

Spacecraft Attitude Determination and Control Toolbox Manual

SPC 413 - Project Report

Table of Contents

1 Angle Conversions	3
1.1 DCM to Quaternion	3
1.2 Quaternion to DCM	4
Non-Homogeneous	4
Homogeneous	5
1.3 Euler to DCM	5
1.4 DCM to Euler	5
1.5 Euler to Quaternion	6
1.6 Quaternion to Euler	6
2 Attitude Determination Methods	6
2.1 Biased Triad	7
2.2 Balanced Triad	7
2.3 Q-method	8
2.4 QUEST method	9
3 Sensor Measurements	9
3.1 Sun Sensor (2D)	9
From Time to Julian Date	10
From Julian Date to sun in the inertial frame	11
4 Equations of Motion Visualization	11
4.1 Non-Linear Equations of Motion	11
4.2 Linear Equations of Motion	12
1. Main	12

2. Angle Conversions	12
2.1 Input and Output	12
2.2 Solved Example	14
3. Attitude Determination	14
3.1 Inputs and Outputs	14
3.2 Solved Example	16
4. Sun Sensor	17
4.1 Inputs and Outputs	17
4.2 Solved Example	18
5. Motion Simulation	19
5.1 Linear	19
5.2 Non-Linear	20

Functions

In this manual, we will discuss our toolbox and the main functions used in it.

1 Angle Conversions

1.1 DCM to Quaternion

We convert from DCM to quaternion by using the following equations:

$$\begin{aligned}q_s &= \sqrt{\frac{1}{4} \cdot (1 + T_{11} + T_{22} + T_{33})} \\q_x &= \sqrt{\frac{1}{4} \cdot (1 + T_{11} - T_{22} - T_{33})} \\q_y &= \sqrt{\frac{1}{4} \cdot (1 - T_{11} + T_{22} - T_{33})} \\q_z &= \sqrt{\frac{1}{4} \cdot (1 - T_{11} - T_{22} + T_{33})}\end{aligned}$$

After calculating those four values, we choose the maximum one and then recalculate all the others based on the following table:

maximum	q_s	q_x	q_y	q_z
q_s	q_s	$\frac{T_{32}-T_{23}}{4 \cdot q_s}$	$\frac{T_{13}-T_{31}}{4 \cdot q_s}$	$\frac{T_{21}-T_{12}}{4 \cdot q_s}$
q_x	$\frac{T_{32}-T_{23}}{4 \cdot q_x}$	q_x	$\frac{T_{21}+T_{12}}{4 \cdot q_x}$	$\frac{T_{13}+T_{31}}{4 \cdot q_x}$
q_y	$\frac{T_{13}-T_{31}}{4 \cdot q_y}$	$\frac{T_{21}+T_{12}}{4 \cdot q_y}$	q_y	$\frac{T_{32}+T_{23}}{4 \cdot q_y}$
q_z	$\frac{T_{21}-T_{12}}{4 \cdot q_z}$	$\frac{T_{13}+T_{31}}{4 \cdot q_z}$	$\frac{T_{32}+T_{23}}{4 \cdot q_z}$	q_z

1.2 Quaternion to DCM

In order to convert from a quaternion to a DCM, we can use the transpose of either of those two matrices:

Non-Homogeneous

$$R = \begin{bmatrix} 1 - 2(q_2^2 + q_3^2) & 2(q_1 q_2 - q_0 q_3) & 2(q_0 q_2 + q_1 q_3) \\ 2(q_1 q_2 + q_0 q_3) & 1 - 2(q_1^2 + q_3^2) & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_0 q_1 + q_2 q_3) & 1 - 2(q_1^2 + q_2^2) \end{bmatrix}$$

Homogeneous

$$R = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_0 q_2 + q_1 q_3) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_0 q_1 + q_2 q_3) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

1.3 Euler to DCM

There are 12 different possibilities for this matrix form depending on the sequence of the transformation. Using the matrices for the transformations shown below, we multiply them based on the required sequence.

