AERO 4540
Spacecraft Attitude Dynamics and Control

Assignment

Steve Ulrich
Carleton University
Ottawa, ON, Canada

PROBA-2 is a follow-on European Space Agency's (ESA) microsatellite technology demonstration mission within the PROBA (PRoject for On-Board Autonomy) program. The PROBA-2 goals are dedicated to space weather, specifically, sun observations and monitoring, and innovative platform subsystems with new advanced technologies such as propulsion systems with cold gas generator, Li-ion batteries, autonomous star tracker, reaction wheels, as well as solar array with concentrator. The overall PROBA-2 mission includes major industrial contributions from Verhaert Space, a Belgian company that led the development of the spacecraft platform, scientific payloads and ground segment, and NGC Aerospace Ltd, a Canadian company who was in charge of innovative Guidance, Navigation, and Control (GNC) algorithms and associated software development. Prof. Ulrich's contribution to the GNC system is a simple, however robust, navigation algorithm to autonomously evaluate and compensate for the attitude torque perturbations due to the interaction of the spacecraft residual magnetic dipole with the orbital magnetic environment. This module, originally carried as a technology demonstration, has been, after an extensive successful evaluation campaign, integrated in the onboard autonomous navigation loop of PROBA-2, to even further improve the spacecraft absolute pointing accuracy performances, for the optimization of the quality of the scientific data.

The GNC system relies primarily on the autonomous CCD star tracker referred to as µASC (micro Advanced Stellar Compass) for attitude sensing. Due to the high accuracy of the µASC instrument (1 arcsec), no rate gyroscope are installed within the spacecraft.

Reaction wheels manufactured in Canada are the main actuators used for attitude control while magnetorquers are used for detumbling purposes. The wheels are mounted along the the axis of the body-fixed reference frame, that is, $\mathbf{a}_{1_B}= [1{,}0{,}0]^{\mathrm{T}}$, $\mathbf{a}_{2_B}= [0{,}1{,}0]^{\mathrm{T}}$, and $\mathbf{a}_{3_B}= [0{,}0{,}1]^{\mathrm{T}}$. The moment of inertia of each wheel about their respective spin axis are: $J_{a_1} = 4.10e-04 \text{ kg}\cdot\text{m}^2$, $J_{a_2} = 4.11e-04 \text{ kg}\cdot\text{m}^2$, $J_{a_3} = 4.12e-04 \text{ kg}\cdot\text{m}^2$. Reaction wheels operate on the basis of momentum exchange within the spacecraft body. As such, Euler's equations of motion are modified as follows

$$\dot{\mathbf{h}}_{tot_B} + \boldsymbol{\omega}_B^{\times}\mathbf{h}_{tot_B} = \boldsymbol{\tau}_B \tag{1}$$

$$\mathbf{h}_{tot_B} = J\boldsymbol{\omega}_B + \sum_i h_{a_i}\mathbf{a}_{i_B} \tag{2}$$

and the dynamics equations for the reaction wheels are

$$\dot{h}_{a_i} = \tau_{a_i} \tag{3}$$

$$h_{a_i} = J_{a_i}\left[\omega_{rel_i} + \mathbf{a}_{i_B}^T\boldsymbol{\omega}_B\right] \tag{4}$$

Equations (1) and (3) can be numerically integrated to obtain the integrated variables $\mathbf{h}_{tot_B}$ and h_{a_i}. Then, Eq. (2) is solved for $\boldsymbol{\omega}_B$ whereas Eq. (4) is solved for ω_{rel_i}.

The 120-kg PROBA-2 spacecraft has dimensions of 60 cm × 70 cm × 85 cm and has an inertia matrix of

$$\mathbf{J} = \begin{bmatrix} 120 & 10 & 50 \\ 10 & 150 & -25 \\ 50 & -25 & 100 \end{bmatrix} \text{kg}\cdot\text{m}^2$$

and the initial attitude states are defined as

$$\boldsymbol{\omega}_{B_0} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}} \text{rad/s} \qquad \mathbf{q}_0 = \begin{bmatrix} 0.57 & 0.57 & 0.57 & 0.159 \end{bmatrix}^{\mathrm{T}}$$

Panel	**Area**, m^2	**Normal unit vector**	**Vector orientation**	**Center of pressure**, m
Panel 1	0.595	n_1	$\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$	$\begin{bmatrix} 0.3 & 0 & -0.2 \end{bmatrix}^{\mathrm{T}}$
Panel 2	0.510	n_2	$\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^{\mathrm{T}}$	$\begin{bmatrix} 0 & 0.35 & -0.2 \end{bmatrix}^{\mathrm{T}}$
Panel 3	0.420	n_3	$\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^{\mathrm{T}}$	$\begin{bmatrix} 0 & 0 & 0.425 \end{bmatrix}^{\mathrm{T}}$
Panel 4	0.595	n_4	$\begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$	$\begin{bmatrix} -0.3 & 0 & -0.2 \end{bmatrix}^{\mathrm{T}}$
Panel 5	0.510	n_5	$\begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^{\mathrm{T}}$	$\begin{bmatrix} 0 & -0.35 & -0.2 \end{bmatrix}^{\mathrm{T}}$
Panel 6	0.420	n_6	$\begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^{\mathrm{T}}$	$\begin{bmatrix} 0 & 0 & -0.425 \end{bmatrix}^{\mathrm{T}}$

Table 1: Spacecraft panel data

and

$$\omega_{rel_1 0} = \omega_{rel_2 0} = \omega_{rel_3 0} = 0 \text{ rad/s}$$

In terms of the spacecraft structure, the panel normal unit vectors are labeled, oriented with respect to the body-fixed reference frame, and have areas as defined in Table 1.
Furthermore, based on their specific coating and material, the drag coefficient and absorption coefficient are estimated to be 2.2 and 0.3, respectively.

The spacecraft was launched on Nov. 2, 2009 as a secondary payload onboard a Rockot launch vehicle from Plesetsk, Russia. The nominal mission duration was two years. As of 2024, the mission continues. The first burn placed the launch vehicle on an elliptical transfer orbit and a second burn circularized the trajectory into its nominal sun-synchronous low-Earth orbit, which, as of recently, is defined by the following orbital elements:

$$a = R_{\oplus} + 713.5 \text{ km} \quad e = 0.0014581 \quad i = 98.2281^o \quad \Omega = 105.3095^o \quad \omega = 72.9288^o \quad t_p = 0 \text{ sec}$$

For such Sun-synchronous orbits, the magnetic perturbation torque created by the relatively high residual magnetic moment of the spacecraft represents the greatest attitude perturbation encountered. Indeed, for PROBA-2, the residual magnetic moment is defined as a 0.5 $\mathrm{A} \cdot \mathrm{m}^2$ torquer located in the center of the spacecraft (considered as a dipole) oriented along the $\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^{\mathrm{T}} / \sqrt{3}$ direction in the body fixed-reference frame.

Question 1. (10 points) Reaction wheels are actuators often used to control the attitude dynamics of a spacecraft through momentum exchange. More specifically, as described by Newton's third law of motion, applying a torque (electrical) to accelerate the wheel in one direction about its fixed axis within the body-fixed frame, causes the spacecraft to experience a torque of the same magnitude but in the opposite direction. This phenomenon can also be explained by the principle of conservation of angular momentum since torques applied on reaction wheels and the spacecraft are internal torques.

Mathematically speaking, since reaction wheels are therefore not contributing to the external torque term in Euler's equation of motion, the only way they can modify the angular acceleration term is by modifying the total angular momentum. Following the same development as in Sec. 4.3.1 of the lecture notes, obtain the attitude dynamics equation of a spacecraft equipped with three reaction wheels. Note that, unlike the dual-spin stabilization analysis in Sec. 4.3.1, there are now three wheels instead of one, and the angular velocity of the wheels are not maintained constant. Specifically, denoting the total angular momentum (spacecraft platform and wheels) in $\mathcal{F}_B$ as $\mathbf{h}_{tot_B}$, the angular momentum of the i-th wheel about its spin axis as h_{a_i}, and the i-th wheel spin axis in $\mathcal{F}_B$ as $\mathbf{a}_{i_B}$, obtain the following expressions:

- $\dot{\mathbf{h}}_{tot_B} = f(\boldsymbol{\omega}_B, \mathbf{h}_{tot_B}, \boldsymbol{\tau}_B)$
- $\mathbf{h}_{tot_B} = f(\boldsymbol{\omega}_B, h_{a_i}, \mathbf{a}_{i_B})$

Utkarsh Sheel Anand, 101191746

AERO 4540

Spacecraft Dynamics and Control Assignment

1.

$\underline{h}_{tot_B} \rightarrow$ total ang. momentum (platform & wheel)

$h_{ai} \rightarrow$ ang. momentum of i-th wheel about its spin axis

$$\vec{h}_{tot_B} = \vec{\mathcal{F}}_B^T \underline{h}_{total_B}$$

$$\vec{h}_{tot_B} = \vec{\mathcal{F}}_B^T \left[\underline{J}\,\underline{\omega}_B + \sum_{i=1}^{3} h_{ai}\,\underline{a}_i\right] \quad \text{—(i)}$$

Using $\left(\vec{\tau} = \dot{\vec{h}} = \overset{\circ}{\vec{h}} + \vec{\omega} \times \vec{h}\right)$

$$\overset{\circ}{\vec{h}} = \vec{\mathcal{F}}_B^T \dot{h}_B = \vec{\mathcal{F}}_B^T \frac{d}{dt}\left[\underline{J}\,\underline{\omega}_B + \sum_{i=1}^{3} h_{ai}\,\underline{a}_i\right]$$

$$\overset{\circ}{\vec{h}} = \vec{\mathcal{F}}_B^T \left(\underline{J}\,\dot{\underline{\omega}}_B + \cancel{\dot{\underline{J}}\,\underline{\omega}_B}^{\,0} + \cancel{\Sigma\,\dot{h}_{ai}\,\underline{a}_i}^{\,0} + \cancel{\Sigma\,h_{ai}\,\dot{\underline{a}}_i}^{\,0}\right]$$

$$\overset{\circ}{\vec{h}} = \vec{\mathcal{F}}_B^T \underline{J}\,\dot{\underline{\omega}}_B$$

$$\therefore\ \underline{\tau}_B = \underline{J}\,\dot{\underline{\omega}}_B + \underline{\omega}_B^{\times}\,\underline{h}_{total_B}$$

$$\tau_B = \dot{\underline{h}}_{total_B} + \omega_B^{\times}\,h_{tot_B}$$

$$\rightarrow\ \dot{h}_{tot_B}(\omega_B, h_{tot_B}, \tau_B) = \tau_B - \omega_B^{\times}\,h_{tot_B}$$

from eqⁿ (i)

$$\rightarrow\ \underline{h}_{tot_B}(\underline{\omega}_B, h_{ai}, \underline{a}_i) = \underline{J}\,\underline{\omega}_B + \Sigma\,h_{ai}\,a_i$$

Question 2. (5 points) Convert the new dynamical equations of motion obtained in the previous question into a block scheme diagram with inputs being $\boldsymbol{\tau}_B$ and $\mathbf{h}_{rw_B}$ and output $\boldsymbol{\omega}_B$, where $\mathbf{h}_{rw_B}$ corresponds to the net angular momentum of all wheels in $\mathcal{F}_B$.

2.

