\phi\cos\theta & -\sin\phi & 0\end{bmatrix}\begin{pmatrix}\dot{\psi}\\ \dot{\theta}\\ \dot{\phi}\end{pmatrix}$$

- The inverse relationship (the kinematic differential equation of the (3-2-1) Euler angles) is found to be

$$\begin{pmatrix}\dot{\psi}\\ \dot{\theta}\\ \dot{\phi}\end{pmatrix} = \frac{1}{\cos\theta}\begin{bmatrix}0 & \sin\phi & \cos\phi\\ 0 & \cos\phi\cos\theta & -\sin\phi\cos\theta\\ \cos\theta & \sin\phi\sin\theta & \cos\phi\sin\theta\end{bmatrix}{}^{\mathcal{B}}\begin{pmatrix}\omega_1\\ \omega_2\\ \omega_3\end{pmatrix}$$

$$= [B(\psi, \theta, \phi)]\,{}^{\mathcal{B}}\boldsymbol{\omega}$$

(3-1-3) Kinematic Differential Eqn

- Similarly, the body angular velocity vector is written in terms of the (3-1-3) Euler angles as

$$ {}^{\mathcal{B}}\boldsymbol{\omega} = \begin{bmatrix} \sin\theta_3 \sin\theta_2 & \cos\theta_3 & 0 \\ \cos\theta_3 \sin\theta_2 & -\sin\theta_3 & 0 \\ \cos\theta_2 & 0 & 1 \end{bmatrix} \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{pmatrix} $$

- with the inverse transformation (the kinematic differential equation of the Euler angles) being

$$ \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{pmatrix} = \frac{1}{\sin\theta_2} \begin{bmatrix} \sin\theta_3 & \cos\theta_3 & 0 \\ \cos\theta_3 \sin\theta_2 & -\sin\theta_3 \sin\theta_2 & 0 \\ -\sin\theta_3 \cos\theta_2 & -\cos\theta_3 \cos\theta_2 & \sin\theta_2 \end{bmatrix} {}^{\mathcal{B}}\boldsymbol{\omega} $$

$$ = [B(\boldsymbol{\theta})]\,{}^{\mathcal{B}}\boldsymbol{\omega} $$

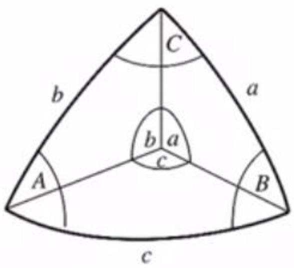
Comments

- Note that it is always the second Euler angle which causes the kinematic differential equations to become singular.
- As with the Euler angle geometric singularities, we find that for
  - Asymmetric Euler angles: differential equations are singular at $\theta_2 = \pm 90^{\circ}$
  - Symmetric Euler angles: differential equations are singular at $\theta_2 = 0^{\circ}$ or 180°
- With Euler angles, one is never more than a 90 degree removed from a singularity. This makes these attitude coordinates less attractive for large reorientations.

Addition of Symmetric Euler Angles

Spherical Law of Sines:

$$ \frac{\sin A}{\sin a} = \frac{\sin B}{\sin b} = \frac{\sin C}{\sin c} $$

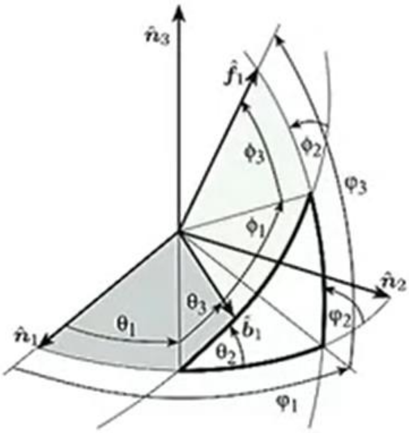


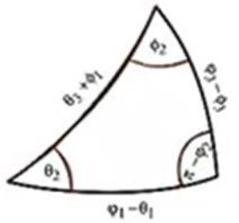
Spherical Law of Cosines:

$$ \cos A = -\cos B \cos C + \sin B \sin C \cos a $$
$$ \cos B = -\cos A \cos C + \sin A \sin C \cos b $$
$$ \cos C = -\cos A \cos B + \sin A \sin B \cos c $$

Quicker way to change Euler angle combinations

3-1-3 Euler Angles:

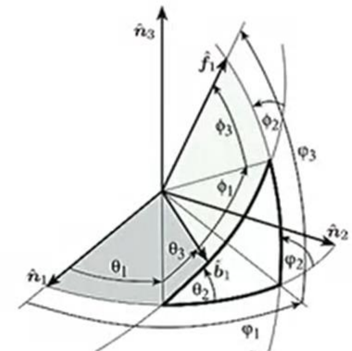


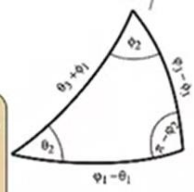


$$\cos(\pi - \varphi_2) = -\cos\theta_2 \cos\phi_2 + \sin\theta_2 \sin\phi_2 \cos(\theta_3 + \phi_1)$$

$$\varphi_2 = \cos^{-1}\left(\cos\theta_2 \cos\phi_2 - \sin\theta_2 \sin\phi_2 \cos(\theta_3 + \phi_1)\right)$$

3-1-3 Euler Angles:





Note: This formula actually holds for *any* symmetric Euler angle set!

Spherical Law of Sines:

$$\sin(\varphi_1 - \theta_1) = \frac{\sin\phi_2}{\sin\varphi_2}\sin(\theta_3 + \phi_1)$$

$$\sin(\varphi_3 - \phi_3) = \frac{\sin\theta_2}{\sin\varphi_2}\sin(\theta_3 + \phi_1)$$

Spherical Law of Cosines:

$$\cos(\varphi_1 - \theta_1) = \frac{\cos\phi_2 - \cos\theta_2\cos\varphi_2}{\sin\theta_2\sin\varphi_2}$$

$$\cos(\varphi_3 - \phi_3) = \frac{\cos\theta_2 - \cos\phi_2\cos\varphi_2}{\sin\phi_2\sin\varphi_2}$$

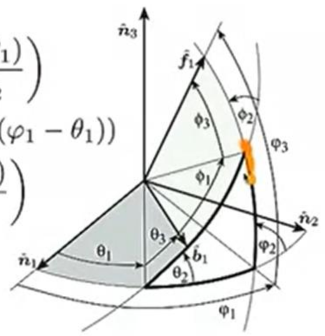
$$\varphi_1 = \theta_1 + \tan^{-1}\left(\frac{\sin\theta_2\sin\phi_2\sin(\theta_3+\phi_1)}{\cos\phi_2 - \cos\theta_2\cos\varphi_2}\right)$$

$$\varphi_3 = \phi_3 + \tan^{-1}\left(\frac{\sin\theta_2\sin\phi_2\sin(\theta_3+\phi_1)}{\cos\theta_2 - \cos\phi_2\cos\varphi_2}\right)$$

Symmetric Euler Angle Subtraction

- Using the equivalent spherical trigonometric formulas as for the EA addition problem, we can find a direct analytical solution to compute the relative symmetric EAs (i.e. EA subtraction).

$$\phi_1 = -\theta_3 + \tan^{-1}\left(\frac{\sin\theta_2 \sin\varphi_2 \sin(\varphi_1 - \theta_1)}{\cos\theta_2 \cos\phi_2 - \cos\varphi_2}\right)$$
$$\phi_2 = \cos^{-1}\left(\cos\theta_2 \cos\varphi_2 + \sin\theta_2 \sin\varphi_2 \cos(\varphi_1 - \theta_1)\right)$$
$$\phi_3 = \varphi_3 - \tan^{-1}\left(\frac{\sin\theta_2 \sin\varphi_2 \sin(\varphi_1 - \theta_1)}{\cos\theta_2 - \cos\phi_2 \cos\varphi_2}\right)$$



Arctan gives the right quadrant

We can use this law because we have one angle repeated

- Finally, we can relate the Euler angle rates and the body angular velocity vector components through:

$$^{\mathcal{B}}\boldsymbol{\omega} = {}^{\mathcal{B}}\begin{pmatrix}\omega_1\\ \omega_2\\ \omega_3\end{pmatrix} = \begin{bmatrix}-\sin\theta & 0 & 1\\ \sin\phi\cos\theta & \cos\phi & 0\\ \cos\phi\cos\theta & -\sin\phi & 0\end{bmatrix}\begin{pmatrix}\dot{\psi}\\ \dot{\theta}\\ \dot{\phi}\end{pmatrix}$$

- The inverse relationship (the kinematic differential equation of the (3-2-1) Euler angles) is found to be

$$\begin{pmatrix}\dot{\psi}\\ \dot{\theta}\\ \dot{\phi}\end{pmatrix} = \frac{1}{\cos\theta}\begin{bmatrix}0 & \sin\phi & \cos\phi\\ 0 & \cos\phi\cos\theta & -\sin\phi\cos\theta\\ \cos\theta & \sin\phi\sin\theta & \cos\phi\sin\theta\end{bmatrix} {}^{\mathcal{B}}\begin{pmatrix}\omega_1\\ \omega_2\\ \omega_3\end{pmatrix}$$
$$= [B(\psi,\theta,\phi)]\,^{\mathcal{B}}\boldsymbol{\omega}$$

February 2, 2017

$$\dot{\underline{X}} = \underline{F}(\underline{X}, t) \qquad \underline{X} = \begin{pmatrix} \psi \\ \theta \\ \phi \end{pmatrix}$$

$$\begin{pmatrix} \dot{\psi} \\ \dot{\theta} \\ \dot{\phi} \end{pmatrix} = [B]\,{}^{B}\underline{\omega}(t) \qquad \underline{X}_n = \underline{X}_0$$

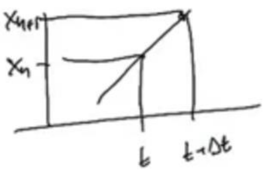
time
$t = 0 \rightarrow T_{final}$

$\underline{f} = \underline{F}(x_n, t)$

$\underline{x}_{n+1} = \underline{x}_n + \underline{f}\,\Delta t$

$\underline{x}_n = \underline{x}_{n+1}$

$$\underline{X}_{n+1} = \underline{x}_n + F \cdot \Delta t$$



Principal Rotation Vector

The building block of many advanced attitude coordinates...

Theorem 3.1 (Euler's Principal Rotation): A rigid body or coordinate reference frame can be brought from an arbitrary initial orientation to an arbitrary final orientation by a single rigid rotation through a principal angle Φ about the principal axis $\hat{e}$; the principal axis is a judicious axis fixed in both the initial and final orientation.