$$R^x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & \sin \theta \\ 0 & -\sin \theta & \cos \theta \end{bmatrix}$$

$$R^y(\theta) = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix}$$

$$R^z(\theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1.4 DCM to Euler

This conversion has 12 different possibilities depending on the sequence of the transformation, so we had to accomodate to that using the following matrices:

Proper Euler angles	Tait-Bryan angles
$X_1 Z_2 X_3 = \begin{bmatrix} c_2 & -c_3 s_2 & s_2 s_3 \\ c_1 s_2 & c_1 c_2 c_3 - s_1 s_3 & -c_3 s_1 - c_1 c_2 s_3 \\ s_1 s_2 & c_1 s_3 + c_2 c_3 s_1 & c_1 c_3 - c_2 s_1 s_3 \end{bmatrix}$	$X_1 Z_2 Y_3 = \begin{bmatrix} c_2 c_3 & -s_2 & c_2 s_3 \\ s_1 s_3 + c_1 c_3 s_2 & c_1 c_2 & c_1 s_2 s_3 - c_3 s_1 \\ c_3 s_1 s_2 - c_1 s_3 & c_2 s_1 & c_1 c_3 + s_1 s_2 s_3 \end{bmatrix}$
$X_1 Y_2 X_3 = \begin{bmatrix} c_2 & s_2 s_3 & c_3 s_2 \\ s_1 s_2 & c_1 c_3 - c_2 s_1 s_3 & -c_1 s_3 - c_2 c_3 s_1 \\ -c_1 s_2 & c_3 s_1 + c_1 c_2 s_3 & c_1 c_2 c_3 - s_1 s_3 \end{bmatrix}$	$X_1 Y_2 Z_3 = \begin{bmatrix} c_2 c_3 & -c_2 s_3 & s_2 \\ c_1 s_3 + c_3 s_1 s_2 & c_1 c_3 - s_1 s_2 s_3 & -c_2 s_1 \\ s_1 s_3 - c_1 c_3 s_2 & c_3 s_1 + c_1 s_2 s_3 & c_1 c_2 \end{bmatrix}$
$Y_1 X_2 Y_3 = \begin{bmatrix} c_1 c_3 - c_2 s_1 s_3 & s_1 s_2 & c_1 s_3 + c_2 c_3 s_1 \\ s_2 s_3 & c_2 & -c_3 s_2 \\ -c_3 s_1 - c_1 c_2 s_3 & c_1 s_2 & c_1 c_2 c_3 - s_1 s_3 \end{bmatrix}$	$Y_1 X_2 Z_3 = \begin{bmatrix} c_1 c_3 + s_1 s_2 s_3 & c_3 s_1 s_2 - c_1 s_3 & c_2 s_1 \\ c_2 s_3 & c_2 c_3 & -s_2 \\ c_1 s_2 s_3 - c_3 s_1 & c_1 c_3 s_2 + s_1 s_3 & c_1 c_2 \end{bmatrix}$
$Y_1 Z_2 Y_3 = \begin{bmatrix} c_1 c_2 c_3 - s_1 s_3 & -c_1 s_2 & c_3 s_1 + c_1 c_2 s_3 \\ c_3 s_2 & c_2 & s_2 s_3 \\ -c_1 s_3 - c_2 c_3 s_1 & s_1 s_2 & c_1 c_3 - c_2 s_1 s_3 \end{bmatrix}$	$Y_1 Z_2 X_3 = \begin{bmatrix} c_1 c_2 & s_1 s_3 - c_1 c_3 s_2 & c_3 s_1 + c_1 s_2 s_3 \\ s_2 & c_2 c_3 & -c_2 s_3 \\ -c_2 s_1 & c_1 s_3 + c_3 s_1 s_2 & c_1 c_3 - s_1 s_2 s_3 \end{bmatrix}$
$Z_1 Y_2 Z_3 = \begin{bmatrix} c_1 c_2 c_3 - s_1 s_3 & -c_3 s_1 - c_1 c_2 s_3 & c_1 s_2 \\ c_1 s_3 + c_2 c_3 s_1 & c_1 c_3 - c_2 s_1 s_3 & s_1 s_2 \\ -c_3 s_2 & s_2 s_3 & c_2 \end{bmatrix}$	$Z_1 Y_2 X_3 = \begin{bmatrix} c_1 c_2 & c_1 s_2 s_3 - c_3 s_1 & s_1 s_3 + c_1 c_3 s_2 \\ c_2 s_1 & c_1 c_3 + s_1 s_2 s_3 & c_3 s_1 s_2 - c_1 s_3 \\ -s_2 & c_2 s_3 & c_2 c_3 \end{bmatrix}$
$Z_1 X_2 Z_3 = \begin{bmatrix} c_1 c_3 - c_2 s_1 s_3 & -c_1 s_3 - c_2 c_3 s_1 & s_1 s_2 \\ c_3 s_1 + c_1 c_2 s_3 & c_1 c_2 c_3 - s_1 s_3 & -c_1 s_2 \\ s_2 s_3 & c_3 s_2 & c_2 \end{bmatrix}$	$Z_1 X_2 Y_3 = \begin{bmatrix} c_1 c_3 - s_1 s_2 s_3 & -c_2 s_1 & c_1 s_3 + c_3 s_1 s_2 \\ c_3 s_1 + c_1 s_2 s_3 & c_1 c_2 & s_1 s_3 - c_1 c_3 s_2 \\ -c_2 s_3 & s_2 & c_2 c_3 \end{bmatrix}$

