Homework 2

Saturday, October 22, 2022 9:36 PM

Gagandeep Thapar

2) $\tau_c = -k_p \, \text{sign}(\eta_e)\varepsilon_e - k_d(1-\varepsilon_e^T\varepsilon_e)\omega$ is asymptotically stable around $\underline{x}^* = \begin{bmatrix} \omega^* \\ \varepsilon^* \\ \eta^* \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}_{7\times1}$

using $V = \frac{1}{4}\omega^T J \omega + \frac{1}{2}k_p \varepsilon_e^T\varepsilon_e + \frac{1}{2}k_p(\text{sign}(\eta_e) - \eta_e)^2$

<u>notice:</u> $\text{sign}(\eta_e) = \pm 1$

if $\text{sign}(\eta_e) = 1$ then $\text{sign}(\eta_e) - \eta_e < 1$

if $\text{sign}(\eta_e) = -1$ then $\text{sign}(\eta_e) - \eta_e > -1$

<u>notice:</u> $|\text{sign}(\eta_e) - \eta_e|^2 > 1$ for all $|\eta_d| \leq 1$

<u>therefore:</u> $(\text{sign}(\eta_e) - \eta_e)^2 = (1-\eta_e)^2$ for all $\eta_e \leq 1$

$V = \frac{1}{4}\omega^T J\omega + \frac{1}{2}k_p\varepsilon_e^T\varepsilon_e + \frac{1}{2}k_p(1-\eta_e)^2$

$\dot{V} = \frac{1}{4}\dot{\omega}^T J\omega + \frac{1}{4}\omega^T J\dot{\omega} + \frac{1}{2}k_p\dot{\varepsilon}_e^T\varepsilon_e + \frac{1}{2}k_p\varepsilon_e^T\dot{\varepsilon}_e - k_p(1-\eta_e)\dot{\eta}_e$

$= \frac{1}{2}\omega^T J\left[-J^{-1}(\omega^\times J\omega + k_p\varepsilon_e + k_d\omega)\right] + k_p\varepsilon_e^T\frac{1}{2}(\eta_e \mathbb{I} + \varepsilon_e^\times)\omega$
$- k_p(1-\eta_e)(-\frac{1}{2}\varepsilon_e^T\omega)$

after combining inverse matrices & simplifying ---

$\dot{V} = \frac{1}{2}\omega^T\varepsilon_e\left[-k_p + \eta_e k_p + k_p - k_p\eta_e\right]\omega - \frac{1}{2}k_d\omega^T\omega$

<u>notice:</u> $\dot{V} = \frac{1}{2}k_d\omega^T\omega \leq 0$ for all $k_d > 0$

<u>$\therefore$ Stable!</u>

for LaSalle's Thm --- prove $V(x(t)) \leq V(x(0))$

using $\dot{V} = -J^{-1}(\omega^\times J\omega + k_p\varepsilon_e + k_d\omega) + k_p\varepsilon_e^T\frac{1}{2}(\eta_e\mathbb{I} + \varepsilon_e^\times)\omega - k_p(1-\eta_e)(-\frac{1}{2}\varepsilon_e^T\omega)$

choose $\Omega = \{\underline{x} \in \mathbb{R}^n \text{ where } V(\underline{x}) \leq V(\underline{x}(0))\}$

and $E = \{\underline{x} \in \Omega \text{ where } \omega = 0\}$

$\dot{V}(\omega=0) = -J^{-1}(\cancel{\omega^\times J\omega}^{\,0} + k_p\varepsilon_e + \cancel{k_d\omega}^{\,0}) + \cancel{k_p\varepsilon_e^T}^{\,0}\frac{1}{2}(\cancel{\eta_e\mathbb{I} + \varepsilon_e^\times})\omega^{\,0} - \cancel{k_p(1-\eta_e)(-\frac{1}{2}\varepsilon_e^T\omega)}^{\,0}$

$\dot{V}(\omega=0) = -J^{-1}k_p\varepsilon_e = 0$

$J \neq 0$ (Trivial case)

$k_p \neq 0$ (Trivial case)

$\therefore \varepsilon_e = 0 \Rightarrow \underline{x}(0) = \varepsilon_e = 0$

and for all t $\dot{V}(\omega=0) \leq 0$

$\therefore V(\underline{x}(t)) \leq V(\underline{x}(0))$ for all $t > 0$! $\therefore$ Asymptotically stable!

