ROTATION CONVERSIONS

S. Visalakshi

DCM to Quaternion:

The Quaternion can be obtained from DCM using the below formulae:

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

However, this formula won't work in cases where β_0 is zero. Hence, Sheppard's approach is used to convert the Direction Cosine Matrix to Quaternion.

1st step: Find largest value of

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

2nd step: Compute the remaining EPs using

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

In the 2nd step, the largest value is used to compute the remaining parameters.

MATLAB function:

```
%DCM to quaternion convertor
%Input: 3x3 DCM matrix
%Output: 4x1 quaternion

function q = DCM2quaternion(DCM)
qt(1,1) = (1 + trace(DCM))/4;
qt(2,1) = (1 + 2*DCM(2, 2) - trace(DCM))/4;
qt(3,1) = (1 + 2*DCM(1, 1) - trace(DCM))/4;
qt(4,1) = (1 + 2*DCM(3, 3) - trace(DCM))/4;
x = max(qt);
if x == qt(1,1)
    q(1,1) = sqrt(qt(1,1));
    q(2,1) = (DCM(2,3) - DCM(3,2))/(4*q(1,1));
    q(3,1) = (DCM(3,1) - DCM(1,3))/(4*q(1,1));
    q(4,1) = (DCM(1,2) - DCM(2,1))/(4*q(1,1));
```

```
elseif x == qt(2,1)
    q(2,1) = sqrt(qt(2,1));
    q(1,1) = (DCM(2,3) - DCM(3,2))/(4*q(2,1));
    q(3,1) = (DCM(1,2) + DCM(2,1))/(4*q(2,1));
    q(4,1) = (DCM(3,1) + DCM(1,3))/(4*q(2,1));
elseif x == qt(3,1)
    q(3,1) = sqrt(qt(3,1));
    q(1,1) = (DCM(3,1) - DCM(1,3))/(4*q(3,1));
    q(2,1) = (DCM(1,2) + DCM(2,1))/(4*q(3,1));
    q(4,1) = (DCM(2,3) + DCM(3,2))/(4*q(3,1));
else
    q(4,1) = sqrt(qt(4,1));
    q(1,1) = (DCM(1,2) - DCM(2,1))/(4*q(4,1));
    q(2,1) = (DCM(3,1) + DCM(1,3))/(4*q(4,1));
    q(3,1) = (DCM(2,3) + DCM(3,2))/(4*q(4,1));
end
```

Quaternion to DCM:

The conversion from quaternion to DCM can be achieved using the following formula:

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

MATLAB function:

```
%Quaternion to DCM converter
%Input: 4x1 Quaternion
%Output: 3x3 DCM Matrix

function DCM = quaternion2DCM(q)
DCM = zeros(3,3);
DCM(1, 1) = q(1,1)*q(1,1) + q(2,1)*q(2,1) - q(3,1)*q(3,1) - q(4,1)*q(4,1);
DCM(1, 2) = 2*((q(2, 1) * q(3, 1)) + (q(1, 1) * q(4, 1)));
DCM(1, 3) = 2*((q(2, 1) * q(4, 1)) - (q(1, 1) * q(3, 1)));
DCM(2, 1) = 2*((q(2, 1) * q(3, 1)) - (q(1, 1) * q(4, 1)));
DCM(2, 2) = q(1,1)*q(1,1) - q(2,1)*q(2,1) + q(3,1)*q(3,1) - q(4,1)*q(4,1);
DCM(2, 3) = 2*((q(3, 1) * q(4, 1)) + (q(1, 1) * q(2, 1)));
DCM(3, 1) = 2*((q(2, 1) * q(4, 1)) + (q(1, 1) * q(3, 1)));
DCM(3, 2) = 2*((q(3, 1) * q(4, 1)) - (q(1, 1) * q(2, 1)));
DCM(3, 3) = q(1,1)*q(1,1) - q(2,1)*q(2,1) - q(3,1)*q(3,1) + q(4,1)*q(4,1);
```

Euler Angles to Quaternion:

Consider the 3-2-1 Euler angle sequence (yaw, pitch, roll). The conversion of Euler angles to quaternion can be achieved by using the axis- angle representation for each individual Euler angle. The formulae are as shown:

$$\beta_0 = \cos\frac{\Psi}{2}\cos\frac{\theta}{2}\cos\frac{\phi}{2} + \sin\frac{\Psi}{2}\sin\frac{\theta}{2}\sin\frac{\phi}{2}$$
$$\beta_1 = \cos\frac{\Psi}{2}\cos\frac{\theta}{2}\sin\frac{\phi}{2} - \sin\frac{\Psi}{2}\sin\frac{\theta}{2}\cos\frac{\phi}{2}$$
$$\beta_2 = \sin\frac{\Psi}{2}\cos\frac{\theta}{2}\sin\frac{\phi}{2} + \cos\frac{\Psi}{2}\sin\frac{\theta}{2}\cos\frac{\phi}{2}$$
$$\beta_3 = \sin\frac{\Psi}{2}\cos\frac{\theta}{2}\cos\frac{\phi}{2} - \cos\frac{\Psi}{2}\sin\frac{\theta}{2}\sin\frac{\phi}{2}$$

where Ψ represents the yaw angle, θ represents the pitch angle, and ϕ represents the roll angle.

MATLAB function:

```
%3-2-1 Euler Angles to Quaternions converter
%Input: 3-2-1 Euler Angles(yaw, pitch, roll)
%Output: 4x1 Quaternion

function q = eulerAngle2quat(yaw, pitch, roll)
q = zeros(4, 1);
yaw = deg2rad(yaw);
pitch = deg2rad(pitch);
roll = deg2rad(roll);
q(1, 1) = (cos(yaw/2) * cos(pitch/2) * cos(roll/2)) + (sin(yaw/2) * sin(pitch/2) * sin(roll/2));
q(2, 1) = -(sin(yaw/2) * sin(pitch/2) * cos(roll/2)) + (cos(yaw/2) * cos(pitch/2) * sin(roll/2));
q(3, 1) = (cos(yaw/2) * sin(pitch/2) * cos(roll/2)) + (sin(yaw/2) * cos(pitch/2) * sin(roll/2));
q(4, 1) = (sin(yaw/2) * cos(pitch/2) * cos(roll/2)) - (cos(yaw/2) * sin(pitch/2) * sin(roll/2));
```

Quaternion to Euler Angles:

Quaternion is converted to 3-2-1 Euler Angle sequence as shown:

$$\Psi = \tan^{-1}\left[\frac{2(\beta_1\beta_2 + \beta_0\beta_3)}{\beta_0^2 + \beta_1^2 - \beta_2^2 - \beta_3^2}\right]$$
$$\theta = \sin^{-1}\left[2(\beta_0\beta_2 - \beta_1\beta_3)\right]$$
$$\phi = \tan^{-1}\left[\frac{2(q_2q_3 + q_0q_1)}{q_0^2 - q_1^2 - q_2^2 + q_3^2}\right]$$

where Ψ represents the yaw angle, θ represents the pitch angle, and ϕ represents the roll angle. The above expressions are obtained, by first converting the quaternion to DCM representation, and then to Euler angles.

MATLAB function:

```
%Quaternion to Euler Angle converter
%Input: 4x1 Quaternion
%Output: 3-2-1 Euler Angles(yaw, pitch, roll)

function [yaw, pitch, roll] = quaternion2eulerAngle(q)
yaw = atan2(2*(q(2, 1) * q(3, 1) + q(1, 1) * q(4, 1))/(q(1, 1) * q(1, 1) + q(2, 1) * q(2, 1) - q(3, 1) * q(3, 1) - q(4, 1) * q(4, 1)), 1);
pitch = asin(2 * (q(1, 1) * q(3, 1) - q(2, 1) * q(4, 1)));
roll = atan2(2*(q(3, 1) * q(4, 1) + q(1, 1) * q(2, 1))/(q(1, 1) * q(1, 1) - q(2, 1) * q(2, 1) - q(3, 1) * q(3, 1) + q(4, 1) * q(4, 1)), 1);
yaw = rad2deg(yaw);
pitch = rad2deg(pitch);
roll = rad2deg(roll);
```

References:

- Spacecraft Dynamics and Control: University of Colorado Boulder
- Euler angles to Quaternions conversion: Mathematics Stack Exchange