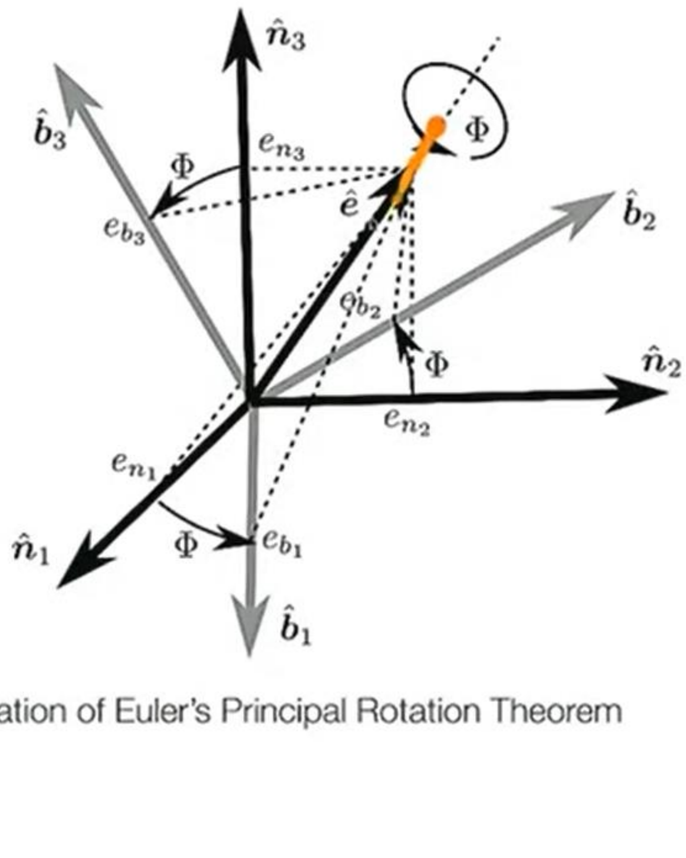
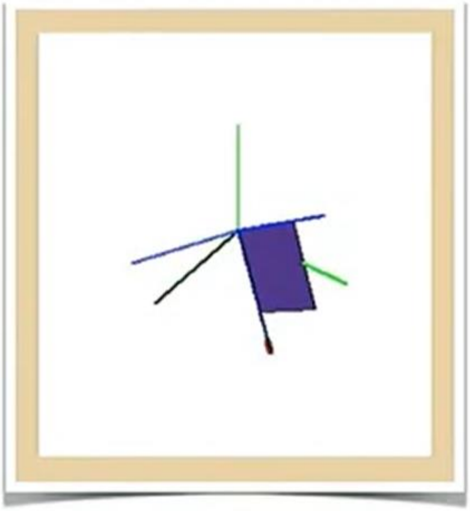
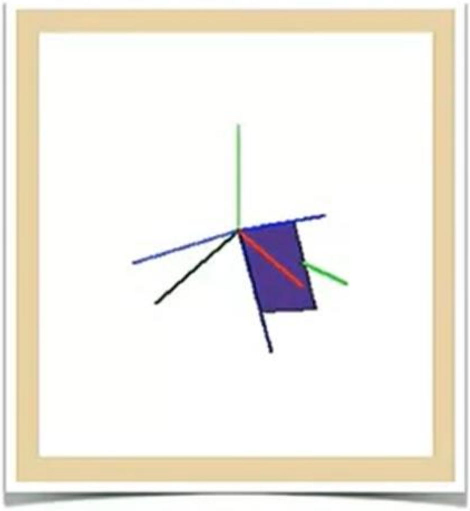


Illustration of Euler's Principal Rotation Theorem



(3-2-1) Euler Angles
(60,50,70) Degrees



Principal Rotation Vector
$\Phi = 80.3385^o$
$\hat{\boldsymbol{e}} = (0.429577, 0.867729, 0.250019)^T$

- Let's study the last statement of this theorem first: "the principal axis is a judicious axis fixed in both the initial and final orientation"

- This means that the principal axis unit vector will have the same vector components in the initial (i.e. inertial) and the final frame (i.e. body frame)

$$\hat{\boldsymbol{e}} = e_{b_1}\hat{\boldsymbol{b}}_1 + e_{b_2}\hat{\boldsymbol{b}}_2 + e_{b_3}\hat{\boldsymbol{b}}_3$$
$$\hat{\boldsymbol{e}} = e_{n_1}\hat{\boldsymbol{n}}_1 + e_{n_2}\hat{\boldsymbol{n}}_2 + e_{n_3}\hat{\boldsymbol{n}}_3 \Rightarrow e_{b_i} = e_{n_i} = e_i$$

- Using the rotation matrix [C], the $\hat{e}$ frame vector components in B and N frame can be related through

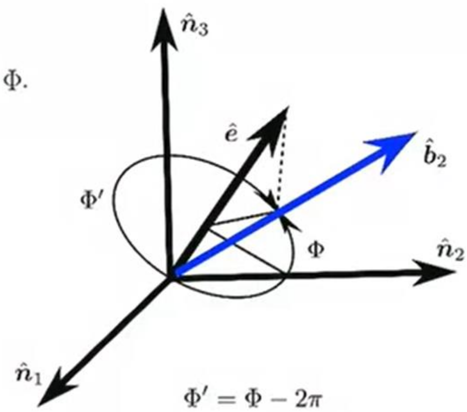
$$\begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = [C] \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix}$$

- From this last equation, it is evident that $\hat{e}$ must be an eigenvector of [C] with an eigenvalue of +1.

$$\begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = [C] \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix}$$

- This eigenvector is unique to within a sign of $\hat{e}$ or Φ.
- The $\hat{e}$ vector is not defined for a zero rotation!
- There are four possible principal rotations:

$$(\hat{\boldsymbol{e}}, \Phi)$$
$$(-\hat{\boldsymbol{e}}, -\Phi)$$
$$(\hat{\boldsymbol{e}}, \Phi')$$
$$(-\hat{\boldsymbol{e}}, -\Phi')$$



$\Phi' = \Phi - 2\pi$

Relationship to DCM

- We can express the [C] matrix in terms of PRV components as

$$[C] = \begin{bmatrix} e_1^2\Sigma + c\Phi & e_1e_2\Sigma + e_3s\Phi & e_1e_3\Sigma - e_2s\Phi \\ e_2e_1\Sigma - e_3s\Phi & e_2^2\Sigma + c\Phi & e_2e_3\Sigma + e_1s\Phi \\ e_3e_1\Sigma + e_2s\Phi & e_3e_2\Sigma - e_1s\Phi & e_3^2\Sigma + c\Phi \end{bmatrix}$$

$$\Sigma = 1 - c\Phi$$

- The inverse transformation from [C] to PRV is found by inspecting the matrix structure:

$$\cos\Phi = \frac{1}{2}(C_{11} + C_{22} + C_{33} - 1) \qquad \Phi' = \Phi - 2\pi$$

$$\hat{\boldsymbol{e}} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \frac{1}{2\sin\Phi}\begin{pmatrix} C_{23} - C_{32} \\ C_{31} - C_{13} \\ C_{12} - C_{21} \end{pmatrix}$$

PRV Addition

- DCM method:

$$[FN(\Phi, \hat{\boldsymbol{e}})] = [FB(\Phi_2, \hat{\boldsymbol{e}}_2)][BN(\Phi_1, \hat{\boldsymbol{e}}_1)]$$

- Direct method:

$$\Phi = 2\cos^{-1}\left(\cos\frac{\Phi_1}{2}\cos\frac{\Phi_2}{2} - \sin\frac{\Phi_1}{2}\sin\frac{\Phi_2}{2}\hat{\boldsymbol{e}}_1 \cdot \hat{\boldsymbol{e}}_2\right)$$

$$\hat{\boldsymbol{e}} = \frac{\cos\frac{\Phi_2}{2}\sin\frac{\Phi_1}{2}\hat{\boldsymbol{e}}_1 + \cos\frac{\Phi_1}{2}\sin\frac{\Phi_2}{2}\hat{\boldsymbol{e}}_2 + \sin\frac{\Phi_1}{2}\sin\frac{\Phi_2}{2}\hat{\boldsymbol{e}}_1 \times \hat{\boldsymbol{e}}_2}{\sin\frac{\Phi}{2}}$$

PRV Subtraction

- DCM method:

$$[FB(\Phi_2, \hat{\boldsymbol{e}}_2)] = [FN(\Phi, \hat{\boldsymbol{e}})][BN(\Phi_1, \hat{\boldsymbol{e}}_1)]^T$$

- Direct method:

$$\Phi_2 = 2\cos^{-1}\left(\cos\frac{\Phi}{2}\cos\frac{\Phi_1}{2} + \sin\frac{\Phi}{2}\sin\frac{\Phi_1}{2}\hat{\boldsymbol{e}} \cdot \hat{\boldsymbol{e}}_1\right)$$

$$\hat{\boldsymbol{e}}_2 = \frac{\cos\frac{\Phi_1}{2}\sin\frac{\Phi}{2}\hat{\boldsymbol{e}} - \cos\frac{\Phi}{2}\sin\frac{\Phi_1}{2}\hat{\boldsymbol{e}}_1 + \sin\frac{\Phi}{2}\sin\frac{\Phi_1}{2}\hat{\boldsymbol{e}} \times \hat{\boldsymbol{e}}_1}{\sin\frac{\Phi_2}{2}}$$

PRV Differential Kinematic Equation

- Mapping from body angular velocity vector to PRV rates:

$$\dot{\gamma} = \left[[I_{3\times3}] + \frac{1}{2}[\tilde{\gamma}] + \frac{1}{\Phi^2}\left(1 - \frac{\Phi}{2}\cot\left(\frac{\Phi}{2}\right)\right)[\tilde{\gamma}]^2 \right] {}^{\mathcal{B}}\boldsymbol{\omega}$$

- Mapping from PRV rates to body angular velocity vector:

$${}^{\mathcal{B}}\boldsymbol{\omega} = \left[[I_{3\times3}] - \left(\frac{1 - \cos\Phi}{\Phi^2}\right)[\tilde{\gamma}] + \left(\frac{\Phi - \sin\Phi}{\Phi^3}\right)[\tilde{\gamma}]^2 \right] \dot{\gamma}$$

Not used in attitude control, four parameters but singularity at 0 and 180 degrees

Conclusion

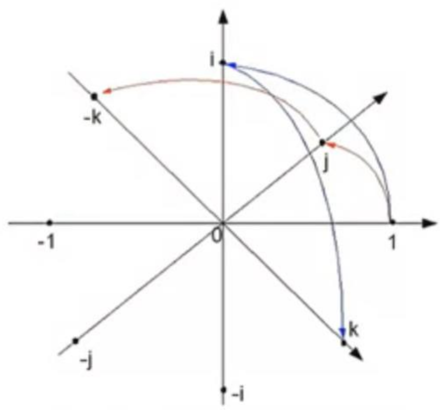
- PRV is based on a very fundamental rotation/orientation property called Euler's principal rotation theorem
- Singular for zero-rotation
- PRVs form the basis for many other attitude coordinates which are very useful for large angle rotations

Euler Parameters (Quaternions)

Voted most popular attitude coordinates in the non-singular category...

Introduction

- Very popular redundant set of attitude coordinates
- Are called either Euler Parameters (EPs) or quaternions
- Major benefits:
  - Non-singular attitude description
  - Linear differential kinematic equation
  - Works well for small and large rotations
- Drawbacks:
  - Constraint equation must be identified as all times
  - Not as simple to visualize



https://en.wikipedia.org/wiki/Quaternion

Definition of EP

- The redundant Euler Parameters are defined using the principal rotation components as

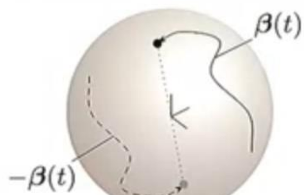
$$\begin{aligned}\beta_0 &= \cos(\Phi/2)\\ \beta_1 &= e_1 \sin(\Phi/2)\\ \beta_2 &= e_2 \sin(\Phi/2)\\ \beta_3 &= e_3 \sin(\Phi/2)\end{aligned}$$

Constraints:

$$e_1^2 + e_2^2 + e_3^2 = 1$$

$$\beta_0^2 + \beta_1^2 + \beta_2^2 + \beta_3^2 = 1$$

- Note that the 4-coordinate set has a single constraint equation! All EPs must lie on the three-dimensional surface of a 4-dimensional hypersphere.



Unit Hypersphere

- Since the PRV components are not unique, we find that the EP also isn't unique:

$(-\hat{e}, -\Phi)$	$\beta_0' = \cos\left(-\frac{\Phi}{2}\right) = \cos\left(\frac{\Phi}{2}\right) = \beta_0$ $\beta_i' = -e_i \sin\left(-\frac{\Phi}{2}\right) = e_i \sin\left(\frac{\Phi}{2}\right) = \beta_i$
$(\hat{e}, \Phi')$	$\beta_0' = \cos\left(\frac{\Phi'}{2}\right) = \cos\left(\frac{\Phi}{2} - \pi\right) = -\cos\left(\frac{\Phi}{2}\right) = -\beta_0$ $\beta_i' = e_i \sin\left(\frac{\Phi'}{2}\right) = e_i \sin\left(\frac{\Phi'}{2} - \pi\right) = -e_i \sin\left(\frac{\Phi}{2}\right) = -\beta_i$

- Note that the alternate EP set corresponds to performing the larger principle rotation angle (i.e., rotating the long way round)

Euler Parameter to DCM Relationship

- The rotation matrix can be expressed in terms or EPs as:

$$[C] = \begin{bmatrix} \beta_0^2+\beta_1^2-\beta_2^2-\beta_3^2 & 2(\beta_1\beta_2+\beta_0\beta_3) & 2(\beta_1\beta_3-\beta_0\beta_2) \\ 2(\beta_1\beta_2-\beta_0\beta_3) & \beta_0^2-\beta_1^2+\beta_2^2-\beta_3^2 & 2(\beta_2\beta_3+\beta_0\beta_1) \\ 2(\beta_1\beta_3+\beta_0\beta_2) & 2(\beta_2\beta_3-\beta_0\beta_1) & \beta_0^2-\beta_1^2-\beta_2^2+\beta_3^2 \end{bmatrix}$$

- The inverse relationship is found by inspection to be

$$\beta_0 = \pm\frac{1}{2}\sqrt{C_{11}+C_{22}+C_{33}+1}$$
$$\beta_1 = \frac{C_{23}-C_{32}}{4\beta_0}$$
$$\beta_2 = \frac{C_{31}-C_{13}}{4\beta_0}$$
$$\beta_3 = \frac{C_{12}-C_{21}}{4\beta_0}$$

Singular if: $\beta_0 \rightarrow 0$

Pick beta0 positive, short rotation.