Table of Contents

Gagandeep Thapar; AERO 560 HW1B

Problem 2

givens

initial conditions

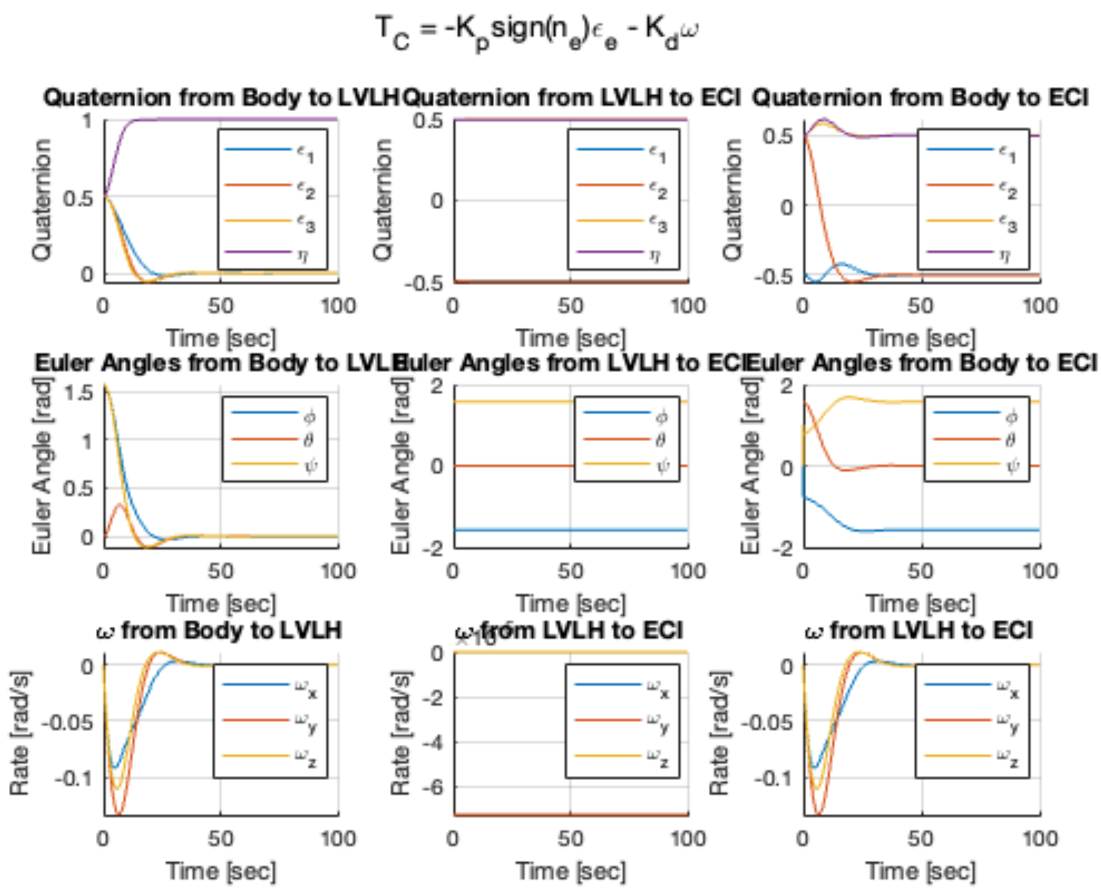
run sim

unpack data: A

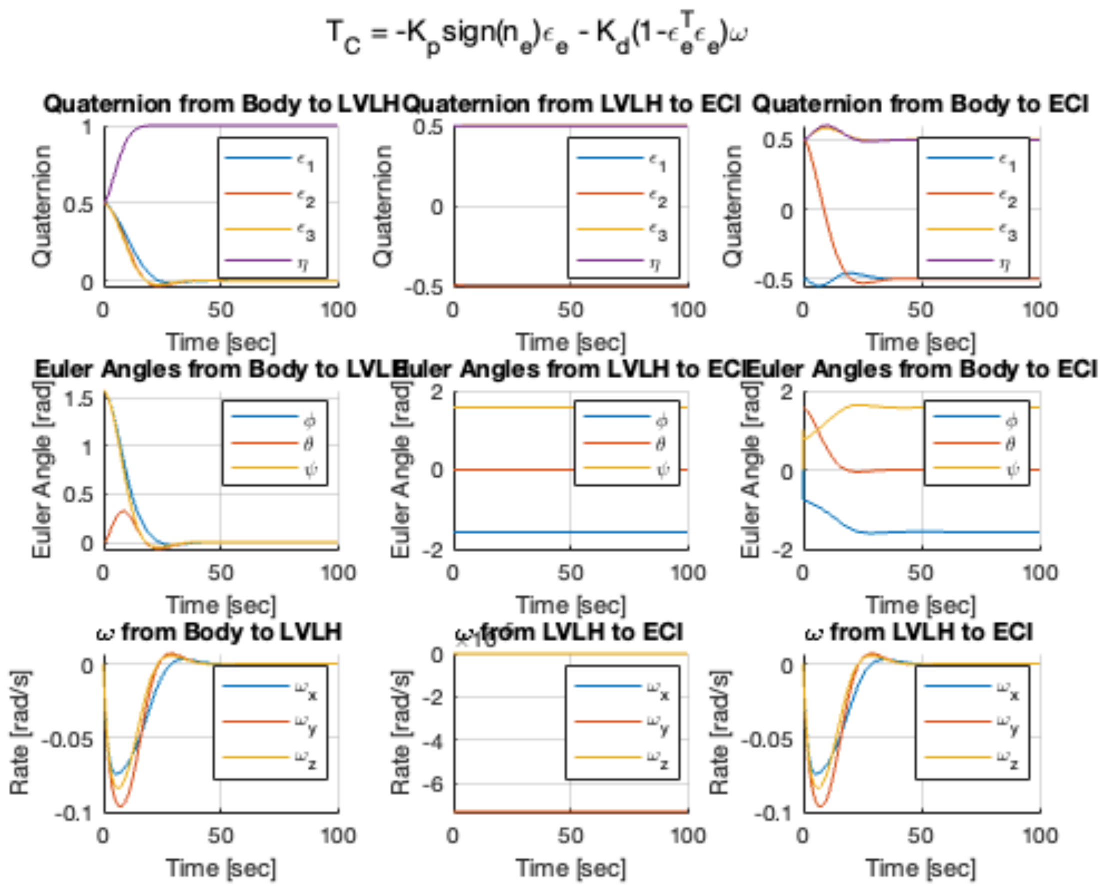
unpack data: B

unpack data: C

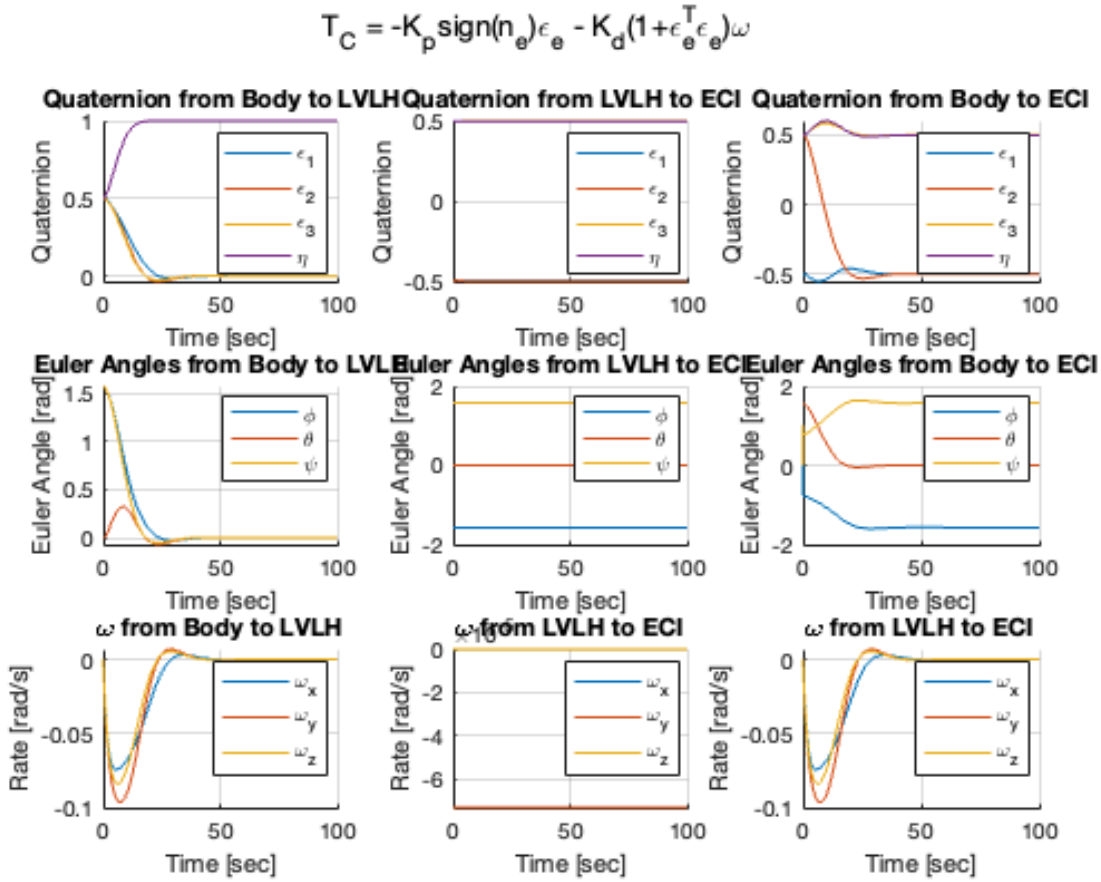
plot data: A



plot data: B


T_C = -K_p sign(n_e)ϵ_e - K_d(1-ϵ_e^T ϵ_e)ω
Quaternion from Body to LVLH
Quaternion from LVLH to ECI
Quaternion from Body to ECI
Quaternion
ϵ1
ϵ2
ϵ3
η
Euler Angles from Body to LVLH
Euler Angles from LVLH to ECI
Euler Angles from Body to ECI
Euler Angle [rad]
ϕ
θ
ψ
ω from Body to LVLH
ω from LVLH to ECI
ω from LVLH to ECI
Rate [rad/s]
ωx
ωy
ωz
Time [sec]

plot data: C



plot misc

~~~~~~~~~~~~~~~~~~~~

4. The satellite can maintain pointing with the disturbance torque. Setting the disturbance torque to 0 results in a similar result with the satellite reaching steady state in less time.

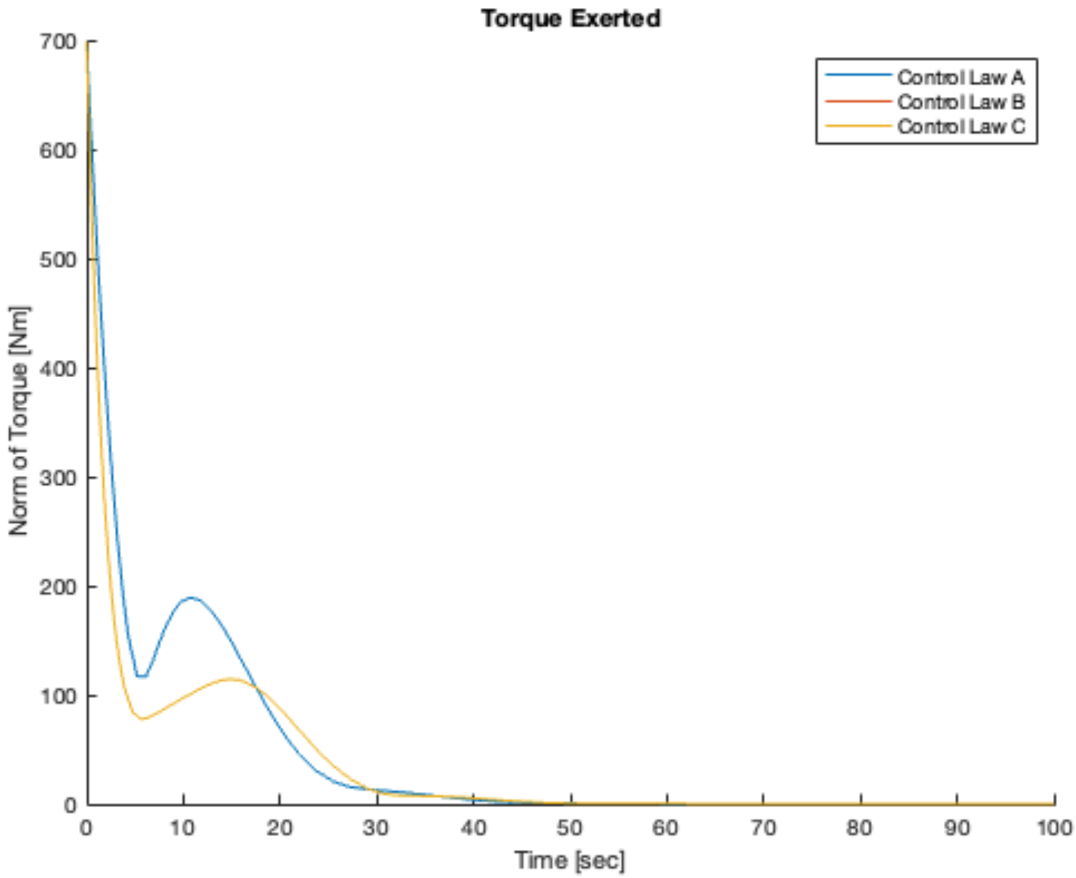
~~~~~~~~~~~~~~~~~~~~

5. The max torque exerted in the first control law is 698.50 Nm. With a height of 10.00 m, the satellite must exert 139.70 N of thrust. No EP is currently capable of producing that much thrust.