1.5 Euler to Quaternion

In order to convert from euler to quaternion, we have to first convert from euler to DCM then from DCM to quaternion.

1.6 Quaternion to Euler

In order to convert from quaternion to euler, we have to first convert from quaternion to DCM then from DCM to euler.

2 Attitude Determination Methods

Given two vectors, one in the inertial frame and another in the body frame, we can calculate the rotation matrix between the body and inertial frame using four

different methods explained here. So far, our input is only two measurements but it should be easy to add more if needed.

2.1 Biased Triad

We assume that the first measurement is more accurate so we use it directly as our first vector and we use it to calculate our second vector. The third vector is then calculated by crossing the two resulting vectors as shown in the steps below.

$$t_{1b} = V_{1b}$$

$$t_{2b} = \frac{V_{1b} \times V_{2b}}{|V_{1b} \times V_{2b}|}$$

$$t_{3b} = t_{1b} \times t_{2b}$$

$$t_{1i} = V_{1i}$$

$$t_{2i} = \frac{V_{1i} \times V_{2i}}{|V_{1i} \times V_{2i}|}$$

$$t_{3i} = t_{1i} \times t_{2i}$$

$$R^{bi} = R^{bt} R^{ti} = [t_{1b} \quad t_{2b} \quad t_{3b}][t_{1i} \quad t_{2i} \quad t_{3i}]^T$$

2.2 Balanced Triad

We assume that both measurements are equally accurate so we use them both directly as follows.

$$t_{1b} = V_{1b}$$

$$t_{2b} = V_{2b}$$

$$t_{3b} = t_{1b} \times t_{2b}$$

$$t_{1i} = V_{1i}$$

$$t_{2i} = V_{2i}$$

$$t_{3i} = t_{1i} \times t_{2i}$$

$$R^{bi} = R^{bt} R^{ti} = [t_{1b} \quad t_{2b} \quad t_{3b}] [t_{1i} \quad t_{2i} \quad t_{3i}]^T$$

2.3 Q-method

In order to use this method, we need to calculate the K matrix as follows:

$$\mathbf{K} = \begin{bmatrix} \mathbf{S} - \sigma \mathbf{I} & \mathbf{Z} \\ \mathbf{Z}^T & \sigma \end{bmatrix}$$

with

$$\begin{aligned} \mathbf{B} &= \sum_{k=1}^N w_k (\mathbf{v}_{kb} \mathbf{v}_{ki}^T) \\ \mathbf{S} &= \mathbf{B} + \mathbf{B}^T \\ \mathbf{Z} &= \begin{bmatrix} B_{23} - B_{32} & B_{31} - B_{13} & B_{12} - B_{21} \end{bmatrix}^T \\ \sigma &= \text{tr}[\mathbf{B}] \end{aligned}$$

Where we assume that the vectors are equally weighted.