DO NOT INCLUDE last formulas on your program because it will cause singularities. 180 degrees.

- Sheppard's method is a robust method to compute the EP from a rotation matrix:

1st step: Find largest value of

$$\beta_0^2 = \frac{1}{4}(1+\text{trace}([C])) \qquad \beta_2^2 = \frac{1}{4}(1+2C_{22}-\text{trace}([C]))$$
$$\beta_1^2 = \frac{1}{4}(1+2C_{11}-\text{trace}([C])) \qquad \beta_3^2 = \frac{1}{4}(1+2C_{33}-\text{trace}([C]))$$

2nd step: Compute the remaining EPs using

$$\beta_0\beta_1 = (C_{23}-C_{32})/4 \qquad \beta_1\beta_2 = (C_{12}+C_{21})/4$$
$$\beta_0\beta_2 = (C_{31}-C_{13})/4 \qquad \beta_3\beta_1 = (C_{31}+C_{13})/4$$
$$\beta_0\beta_3 = (C_{12}-C_{21})/4 \qquad \beta_2\beta_3 = (C_{23}+C_{32})/4$$

Take the positive value of the largest value. In the end you have to check if beta0 turns out to be positive or negative. If it is negative you chose the long rotation so flip everything to have a consistent attitude determination.

Adding Euler Parameters

- A very useful advantage of EPs is how you can add or subtract two orientations using them. Using DCMs, we can add two rotations using:

$$[FN(\boldsymbol{\beta})] = [FB(\boldsymbol{\beta}'')][BN(\boldsymbol{\beta}')]$$

- However, using EPs directly, we find the elegant result:

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

Orthogonal matrix, inverse is equal to transpose

- By reshuffling the terms (i.e. permutation) in the last EP addition equation, we can also write this as

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

- To subtract two orientations described through EPs, we can use the last two equations and exploit the orthogonality property of the 4x4 matrix to invert it and solve for either $\boldsymbol{\beta}'$ or $\boldsymbol{\beta}''$.

Euler Parameter Differential Equation

- Using the differential equation of DCMs, and the relationship between EPs and DCMs, we can derive the differential kinematic equations of Euler parameters.
- However, this is a rather lengthy and algebraically complex task. The end result is the amazingly simple bi-linear result:

$$\begin{pmatrix}\dot\beta_0\\ \dot\beta_1\\ \dot\beta_2\\ \dot\beta_3\end{pmatrix} = \frac{1}{2}\begin{bmatrix}\beta_0 & -\beta_1 & -\beta_2 & -\beta_3\\ \beta_1 & \beta_0 & -\beta_3 & \beta_2\\ \beta_2 & \beta_3 & \beta_0 & -\beta_1\\ \beta_3 & -\beta_2 & \beta_1 & \beta_0\end{bmatrix} \begin{pmatrix}0\\ \omega_1\\ \omega_2\\ \omega_3\end{pmatrix}$$

Omegas are given in body frame

- Rearranging the terms on the right hand side of this differential equation, we can also write this as

$$\begin{pmatrix} \dot{\beta}_0 \\ \dot{\beta}_1 \\ \dot{\beta}_2 \\ \dot{\beta}_3 \end{pmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_3 & -\omega_2 \\ \omega_2 & -\omega_3 & 0 & \omega_1 \\ \omega_3 & \omega_2 & -\omega_1 & 0 \end{bmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix}$$

- Having the differential equation depend linearly on the EPs is important in estimation theory. This makes the EPs ideal coordinate candidates to be used in a Kalman filter (Spacecraft orientation estimator).

2nd Euler Parameter Differential Kinematic Eqs.

- The EP differential equations can also be written in the following convenient form for numerical integration:

$$\dot{\boldsymbol{\beta}} = \frac{1}{2}[B(\boldsymbol{\beta})]\boldsymbol{\omega} \qquad [B(\boldsymbol{\beta})] = \begin{bmatrix} -\beta_1 & -\beta_2 & -\beta_3 \\ \beta_0 & -\beta_3 & \beta_2 \\ \beta_3 & \beta_0 & -\beta_1 \\ -\beta_2 & \beta_1 & \beta_0 \end{bmatrix}$$

- The $[B]$ matrix satisfies the following useful identities:

$$[B(\boldsymbol{\beta})]^T \boldsymbol{\beta} = \mathbf{0}$$
$$[B(\boldsymbol{\beta})]^T \boldsymbol{\beta}' = -[B(\boldsymbol{\beta}')]^T \boldsymbol{\beta}$$

$$(\beta_0, \underbrace{\beta_1, \beta_2, \beta_3}_{\underline{\varepsilon}})$$

$\beta_0 = \cos\frac{\Phi}{2}$

$\beta_1 = e_1 \sin\frac{\Phi}{2}$

$\beta_2 = e_2$ "

$\beta_3 = e_3$ "

$\hat{e} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix}$

$\underline{\beta} \equiv -\underline{\beta}$

3rd Euler Parameter Differential Kinematic Eqs.

- In control applications, the scalar and vector components of the Euler parameters are sometimes treated separately.

Define: $$\boldsymbol{\epsilon} \equiv (\beta_1, \beta_2, \beta_3)^T$$

Define: $$[T(\beta_0, \boldsymbol{\epsilon})] = \beta_0 [I_{3\times 3}] + [\tilde{\boldsymbol{\epsilon}}]$$

Differential Equation:
$$\dot{\beta}_0 = -\frac{1}{2}\boldsymbol{\epsilon}^T \boldsymbol{\omega} = -\frac{1}{2}\boldsymbol{\omega}^T \boldsymbol{\epsilon}$$
$$\dot{\boldsymbol{\epsilon}} = \frac{1}{2}[T]\boldsymbol{\omega}$$

Classical Rodrigues Parameters (Gibbs Vector or CRPs)

Popular coordinates for large rotations and robotics....

CRP Definitions

Euler parameter relationship:

$$q_i = \frac{\beta_i}{\beta_0} \qquad i = 1,2,3$$

Singular if 0 (±180° case)

$$\beta_0 = \frac{1}{\sqrt{1 + \boldsymbol{q}^T \boldsymbol{q}}}$$

$$\beta_i = \frac{q_i}{\sqrt{1 + \boldsymbol{q}^T \boldsymbol{q}}} \qquad i = 1,2,3$$

Singular if ∞ (±180° case)

Principal rotation parameter relationship:

$$\boldsymbol{q} = \tan\frac{\Phi}{2}\hat{\boldsymbol{e}}$$

Singular for ±180°

$$\boldsymbol{q} \approx \frac{\Phi}{2}\hat{\boldsymbol{e}}$$

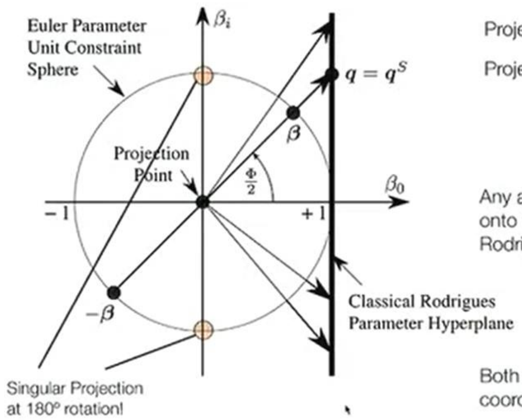
Linearizes to angles over 2.

These parameters are much better suited for large spacecraft rotations than Euler angles, while remaining a minimal coordinate set.
Only the upside down description is singular.

Relationship to DCM: $$[\tilde{q}] = \frac{[C]^T - [C]}{\zeta^2} \qquad \zeta = \sqrt{\operatorname{trace}([C]) + 1} = 2\beta_0$$

$$\boldsymbol{q} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \frac{1}{\zeta^2}\begin{pmatrix} C_{23} - C_{32} \\ C_{31} - C_{13} \\ C_{12} - C_{21} \end{pmatrix}$$

Stereographic Projection



Projection Point: $(0,0,0,0)$

Projection Plane: $\beta_0 = +1$

Any attitude (surface point) is projected onto the hyper-plane to form the classical Rodrigues parameters.

Both EP sets yield the identical CRP coordinates.

Unique, singular at 180

Direction Cosine Matrix

Matrix components:

$$[C] = \frac{1}{1+\boldsymbol{q}^T\boldsymbol{q}} \begin{bmatrix} 1+q_1^2-q_2^2-q_3^2 & 2(q_1q_2+q_3) & 2(q_1q_3-q_2) \\ 2(q_2q_1-q_3) & 1-q_1^2+q_2^2-q_3^2 & 2(q_2q_3+q_1) \\ 2(q_3q_1+q_2) & 2(q_3q_2-q_1) & 1-q_1^2-q_2^2+q_3^2 \end{bmatrix}$$

$$[C] = \frac{1}{1+\boldsymbol{q}^T\boldsymbol{q}}\left((1-\boldsymbol{q}^T\boldsymbol{q})[I_{3\times3}] + 2\boldsymbol{q}\boldsymbol{q}^T - 2[\tilde{\boldsymbol{q}}]\right)$$

$$[C(\boldsymbol{q})]^{-1} = [C(\boldsymbol{q})]^T = [C(-\boldsymbol{q})]$$

$$\underline{q}_{B/N} = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.3 \end{pmatrix} \qquad \underline{q}_{N/B} = \begin{pmatrix} -0.1 \\ -0.2 \\ -0.3 \end{pmatrix}$$

Attitude Addition/Subtraction

- DCM method:

$$[FN(\boldsymbol{q})] = [FB(\boldsymbol{q}'')][BN(\boldsymbol{q}')]$$

- Direct method:

$$\boldsymbol{q} = \frac{\boldsymbol{q}'' + \boldsymbol{q}' - \boldsymbol{q}'' \times \boldsymbol{q}'}{1 - \boldsymbol{q}'' \cdot \boldsymbol{q}'}$$

Attitude Addition

$$\boldsymbol{q}'' = \frac{\boldsymbol{q} - \boldsymbol{q}' + \boldsymbol{q} \times \boldsymbol{q}'}{1 + \boldsymbol{q} \cdot \boldsymbol{q}'}$$

Relative Attitude (Subtraction)

Note: Using $\delta\boldsymbol{q} = \boldsymbol{q} - \boldsymbol{q}'$ to compute the relative attitude, or attitude error, still yields a result that is a proper CRP attitude measure. However, also note that the approximation $\delta\boldsymbol{q} \approx \boldsymbol{q}''$only holds for small attitude differences.

Differential Kinematic Equations

Matrix components:

$$\dot{\boldsymbol{q}} = \frac{1}{2}\begin{bmatrix} 1+q_1^2 & q_1q_2-q_3 & q_1q_3+q_2 \\ q_2q_1+q_3 & 1+q_2^2 & q_2q_3-q_1 \\ q_3q_1-q_2 & q_3q_2+q_1 & 1+q_3^2 \end{bmatrix} {}^{\mathcal{B}}\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix}$$

Vector computation:

$$\dot{\boldsymbol{q}} = \frac{1}{2}\left[[I_{3\times3}] + [\tilde{\boldsymbol{q}}] + \boldsymbol{q}\boldsymbol{q}^T\right] {}^{\mathcal{B}}\boldsymbol{\omega}$$

$${}^{\mathcal{B}}\boldsymbol{\omega} = \frac{2}{1+\boldsymbol{q}^T\boldsymbol{q}}\left([I_{3\times3}] - [\tilde{\boldsymbol{q}}]\right)\dot{\boldsymbol{q}}$$

Note: Only contains quadratic nonlinearities, but is singular for $\Phi = 180^\circ$.