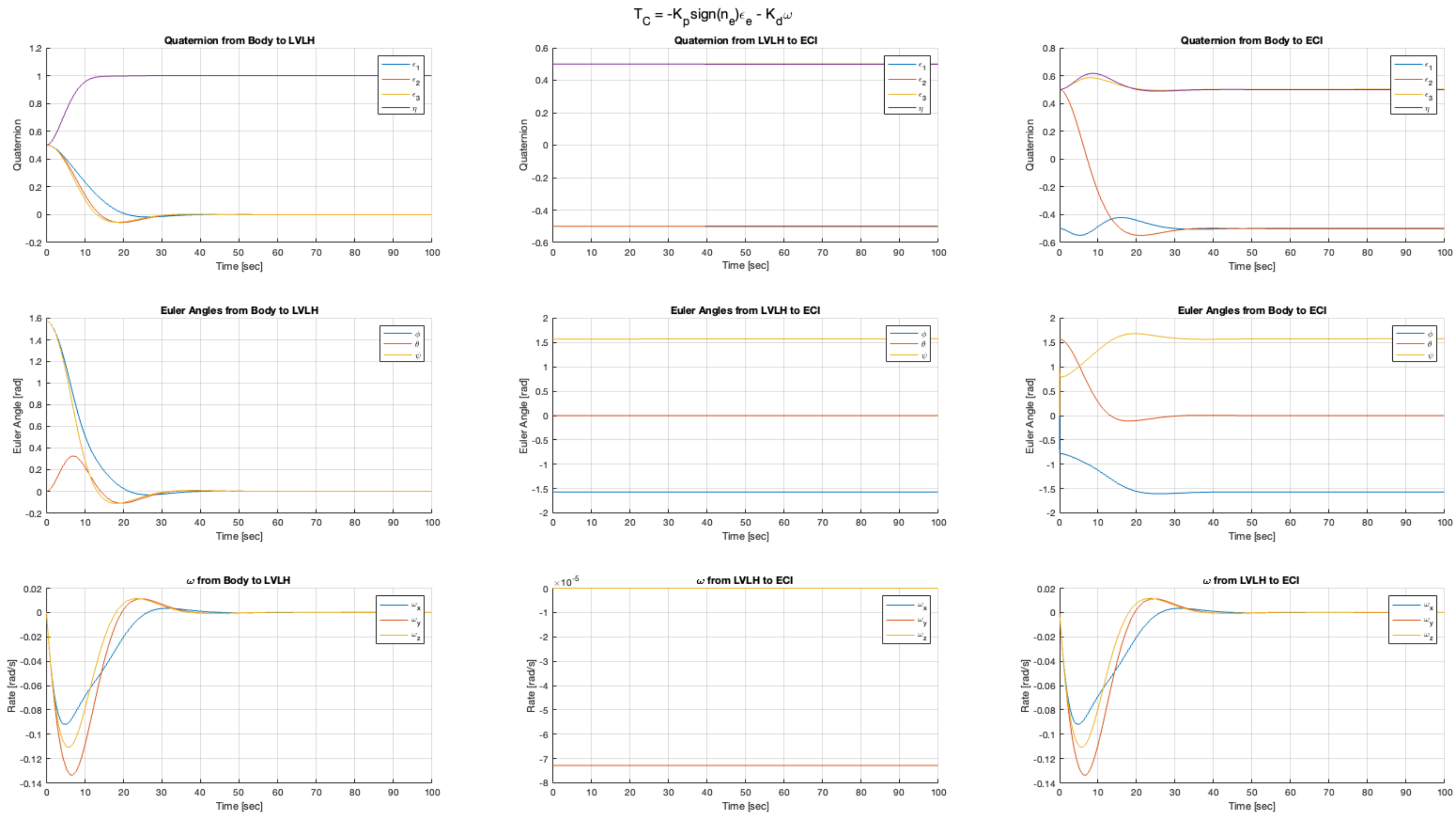
~~~~~~~~~~~~~~~~~~~~

6. The first control law which only considers Kd and w are an approximation of the other control laws but overall are similar. The second two control laws are identical as expected. All reach steady state in a similar time however the approximation (Control Law A) requires much more thrust than its counterparts over the course of the mission.

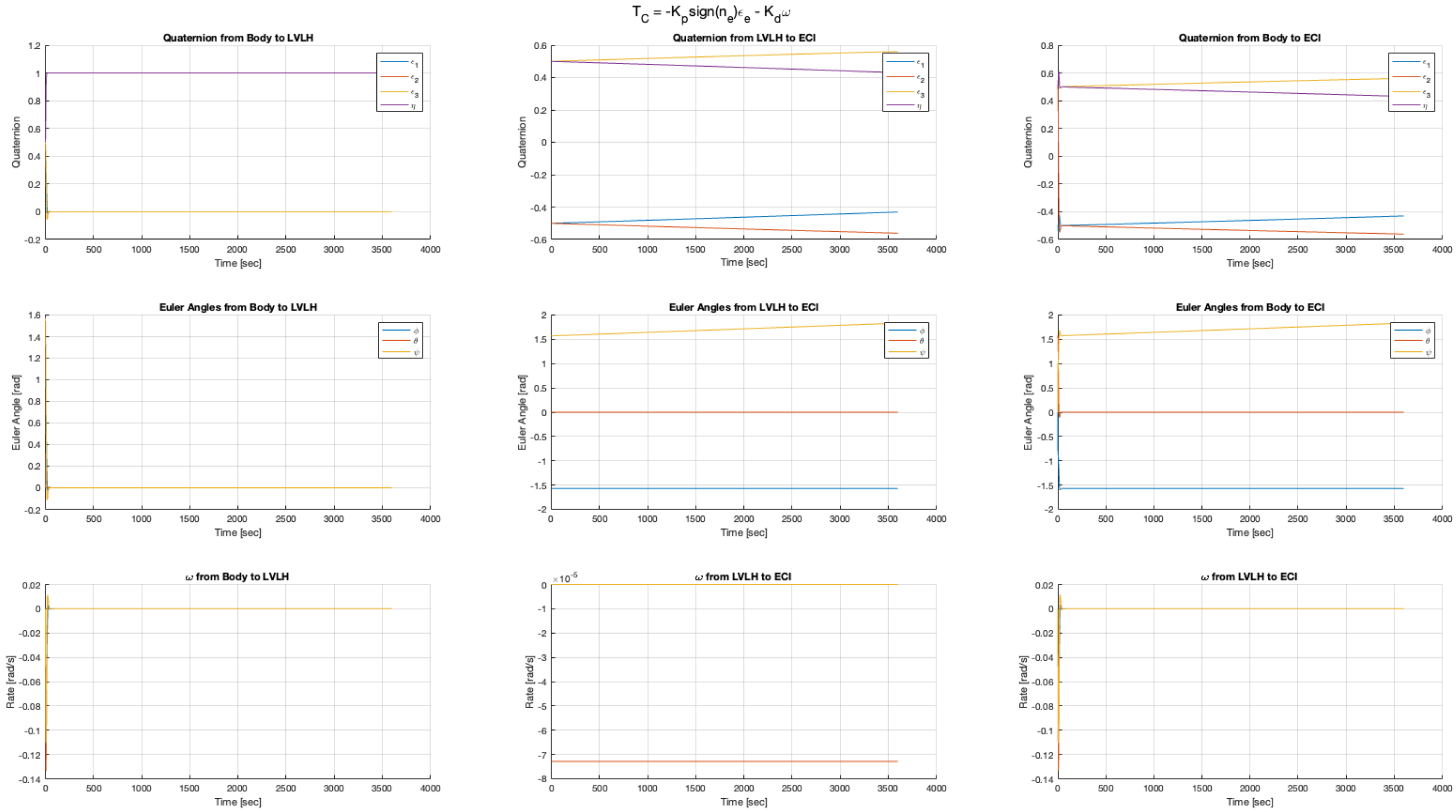
~~~~~~~~~~~~~~~~~~~~