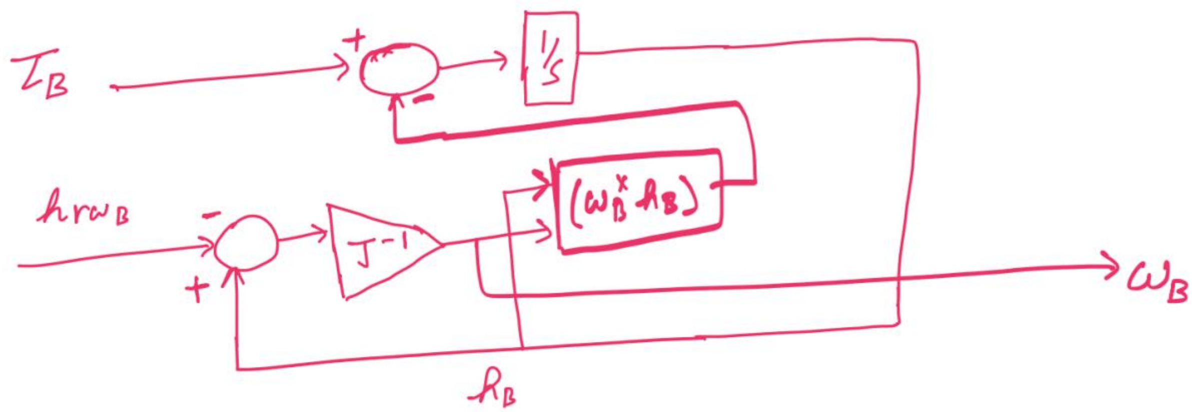


Fig.1. Dynamic equation of motion in block scheme diagram

3.

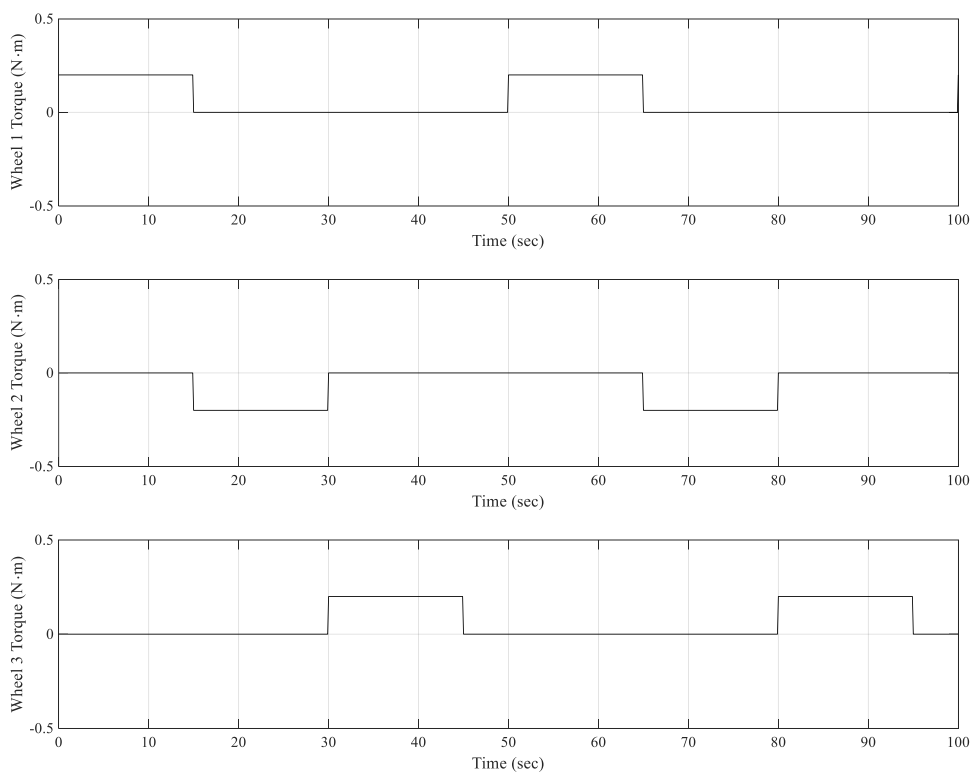


Fig.2. Wheel's Torque Input

Question 3. (20 points) Given the numerical data provided, define all initial conditions and dynamical parameters in the initialization file `ini_SPACECRAFT_ATTITUDE.m`. Those parameters are required for the nonlinear attitude dynamics and kinematics models implementation in MATLAB/Simulink. This has been partially done in the Simulink file `SPACECRAFT_ATTITUDE_incomplete.slx` available on Brightspace. However, the red blocks in the model are incomplete. Specifically,

- Eqs. (3) and (4) must be implemented in each `ACT_RW` block;
- the `ACT_RW` blocks must be interfaced with the outputs of the `ACT` block;
- Eqs. (1) and (2) must be implemented in the `DYN` block and interfaced with the `Attitude Kinematics` block;
- the quaternion differential kinematics equations must be implemented in the `Quaternion Kinematics` user-defined function within the `Attitude Kinematics` block; and
- the conversion of the actual quaternion to the attitude matrix must be implemented in the `QUA2CBI` user-defined function within the `Star Tracker` block.

Make sure you understand the entire Simulink diagram. Once all equations in the red blocks are implemented, do the following:

- Rename the Simulink file to `SPACECRAFT_ATTITUDE.slx`.
- Simulate the model by running the file `ini_SPACECRAFT_ATTITUDE.m`. This file initializes all parameters and calls `run_SPACECRAFT_ATTITUDE.m`, which runs the simulation from 0 to 100 seconds and plots the results.
- Provide your graphs for the wheels' torque inputs and spacecraft quaternions and angular rates.

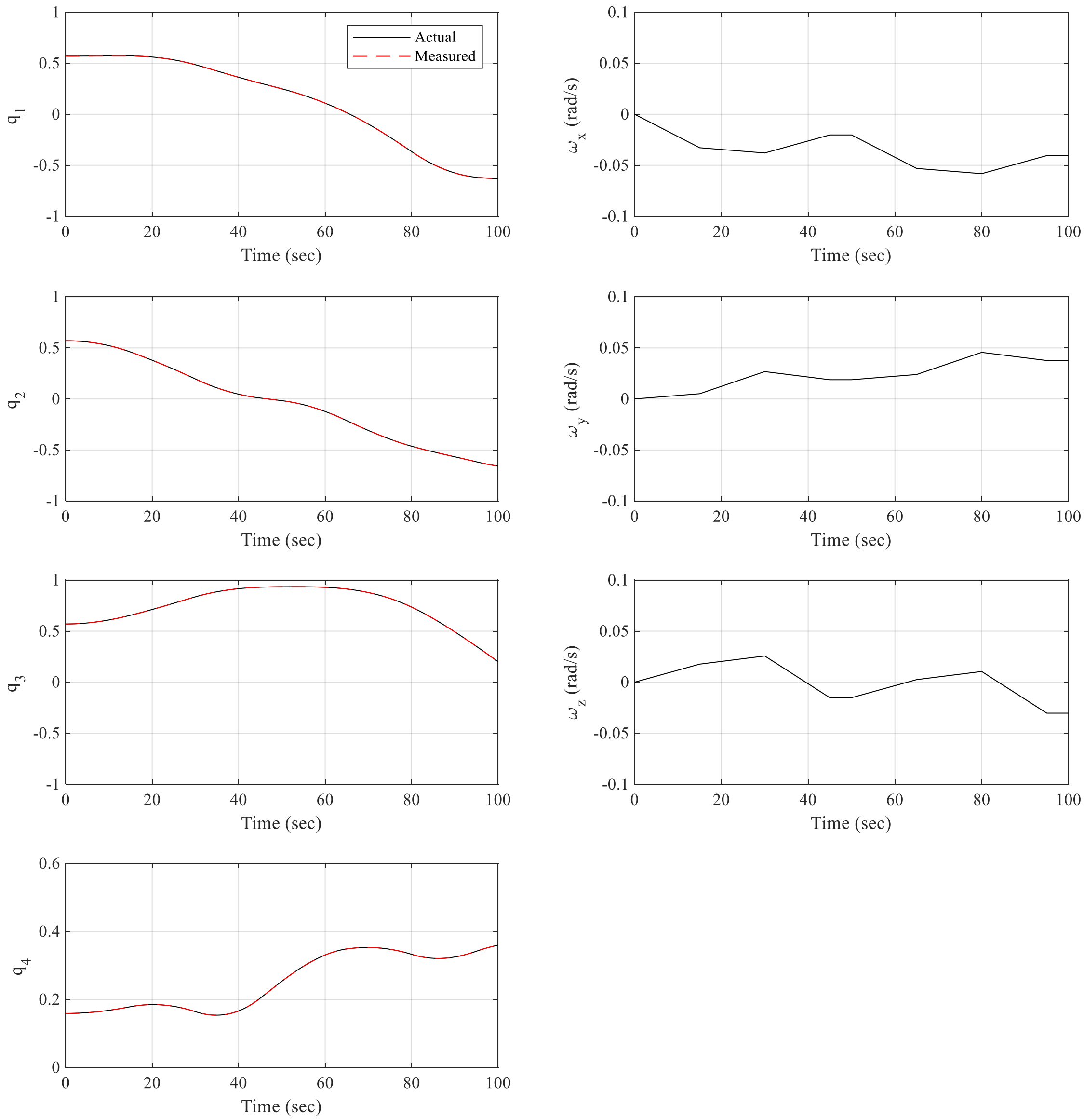


Fig.3. Spacecraft Quaternions and Angular Rates V/s Time

Question 4. (30 points) In this question, closed-loop simulations will be performed to evaluate the performance of the nonlinear quaternion feedback control law described in the lectures with $\mathbf{q}_d = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$. To do so,

- Using the four selection methodologies presented in class, calculate and provide the control gain $\mathbf{K}_p$ for a desired reorientation time or settling time of 50 s and a natural frequency $\omega_n = 0.156$ rad/s. Normalize the proportional gain matrices so that $K_2 = 11$. Show your calculations.
- Calculate and provide the rate feedback gain matrix $\mathbf{K}_d$.
- Define all control gains, and the desired quaternion within the MATLAB file `ini_SPACECRAFT_ATTITUDE.m`.
- Create a new subsystem in the Simulink diagram `SPACECRAFT_ATTITUDE.slx` that takes as input the error quaternion and the calculated angular rates and has a single output corresponding to the control input torque. This is your control block, to be named `CTL`.
- Within this `CTL` block, implement the nonlinear quaternion feedback control law.
- Create another subsystem in the Simulink diagram `SPACECRAFT_ATTITUDE.slx` that takes as input the measured quaternion from the star tracker and that has the angular rates in body-fixed frame (calculated from the measured quaternion) and the error quaternion as the outputs. This is your guidance block, to be named `GDC`.
- Within this `GDC` block, implement the error quaternion calculation and angular rates calculation based on quaternions.
- Interface the `CTL` block with the `GDC` and the `ACT` blocks.
- Disconnect the open-loop torque wave generators within the `ACT` block, and interface the `ACT_RW` blocks with the inputs of the `ACT` block
- Interface the `GDC` block with the `SEN` block.
- Provide the graphs of the spacecraft quaternions as function of time, and the q_i vs. q_j plots (only provide q1 vs q2, q1 vs q3 and q2 vs q3).

4.