Cayley Transform

- Amazingly elegant matrix transformation, that allows us to use attitude parameters in higher dimensional spaces.

- Let [*Q*] be a skew-symmetric matrix, [*C*] be a proper orthogonal matrix, and [*I*] be a identity matrix. These matrices can be of any dimension *N*. The Cayley Transform is then defined as:

$$\boxed{[C] = ([I] - [Q])([I] + [Q])^{-1} = ([I] + [Q])^{-1}([I] - [Q])}$$

$$[Q] = ([I] - [C])([I] + [C])^{-1} = ([I] + [C])^{-1}([I] - [C])$$

Example:

- For 3D space, the proper orthogonal [*C*] matrix is also a rotation or direction cosine matrix. In this case we find that

$$[Q] = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix}$$

where the unique matrix elements are the CRP!

$$[C] = \begin{bmatrix} 0.813797 & 0.296198 & -0.5 \\ 0.235888 & 0.617945 & 0.75 \\ 0.531121 & -0.728292 & 0.433012 \end{bmatrix}$$

$$[Q] = \begin{bmatrix} 0 & -0.021052 & 0.359933 \\ 0.021052 & 0 & -0.516027 \\ -0.359933 & 0.516027 & 0 \end{bmatrix} \qquad \boldsymbol{q} = \begin{pmatrix} 0.516027 \\ 0.359933 \\ 0.021052 \end{pmatrix}$$

CRP vector

- Higher Dimensional Example:

$$[C] = \begin{bmatrix} 0.505111 & -0.503201 & -0.215658 & 0.667191 \\ 0.563106 & -0.034033 & -0.538395 & -0.626006 \\ 0.560111 & 0.748062 & 0.272979 & 0.228387 \\ -0.337714 & 0.431315 & -0.767532 & 0.332884 \end{bmatrix}$$

$$[Q] = \begin{bmatrix} 0 & 0.5 & 0.2 & -0.3 \\ -0.5 & 0 & 0.7 & 0.6 \\ -0.2 & -0.7 & 0 & -0.4 \\ 0.3 & -0.6 & 0.4 & 0 \end{bmatrix}$$

- Recall that regardless of the dimensionality of the orthogonal matrix [*C*(*t*)], it must evolve according to:

$$[\dot{C}] = -[\tilde{\boldsymbol{\omega}}][C]$$

- These higher-dimensional "body angular velocities" can be related to the higher dimensional CRPs using:

$$[\dot{Q}] = \frac{1}{2}\left([I] + [Q]\right)[\tilde{\omega}]\left([I] - [Q]\right)$$
$$[\tilde{\omega}] = 2\left([I] + [Q]\right)^{-1}[\dot{Q}]\left([I] - [Q]\right)^{-1}$$

- Thus, can solve for the [*C*(*t*)] using a reduced coordinate set.

- This parameterization is singular whenever a principal rotation of 180° is performed.

- **Physical Example:**

Consider a typical mechanical system. The EOM can be written in the form

$$[M(\boldsymbol{x},t)]\ddot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x},\dot{\boldsymbol{x}},t)$$

To solve this system for the state accelerations, the system positive definite system mass matrix must be inverted, a numerically expensive operations for large dimensions.

This inverse could be avoided by using the spectral decomposition:

$$[M] = [V][D][V]^T \qquad\qquad [M]^{-1} = [V]^T[D]^{-1}[V]$$

where [*V*] is a proper orthogonal eigenvector matrix and [*D*] is a diagonal eigenvalue matrix. To determine [*V*(*t*)] the Cayley transform could be used to track a reduced parameter set:

$$[Q] = ([I] - [V])([I] + [V])^{-1}$$

Modified Rodrigues Parameters (MRPs)

The "cool" new attitude coordinates...

MRP Definitions

Euler parameter relationship:

$$\sigma_i = \frac{\beta_i}{1+\beta_0} \qquad i = 1,2,3$$

Singular if -1
(±360° case)

$$\beta_0 = \frac{1-\sigma^2}{1+\sigma^2}$$

$$\beta_i = \frac{2\sigma_i}{1+\sigma^2} \qquad i = 1,2,3$$

Singular if ∞
(±360° case)

PRV relationship:

$$\boldsymbol{\sigma} = \tan\frac{\Phi}{4}\hat{e}$$

Singular for ±360°

$$\boldsymbol{\sigma} \approx \frac{\Phi}{4}\hat{e}$$

Linearizes to angles over 4.

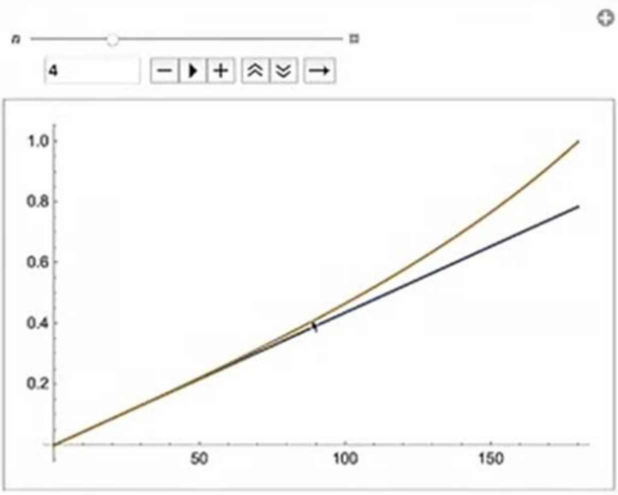
(Show Mathematica Example)

CRP relationship:

$$\boldsymbol{q} = \frac{2\boldsymbol{\sigma}}{1-\sigma^2}$$

$$\boldsymbol{\sigma} = \frac{\boldsymbol{q}}{1+\sqrt{1+\boldsymbol{q}^T\boldsymbol{q}}}$$

```
Manipulate[
 Plot[{ϕ / n Degree, Tan[ϕ / n Degree]}, {ϕ, 0, 180}],
 {n, 2, 10, 1}]
```



Good approximation up to 80 deg

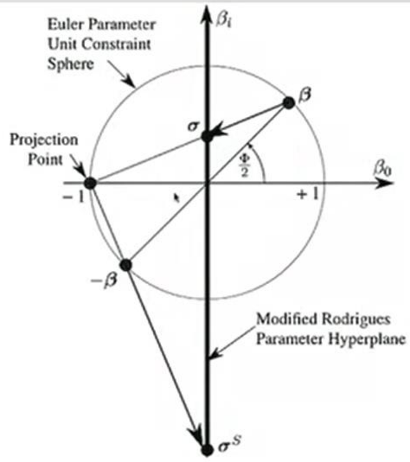
MRP Definitions

Relationship to DCM:

$$[\tilde{\sigma}] = \frac{[C]^T - [C]}{\zeta(\zeta+2)} \qquad \zeta = \sqrt{\mathrm{trace}([C]) + 1} = \beta_0/2$$

$$\boldsymbol{\sigma} = \begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \end{pmatrix} = \frac{1}{\zeta(\zeta+2)} \begin{pmatrix} C_{23} - C_{32} \\ C_{31} - C_{13} \\ C_{12} - C_{21} \end{pmatrix}$$

Stereographic Projection



Projection Point: $(-1, 0, 0, 0)$

Projection Plane: $\beta_0 = 0$

Any attitude (surface point) is projected onto the hyper-plane to form the modified Rodrigues parameters.

The two EP sets yield *distinct* MRP coordinate values with different singular behaviors.

Shadow MRP Set

- Using the alternate set of Euler parameters, we can find the "shadow" set of MRP parameters:

$$\sigma_i^S = \frac{-\beta_i}{1-\beta_0} = \frac{-\sigma_i}{\sigma^2} \quad i = 1, 2, 3$$

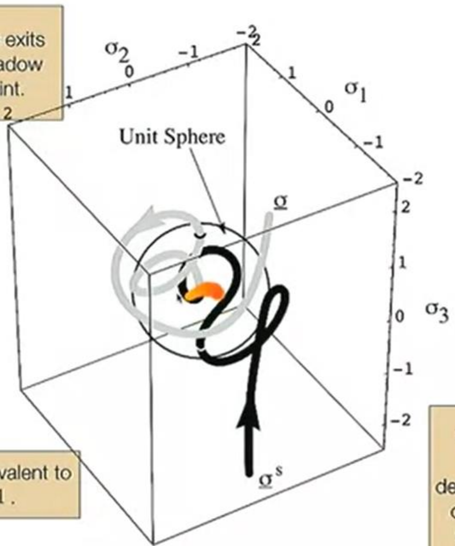
Unique MRP Parameters

A common switching surface is $\sigma^2 = \boldsymbol{\sigma} \cdot \boldsymbol{\sigma} = 1$. Note that

$$|\boldsymbol{\sigma}| \leq 1 \quad \text{if} \quad \Phi \leq 180^o$$
$$|\boldsymbol{\sigma}| \geq 1 \quad \text{if} \quad \Phi \geq 180^o$$
$$|\boldsymbol{\sigma}| = 1 \quad \text{if} \quad \Phi = 180^o$$

$$\boldsymbol{\sigma}^S = \tan\left(\frac{\Phi - 2\pi}{4}\right)\hat{\boldsymbol{e}}$$
$$\boldsymbol{\sigma}^S = \tan\left(\frac{\Phi'}{4}\right)\hat{\boldsymbol{e}}$$

As one set of MRP coordinates exits the unit sphere surface, the shadow set enters at the opposite point.



Setting $\beta_0 \geq 0$ is equivalent to enforcing $|\boldsymbol{\sigma}| \leq 1$.

The original shadow set of MRPs are convenient to describe tumbling bodies. The coordinates always point to the zero attitude along the shortest rotational path

Direction Cosine Matrix

Matrix components:

$$[C] = \frac{1}{(1+\sigma^2)^2} \begin{bmatrix} 4\left(\sigma_1^2-\sigma_2^2-\sigma_3^2\right)+(1-\sigma^2)^2 & 8\sigma_1\sigma_2+4\sigma_3(1-\sigma^2) & \cdots \\ 8\sigma_2\sigma_1-4\sigma_3(1-\sigma^2) & 4\left(-\sigma_1^2+\sigma_2^2-\sigma_3^2\right)+(1-\sigma^2)^2 & \cdots \\ 8\sigma_3\sigma_1+4\sigma_2(1-\sigma^2) & 8\sigma_3\sigma_2-4\sigma_1(1-\sigma^2) & \end{bmatrix}$$

$$\cdots \begin{bmatrix} 8\sigma_1\sigma_3-4\sigma_2(1-\sigma^2) \\ 8\sigma_2\sigma_3+4\sigma_1(1-\sigma^2) \\ 4\left(-\sigma_1^2-\sigma_2^2+\sigma_3^2\right)+(1-\sigma^2)^2 \end{bmatrix}$$

Vector computation:

$$[C] = [I_{3\times3}] + \frac{8[\tilde{\boldsymbol{\sigma}}]^2 - 4\left(1-\sigma^2\right)[\tilde{\boldsymbol{\sigma}}]}{\left(1+\sigma^2\right)^2}$$

Interesting property:

$$[C(\boldsymbol{\sigma})]^{-1} = [C(\boldsymbol{\sigma})]^T = [C(-\boldsymbol{\sigma})]$$

Attitude Addition/Subtraction

- DCM method:

$$[FN(\boldsymbol{\sigma})] = [FB(\boldsymbol{\sigma}'')][BN(\boldsymbol{\sigma}')]$$