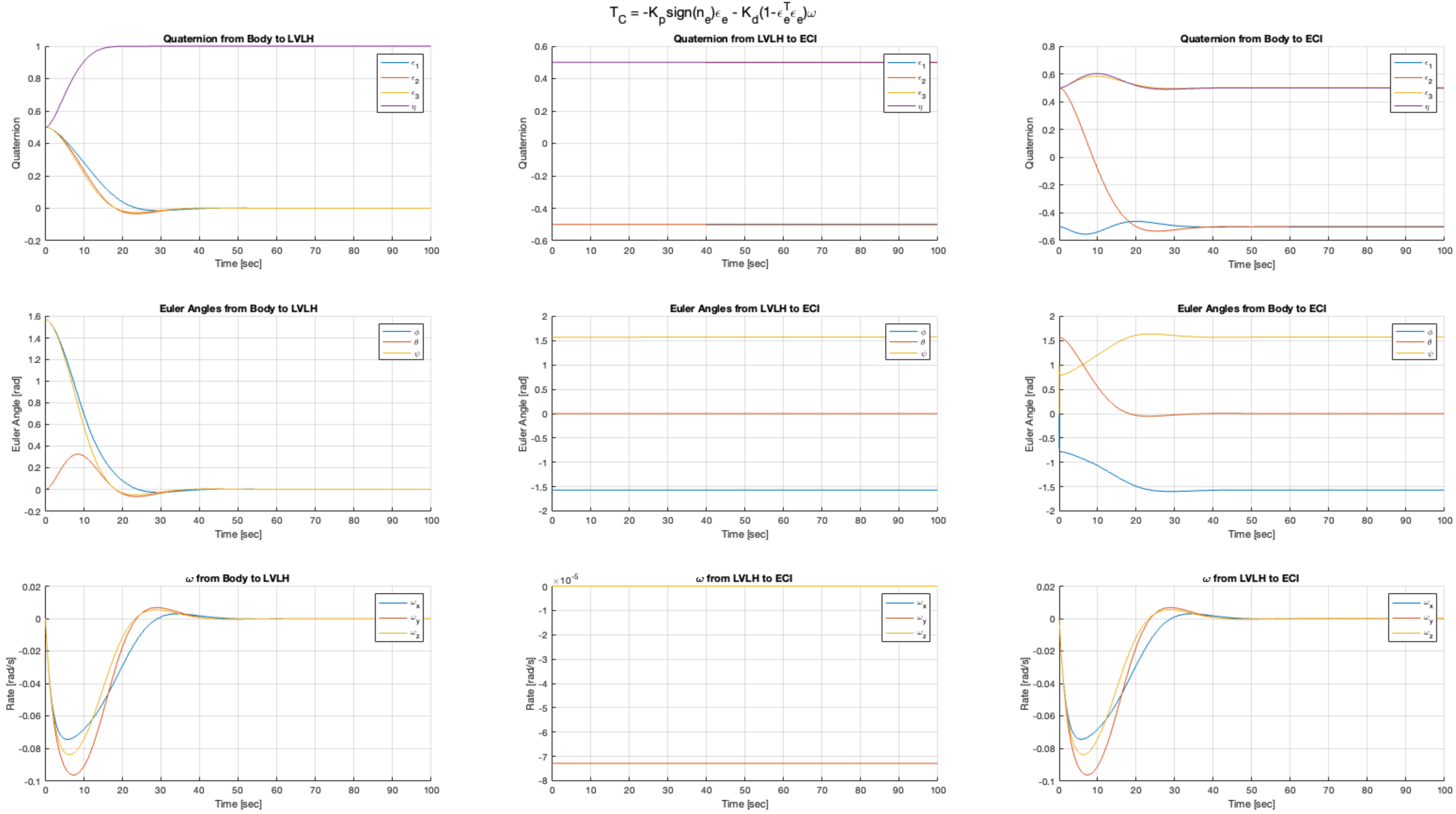


Published with MATLAB® R2022b

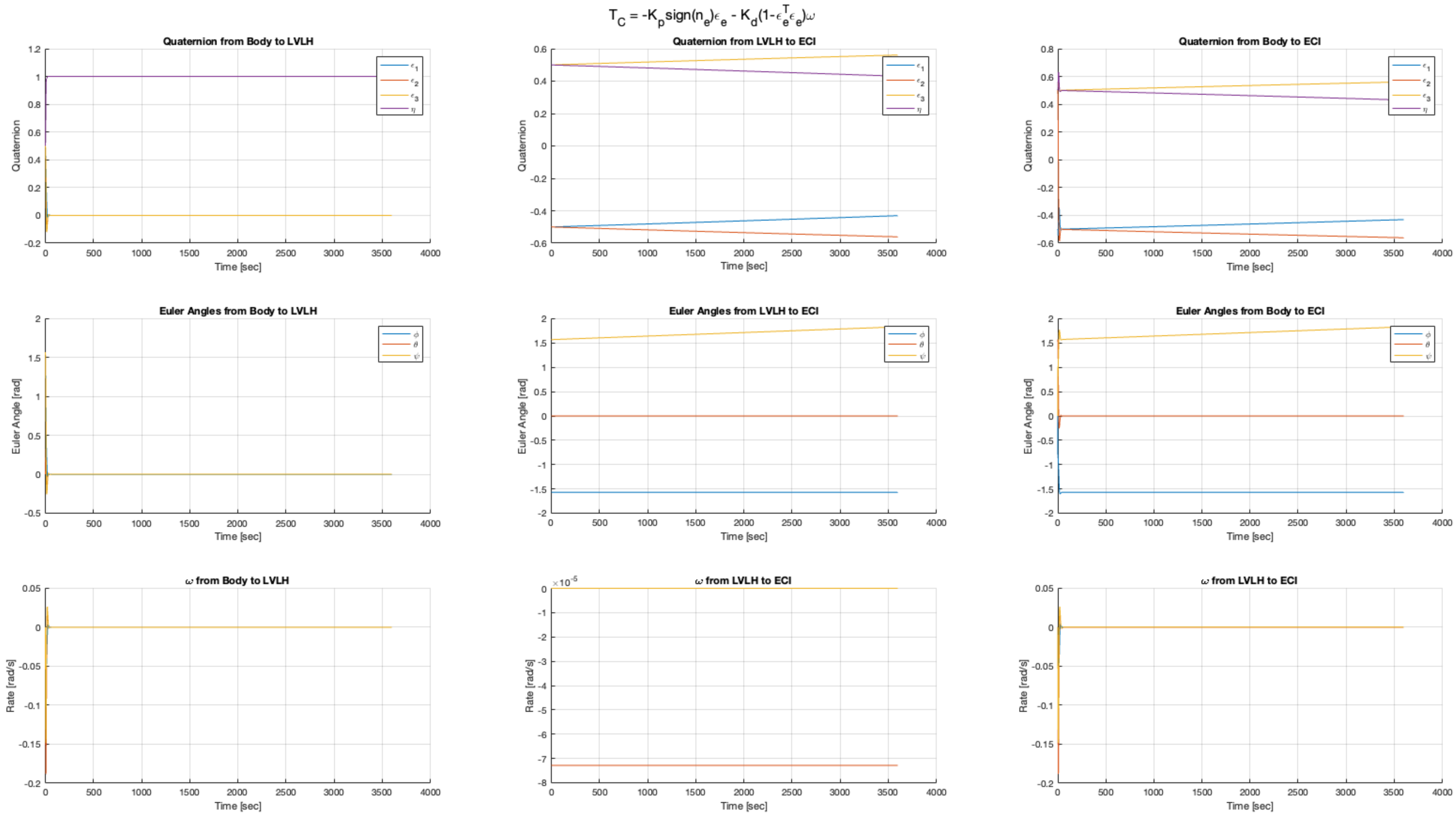


$T_C = -K_p \mathrm{sign}(n_e)\epsilon_e - K_d\omega$
Quaternion from Body to LVLH
Quaternion from LVLH to ECI
Quaternion from Body to ECI
Euler Angles from Body to LVLH
Euler Angles from LVLH to ECI
Euler Angles from Body to ECI
ω from Body to LVLH
ω from LVLH to ECI
ω from LVLH to ECI
Quaternion
Euler Angle [rad]
Rate [rad/s]
Time [sec]
ϵ_1
ϵ_2
ϵ_3
η
ϕ
θ
ψ
ω_x
ω_y
ω_z
$\times 10^{-5}$


T_C = -K_p sign(n_e)ε_e - K_d ω
Quaternion from Body to LVLH
Quaternion from LVLH to ECI
Quaternion from Body to ECI
Quaternion
Time [sec]
ε_1
ε_2
ε_3
η
Euler Angles from Body to LVLH
Euler Angles from LVLH to ECI
Euler Angles from Body to ECI
Euler Angle [rad]
φ
θ
ψ
ω from Body to LVLH
ω from LVLH to ECI
ω from LVLH to ECI
Rate [rad/s]
×10^-5