(i) $(K_p)_{scaled} = K I_3 = 11 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$= \begin{bmatrix} 11 & 0 & 0 \\ 0 & 11 & 0 \\ 0 & 0 & 11 \end{bmatrix}$$

(ii) $\underline{(K_p)_{inverse-inertia}}$

$$K_{pi} = K/J_i$$

as $K_2 = 11 \longrightarrow K_{p_2} = \dfrac{K}{J_2} = 11$

$$K = 11 \cdot J_2$$

$$K_{p_1} = \frac{K}{J_1} = \frac{11\,J_2}{J_1} = \frac{11 \times 150}{120} = 13.75$$

$$K_{p_3} = \frac{K}{J_3} = \frac{11\,J_2}{J_3} = \frac{11 \times 150}{100} = 16.5$$

$$(K_p)_{\substack{inv. \\ inertia}} = \begin{bmatrix} 13.75 & 0 & 0 \\ 0 & 11 & 0 \\ 0 & 0 & 16.5 \end{bmatrix}$$

(iii) $\underline{(K_p)_{eigen-axis}}$

$$\alpha = \frac{\left[9 - \left(\frac{1}{J_1} + \frac{1}{J_2} + \frac{1}{J_3}\right)(J_1 + J_2 + J_3)\right]}{\left(3(J_1^2 + J_2^2 + J_3^2) - (J_1 + J_2 + J_3)^2\right]}$$

$$\alpha = -65.78 \times 10^{-6}$$

$$\beta = \frac{\left[\left(\frac{1}{J_1} + \frac{1}{J_2} + \frac{1}{J_3}\right)\left(J_1^2 + J_2^2 + J_3^2\right) - 3\left(J_1 + J_2 + J_3\right)\right]}{\left[3\left(J_1^2 + J_2^2 + J_3^2\right) - \left(J_1^2 + J_2^2 + J_3^2\right)\right]}$$

$$\beta = 16.47 \times 10^{-3}$$

$$(K_p)_{eign} = (\alpha J^* + \beta I_3)^{-1} = \left(\alpha \begin{bmatrix} J_1 & 0 & 0 \\ 0 & J_2 & 0 \\ 0 & 0 & J_3 \end{bmatrix} + \beta \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right)^{-1}$$

$$(K_p)_{eigen-axis} = \begin{bmatrix} 116.6 & 0 & 0 \\ 0 & 151.4 & 0 \\ 0 & 0 & 101.09 \end{bmatrix}$$

Normalizing for $k_2 = 11$

$$\Rightarrow K_p \rightarrow \left(K_p \times \frac{11}{151.4}\right)$$

$$(K_p)_{eign-axis} = \begin{bmatrix} 8.47 & 0 & 0 \\ 0 & 11 & 0 \\ 0 & 0 & 7.34 \end{bmatrix}$$

(iv) 4th Method

$$t_s = 50 \text{ sec}, \quad \omega_n = 0.156 \text{ rad/s}$$

$$k = 2\omega_n^2 = 0.0487$$

$$K_p = kJ^* = 0.0487 \begin{bmatrix} 120 & 0 & 0 \\ 0 & 150 & 0 \\ 0 & 0 & 100 \end{bmatrix}$$

$$K_p = \begin{bmatrix} 5.844 & 0 & 0 \\ 0 & 7.305 & 0 \\ 0 & 0 & 4.87 \end{bmatrix}$$

Normalizing for $k_2 = 11$

$$\Rightarrow K_p \rightarrow K_p \times \frac{11}{7.305}$$

$$(K_p)_{4\text{-}th} = \begin{bmatrix} 8.8 & 0 & 0 \\ 0 & 11 & 0 \\ 0 & 0 & 7.33 \end{bmatrix}$$

$$t_s = \frac{8}{\zeta \omega_n} \rightarrow \zeta = \frac{8}{0.156 \times 50}$$

$$\zeta = 1.0256$$

$$d = 2\zeta\omega_n = 0.319$$

$$K_d = dJ^* = \begin{bmatrix} 38.4 & 0 & 0 \\ 0 & 48 & 0 \\ 0 & 0 & 32 \end{bmatrix}$$