- Direct method:

$$\boldsymbol{\sigma} = \frac{(1-|\boldsymbol{\sigma}'|^2)\boldsymbol{\sigma}'' + (1-|\boldsymbol{\sigma}''|^2)\boldsymbol{\sigma}' - 2\boldsymbol{\sigma}''\times\boldsymbol{\sigma}'}{1+|\boldsymbol{\sigma}'|^2|\boldsymbol{\sigma}''|^2 - 2\boldsymbol{\sigma}'\cdot\boldsymbol{\sigma}''}$$

Attitude Addition

$$\boldsymbol{\sigma}'' = \frac{(1-|\boldsymbol{\sigma}'|^2)\boldsymbol{\sigma} - (1-|\boldsymbol{\sigma}|^2)\boldsymbol{\sigma}' + 2\boldsymbol{\sigma}\times\boldsymbol{\sigma}'}{1+|\boldsymbol{\sigma}'|^2|\boldsymbol{\sigma}|^2 + 2\boldsymbol{\sigma}'\cdot\boldsymbol{\sigma}}$$

Relative Attitude (Subtraction)

Differential Kinematic Equations

Matrix components:

$$\dot{\boldsymbol{\sigma}} = \frac{1}{4}\begin{bmatrix} 1-\sigma^2+2\sigma_1^2 & 2\left(\sigma_1\sigma_2-\sigma_3\right) & 2\left(\sigma_1\sigma_3+\sigma_2\right) \\ 2\left(\sigma_2\sigma_1+\sigma_3\right) & 1-\sigma^2+2\sigma_2^2 & 2\left(\sigma_2\sigma_3-\sigma_1\right) \\ 2\left(\sigma_3\sigma_1-\sigma_2\right) & 2\left(\sigma_3\sigma_2+\sigma_1\right) & 1-\sigma^2+2\sigma_3^2 \end{bmatrix} {}^{\mathcal{B}}\!\begin{pmatrix}\omega_1\\ \omega_2\\ \omega_3\end{pmatrix}$$

Vector computation:

$$\dot{\boldsymbol{\sigma}} = \frac{1}{4}\left[\left(1-\sigma^2\right)[I_{3\times3}] + 2[\tilde{\boldsymbol{\sigma}}] + 2\boldsymbol{\sigma}\boldsymbol{\sigma}^T\right]{}^{\mathcal{B}}\boldsymbol{\omega} = \frac{1}{4}[B(\boldsymbol{\sigma})]\,{}^{\mathcal{B}}\boldsymbol{\omega}$$

Note: Only contains quadratic nonlinearities, but is singular for $\Phi = \pm 360^{\circ}$.

- Now, let's invert the differential kinematic equation and find:

$$\boldsymbol{\omega} = 4[B]^{-1}\dot{\boldsymbol{\sigma}}$$

- Note the near-orthogonal property of the [*B*] matrix:

$$[B]^{-1} = \frac{1}{(1+\sigma^2)^2}[B]^T$$

  You can proof this by investigating $[B][B]^T$.

- This leads to the elegant inverse transformation

$$\boldsymbol{\omega} = \frac{4}{(1+\sigma^2)^2}[B]^T\dot{\boldsymbol{\sigma}}$$

$$\boldsymbol{\omega} = \frac{4}{(1+\sigma^2)^2}\left[(1-\sigma^2)\,[I_{3\times3}] - 2[\tilde{\boldsymbol{\sigma}}] + 2\boldsymbol{\sigma}\boldsymbol{\sigma}^T\right]\dot{\boldsymbol{\sigma}}$$

$\sigma_0 = \sigma(t_0)$

$x = \sigma_0$

time $t_0 \to t_f$

$$\boxed{\begin{aligned} k_1 &= F(\ldots) \\ k_2 &= F(\ldots) \\ k_3 &= F(\ldots) \\ k_4 &= F(\ldots) \\ x_{n+1} &= x_n + (k\text{'s}) \end{aligned}}$$

integrator

$\to$ if $(|\sigma| > 1) \to \sigma = \dfrac{-\sigma}{\sigma^2}$

$x = \sigma$

$\dot{x} = f(x) = \frac{1}{4}[B]\omega$

Cayley Transform

- Let [*S*] be a skew-symmetric matrix, [*C*] be a proper orthogonal matrix, and [*I*] be a identity matrix. These matrices can be of any dimension *N*. The **extended Cayley Transform** is then defined as:

$$\boxed{[C] = ([I]-[S])^2([I]+[S])^{-2} = ([I]+[S])^{-2}([I]-[S])^2}$$

Unfortunately no equivalent inverse transformation exists. Instead, we define [*W*] to be the "square root" of [*C*]:

$$[C] = [W][W]$$

$$[C] = [V][D][V]^* \text{ — Adjoint Operator}$$

- The “matrix square root” can then be defined as

$$[W] = [V] \begin{bmatrix} \ddots & & 0 \\ & \sqrt{[D]_{ii}} & \\ 0 & & \ddots \end{bmatrix} [V]^*$$

$$[W] = [V] \begin{bmatrix} e^{+i\frac{\theta_1}{2}} & 0 & \cdots & & & 0 \\ 0 & e^{-i\frac{\theta_1}{2}} & \cdots & & & 0 \\ \vdots & \vdots & \ddots & & & 0 \\ & & & e^{+i\frac{\theta_{N-1}}{2}} & 0 & 0 \\ & & & 0 & e^{-i\frac{\theta_{N-1}}{2}} & 0 \\ & & & 0 & 0 & +1 \end{bmatrix} [V]^* \quad \text{Odd dimension}$$

$$[W] = [V] \begin{bmatrix} e^{+i\frac{\theta_1}{2}} & 0 & \cdots & & 0 \\ 0 & e^{-i\frac{\theta_1}{2}} & \cdots & & 0 \\ \vdots & \vdots & \ddots & & 0 \\ & & & e^{+i\frac{\theta_{N-1}}{2}} & 0 \\ & & & 0 & e^{-i\frac{\theta_{N-1}}{2}} \end{bmatrix} [V]^* \quad \text{Even dimension}$$

- The standard Cayley transform can now be used to between between the skew-symmetric [*S*] matrix and the orthogonal [*W*] matrix:

$$[W] = ([I] - [S])([I] + [S])^{-1} \qquad = ([I] + [S])^{-1}([I] - [S])$$
$$[S] = ([I] - [W])([I] + [W])^{-1} \qquad = ([I] + [W])^{-1}([I] - [W])$$

- As with the CRP coordinates, for the 3D case the [*S*] matrix elements are MRP attitude coordinates. For higher dimensional cases, this allows us to parameterize *N*-dimensional proper orthogonal matrices using higher dimensional MRP coordinates.

- Recall that regardless of the dimensionality of the orthogonal matrix [*W*(*t*)], it must evolve according

$$[\dot{W}] = -[\tilde{\mathbf{\Omega}}][W]$$

These higher-dimensional “body angular velocities” can be related to the higher dimensional MRPs using:

$$[\tilde{\omega}] = [\tilde{\mathbf{\Omega}}] + [W][\tilde{\mathbf{\Omega}}][W]^T$$
$$[\dot{S}] = \frac{1}{2}([I] + [S])[\tilde{\mathbf{\Omega}}]([I] - [S])$$

- This parameterization is singular whenever a principal rotation of 360° is performed.

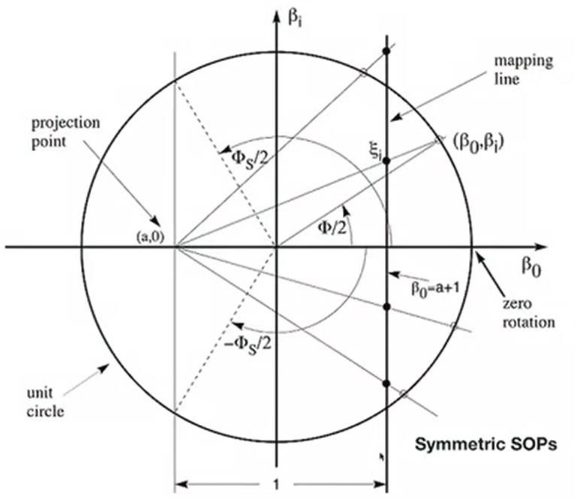
Stereographic Orientation Parameters (SOPs)

Elegant family of attitude coordinates...

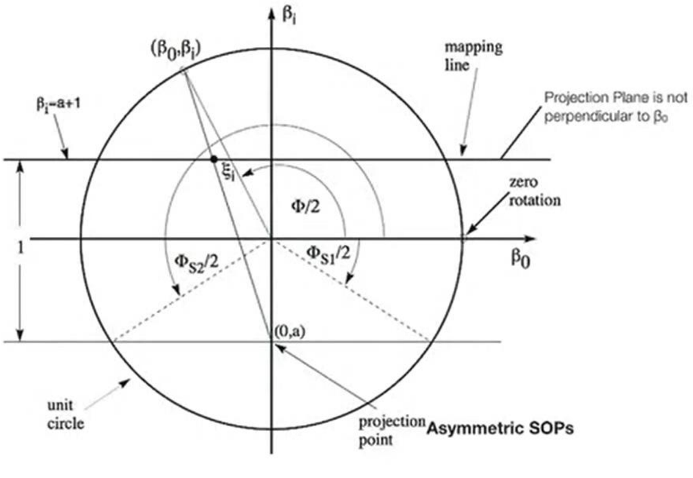
Quick facts...

- The Stereographic Orientation Parameters are a class of attitude parameters that generalize the previously discussed classical and modified Rodrigues parameters.
- The are two types of SOPs:
  - Symmetric Set: Goes singular if a $\pm\Phi$ principal rotation is performed.
  - Asymmetric Set: Goes singular at either Φ_1 or Φ_2, and this rotation must be about a particular axis.
- References:

- H. Schaub and J.L. Junkins. "Stereographic Orientation Parameters for Attitude Dynamics: A Generalization of the Rodrigues Parameters." *Journal of the Astronautical Sciences*, Vol. 44, No. 1, Jan.–Mar. 1996, pp. 1–19.
- C. M. Southward, J. Ellis and H. Schaub, "Spacecraft Attitude Control Using Symmetric Stereographic Orientation Parameters," *Journal of Astronautical Sciences*, Vol. 55, No. 3, July–September, 2007, pp. 389–405.

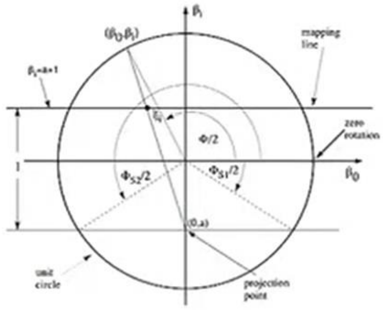


Symmetric SOPs



Asymmetric SOPs

Example: asymmetric SOP



Projection plane: $\beta_1 = 0$

Projection point: $\beta_1 = -1$

Mapping from EP:

$$\eta_1 = \frac{\beta_0}{1+\beta_1} \quad \eta_2 = \frac{\beta_2}{1+\beta_1} \quad \eta_3 = \frac{\beta_3}{1+\beta_1}$$

Mapping to EP:

$$\beta_0 = \frac{2\eta_1}{1+\eta^2} \quad \beta_1 = \frac{1-\eta^2}{1+\eta^2} \quad \beta_2 = \frac{2\eta_2}{1+\eta^2} \quad \beta_3 = \frac{2\eta_3}{1+\eta^2} \qquad \eta^2 = \boldsymbol{\eta}^T\boldsymbol{\eta}$$

Singular behavior:

$$\beta_1 \to -1 \begin{cases} \Phi_1 = -180^o \\ \Phi_2 = +540^o \end{cases}$$

Shadow set:

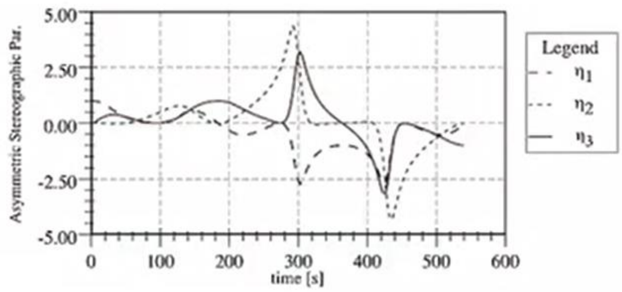
$$\boldsymbol{\eta}^S = -\frac{\boldsymbol{\eta}}{\boldsymbol{\eta}^T\boldsymbol{\eta}}$$