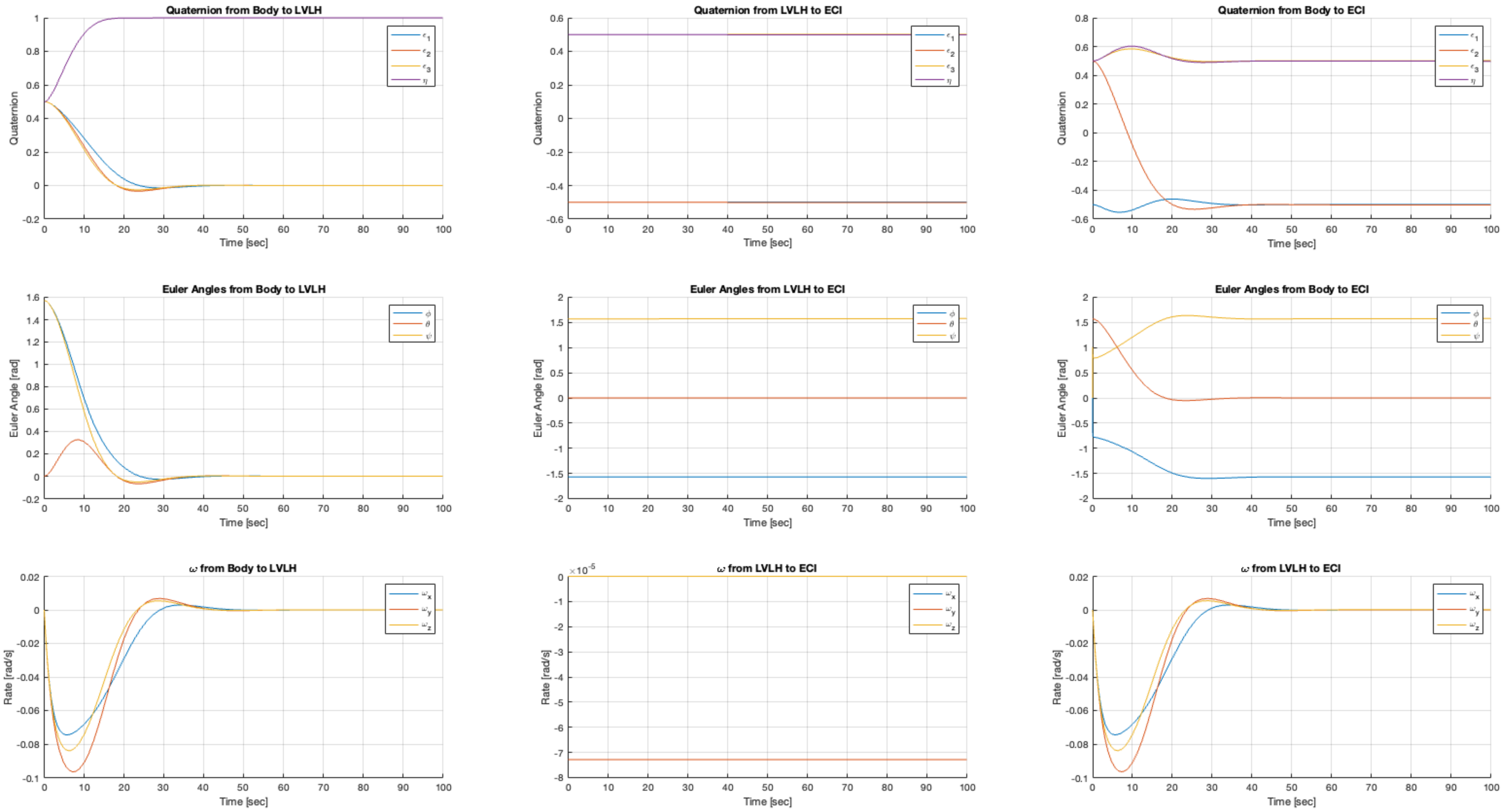


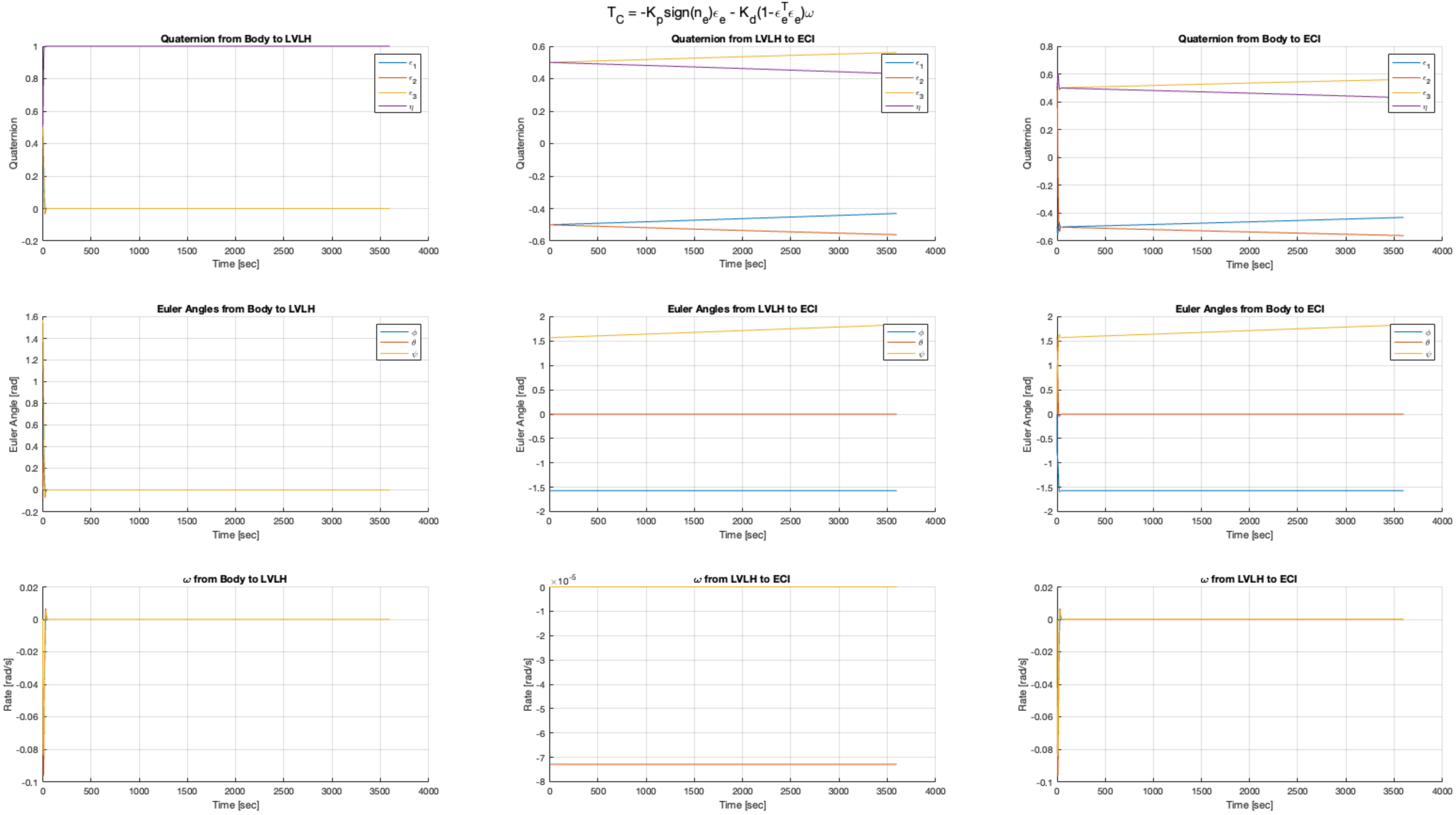
$T_C = -K_p \mathrm{sign}(n_e)\epsilon_e - K_d(1-\epsilon_e^T\epsilon_e)\omega$
Quaternion from Body to LVLH
Quaternion from LVLH to ECI
Quaternion from Body to ECI
Quaternion
ϵ_1
ϵ_2
ϵ_3
η
Euler Angles from Body to LVLH
Euler Angles from LVLH to ECI
Euler Angles from Body to ECI
Euler Angle [rad]
ϕ
θ
ψ
ω from Body to LVLH
ω from LVLH to ECI
ω from LVLH to ECI
Rate [rad/s]
$\times 10^{-5}$
ω_x
ω_y
ω_z
Time [sec]



$T_C = -K_p sign(n_e)\epsilon_e - K_d(1-\epsilon_e^T\epsilon_e)\omega$
Quaternion from Body to LVLH
Quaternion from LVLH to ECI
Quaternion from Body to ECI
Quaternion
Time [sec]
ϵ_1
ϵ_2
ϵ_3
η
Euler Angles from Body to LVLH
Euler Angles from LVLH to ECI
Euler Angles from Body to ECI
Euler Angle [rad]
ϕ
θ
ψ
ω from Body to LVLH
ω from LVLH to ECI
ω from LVLH to ECI
Rate [rad/s]
$\times 10^{-5}$

$$T_C = -K_p \mathrm{sign}(n_e)\epsilon_e - K_d(1+\epsilon_e^T\epsilon_e)\omega$$





$T_C = -K_p \mathrm{sign}(n_e)\epsilon_e - K_d(1-\epsilon_e^T\epsilon_e)\omega$
Quaternion from Body to LVLH
Quaternion from LVLH to ECI
Quaternion from Body to ECI
Quaternion
ϵ_1
ϵ_2
ϵ_3
η
Euler Angles from Body to LVLH
Euler Angles from LVLH to ECI
Euler Angles from Body to ECI
Euler Angle [rad]
ϕ
θ
ψ
ω from Body to LVLH
ω from LVLH to ECI
ω from LVLH to ECI
Rate [rad/s]
$\times 10^{-5}$
Time [sec]

Table of Contents

Gagandeep Thapar; AERO 560 HW1B

```
% housekeeping
clc;
clearvars;
close all;
```

Problem 2

givens

```
orbit.h = 129654.3;     % [kg/m2] angular momentum
orbit.ecc = 0;  % [~] eccentricity
orbit.raan = 0;     % [deg] raan
orbit.inc = 0;      % [deg] inc
orbit.omega = 0;    % [deg] arg of perigee
orbit.theta = 0;     % [deg] true anomaly
orbit.mu = 398600;

sat.mass = 1200;     % [kg]  mass of sat
sat.r = 1.5;         % [m] radius of sat
sat.h = 10;          % [m] height of sat

sat.J = sat.mass/12 * [sat.h^2 + 3*sat.r^2, 0, 0;
                        0, sat.h^2 + 3*sat.r^2, 0;
                        0, 0, 6*sat.r^2];    % principal inertial matrix

sat.des_e_lvlh = [0;0;0];
sat.des_n_lvlh = 1;
sat.des_q_lvlh = [sat.des_e_lvlh;sat.des_n_lvlh];
sat.zeta = 0.65;
sat.ts = 30;    % [sec] settling time
```

initial conditions

```
[orbit.R0, orbit.V0] = COES2STATE(orbit.h, orbit.ecc, orbit.inc, orbit.raan,
 orbit.omega, orbit.theta, 398600);
orbit.R = orbit.R0;
orbit.V = orbit.V0;
orbit.T = 2*pi*norm(orbit.R0)^(1.5)/sqrt(orbit.mu);
orbit.n = 2*pi/orbit.T;

orbit.state0 = [orbit.R;orbit.V];

sat.w0_lvlh = [0.5;-7.27;3.0]*10^(-5); % initial ang vel
sat.e0_lvlh = [0.5;0.5;0.5];   % initial quat
sat.n0_lvlh = 0.5;
sat.q0_lvlh = [sat.e0_lvlh;sat.n0_lvlh];
sat.euls0_lvlh = quat_eul([sat.e0_lvlh;sat.n0_lvlh]); % initial euler

sat.state0_lvlh = [sat.w0_lvlh;sat.e0_lvlh;sat.n0_lvlh;sat.euls0_lvlh];  %
 initial state

sat.wn = log(0.02*sqrt(1 - sat.zeta^2))/(-1*sat.zeta*sat.ts);
sat.zeta = sat.zeta*eye(3);
sat.wn = sat.wn*eye(3);

sat.Kd = 2*sat.J*sat.zeta*sat.wn;
sat.Kp = 2*(sat.J)*(sat.wn^2);
```

run sim

```
out_A = sim('hw2n_lawA', 100);
out_B = sim('hw2n_lawB', 100);
out_C = sim('hw2n_lawC', 100);
```

unpack data: A

```
A_t = squeeze(out_A.tout)';

A_orbit.w_lvlh = squeeze(out_A.w_LVLH_ECI)';
A_orbit.q_lvlh = squeeze(out_A.q_LVLH_ECI)';
A_orbit.eul_lvlh = squeeze(out_A.eul_LVLH_ECI)';

A_orbit.dist_torque_lvlh = squeeze(out_A.dist_torque)';

A_sat.w_lvlh = squeeze(out_A.w_Body_LVLH)';
A_sat.q_lvlh = squeeze(out_A.q_Body_LVLH)';
A_sat.eul_lvlh = squeeze(out_A.eul_Body_LVLH)';

A_sat.w_eci = squeeze(out_A.w_Body_ECI)';
A_sat.q_eci = squeeze(out_A.q_Body_ECI)';
A_sat.eul_eci = squeeze(out_A.eul_Body_ECI)';
```

```
A_command_torque = squeeze(out_A.command_torque)';
```