And then we calculate the eigenvalues and vectors of the matrix K. We then take the maximum eigenvector (corresponding to the maximum eigenvalue) to be our quaternion. Then we convert this quaternion into a DCM to get our transformation matrix.

2.4 QUEST method

We calculate the matrix K elements as in the q-method, but the eigenvalue problem is solved differently. We need to solve for p as shown below:

$$\mathbf{p} = [(\lambda_{opt} + \sigma)\mathbf{1} - \mathbf{S}]^{-1} \mathbf{Z} \quad \text{where} \quad \lambda_{opt} \approx \sum w_k$$

Where our quaternion is:

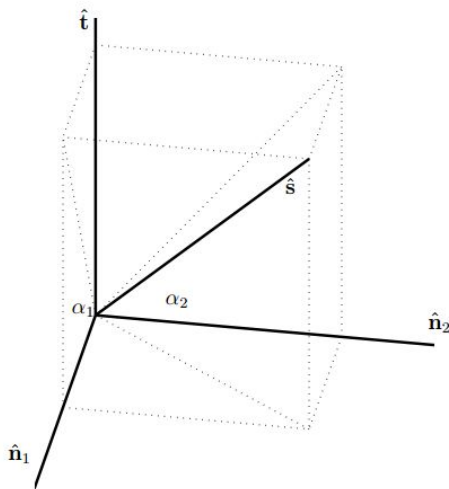
$$\bar{\mathbf{q}} = \frac{1}{\sqrt{1 + \mathbf{p}^T \mathbf{p}}} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$

We then convert this quaternion into a DCM to get our transformation matrix.

3 Sensor Measurements

3.1 Sun Sensor (2D)

For a four-photocell sun sensor configured as the following figure, given α_1 , α_2 , and the sun sensor frame with respect to the body frame as quaternion, we can calculate the transformation matrix between body and sun frames using the equations below.



$$\mathbf{s}_s^* = [1 \quad \tan \alpha_1 / \tan \alpha_2 \quad \tan \alpha_1]^T$$

$$\mathbf{s}_s = \frac{[1 \quad \tan \alpha_1 / \tan \alpha_2 \quad \tan \alpha_1]^T}{\sqrt{\mathbf{s}_s^{*T} \mathbf{s}_s^*}}$$

$$\mathbf{R} = (q_4^2 - \mathbf{q}^T \mathbf{q}) \mathbf{1} + 2\mathbf{q}\mathbf{q}^T - 2q_4\mathbf{q}^\times$$

Where \mathbf{q}^\times is the skew matrix of the vector part of the quaternion.

Then we can calculate S in the body frame using: $S_b = R_{bs} S_s$

From Time to Julian Date

Using the following equation, we can convert from time in year-month-hour-minute-second to julian date.

$$JD = 367year - INT \left\{ \frac{7 \left[year + INT \left(\frac{month+9}{12} \right) \right]}{4} \right\} + INT \left(\frac{275month}{9} \right) + day \\ + 1,721,013.5 + \frac{hour}{24} + \frac{minute}{1440} + \frac{s}{86,400}$$

From Julian Date to sun in the inertial frame

Using those set of equations, we can convert from julian date to S_i

$$\begin{aligned}
T_{UT1} &= \frac{JD_{UT1} - 2,451,545.0}{36,525} \\
\lambda_{M_{Sun}} &= 280.4606184^\circ + 36,000.77005361T_{UT1} \\
&\quad \text{mean longitude of Sun} \\
\text{Let } T_{TDB} &\approx T_{UT1} \\
M_{Sun} &= 357.5277233^\circ + 35,999.05034T_{TDB} \\
&\quad \text{mean anomaly of Sun} \\
\lambda_{ecliptic} &= \lambda_{M_{Sun}} + 1.914666471^\circ \sin M_{Sun} + 0.918994643 \sin 2M_{Sun} \\
&\quad \text{ecliptic longitude of Sun} \\
s^* &= 1.000140612 - 0.016708617 \cos M_{Sun} - 0.000139589 \cos 2M_{Sun} \\
&\quad \text{distance to Sun in AUs} \\
\varepsilon &= 23.439291^\circ - 0.0130042T_{TDB} \\
\mathbf{s}_i &= \begin{bmatrix} \cos \lambda_{ecliptic} \\ \cos \varepsilon \sin \lambda_{ecliptic} \\ \sin \varepsilon \sin \lambda_{ecliptic} \end{bmatrix}
\end{aligned}$$

4 Equations of Motion Visualization

4.1 Non-Linear Equations of Motion

This was based on the equations given in class and implemented in simulink.