Prescribed 3-1-3 Euler Angle time histories:

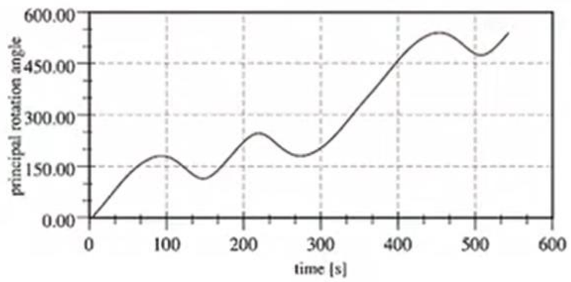
$$\theta_1(t) = t$$

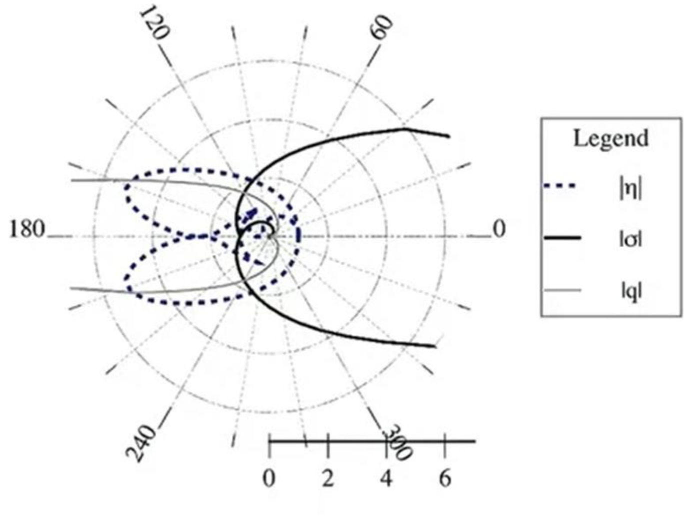
$$\theta_2(t) = (1-\cos 2t)\frac{\pi}{2}$$

$$\theta_3(t) = (\sin 2t)\frac{\pi}{4}$$

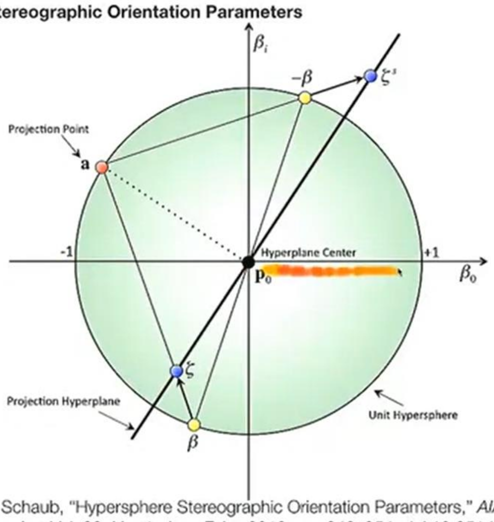


The body is essentially doing a tumble about the 1st body axis, while doing sinusoidal wobbles about the other axes.





Hyper-Surface Stereographic Orientation Parameters



J. Mullen and H. Schaub, "Hypersphere Stereographic Orientation Parameters," *AIAA Journal of Guidance, Control and Dynamics*, Vol. 33, No. 1, Jan.–Feb., 2010, pp. 249–254. doi:10.2514/1.46783

Attitude Determination

ASEN 5010

Dr. Hanspeter Schaub
hanspeter.schaub@colorado.edu

Introduction

- Attitude determination is broken up into two areas
  - **Static attitude determination**: All measurements are taken at the same time. Using this snap shot in time concept, the problem becomes up of optimally solving the geometry of the measurements
  - **Dynamic attitude determination**: Here measurements are taken over time. This is a much harder problem, in that attitude measurements are taken over time, along with some gyro (rotation rate) measurements, which then need to be optimally blended together (Kalman filter).

Basic Concept

- Consider the 2D attitude problem. How many direction measurements (unit direction vectors) does it take to determine your heading?

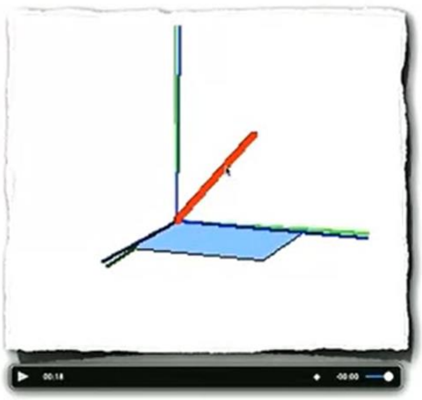
Answer: You only need one direction measurement for the 2D case.

Explanation: Headings in a 2D environment is a 1D measure. The unit direction vector (with the unit length constraint) provides all the required information.

- Next, let us consider the three dimensional orientation measurement. How many observation vectors (unit direction vectors) are required here?

Answer: You will need a minimum of two observation vectors.

Explanation: With only one measurement, you cannot sense rotations about this axis. Measuring a second direction will fix the complete three dimension orientation in space.



- To determine attitude, we assume you already know the inertial direction to certain objects (sun, Earth, magnetic field direction, stars, moon, etc.)

- Assume the sun direction is given by $\hat{s}$ and the local magnetic field direction is given by $\hat{m}$.

- If the vehicle has sensors on board that measure these directions, then these unit vectors are measured with components taken in the vehicle fixed body frame *B*.

Measured: ${}^{\mathcal{B}}\hat{m}$ ${}^{\mathcal{B}}\hat{s}$

Given: ${}^{\mathcal{N}}\hat{m}$ ${}^{\mathcal{N}}\hat{s}$

Mapping:

$$ {}^{\mathcal{B}}\hat{m} = [\bar{B}N]\,{}^{\mathcal{N}}\hat{m} $$

$$ {}^{\mathcal{B}}\hat{s} = [\bar{B}N]\,{}^{\mathcal{N}}\hat{s} $$

Challenge: How do we find [*BN*]?

Under or Over?

- Note that each observation vector (unit direction vector) contains two independent degrees of freedom.
- The 3D attitude problem is a three-degree of freedom problem.
- Thus, by measuring two observation directions, the attitude determination problem is always an over-determined problem!

Deterministic Attitude Estimation

Vector Triad Method

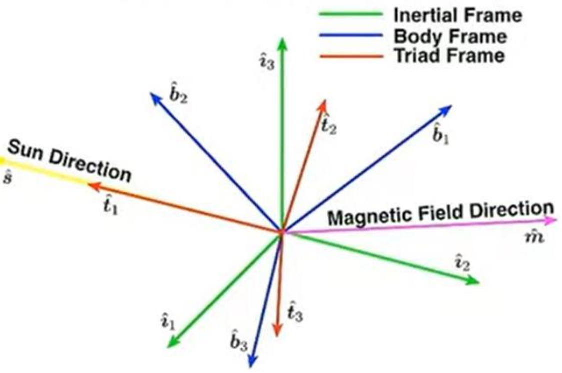
- To determine the desired [*BI*] matrix, we first introduce the triad coordinate frame *T*.

Assume: $\hat{t}_1 = \hat{s}$

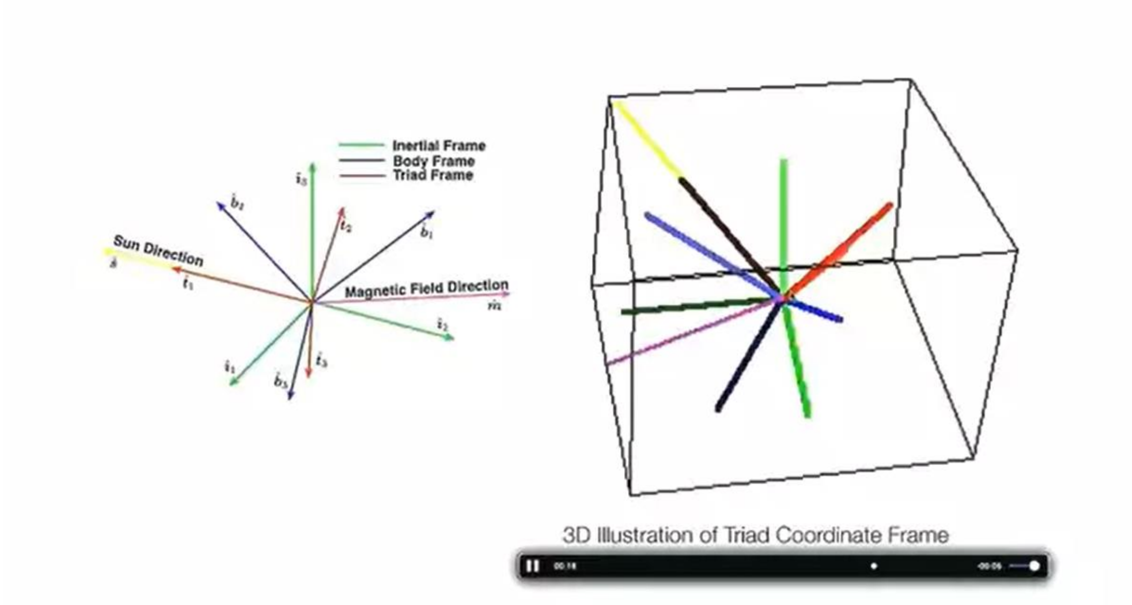
Then define:

$$ \hat{t}_2 = \frac{\hat{s} \times \hat{m}}{|\hat{s} \times \hat{m}|} $$

$$ \hat{t}_3 = \hat{t}_1 \times \hat{t}_2 $$



Sun direction is more precise. First axis is the measurement that you have more reliability on.



This method does not work if the two measurements are collinear

- We can compute the T frame direction axes using both B and I frame components using

$$
\begin{aligned}
{}^{\mathcal{B}}\hat{\boldsymbol{t}}_1 &= {}^{\mathcal{B}}\hat{\boldsymbol{s}} \\
{}^{\mathcal{B}}\hat{\boldsymbol{t}}_2 &= \frac{({}^{\mathcal{B}}\hat{\boldsymbol{s}}) \times ({}^{\mathcal{B}}\hat{\boldsymbol{m}})}{|({}^{\mathcal{B}}\hat{\boldsymbol{s}}) \times ({}^{\mathcal{B}}\hat{\boldsymbol{m}})|} \\
{}^{\mathcal{B}}\hat{\boldsymbol{t}}_3 &= ({}^{\mathcal{B}}\hat{\boldsymbol{t}}_1) \times ({}^{\mathcal{B}}\hat{\boldsymbol{t}}_2)
\end{aligned}
$$

Body Frame Triad Vectors

$$
\begin{aligned}
{}^{\mathcal{N}}\hat{\boldsymbol{t}}_1 &= {}^{\mathcal{N}}\hat{\boldsymbol{s}} \\
{}^{\mathcal{N}}\hat{\boldsymbol{t}}_2 &= \frac{({}^{\mathcal{N}}\hat{\boldsymbol{s}}) \times ({}^{\mathcal{N}}\hat{\boldsymbol{m}})}{|({}^{\mathcal{N}}\hat{\boldsymbol{s}}) \times ({}^{\mathcal{N}}\hat{\boldsymbol{m}})|} \\
{}^{\mathcal{N}}\hat{\boldsymbol{t}}_3 &= ({}^{\mathcal{N}}\hat{\boldsymbol{t}}_1) \times ({}^{\mathcal{N}}\hat{\boldsymbol{t}}_2)
\end{aligned}
$$

Inertial Frame Triad Vectors

- In the absence of measurement errors, both sets of Triad frame representations should be the same.
- We can write the various rotation matrices as

$$
[\bar{B}T] = \begin{bmatrix} {}^{\mathcal{B}}\hat{\boldsymbol{t}}_1 & {}^{\mathcal{B}}\hat{\boldsymbol{t}}_2 & {}^{\mathcal{B}}\hat{\boldsymbol{t}}_3 \end{bmatrix} \qquad [NT] = \begin{bmatrix} {}^{\mathcal{N}}\hat{\boldsymbol{t}}_1 & {}^{\mathcal{N}}\hat{\boldsymbol{t}}_2 & {}^{\mathcal{N}}\hat{\boldsymbol{t}}_3 \end{bmatrix}
$$

- Finally, we can compute the desired DCM matrix using

$$
[\bar{B}N] = [\bar{B}T][NT]^T
$$

- From the rotation matrix, we can now extract any desired set of attitude coordinates!
- Note that with this method we do not use the full magnetic field direction vector $\hat{\boldsymbol{m}}$. If this measurement were more accurate, then we could modify this method to define $\hat{\boldsymbol{t}}_1 = \hat{\boldsymbol{m}}$ instead.