unpack data: B

```
B_t = squeeze(out_B.tout)';

B_orbit.w_lvlh = squeeze(out_B.w_LVLH_ECI)';
B_orbit.q_lvlh = squeeze(out_B.q_LVLH_ECI)';
B_orbit.eul_lvlh = squeeze(out_B.eul_LVLH_ECI)';

B_orbit.dist_torque_lvlh = squeeze(out_B.dist_torque)';

B_sat.w_lvlh = squeeze(out_B.w_Body_LVLH)';
B_sat.q_lvlh = squeeze(out_B.q_Body_LVLH)';
B_sat.eul_lvlh = squeeze(out_B.eul_Body_LVLH)';

B_sat.w_eci = squeeze(out_B.w_Body_ECI)';
B_sat.q_eci = squeeze(out_B.q_Body_ECI)';
B_sat.eul_eci = squeeze(out_B.eul_Body_ECI)';

B_command_torque = squeeze(out_B.command_torque)';
```

unpack data: C

```
C_t = squeeze(out_C.tout)';

C_orbit.w_lvlh = squeeze(out_C.w_LVLH_ECI)';
C_orbit.q_lvlh = squeeze(out_C.q_LVLH_ECI)';
C_orbit.eul_lvlh = squeeze(out_C.eul_LVLH_ECI)';

C_orbit.dist_torque_lvlh = squeeze(out_C.dist_torque)';

C_sat.w_lvlh = squeeze(out_C.w_Body_LVLH)';
C_sat.q_lvlh = squeeze(out_C.q_Body_LVLH)';
C_sat.eul_lvlh = squeeze(out_C.eul_Body_LVLH)';

C_sat.w_eci = squeeze(out_C.w_Body_ECI)';
C_sat.q_eci = squeeze(out_C.q_Body_ECI)';
C_sat.eul_eci = squeeze(out_C.eul_Body_ECI)';

C_command_torque = squeeze(out_C.command_torque)';
```

plot data: A

```
figure

subplot(3,3,1)
hold on
plot(A_t, A_sat.q_lvlh(:,1))
plot(A_t, A_sat.q_lvlh(:,2))
plot(A_t, A_sat.q_lvlh(:,3))
```

```
plot(A_t, A_sat.q_lvlh(:,4))
hold off
grid on
title('Quaternion from Body to LVLH')
xlabel('Time [sec]')
ylabel('Quaternion')
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,2)
hold on
plot(A_t, A_orbit.q_lvlh(:,1))
plot(A_t, A_orbit.q_lvlh(:,2))
plot(A_t, A_orbit.q_lvlh(:,3))
plot(A_t, A_orbit.q_lvlh(:,4))
hold off
grid on
title('Quaternion from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Quaternion')
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,3)
hold on
plot(A_t, A_sat.q_eci(:,1))
plot(A_t, A_sat.q_eci(:,2))
plot(A_t, A_sat.q_eci(:,3))
plot(A_t, A_sat.q_eci(:,4))
hold off
grid on
title('Quaternion from Body to ECI')
xlabel('Time [sec]')
ylabel('Quaternion')
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,4)
hold on
plot(A_t, A_sat.eul_lvlh(:,1))
plot(A_t, A_sat.eul_lvlh(:,2))
plot(A_t, A_sat.eul_lvlh(:,3))
hold off
grid on
title('Euler Angles from Body to LVLH')
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
legend('\phi', '\theta', '\psi')

subplot(3,3,5)
hold on
plot(A_t, A_orbit.eul_lvlh(:,1))
plot(A_t, A_orbit.eul_lvlh(:,2))
plot(A_t, A_orbit.eul_lvlh(:,3))
hold off
grid on
title('Euler Angles from LVLH to ECI')
```

```
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
legend('\phi', '\theta', '\psi')

subplot(3,3,6)
hold on
plot(A_t, A_sat.eul_eci(:,1))
plot(A_t, A_sat.eul_eci(:,2))
plot(A_t, A_sat.eul_eci(:,3))
hold off
grid on
title('Euler Angles from Body to ECI')
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
legend('\phi', '\theta', '\psi')


subplot(3,3,7)
hold on
plot(A_t, A_sat.w_lvlh(:,1))
plot(A_t, A_sat.w_lvlh(:,2))
plot(A_t, A_sat.w_lvlh(:,3))
hold off
grid on
title('\omega from Body to LVLH')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

subplot(3,3,8)
hold on
plot(A_t, A_orbit.w_lvlh(:,1))
plot(A_t, A_orbit.w_lvlh(:,2))
plot(A_t, A_orbit.w_lvlh(:,3))
hold off
grid on
title('\omega from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

subplot(3,3,9)
hold on
plot(A_t, A_sat.w_eci(:,1))
plot(A_t, A_sat.w_eci(:,2))
plot(A_t, A_sat.w_eci(:,3))
hold off
grid on
title('\omega from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

sgtitle('T_{C} = -K_psign(n_e)\epsilon_e - K_d\omega')
```

plot data: B

```
figure

subplot(3,3,1)
hold on
plot(B_t, B_sat.q_lvlh(:,1))
plot(B_t, B_sat.q_lvlh(:,2))
plot(B_t, B_sat.q_lvlh(:,3))
plot(B_t, B_sat.q_lvlh(:,4))
hold off
grid on
title('Quaternion from Body to LVLH')
xlabel('Time [sec]')
ylabel('Quaternion')
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,2)
hold on
plot(B_t, B_orbit.q_lvlh(:,1))
plot(B_t, B_orbit.q_lvlh(:,2))
plot(B_t, B_orbit.q_lvlh(:,3))
plot(B_t, B_orbit.q_lvlh(:,4))
hold off
grid on
title('Quaternion from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Quaternion')
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,3)
hold on
plot(B_t, B_sat.q_eci(:,1))
plot(B_t, B_sat.q_eci(:,2))
plot(B_t, B_sat.q_eci(:,3))
plot(B_t, B_sat.q_eci(:,4))
hold off
grid on
title('Quaternion from Body to ECI')
xlabel('Time [sec]')
ylabel('Quaternion')
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,4)
hold on
plot(B_t, B_sat.eul_lvlh(:,1))
plot(B_t, B_sat.eul_lvlh(:,2))
plot(B_t, B_sat.eul_lvlh(:,3))
hold off
grid on
title('Euler Angles from Body to LVLH')
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
```

```
legend('\phi', '\theta', '\psi')

subplot(3,3,5)
hold on
plot(B_t, B_orbit.eul_lvlh(:,1))
plot(B_t, B_orbit.eul_lvlh(:,2))
plot(B_t, B_orbit.eul_lvlh(:,3))
hold off
grid on
title('Euler Angles from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
legend('\phi', '\theta', '\psi')

subplot(3,3,6)
hold on
plot(B_t, B_sat.eul_eci(:,1))
plot(B_t, B_sat.eul_eci(:,2))
plot(B_t, B_sat.eul_eci(:,3))
hold off
grid on
title('Euler Angles from Body to ECI')
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
legend('\phi', '\theta', '\psi')


subplot(3,3,7)
hold on
plot(B_t, B_sat.w_lvlh(:,1))
plot(B_t, B_sat.w_lvlh(:,2))
plot(B_t, B_sat.w_lvlh(:,3))
hold off
grid on
title('\omega from Body to LVLH')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

subplot(3,3,8)
hold on
plot(B_t, B_orbit.w_lvlh(:,1))
plot(B_t, B_orbit.w_lvlh(:,2))
plot(B_t, B_orbit.w_lvlh(:,3))
hold off
grid on
title('\omega from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

subplot(3,3,9)
hold on
plot(B_t, B_sat.w_eci(:,1))
```

```
plot(B_t, B_sat.w_eci(:,2))
plot(B_t, B_sat.w_eci(:,3))
hold off
grid on
title('\omega from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

sgtitle('T_{C} = -K_psign(n_e)\epsilon_e - K_d(1-\epsilon_e^T
\epsilon_e)\omega')
```

plot data: C

```
figure

subplot(3,3,1)
hold on
plot(C_t, C_sat.q_lvlh(:,1))
plot(C_t, C_sat.q_lvlh(:,2))
plot(C_t, C_sat.q_lvlh(:,3))
plot(C_t, C_sat.q_lvlh(:,4))
hold off
grid on
title('Quaternion from Body to LVLH')
xlabel('Time [sec]')
ylabel('Quaternion')
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,2)
hold on
plot(C_t, C_orbit.q_lvlh(:,1))
plot(C_t, C_orbit.q_lvlh(:,2))
plot(C_t, C_orbit.q_lvlh(:,3))
plot(C_t, C_orbit.q_lvlh(:,4))
hold off
grid on
title('Quaternion from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Quaternion')
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,3)
hold on
plot(C_t, C_sat.q_eci(:,1))
plot(C_t, C_sat.q_eci(:,2))
plot(C_t, C_sat.q_eci(:,3))
plot(C_t, C_sat.q_eci(:,4))
hold off
grid on
title('Quaternion from Body to ECI')
xlabel('Time [sec]')
ylabel('Quaternion')
```

```
legend('\epsilon_1','\epsilon_2','\epsilon_3','\eta')

subplot(3,3,4)
hold on
plot(C_t, C_sat.eul_lvlh(:,1))
plot(C_t, C_sat.eul_lvlh(:,2))
plot(C_t, C_sat.eul_lvlh(:,3))
hold off
grid on
title('Euler Angles from Body to LVLH')
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
legend('\phi', '\theta', '\psi')

subplot(3,3,5)
hold on
plot(C_t, C_orbit.eul_lvlh(:,1))
plot(C_t, C_orbit.eul_lvlh(:,2))
plot(C_t, C_orbit.eul_lvlh(:,3))
hold off
grid on
title('Euler Angles from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
legend('\phi', '\theta', '\psi')

subplot(3,3,6)
hold on
plot(C_t, C_sat.eul_eci(:,1))
plot(C_t, C_sat.eul_eci(:,2))
plot(C_t, C_sat.eul_eci(:,3))
hold off
grid on
title('Euler Angles from Body to ECI')
xlabel('Time [sec]')
ylabel('Euler Angle [rad]')
legend('\phi', '\theta', '\psi')


subplot(3,3,7)
hold on
plot(C_t, C_sat.w_lvlh(:,1))
plot(C_t, C_sat.w_lvlh(:,2))
plot(C_t, C_sat.w_lvlh(:,3))
hold off
grid on
title('\omega from Body to LVLH')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

subplot(3,3,8)
hold on
plot(C_t, C_orbit.w_lvlh(:,1))
```