<u>Scaled Design</u>

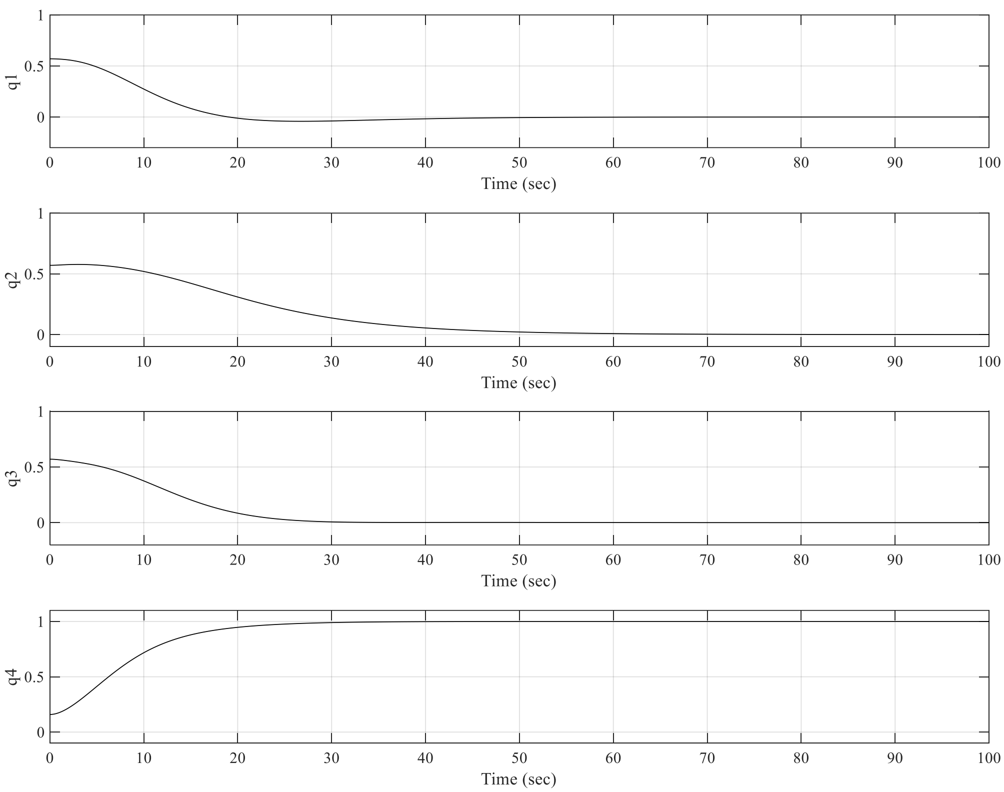


Fig.4. Quaternions V/s Time for Scaled Design

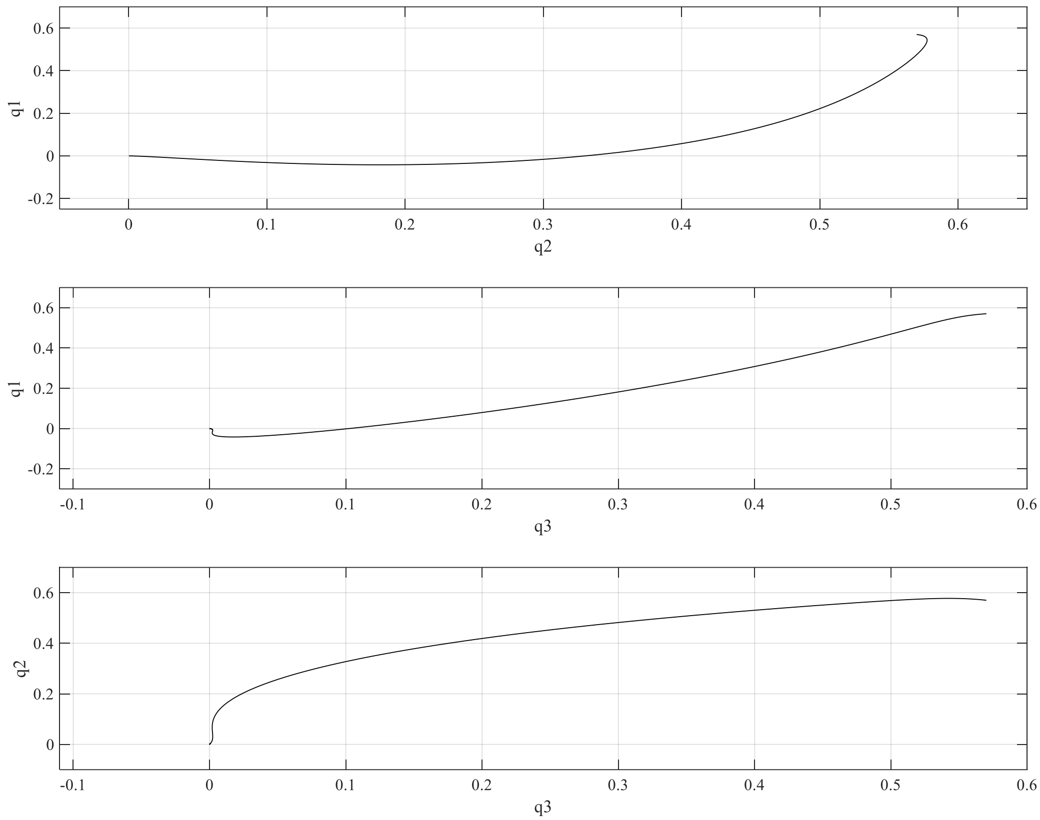


Fig.5. q_i V/s q_j plots for Scaled Design

<u>Inverse Inertia Design</u>

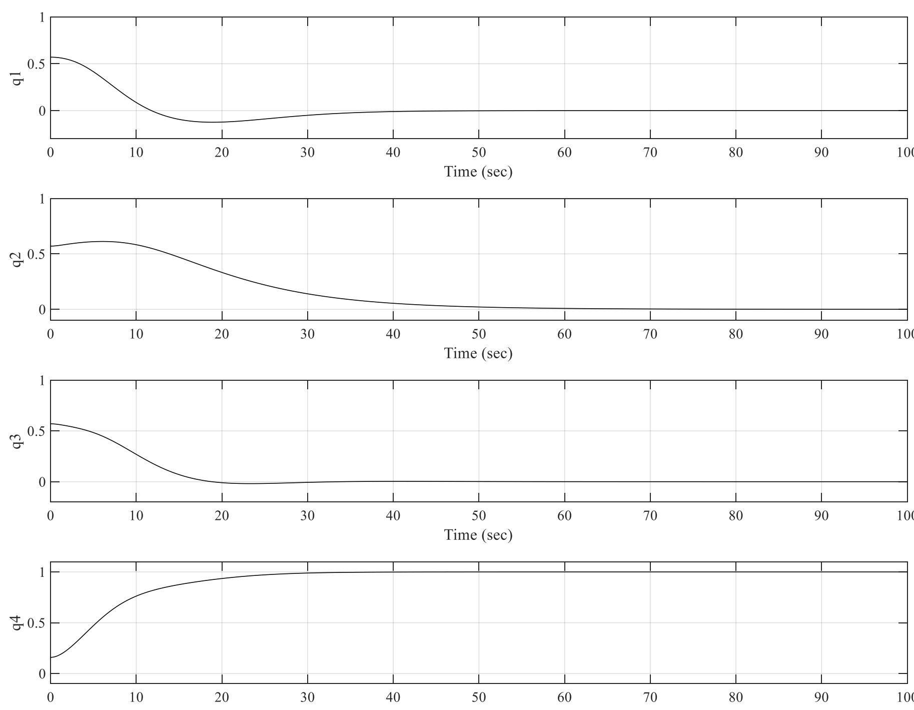


Fig.6. Quaternions V/s Time for Inverse Inertia Design

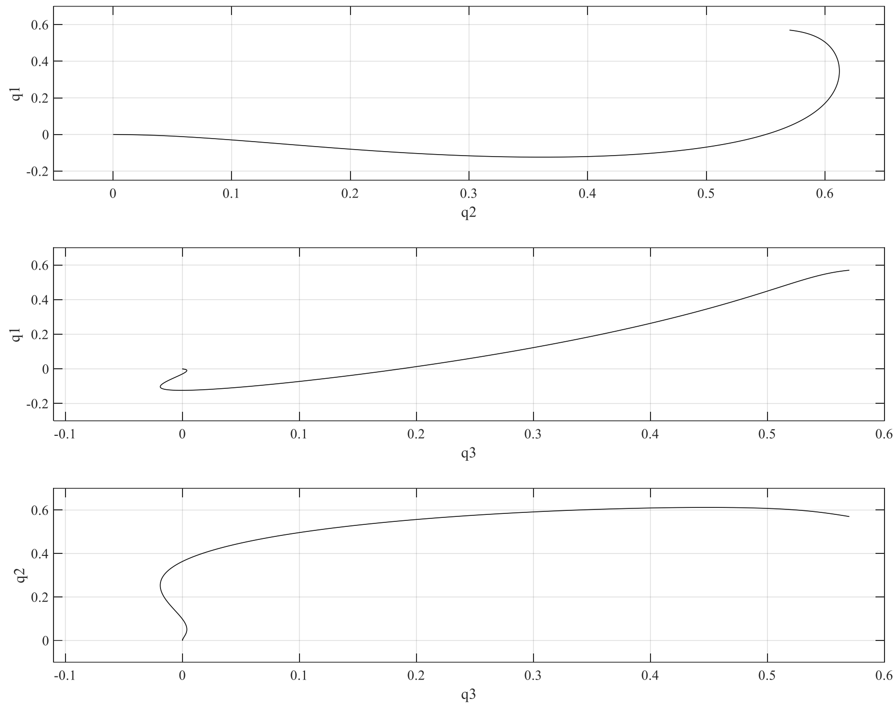


Fig.7. q_i V/s q_j plots for Inverse Inertia Design

Eigenaxis Design

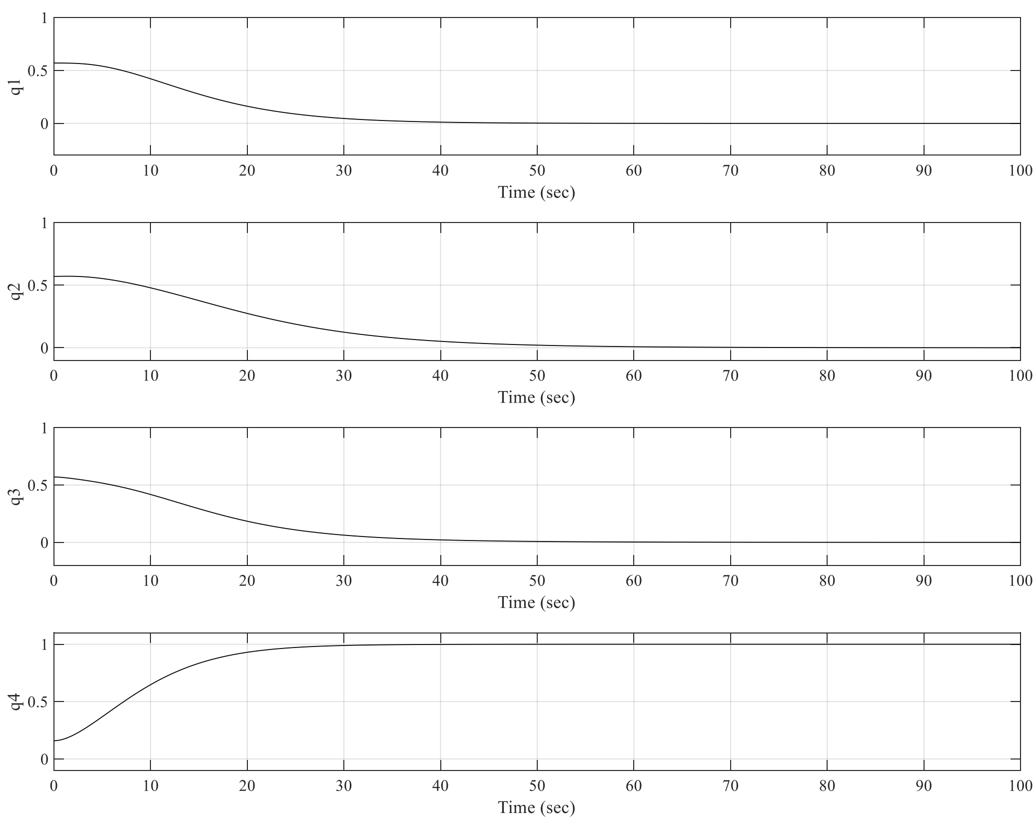


Fig.8. Quaternions V/s Time for Eigenaxis Design

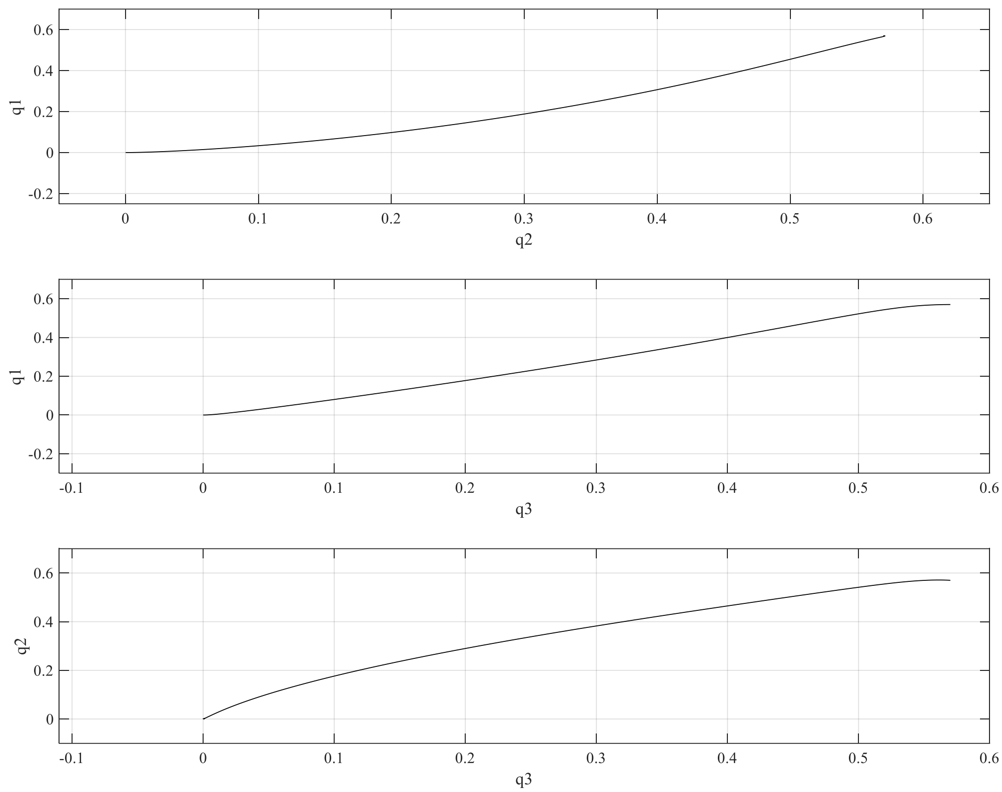


Fig.9. q_i V/s q_j plots for Eigenaxis Design

4th Method

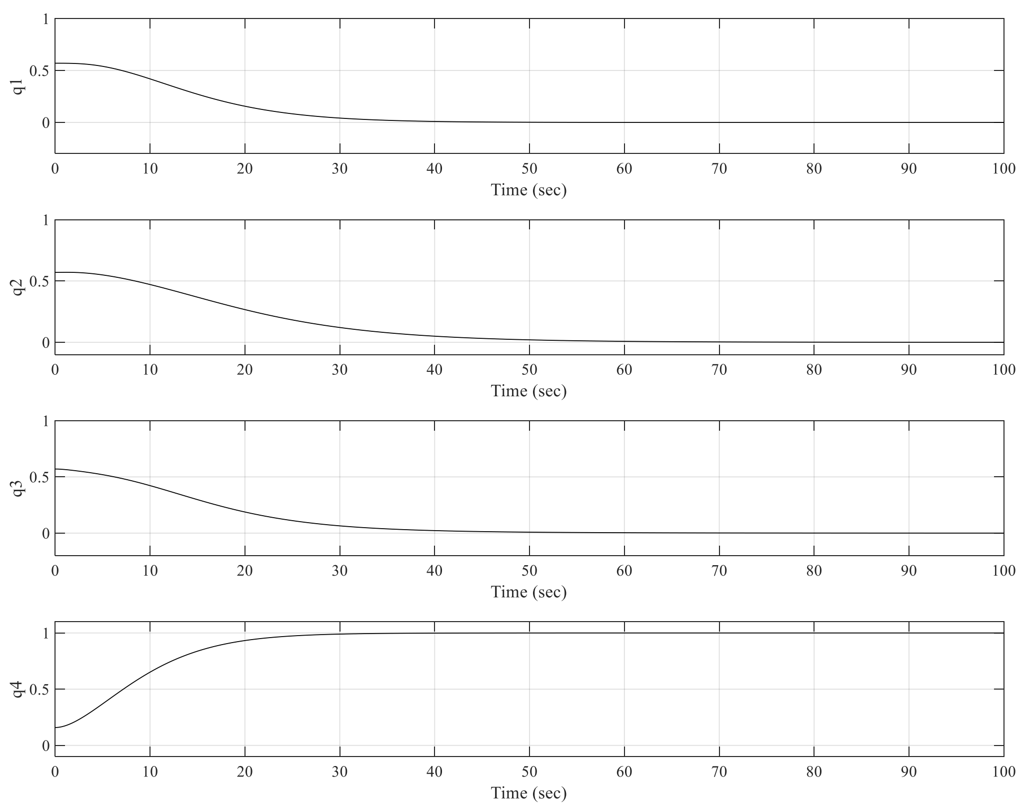


Fig.10. Quaternions V/s Time for 4th Method

Question 5. (35 points) Last, but not least, a closed-loop simulation under attitude perturbations will be performed in this question. To do so,

- Create a new subsystem in the Simulink diagram `SPACECRAFT_ATTITUDE.slx` that outputs the spacecraft position and velocity in the Earth-centered inertial reference frame. This is your two-body orbit propagator block, to be named `ORB`.
- Implement the two-body equation of motion within the `ORB` block.
- Provide graphs of orbital distance and speed as function of time over a complete orbit.
- Create another new subsystem in the Simulink diagram `SPACECRAFT_ATTITUDE.slx` that takes as inputs the simulation time, the actual spacecraft position and velocity and outputs the net external perturbation torque. This is your attitude perturbation torque block, to be named `TEX`.
- Within this `TEX` block, implement all four external perturbation torque models seen in class, i.e., drag torque, solar radiation pressure, gravity gradient, and magnetic torque.
- Interface the `ORB` block with the `TEX` block, and the `TEX` block with the the `DYN` block.
- Using the alpha-beta control gain tuning methodology, provide a graph of the net perturbing torque, i.e., the output of `TEX`, as function of time.
- Using the alpha-beta control gain tuning methodology, provide the graphs of the spacecraft quaternions vs time.
- For all 4 control tuning methods, provide the q_i vs. q_j plots.
- Happy holidays!!