4.2 Linear Equations of Motion

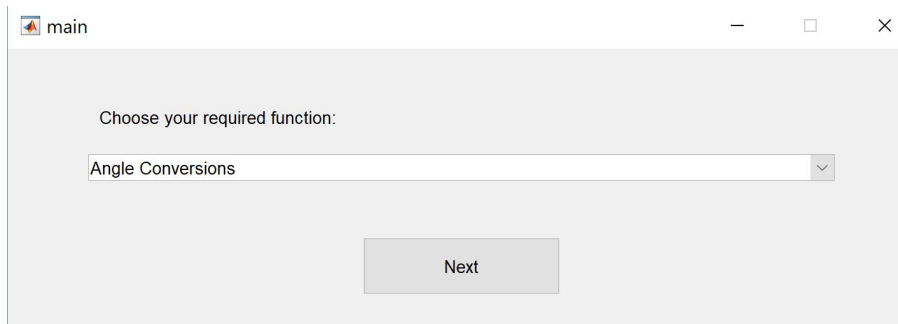
The simulink was implemented based on those equations:

$$\begin{aligned}
T_{dx} &= I_x \ddot{\phi} + 4\omega_o^2(I_y - I_z)\phi - \omega_o(I_x + I_z - I_y)\dot{\psi}, \\
T_{dz} &= I_z \ddot{\psi} + \omega_o^2(I_y - I_x)\psi + \omega_o(I_z + I_x - I_y)\dot{\phi}, \\
T_{dy} &= I_y \ddot{\theta} + 3\omega_o^2(I_x - I_z)\theta.
\end{aligned}$$

The Toolbox

1. Main

We start with a simple GUI where the user chooses the task and clicks next.



2. Angle Conversions

2.1 Input and Output

Here there are three input possibilities and three output possibilities: quaternions, euler angles, or direction cosine matrix. Using radio buttons, the user will choose one input and one output. For quaternions, they should input the scalar and vector components, in order of x y z components. For euler angles, they should input three angles and specify the axis for these angles from the 12 possible rotations defined before. For the direction cosine matrix, they should enter the 9 elements of the matrix. If the user choose euler angles as the output, they should specify the sequence for the output as well. The output shows up in the boxes to the left.

3. Attitude Determination

3.1 Inputs and Outputs

Here, the user inputs four vectors, two in the inertial frame and two in body frame. Then, they choose a calculation method and the final transformation matrix shows up in the boxes to the lower right.

The screenshot shows a software window titled "attdet" with standard window controls (minimize, maximize, close). The main window is titled "Attitude Determination" and contains the following elements:

- Instruction:** "Enter your vectors, select your preferred method of calculation, and click calculate."
- Input Vectors:** A section on the left containing two sub-sections:
 - Inertial Frame:** A 2x2 grid of input boxes for vectors.
 - Body Frame:** A 2x2 grid of input boxes for vectors.
- Calculation Method:** A dropdown menu on the right with the following options:
 - Biased Triad (highlighted)
 - Balanced Triad
 - Quest Method
 - Q-method
- Output Rbi:** A 3x3 grid of boxes on the right for the resulting transformation matrix.
- Calculate:** A button at the bottom center of the window.

3.2 Solved Example

attdet

Attitude Determination

Enter your vectors, select your preferred method of calculation, and click calculate.

Input Vectors

Inertial Frame

0.2673	-0.3124
0.5345	0.9370
0.8018	0.1562

Body Frame

0.7814	0.6163
0.3751	0.7075
0.4987	-0.3459

Calculation Method

Biased Triad
Balanced Triad
Quest Method
Q-method

Output Rbi

0.5569	0.7897	0.2574
-0.7950	0.4172	0.4402
0.2402	-0.4499	0.8602

Calculate

attdet

Attitude Determination

Enter your vectors, select your preferred method of calculation, and click calculate.