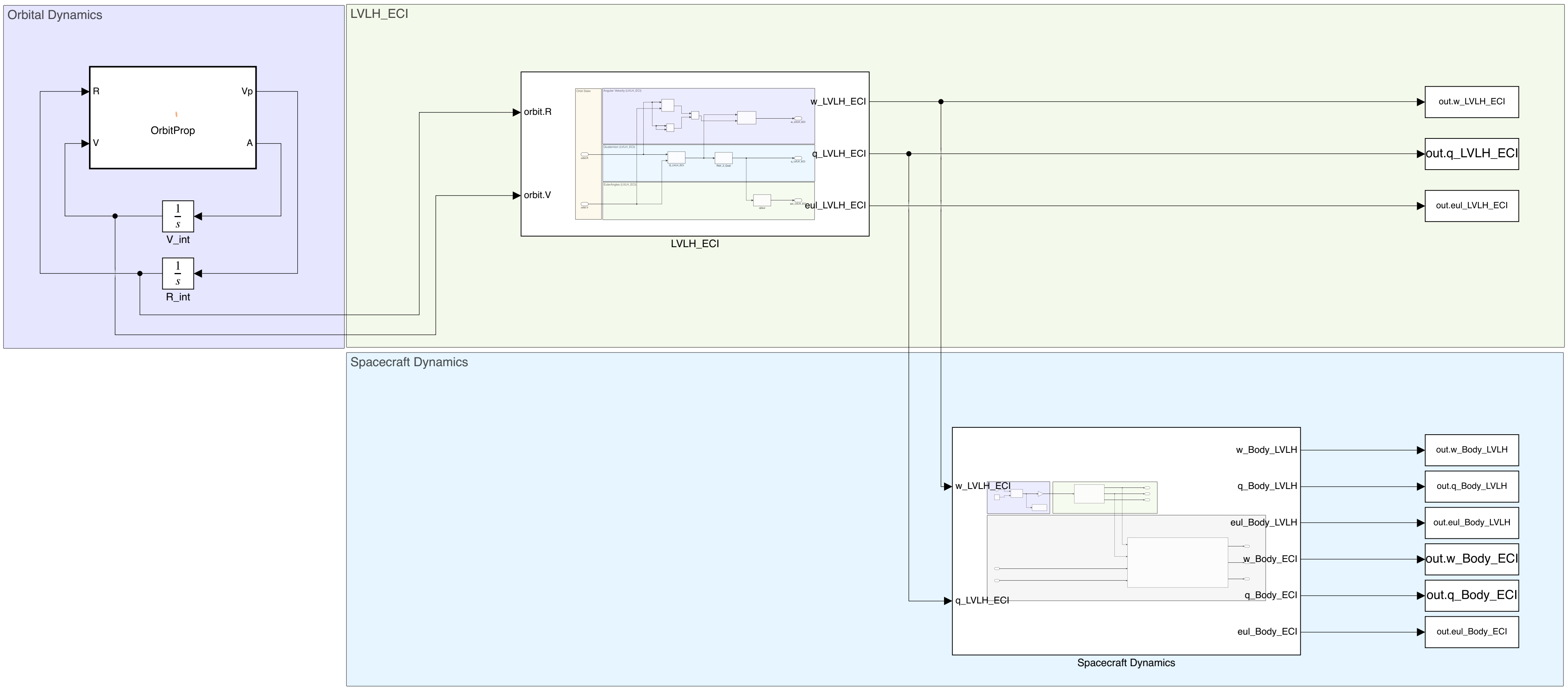
```
plot(C_t, C_orbit.w_lvlh(:,2))
plot(C_t, C_orbit.w_lvlh(:,3))
hold off
grid on
title('\omega from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

subplot(3,3,9)
hold on
plot(C_t, C_sat.w_eci(:,1))
plot(C_t, C_sat.w_eci(:,2))
plot(C_t, C_sat.w_eci(:,3))
hold off
grid on
title('\omega from LVLH to ECI')
xlabel('Time [sec]')
ylabel('Rate [rad/s]')
legend('\omega_x', '\omega_y', '\omega_z')

sgtitle('T_{C} = -K_psign(n_e)\epsilon_e - K_d(1+\epsilon_e^T
\epsilon_e)\omega')
```

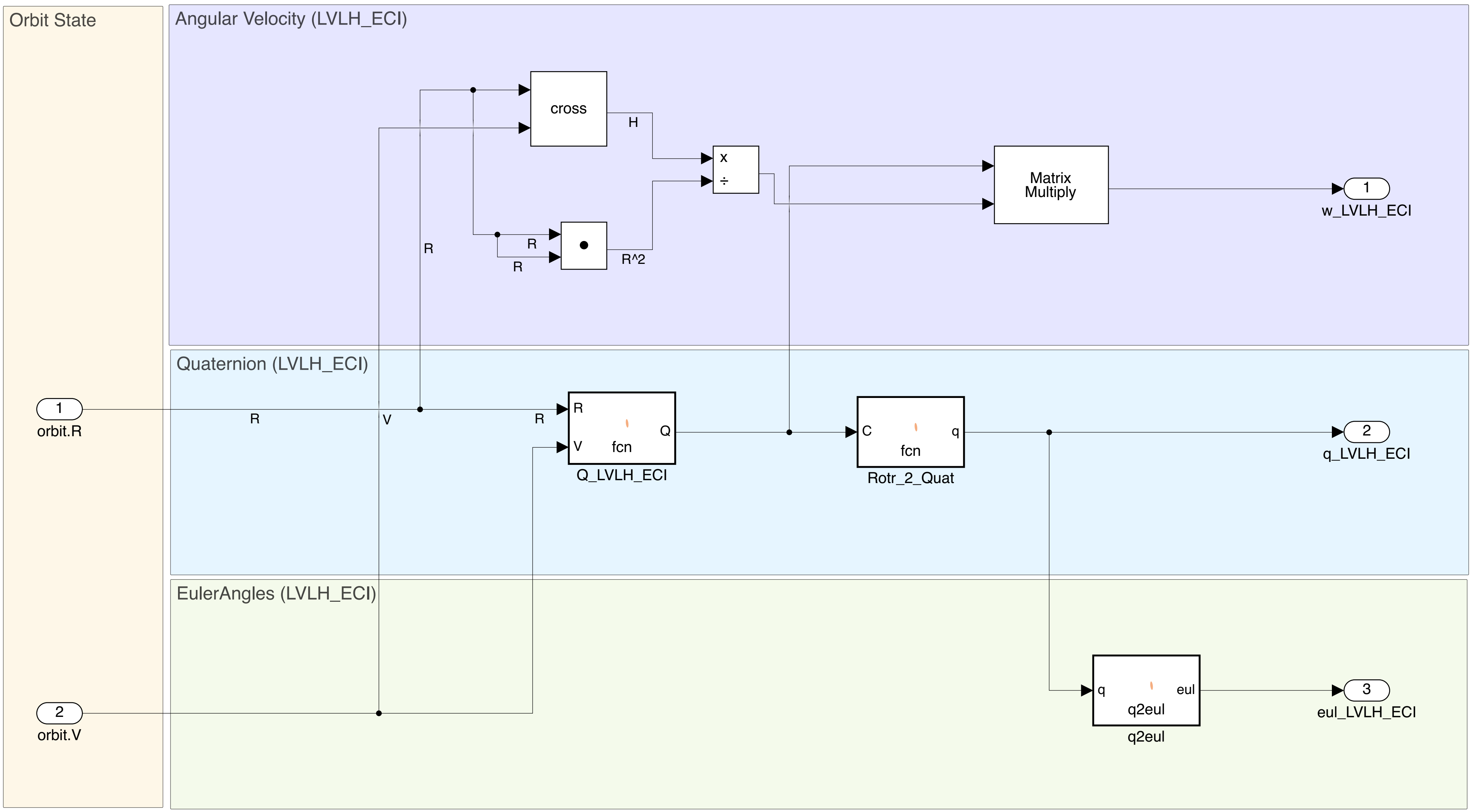
plot misc

```
for i = 1:length(A_command_torque)
    A_tq(i) = norm(A_command_torque(i,:));
end

for i = 1:length(B_command_torque)
    B_tq(i) = norm(B_command_torque(i,:));
end

for i = 1:length(C_command_torque)
    C_tq(i) = norm(C_command_torque(i,:));
end

figure
hold on
plot(A_t, A_tq)
plot(B_t, B_tq)
plot(C_t, C_tq)
hold off
title('Torque Exerted')
xlabel('Time [sec]')
ylabel('Norm of Torque [Nm]')
legend('Control Law A', 'Control Law B', 'Control Law C')

fprintf('~~~~~~~~~~~~~~~~~~~~\n')
fprintf('4. The satellite can maintain pointing with the disturbance torque.
 Setting the disturbance torque to 0 results in a similar result with the
 satellite reaching steady state in less time.\n')
```

```
fprintf('~~~~~~~~~~~~~~~~~~~~~\n')
fprintf('5. The max torque exerted in the first control law is %.2f Nm.
 With a height of %.2f m, the satellite must exert %.2f N of thrust. No EP
 is currently capable of producing that much thrust.\n', max(A_tq), sat.h,
 max(A_tq)/(sat.h/2));

fprintf('~~~~~~~~~~~~~~~~~~~~~\n')
fprintf('6. The first control law which only considers Kd and w are an
 approximation of the other control laws but overall are similar. The second
 two control laws are identical as expected. All reach steady state in a
 similar time however the approximation (Control Law A) requires much more
 thrust than its counterparts over the course of the mission.\n');

fprintf('~~~~~~~~~~~~~~~~~~~~~\n')
function eul = quat_eul(q)

    n = q(4);
    e = q(1:3);

    q = [n, e(1), e(2), e(3)];

    phi = atan2(2*(q(1)*q(2) + q(3)*q(4)), 1 - 2*(q(2)^2 + q(3)^2));
    theta = asin(2*(q(1)*q(3) - q(4)*q(2)));
    psi = atan2(2*(q(1)*q(4) + q(2)*q(3)), 1-2*(q(3)^2 + q(4)^2));

    eul = [phi; theta; psi];

end
```