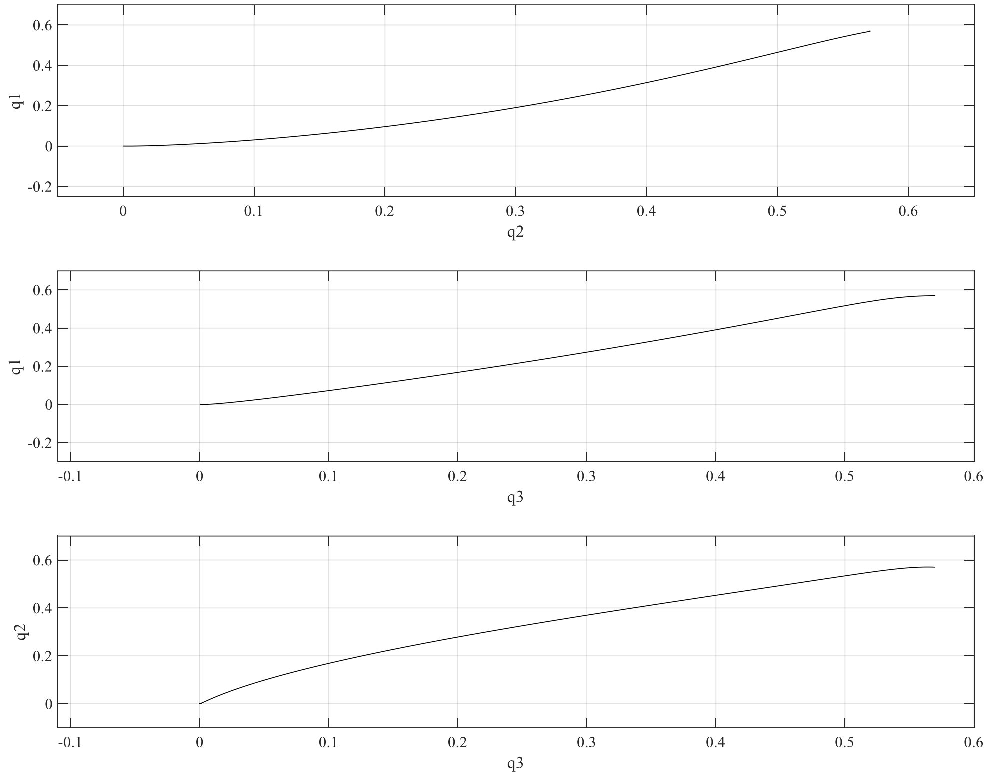


Fig.11. q_i V/s q_j plots for 4[th] Method

5.