Example 3.14 (Triad)

Setup the true attitude states:

```
θtrue = {30., 20., -10.} Degree;
BNtrue = Euler3212C[θtrue]

{{0.813798, 0.469846, -0.34202},
 {-0.543838, 0.823173, -0.163176}, {0.204874, 0.318796, 0.925417}}

v1N = {1, 0, 0};
v2N = {0, 0, 1};

MatrixForm[v1Btrue = BNtrue.v1N ]
```

$$\begin{pmatrix} 0.813798 \\ -0.543838 \\ 0.204874 \end{pmatrix}$$

```
MatrixForm[v2Btrue = BNtrue.v2N ]
```

$$\begin{pmatrix} -0.34202 \\ -0.163176 \\ 0.925417 \end{pmatrix}$$

Setup the measured attitude states:

```
v1B = {0.8190, -0.5282, 0.2242};
v2B = {-0.3138, -0.1584, 0.9362};
v1B = v1B / Norm[v1B];
v2B = v2B / Norm[v2B];
```

Assuming suppressed output | Use as a vector instead

show output

Develop Triad Frame

From measured states

```
MatrixForm[t1B = v1B]
t2B = Cross[v1B, v2B];
MatrixForm[t2B = t2B / Norm[t2B]]
MatrixForm[t3B = Cross[t1B, t2B]]
```

$$\begin{pmatrix} 0.818991 \\ -0.528194 \\ 0.224198 \end{pmatrix}$$

$$\begin{pmatrix} -0.459282 \\ -0.837639 \\ -0.295669 \end{pmatrix}$$

$$\begin{pmatrix} 0.343967 \\ 0.13918 \\ -0.928609 \end{pmatrix}$$

From inertial states of the measurements

```
In[17]:= MatrixForm[t1N = v1N]
         t2N = Cross[t1N, v2N ];
         MatrixForm[t2N = t2N / Norm[t2N]]
         MatrixForm[t3N = Cross[t1N, t2N]]
```

Out[17]//MatrixForm=

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

Out[19]//MatrixForm=

$$\begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix}$$

Out[20]//MatrixForm=

$$\begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix}$$

Find Estimated Attitude

In[21]:=
```
MatrixForm[BbarT = Transpose[{t1B, t2B, t3B}]]
```

Out[21]//MatrixForm=

$$\begin{pmatrix} 0.818991 & -0.459282 & 0.343967 \\ -0.528194 & -0.837639 & 0.13918 \\ 0.224198 & -0.295669 & -0.928609 \end{pmatrix}$$

```
NT = Transpose[{t1N, t2N, t3N}]
```

```
{{1, 0, 0}, {0, -1, 0}, {0, 0, -1}}
```

```
MatrixForm[BbarN = BbarT.Transpose[NT]]
```

$$\begin{pmatrix} 0.818991 & 0.459282 & -0.343967 \\ -0.528194 & 0.837639 & -0.13918 \\ 0.224198 & 0.295669 & 0.928609 \end{pmatrix}$$

Check accuracy of estimate

```
MatrixForm[BbarB = BbarN.Transpose[BNtrue]]
```

$$\begin{pmatrix} 0.999929 & -0.0112026 & -0.00410552 \\ 0.0113209 & 0.999485 & 0.0300232 \\ 0.00376707 & -0.0300675 & 0.999541 \end{pmatrix}$$

```
p = C2PRV[BbarB]
```

```
{0.0300506, 0.00393698, -0.0112637}
```

```
Norm[p] / Degree
```

```
1.85253
```

Statistical Attitude Determination

Wahba's Problem

- Assume we have *N*>1 observation measurements (i.e. measured directions to sun, magnetic field, stars, etc.), and we know the corresponding inertial vector directions. Then we can write attitude determination problem as

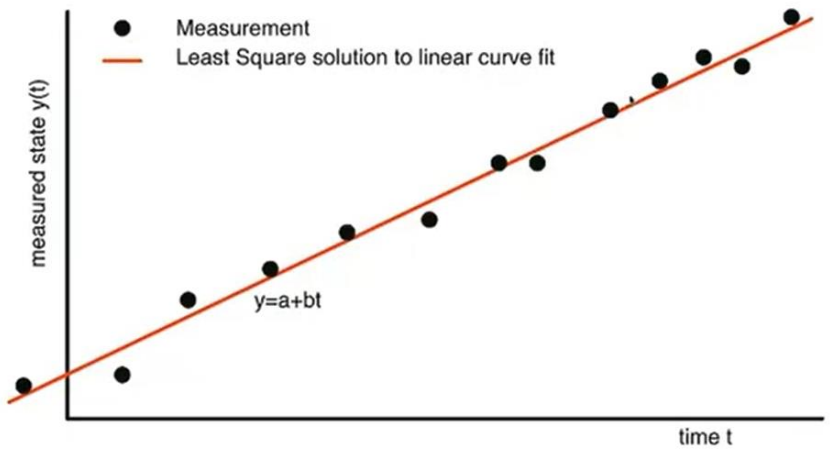
$$ {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k = [\bar{B}N]\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k \qquad \text{for } k = 1, \cdots, N $$

with the goal to find the rotation matrix [*BN*] such that the following loss function is minimized:

- If all measurements are perfect, then *J* = 0.

$$ J([\bar{B}N]) = \frac{1}{2}\sum_{k=1}^{N} w_k \left| {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k - [\bar{B}N]\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k \right|^2 $$

- Think of the cost function *J* as the error of the common least squares curve fitting problem:



Devenport's *q*-Method

- Let the 4-D Euler parameter (quaternion) vector be defined as

$$ \bar{\boldsymbol{\beta}} = (\beta_0, \beta_1, \beta_2, \beta_3)^T $$

- The cost function can be rewritten

$$ J = \frac{1}{2}\sum_{k=1}^{N} w_k \left({}^{\mathcal{B}}\hat{\boldsymbol{v}}_k - [\bar{B}N]\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k \right)^T \left({}^{\mathcal{B}}\hat{\boldsymbol{v}}_k - [\bar{B}N]\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k \right) $$

$$ J = \frac{1}{2}\sum_{k=1}^{N} w_k \left({}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T\ {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k + {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k^T\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k - 2\ {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T [\bar{B}N]\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k \right) $$

$$ J = \sum_{k=1}^{N} w_k \left(1 - {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T\ [\bar{B}N]\ {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k \right) $$

- Minimizing J is equivalent to maximizing the gain function g:

$$g = \sum_{k=1}^{N} w_k \, {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T \; [\bar{B}N] \; {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k$$

- The rotation matrix can be written in terms of Euler parameters as

$$[\bar{B}N] = (\beta_0^2 - \boldsymbol{\epsilon}^T \boldsymbol{\epsilon})[I_{3\times 3}] + 2\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T - 2\beta_0[\tilde{\boldsymbol{\epsilon}}] \qquad \boldsymbol{\epsilon} = (\beta_1, \beta_2, \beta_3)$$

- This allows us to rewrite the gain function $g()$ using the 4x4 matrix $[K]$

$$\boxed{g(\bar{\boldsymbol{\beta}}) = \bar{\boldsymbol{\beta}}^T [K] \bar{\boldsymbol{\beta}}} \qquad [K] = \begin{bmatrix} \sigma & Z^T \\ Z & S - \sigma I_{3\times 3} \end{bmatrix}$$

$$[B] = \sum_{k=1}^{N} w_k \, {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k \, {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k^T \qquad [S] = [B] + [B]^T$$

$$\sigma = \mathrm{tr}\,([B])$$

$$[Z] = \begin{bmatrix} B_{23} - B_{32} & B_{31} - B_{13} & B_{12} - B_{21} \end{bmatrix}^T$$

- However, since the Euler parameter vector must abide by the unit length constraint, we cannot solve this gain function directly. Instead, we use Lagrange multipliers to yield a new gain function g'

$$g'(\bar{\boldsymbol{\beta}}) = \bar{\boldsymbol{\beta}}^T [K] \bar{\boldsymbol{\beta}} - \lambda(\bar{\boldsymbol{\beta}}^T \bar{\boldsymbol{\beta}} - 1)$$

- We differentiate g' and set it equal to zero to find the extrema point of this function.

$$\frac{\mathrm{d}}{\mathrm{d}\bar{\boldsymbol{\beta}}}(g'(\bar{\boldsymbol{\beta}})) = 2[K]\bar{\boldsymbol{\beta}} - 2\lambda\bar{\boldsymbol{\beta}} = 0 \quad \Rightarrow \quad \boxed{[K]\bar{\boldsymbol{\beta}} = \lambda\bar{\boldsymbol{\beta}}}$$

- To maximize the gain function, we need to choose the largest eigenvalue of the $[K]$ matrix.

$$g(\bar{\boldsymbol{\beta}}) = \bar{\boldsymbol{\beta}}^T [K] \bar{\boldsymbol{\beta}} = \bar{\boldsymbol{\beta}}^T \lambda \bar{\boldsymbol{\beta}} = \lambda \bar{\boldsymbol{\beta}}^T \bar{\boldsymbol{\beta}} = \lambda$$

- In summary, to use the q-Method, we must
  - Compute the 4x4 matrix $[K]$
  - Find the eigenvalue and eigenvector of the $[K]$ matrix
  - Choose the largest eigenvalue and associated eigenvector.
  - This eigenvector is the Euler parameter vector

- Note that solving this eigenvalue, eigenvector problem is numerically rather intensive for real-time applications.

Example 3.15 (Devenport's q-Method)

Setup the true attitude states:

```
θtrue = {30., 20., -10.} Degree;
BNtrue = Euler3212C[θtrue]
```

```
{{0.813798, 0.469846, -0.34202},
 {-0.543838, 0.823173, -0.163176}, {0.204874, 0.318796, 0.925417}}
```

```
v1N = {1, 0, 0};
v2N = {0, 0, 1};
```

```
MatrixForm[v1Btrue = BNtrue.v1N ]
```

$$\begin{pmatrix} 0.813798 \\ -0.543838 \\ 0.204874 \end{pmatrix}$$

```
MatrixForm[v2Btrue = BNtrue.v2N ]
```

$$\begin{pmatrix} -0.34202 \\ -0.163176 \\ 0.925417 \end{pmatrix}$$

Setup the measured attitude states:

```
v1B = {0.8190, -0.5282, 0.2242};
v2B = {-0.3138, -0.1584, 0.9362};
v1B = v1B / Norm[v1B];
v2B = v2B / Norm[v2B];
```

Setup q-method parameters

```
w1 = 1;
w2 = 1;
```

In[101]:=

```
B = w1 Outer[Times, v1B, v1N] + w2 Outer[Times, v2B, v2N]
```

Out[101]=

```
{{0.818991, 0., -0.313795},
 {-0.528194, 0., -0.158398}, {0.224198, 0., 0.936185}}
```

In[102]:=

```
S = B + Transpose[B]
```

Out[102]=

```
{{1.63798, -0.528194, -0.0895975},
 {-0.528194, 0., -0.158398}, {-0.0895975, -0.158398, 1.87237}}
```

In[103]:=

```
σ = B[[1, 1]] + B[[2, 2]] + B[[3, 3]]
```

Out[103]=

```
1.75518
```

In[104]:=

```
Z = {B[[2, 3]] - B[[3, 2]],
  B[[3, 1]] - B[[1, 3]],
  B[[1, 2]] - B[[2, 1]]}
```

Out[104]=

```
{-0.158398, 0.537993, 0.528194}
```

In[105]:=

```
MatrixForm[
 K = {{σ, Z[[1]], Z[[2]], Z[[3]]},
   {Z[[1]], S[[1, 1]] - σ, S[[1, 2]], S[[1, 3]]},
   {Z[[2]], S[[2, 1]], S[[2, 2]] - σ, S[[2, 3]]},
   {Z[[3]], S[[3, 1]], S[[3, 2]], S[[3, 3]] - σ}}
]
```