Input Vectors

Inertial Frame

0.2673	-0.3124
0.5345	0.9370
0.8018	0.1562

Body Frame

0.7814	0.6163
0.3751	0.7075
0.4987	-0.3459

Calculation Method

Biased Triad
Balanced Triad
Quest Method
Q-method

Output Rbi

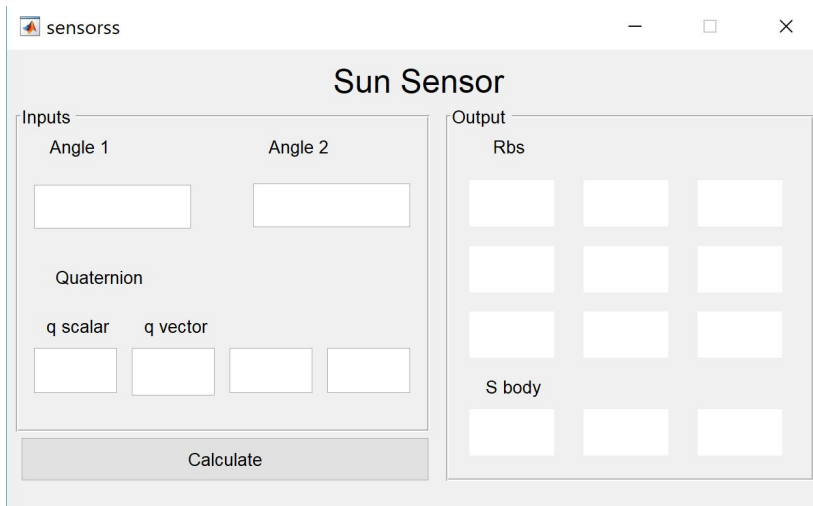
0.5571	0.7895	0.2574
-0.7950	0.4174	0.4401
0.2400	-0.4498	0.8603

Calculate

4. Sun Sensor

4.1 Inputs and Outputs

The used has to inputs two alpha angles, and a quaternion. The function outputs the transformation matrix and the calculated position of the sun in the body frame.