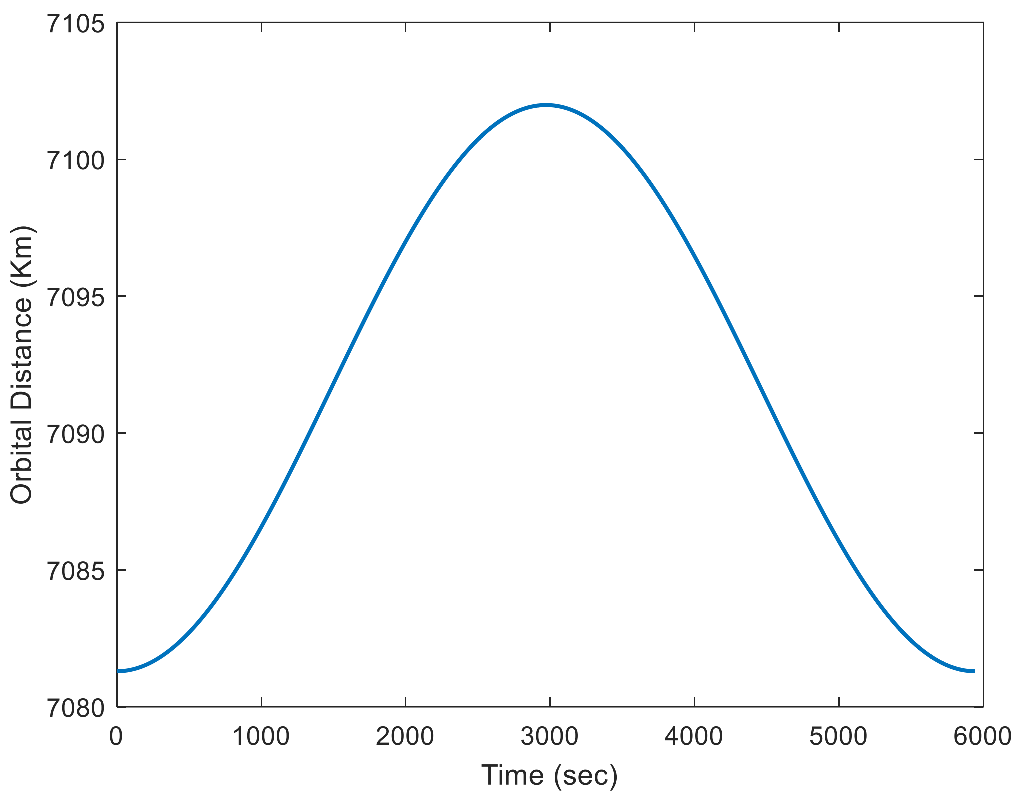


Fig.12. Orbital Distance V/s Time for a complete orbit

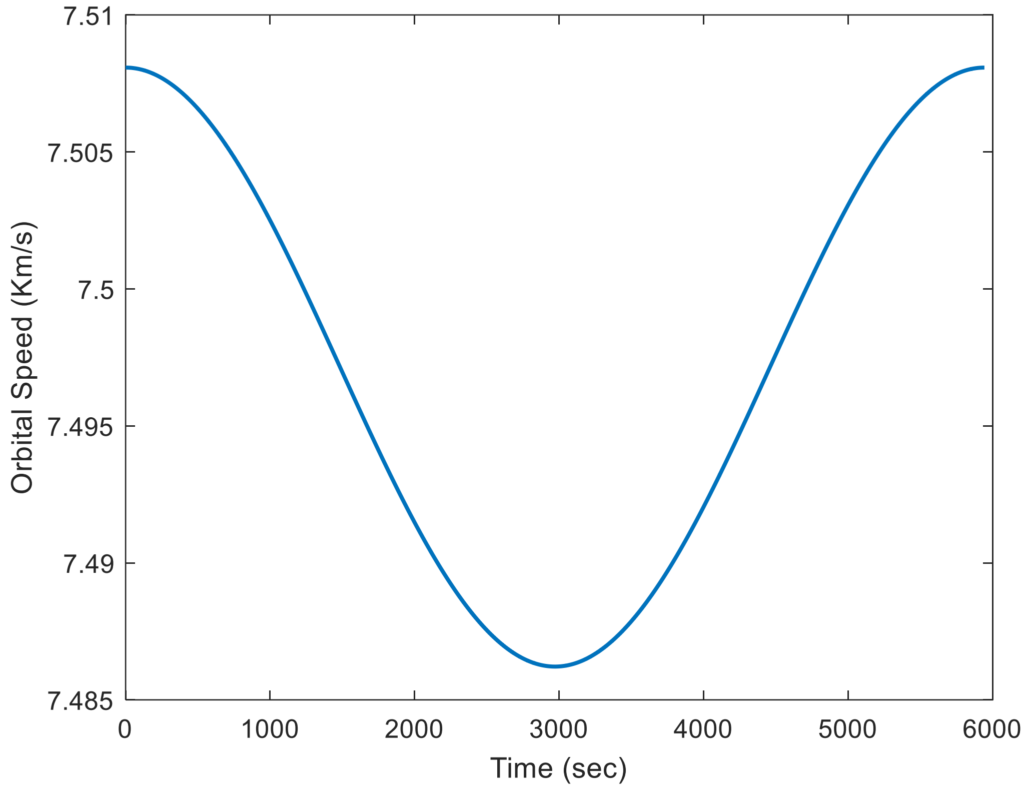


Fig.13. Orbital Speed V/s Time for a complete orbit

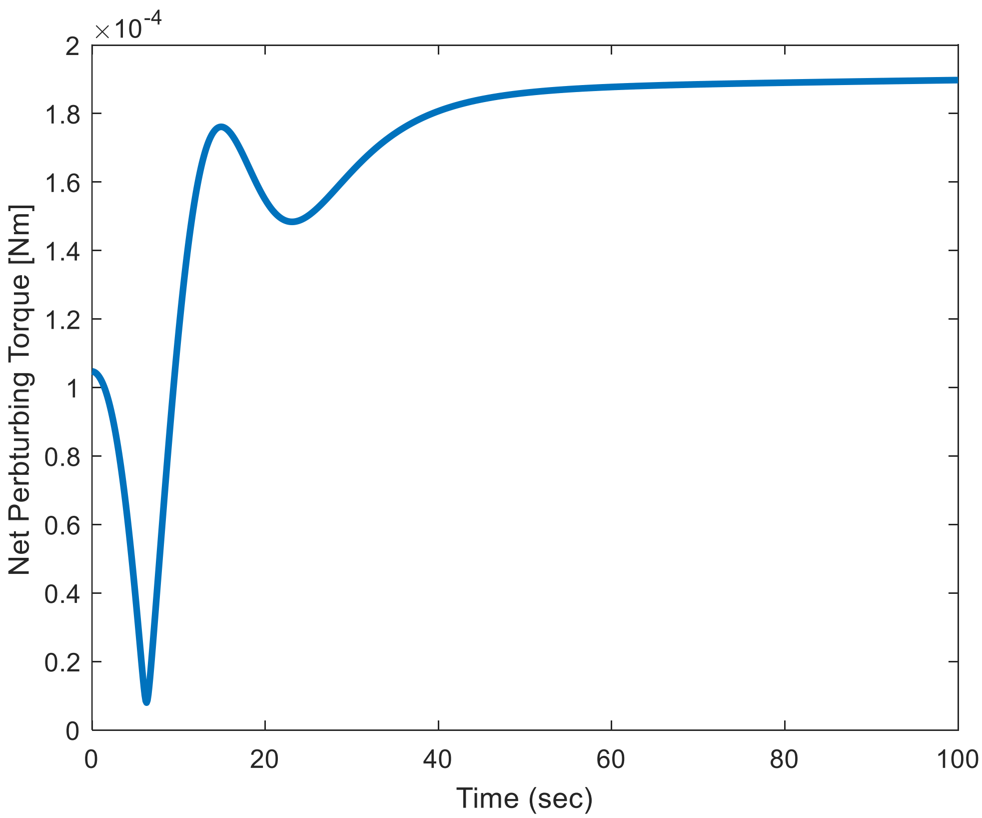


Fig.14. Net Perturbing Torque V/s Time

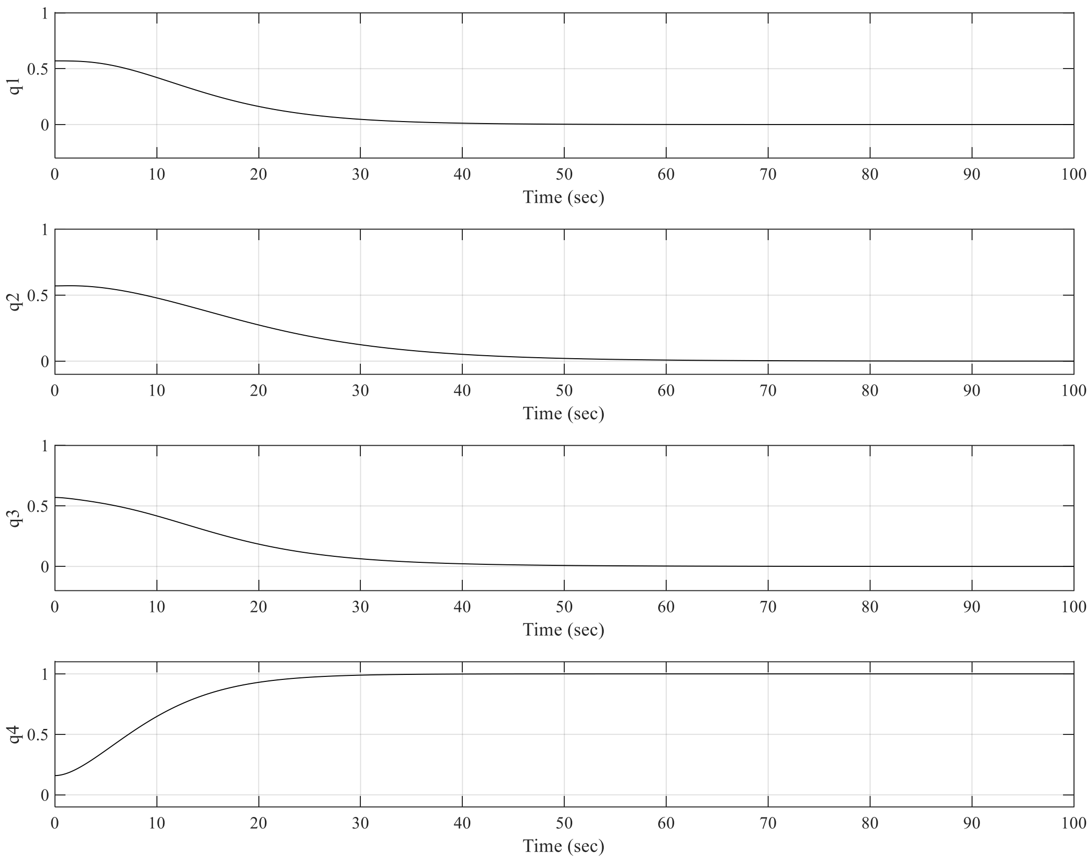


Fig.15. Spacecraft Quaternions V/s Time with alpha-beta control gain tuning

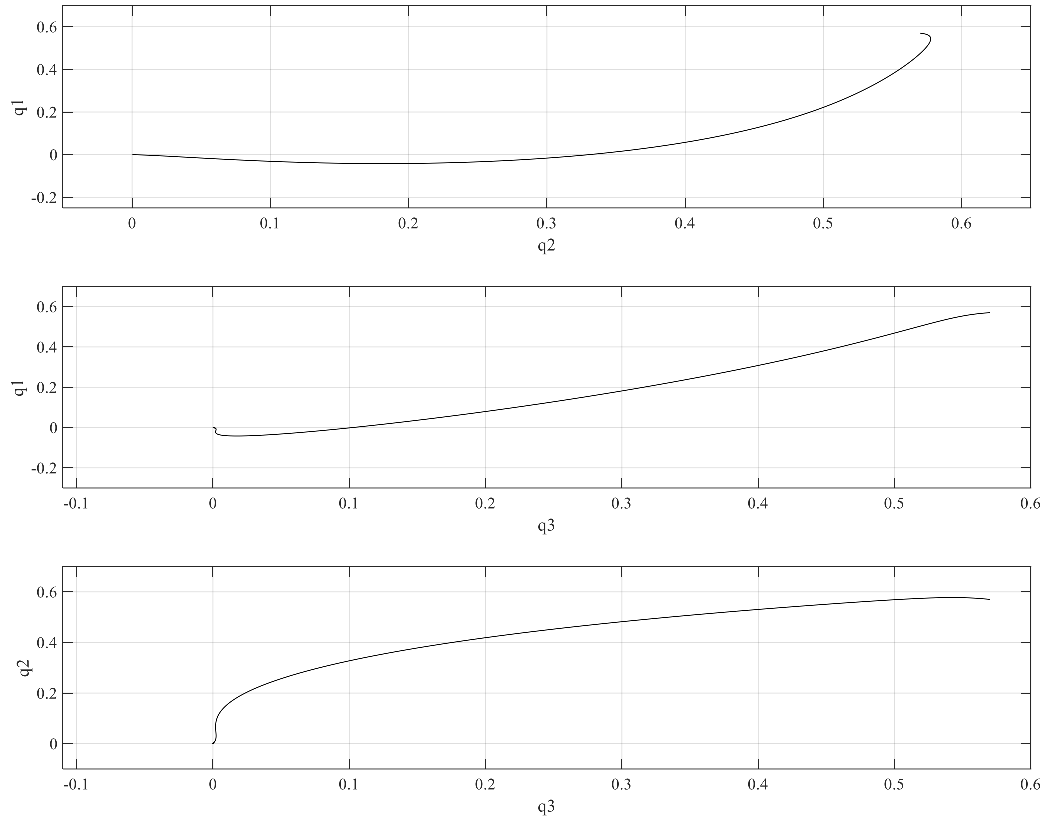


Fig.16. q_i V/s q_j plots for Scaled Design with alpha-beta control gain tuning

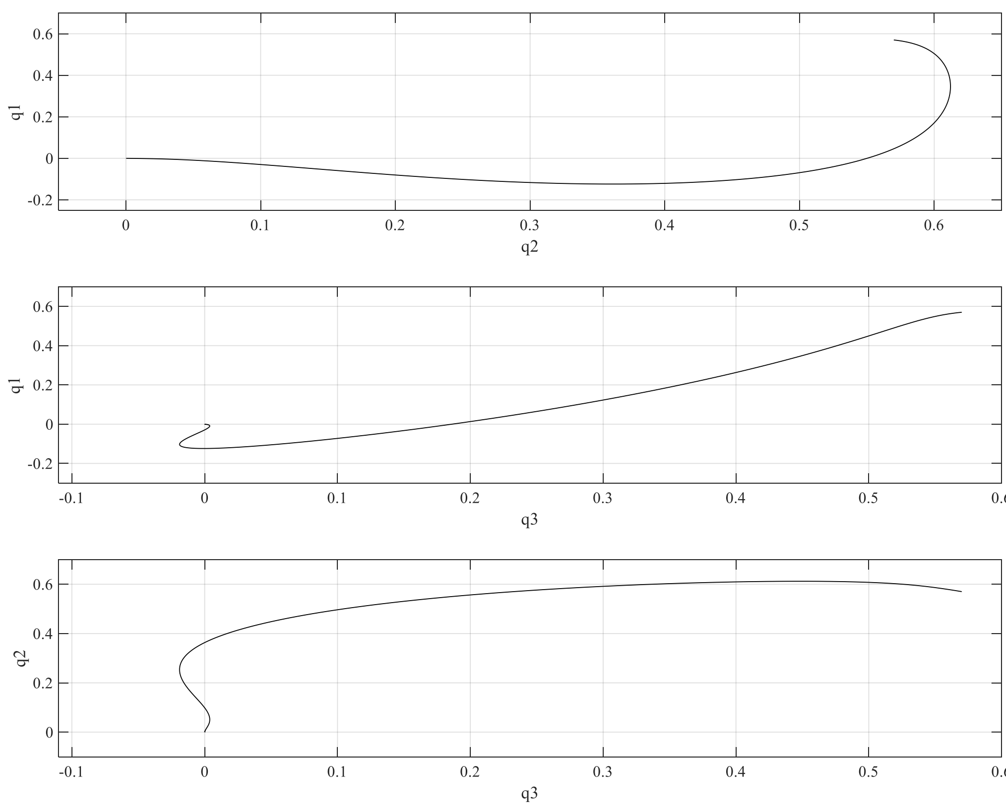


Fig.17. q_i V/s q_j plots for Inverse Inertia Design with alpha-beta control gain tuning

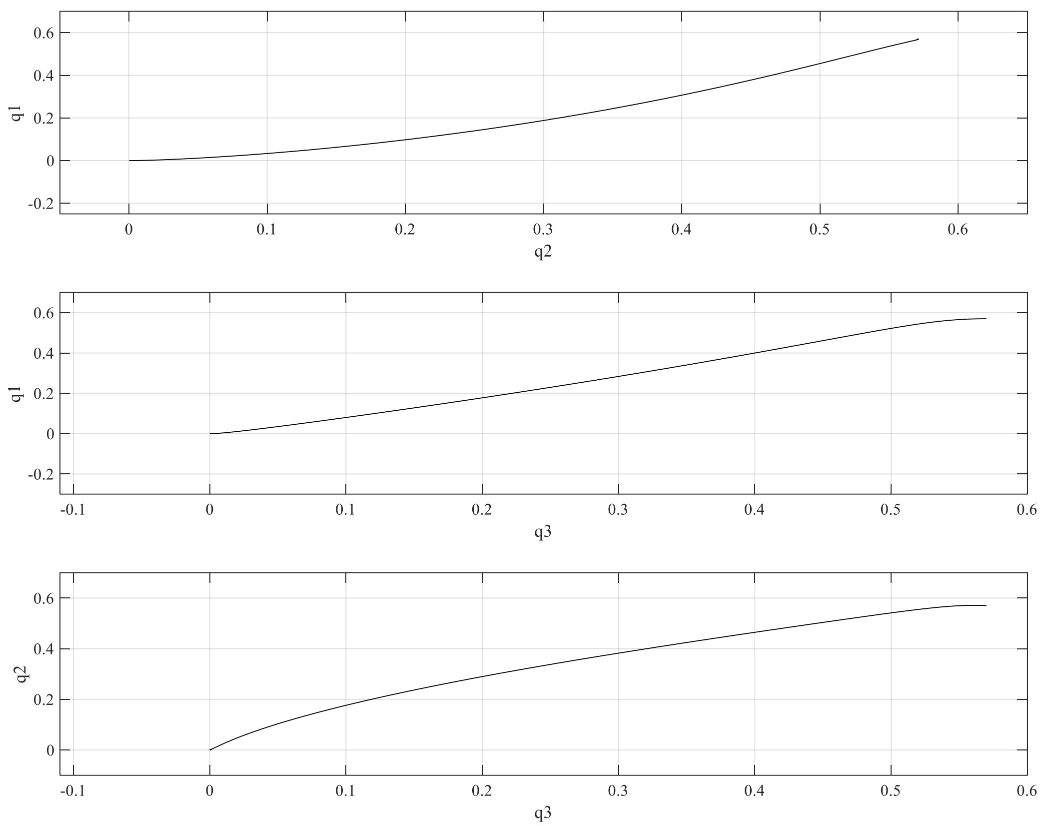


Fig.18. q_i V/s q_j plots for Eigenaxis Design with alpha-beta control gain tuning