Out[105]//MatrixForm=

$$\begin{pmatrix} 1.75518 & -0.158398 & 0.537993 & 0.528194 \\ -0.158398 & -0.117194 & -0.528194 & -0.0895975 \\ 0.537993 & -0.528194 & -1.75518 & -0.158398 \\ 0.528194 & -0.0895975 & -0.158398 & 0.117194 \end{pmatrix}$$

Solve for optimal attitude

In[106]:=

```
{λ, βvectors} = Eigensystem[K];
```

In[107]:=

```
λ
```

Out[107]=

```
{1.99967, -1.99967, 0.0365659, -0.0365659}
```

In[108]:=

```
β = βvectors[[1]]
```

Out[108]=

```
{0.948069, -0.117207, 0.141371, 0.259697}
```

In[109]:=

```
MatrixForm[BbarN = EP2C[β]]
```

Out[109]//MatrixForm=

$$\begin{pmatrix} 0.825143 & 0.459282 & -0.328936 \\ -0.525561 & 0.837639 & -0.148814 \\ 0.207182 & 0.295669 & 0.932553 \end{pmatrix}$$

Check accuracy of estimate

In[110]:=

```
MatrixForm[BbarB = BbarN.Transpose[BNtrue]]
```

Out[110]//MatrixForm=

$$\begin{pmatrix} 0.999794 & -0.0170009 & 0.0110648 \\ 0.0167585 & 0.999625 & 0.0216474 \\ -0.0114287 & -0.0214576 & 0.999704 \end{pmatrix}$$

In[111]:=

```
q = C2PRV[BbarB]
```

Out[111]=

```
{0.0215556, -0.0112484, -0.0168821}
```

In[112]:=

```
θErrorQmethod = Norm[q] / Degree
```

Out[112]=

```
1.69597
```

QUEST

- Recall the cost function J and the gain function g

$$J = \sum_{k=1}^{N} w_k \left(1 - {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T \; [\bar{B}N] \; {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k\right)$$

$$g = \sum_{k=1}^{N} w_k {}^{\mathcal{B}}\hat{\boldsymbol{v}}_k^T \; [\bar{B}N] \; {}^{\mathcal{N}}\hat{\boldsymbol{v}}_k$$

- Further, we found that the optimal g will be

$$g(\bar{\boldsymbol{\beta}}) = \lambda_{\text{opt}}$$

- This can now be rewritten in the useful form

$$J = \sum_{k=1}^{N} w_k - g = \sum_{k=1}^{N} w_k - \lambda_{\text{opt}}$$

- Finally, the optimality condition can be written as

$$\lambda_{\text{opt}} = \sum_{k=1}^{N} w_k - J$$

- Note that J should be small for an optimal solution. This assumes that the measurement noise is reasonable small and Gaussian. The QUEST method then makes the elegant assumption that

$$\lambda_{\text{opt}} \approx \sum_{k=1}^{N} w_k$$

- This allows us to avoid the numerically intensive eigenvalue problem!
- However, we still need to find a solution for the eigenvector.

- The eigenvalues of $[K]$ must satisfy the characteristic equation:

$$f(s) = \det([K] - s[I_{4\times4}]) = 0$$

- The desire root can be solve using a classic Newton-Raphson iteration method:

$$\lambda_0 = \sum_{k=1}^{N} w_k$$

$$\lambda_1 = \lambda_0 - \frac{f(\lambda_0)}{f'(\lambda_0)}$$

$$\vdots$$

$$\lambda_{\max} = \lambda_i = \lambda_{i-1} - \frac{f(\lambda_{i-1})}{f'(\lambda_{i-1})}$$

- Let use introduce the classical Rodrigues parameter vector $\boldsymbol{q}$

$$\boldsymbol{q} = \hat{\boldsymbol{e}} \tan\left(\frac{\Phi}{2}\right) = \frac{1}{\beta_0}\begin{bmatrix}\beta_1\\ \beta_2\\ \beta_3\end{bmatrix} = \frac{\boldsymbol{\epsilon}}{\beta_0}$$

- Note that

$$\frac{\bar{\boldsymbol{\beta}}}{\beta_0} = \begin{bmatrix}1\\ \bar{\boldsymbol{q}}\end{bmatrix}$$

- The eigenvector problem is now re-written as

$$[K]\frac{\bar{\boldsymbol{\beta}}}{\beta_0} = \lambda_{\text{opt}}\frac{\bar{\boldsymbol{\beta}}}{\beta_0}$$

$$\begin{bmatrix}\sigma & Z^T\\ Z & S-\sigma I_{3\times 3}\end{bmatrix}\begin{bmatrix}1\\ \bar{\boldsymbol{q}}\end{bmatrix} = \lambda_{\text{opt}}\begin{bmatrix}1\\ \bar{\boldsymbol{q}}\end{bmatrix}$$

$$([S]-\sigma[I_{3\times 3}])\,\bar{\boldsymbol{q}} + [Z] = \lambda_{\text{opt}}\bar{\boldsymbol{q}}$$

- Finally, the classical Rodrigues parameter vector is found

$$\bar{\boldsymbol{q}} = \left((\lambda_{\text{opt}}+\sigma)[I_{3\times 3}]-[S]\right)^{-1}[Z]$$

- Note that we still have to take an inverse of a 3x3 matrix here. However, this is numerically a very fast process.

- To solve for the corresponding 4-D Euler parameter vector, we use

$$\bar{\boldsymbol{\beta}} = \frac{1}{\sqrt{1+\bar{\boldsymbol{q}}^T\bar{\boldsymbol{q}}}}\begin{bmatrix}1\\ \bar{\boldsymbol{q}}\end{bmatrix}$$

Setup the true attitude states:

Setup the measured attitude states:

Setup q-method parameters

Setup QUEST parameters

In[154]:=

```
λ = w1 + w2
```

Out[154]=

```
2
```

In[155]:=

```
q = Inverse[ (λ + σ) IdentityMatrix[3] - S].Z
```

Out[155]=

```
{-0.123602, 0.1491, 0.273874}
```

In[156]:=

```
MatrixForm[BbarN = Gibbs2C[q]]
```

Out[156]//MatrixForm=

$$\begin{pmatrix} 0.825193 & 0.45922 & -0.328897 \\ -0.525482 & 0.837693 & -0.148793 \\ 0.207186 & 0.295613 & 0.93257 \end{pmatrix}$$

In[157]:=

```
MatrixForm[BbarB = BbarN.Transpose[BNtrue]]
```

Out[157]//MatrixForm=

$$\begin{pmatrix} 0.999793 & -0.0170853 & 0.0110912 \\ 0.0168417 & 0.999623 & 0.0216995 \\ -0.0114577 & -0.0215082 & 0.999703 \end{pmatrix}$$

In[158]:=

```
q = C2PRV[BbarB]
```

Out[158]=

```
{0.021607, -0.0112761, -0.016966}
```

In[159]:=

```
Norm[q] / Degree
```

Out[159]=

```
1.70146
```

show all digits | integer part | digits | root approximant | more...

Iterate for Optimal Attitude

In[160]:=

```
Det[K - s * IdentityMatrix[4]]
```

Out[160]=

$$0.00534646 + 7.35523 \times 10^{-16}\, s - 4.\, s^2 - 2.22045 \times 10^{-16}\, s^3 + s^4$$

In[161]:=

```
f[s_] := Det[K - s * IdentityMatrix[4]]
```

In[162]:=

```
λ0 = λ
```

Out[162]=

```
2
```

In[163]:=

```
f[λ0]
```

Out[163]=

0.00534646

In[164]:=

```
λ1 = λ0 - f[λ0] / f'[λ0]
```

Out[164]=

1.99967

In[165]:=

```
f[λ1]
```

Out[165]=

2.23288×10^{-6}

In[166]:=

```
λ2 = λ1 - f[λ1] / f'[λ1]
```

Out[166]=

1.99967

In[167]:=

```
f[λ2]
```

Out[167]=

3.90559×10^{-13}

In[168]:=

```
λ3 = λ2 - f[λ2] / f'[λ2]
```

Out[168]=

1.99967

In[169]:=

```
f[λ3]
```

Out[169]=

-3.45856×10^{-16}

Optimal Linear Attitude Estimator (OLAE)

Cayley Transform

$$[\bar{B}N] = ([I_{3\times3}] + [\tilde{\bar{q}}])^{-1}([I_{3\times3}] - [\tilde{\bar{q}}])$$

$$\Downarrow$$

$${}^{\mathcal{B}}\hat{v}_i = [\bar{B}N]\,{}^{\mathcal{N}}\hat{v}_i$$

$$([I_{3\times3}] + [\tilde{\bar{q}}])\,{}^{\mathcal{B}}\hat{v}_i = ([I_{3\times3}] - [\tilde{\bar{q}}])\,{}^{\mathcal{N}}\hat{v}_i$$

$${}^{\mathcal{B}}\hat{v}_i - {}^{\mathcal{N}}\hat{v}_i = -[\tilde{\bar{q}}]({}^{\mathcal{B}}\hat{v}_i + {}^{\mathcal{N}}\hat{v}_i)$$

$$\Downarrow$$

Define:

$$s_i = {}^{\mathcal{B}}\hat{v}_i + {}^{\mathcal{N}}\hat{v}_i$$
$$d_i = {}^{\mathcal{B}}\hat{v}_i - {}^{\mathcal{N}}\hat{v}_i$$

$$\Downarrow$$

$$d_i = [\tilde{s}_i]\bar{q}$$

$$d = \begin{bmatrix} d_1 \\ \cdots \\ d_N \end{bmatrix} \qquad [S] = \begin{bmatrix} \tilde{s}_1 \\ \cdots \\ \tilde{s}_N \end{bmatrix}$$

$$[W] = \begin{bmatrix} w_1 I_{3\times3} & 0_{3\times3} & \ddots \\ 0_{3\times3} & \ddots & 0_{3\times3} \\ \ddots & 0_{3\times3} & w_N I_{3\times3} \end{bmatrix}$$

$$\Downarrow$$

$$\boxed{\bar{q} = \left([S]^T[W][S]\right)^{-1}[S]^T[W]d}$$

Example 3.17 (OLAE)

Setup the true attitude states:

Setup the measured attitude states:

Evaluate OLAE states

In[180]:=
```
W = IdentityMatrix[6];
```

In[181]:=
```
d = Join[v1B - v1N, v2B - v2N]
```

Out[181]=
```
{-0.181009, -0.528194, 0.224198, -0.313795, -0.158398, -0.0638147}
```

In[182]:=
```
S = Join[tilde[v1B + v1N], tilde[v2B + v2N]]
```

Out[182]=
```
{{0, -0.224198, -0.528194}, {0.224198, 0, -1.81899},
 {0.528194, 1.81899, 0}, {0, -1.93619, -0.158398},
 {1.93619, 0, 0.313795}, {0.158398, -0.313795, 0}}
```

In[183]:=
```
qBar = Inverse[Transpose[S].W.S].Transpose[S].W.d
```

Out[183]=
```
{-0.12359, 0.148759, 0.274255}
```

In[184]:=
```
BbarN = Gibbs2C[qBar]
```

Out[184]=
```
{{0.825016, 0.459942, -0.328332},
 {-0.526039, 0.837338, -0.148823}, {0.206474, 0.295497, 0.932765}}
```

Check accuracy of estimate

In[185]:=
```
BbarB = BbarN.Transpose[BNtrue]
```

Out[185]=
```
{{0.999794, -0.0164875, 0.0118086},
 {0.0162316, 0.999638, 0.0214443}, {-0.0121579, -0.0212482, 0.9997}}
```

In[186]:=
```
q = C2PRV[BbarB]
```

Out[186]=
```
{0.0213494, -0.011985, -0.0163619}
```

In[187]:=
```
ErrorQmethod = Norm[q] / Degree
```

Out[187]=
```
1.68721
```