Sun Sensor

Inputs

Angle 1

Angle 2

Quaternion

q scalar

q vector

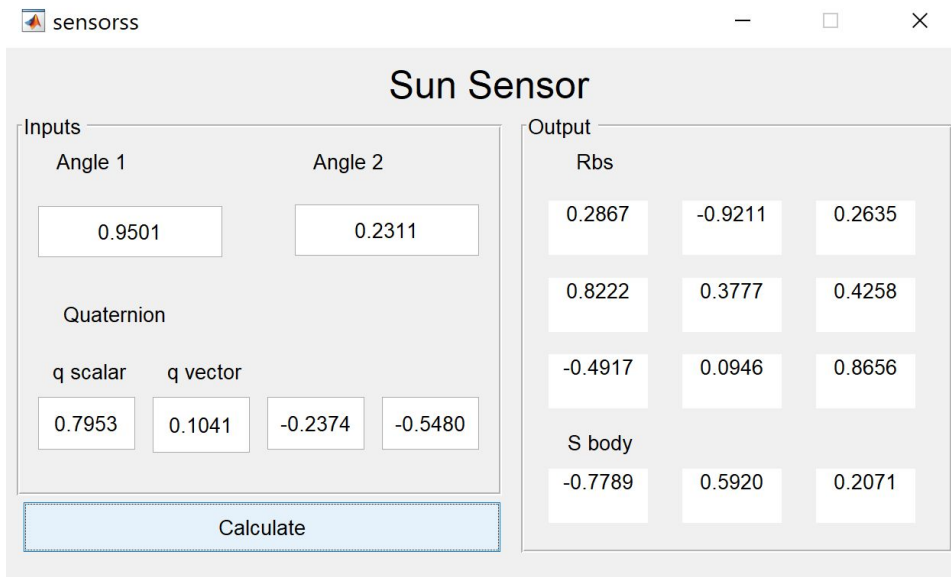
Calculate

Output

Rbs

S body

4.2 Solved Example



Sun Sensor

Inputs

Angle 1

Angle 2

Quaternion

q scalar

q vector

Calculate

Output

Rbs

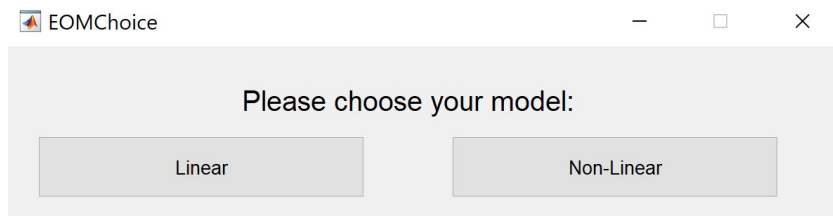
S body

0.2867	-0.9211	0.2635
0.8222	0.3777	0.4258
-0.4917	0.0946	0.8656

-0.7789	0.5920	0.2071
---------	--------	--------

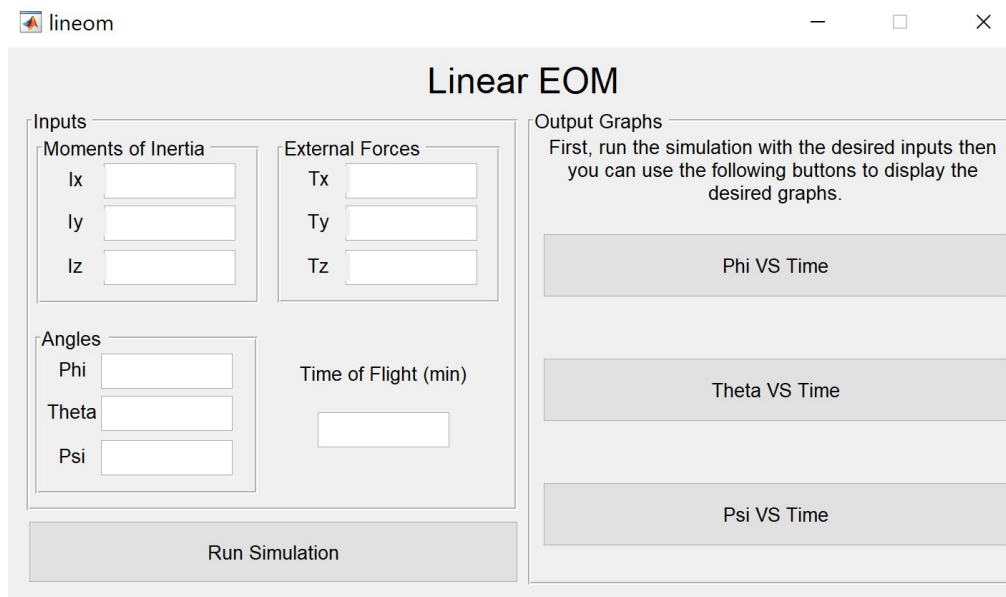
5. Motion Simulation

First, the user has to choose their model:



5.1 Linear


The user inputs the moments of inertia, any external forces, initial conditions for the three angles, and the time period of the orbit. The Run Simulation button will display the 6-DOF animation for the satellite. After the simulation finishes, the user can click any of the buttons to the left to show the graphs as stated.



5.2 Non-Linear

The user inputs the moments of inertia, control parameters, and initial conditions for both the angles and the omegas. The Run Simulation button will display the

6-DOF animation for the satellite. After the simulation finishes, the user can click any of the buttons to the left to show the graphs as stated.


nonlineom

Non-Linear EOM

Inputs

Moments of Inertia

lx

ly

lz

Control Parameters

hwx

hwy

hwz

Initial Conditions

Omega

w1

w2

w3

Angles

Phi

Theta

Psi

Output Graphs

First, run the simulation with the desired inputs then you can use the following buttons to display the desired graphs.

Omega

w1 VS Time

w2 VS Time

w3 VS Time

Angles

Phi VS Time

Theta VS Time

Psi VS Time

Run