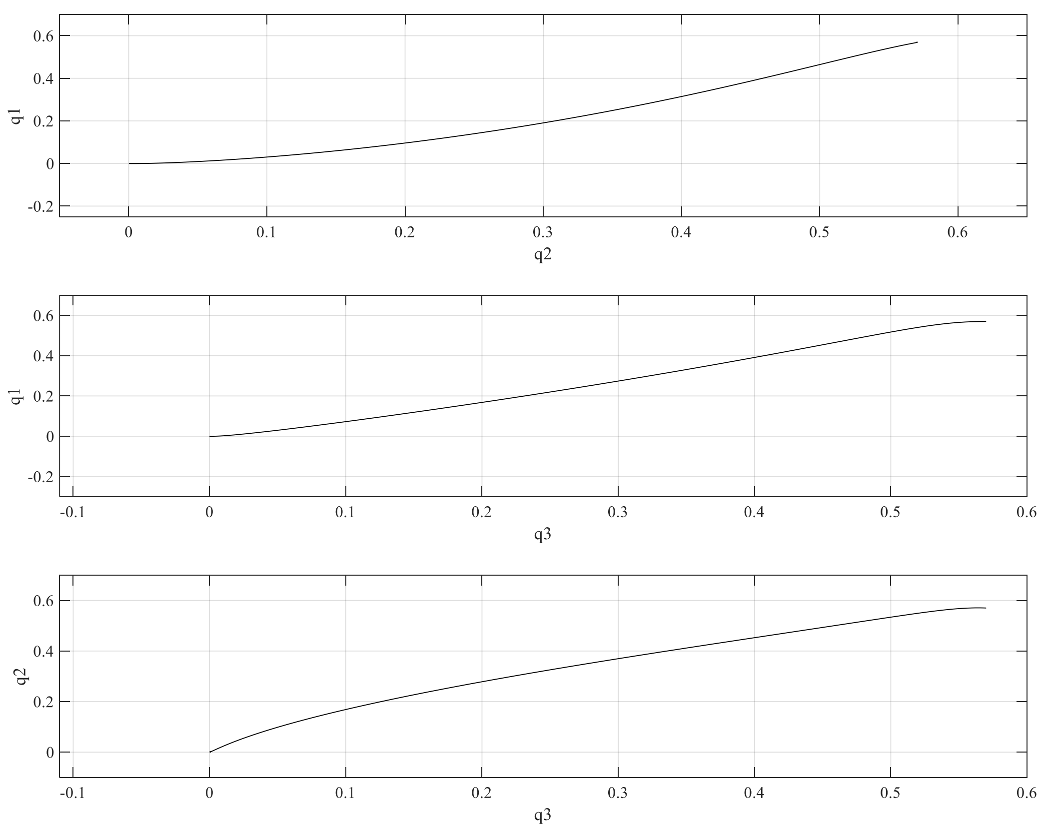
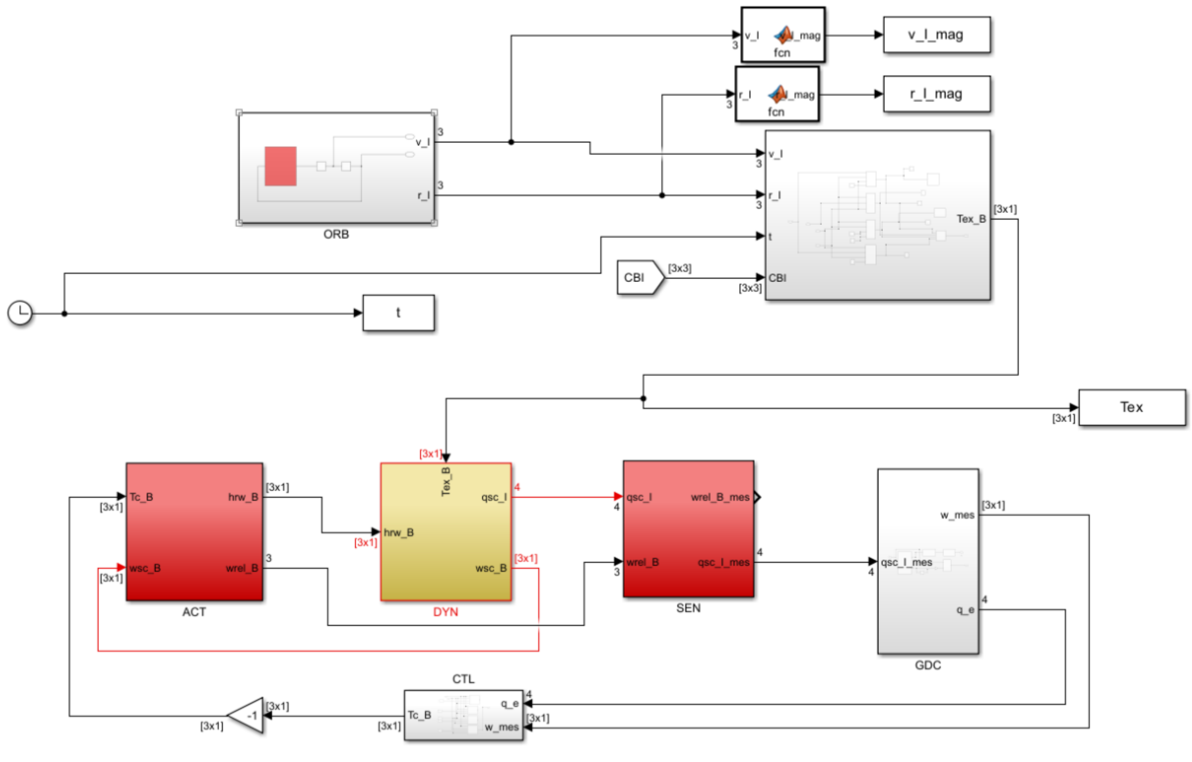


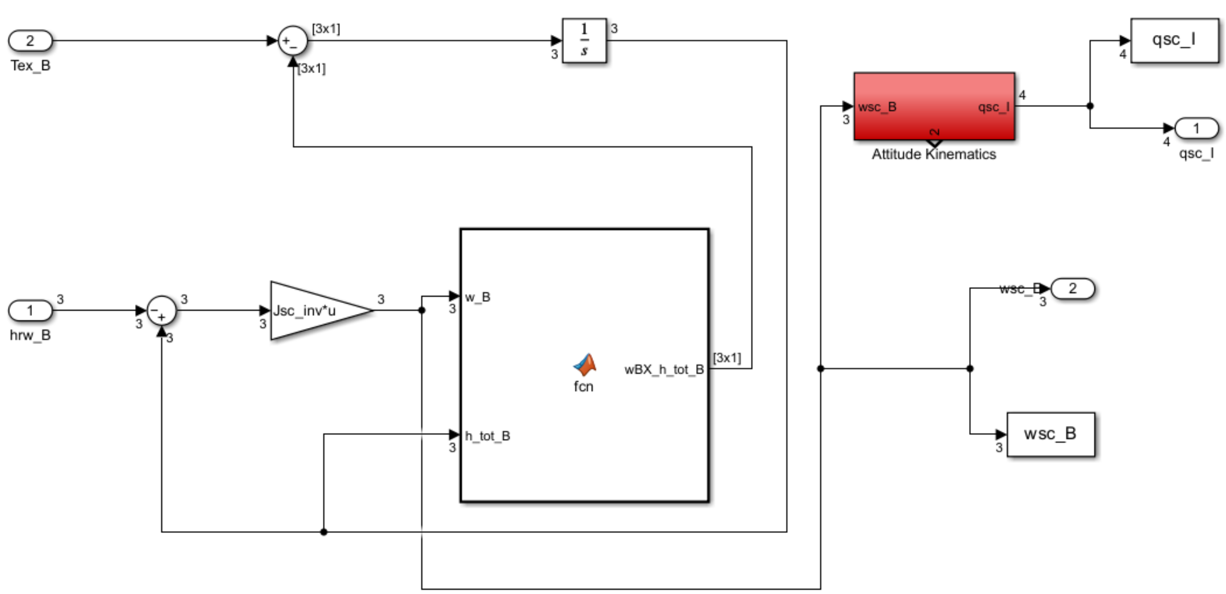
Fig.19. q_i V/s q_j plots for 4th method with alpha-beta control gain tuning

Appendix

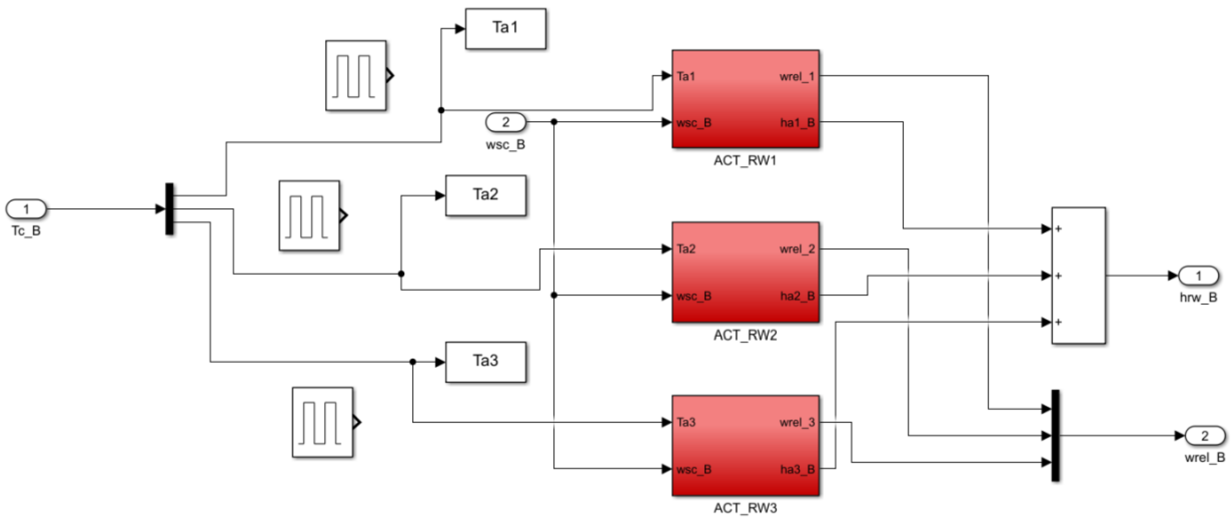
Spacecraft Attitude



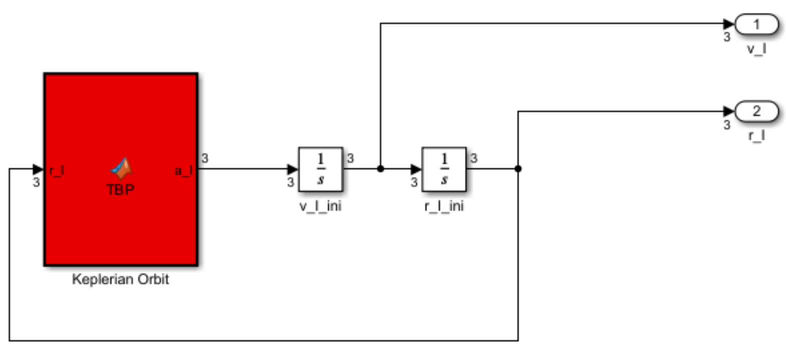
DYN Block



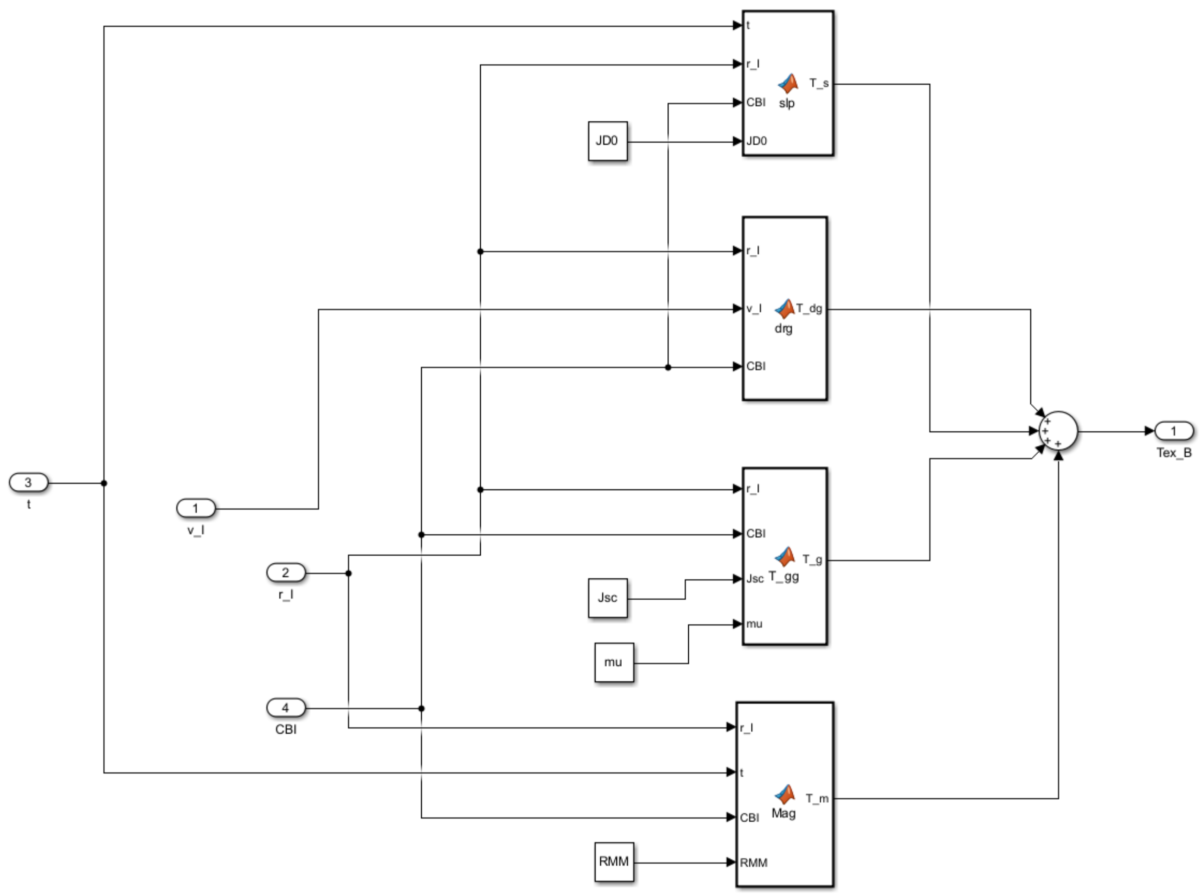
ACT Block



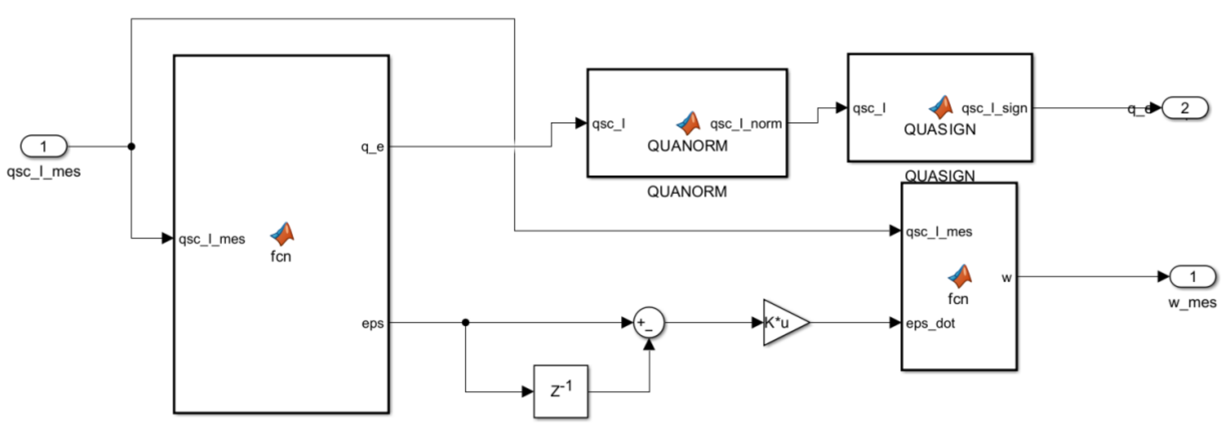
ORB Block



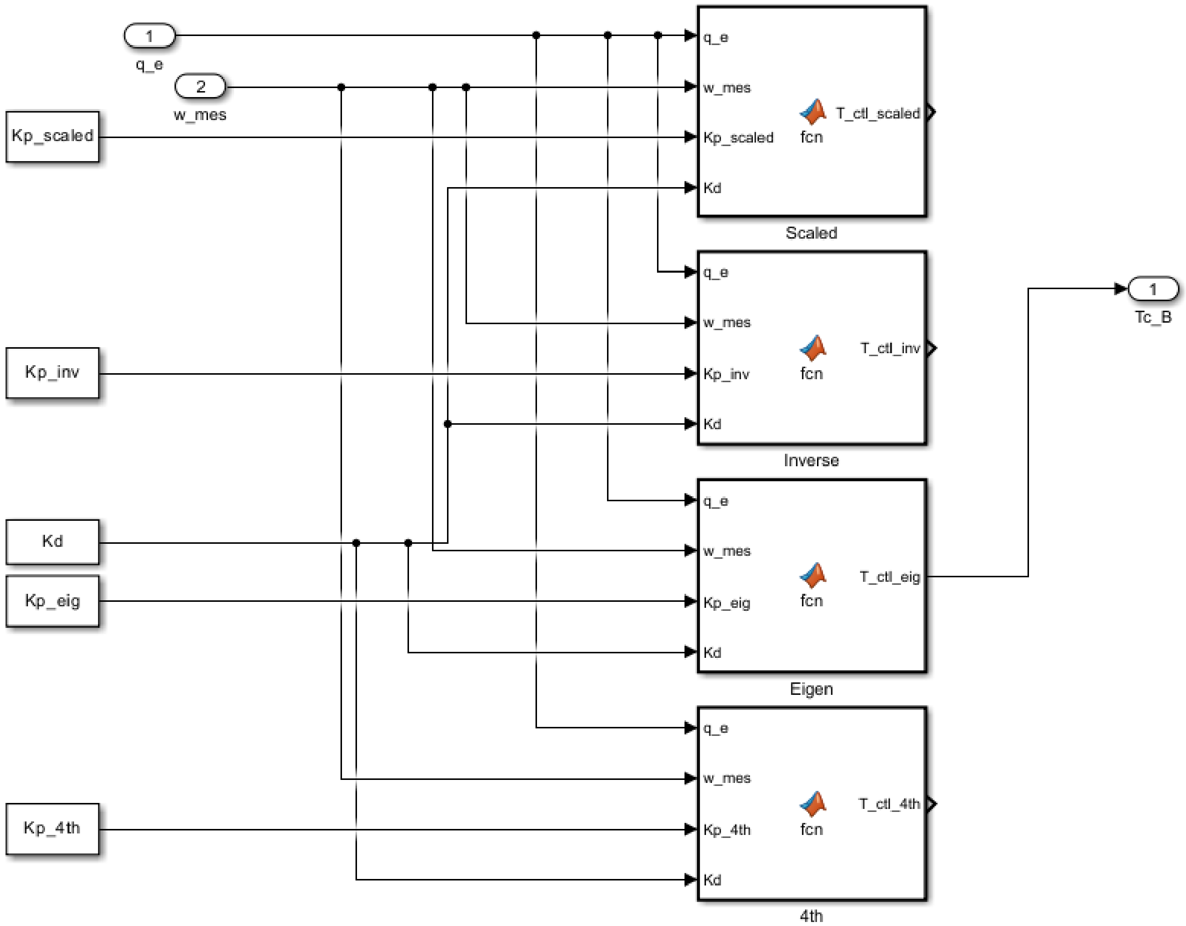
Tex Block



GDC Block



CTL Block


1
q_e
2
w_mes
Kp_scaled
Kp_inv
Kd
Kp_eig
Kp_4th
q_e
w_mes
Kp_scaled
Kd
fcn
T_ctl_scaled
Scaled
q_e
w_mes
Kp_inv
Kd
fcn
T_ctl_inv
Inverse
q_e
w_mes
Kp_eig
Kd
fcn
T_ctl_eig
Eigen
q_e
w_mes
Kp_4th
Kd
fcn
T_ctl_4th
4th
1
Tc_B

Attitude Kinematics – Quaternion Kinematics

```
function qsc_I_dot  = qdot(wsc_B,qsc_I)

% --------------------------------------------------------------
% qdot
% --------------------------------------------------------------
% Description:
% Calculate the quaternion differential kinematics
% --------------------------------------------------------------
% Inputs:
% qsc_I => quaternion (4 x 1 matrix)
% wsc_B => angular rates (3 x 1 matrix)
% --------------------------------------------------------------
% Outputs:
% qsc_I_dot => quaternion derivatives (4 x 1 matrix)
% --------------------------------------------------------------
% Parameters:
% None
% --------------------------------------------------------------
% Copyright:
% Steve Ulrich, 2023
% --------------------------------------------------------------


% Calculate

q4 = qsc_I(4);
eps = qsc_I(1:3);
eta = q4;
Skew_eps = [0 -eps(3) eps(2);
    eps(3) 0 -eps(1);
    -eps(2) eps(1)  0];

eps_dot = 0.5*((eta*eye(3))+ Skew_eps)*wsc_B;
eta_dot = -0.5*eps'*wsc_B;

qsc_I_dot = [eps_dot;eta_dot];
```

Star Tracker – QUA2CBI

```
function CBI  =  QUA2CBI(qsc_I)

% ---------------------------------------------------------------
% QUA2CBI
% ---------------------------------------------------------------
% Description:
% Convert a quaternion into the attitude matrix
% ---------------------------------------------------------------
% Inputs:
% qsc_I => quaternion (4 x 1 matrix)
% ---------------------------------------------------------------
% Outputs:
% CBI => attitude matrix (3 x 3 matrix)
% ---------------------------------------------------------------
% Parameters:
% None
% ---------------------------------------------------------------
% Copyright:
% Steve Ulrich, 2023
% ---------------------------------------------------------------

eps = qsc_I(1:3);
eta = qsc_I(4);
Skew_eps = [0 -eps(3) eps(2);
    eps(3) 0 -eps(1);
    -eps(2) eps(1)  0];

% Calculate
CBI = ((eta*eta) -eps'*eps)*eye(3) + (2*eps*eps') - 2*eta*Skew_eps;
```