Published with MATLAB® R2022b


Orbital Dynamics
R
V
Vp
A
OrbitProp
1/s
V_int
1/s
R_int
LVLH_ECI
orbit.R
orbit.V
w_LVLH_ECI
q_LVLH_ECI
eul_LVLH_ECI
LVLH_ECI
out.w_LVLH_ECI
out.q_LVLH_ECI
out.eul_LVLH_ECI
Spacecraft Dynamics
w_LVLH_ECI
q_LVLH_ECI
w_Body_LVLH
q_Body_LVLH
eul_Body_LVLH
w_Body_ECI
q_Body_ECI
eul_Body_ECI
Spacecraft Dynamics
out.w_Body_LVLH
out.q_Body_LVLH
out.eul_Body_LVLH
out.w_Body_ECI
out.q_Body_ECI
out.eul_Body_ECI



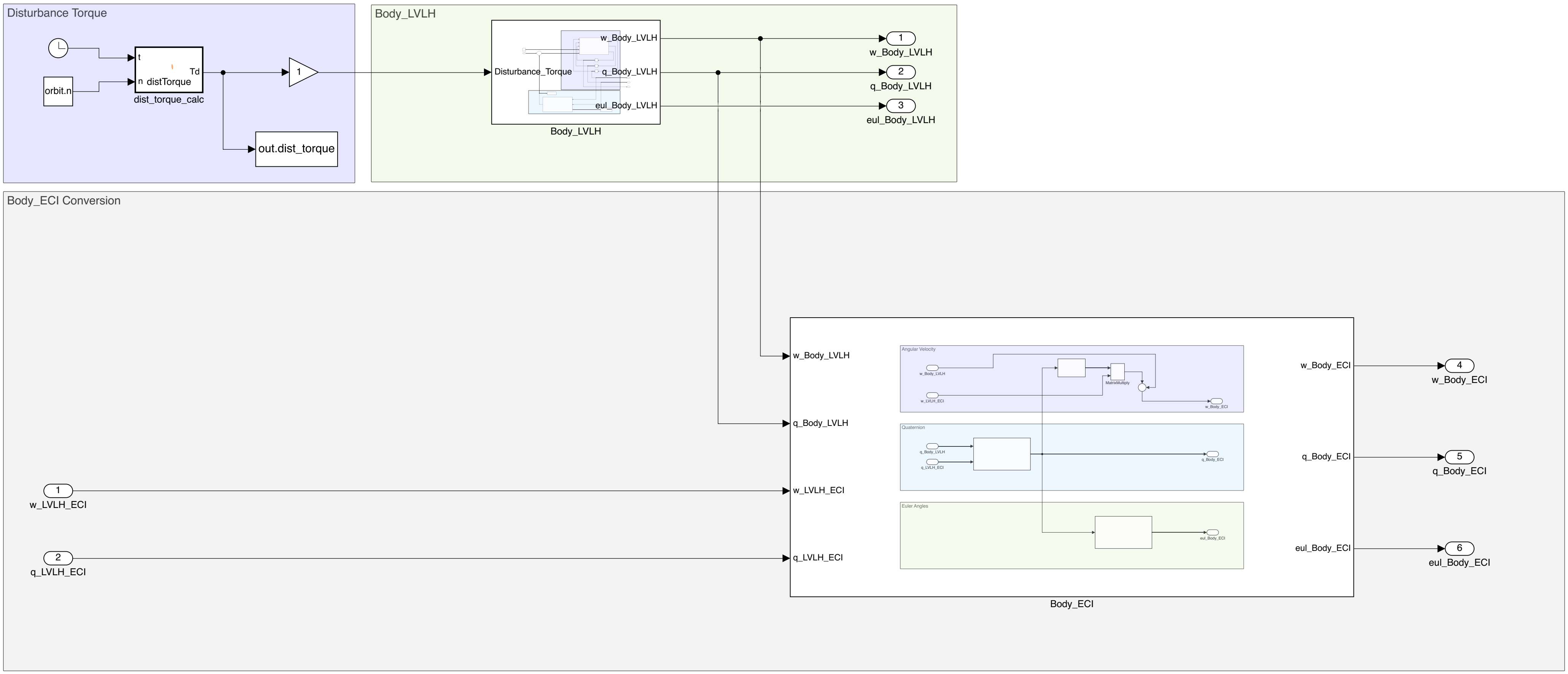
Orbit State
Angular Velocity (LVLH_ECI)
Quaternion (LVLH_ECI)
EulerAngles (LVLH_ECI)
1
orbit.R
2
orbit.V
cross
H
x
÷
R
R
R
R^2
Matrix Multiply
1
w_LVLH_ECI
R
V
R
R
V
Q
fcn
Q_LVLH_ECI
C
q
fcn
Rotr_2_Quat
2
q_LVLH_ECI
q
eul
q2eul
q2eul
3
eul_LVLH_ECI

```
function Q = fcn(R,V)


z = -1*R / norm(R);
y = -1 * cross(R,V) / norm(cross(R,V));

x = cross(y,z);

Q = [x,y,z]';
```

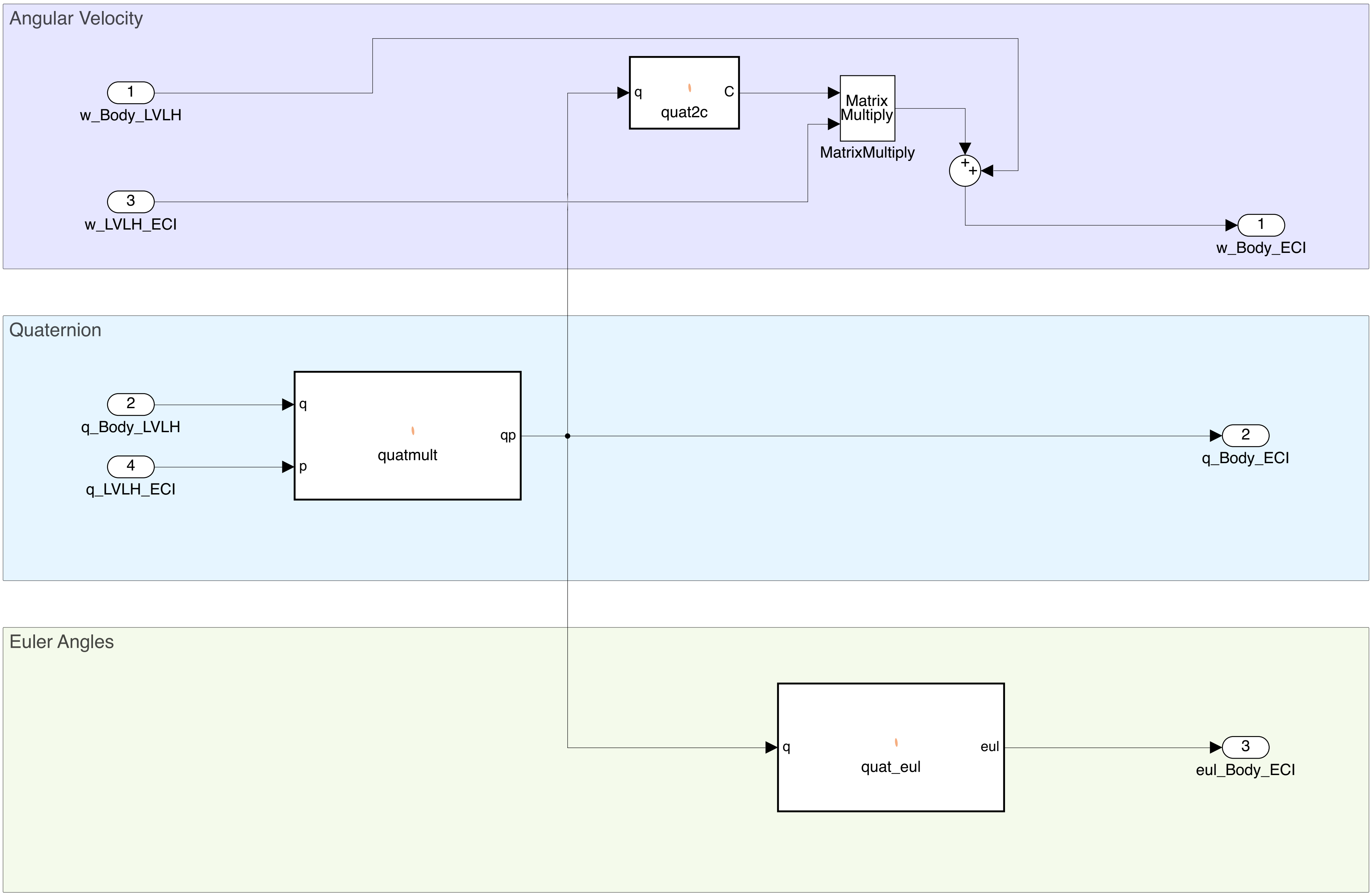
```
function q = fcn(C)

        e = zeros(3,1);
        n = 0.5 * sqrt(1 + trace(C));
        e(1) = 0.25 * (C(2,3) - C(3,2))/n;
        e(2) = 0.25 * (C(3,1) - C(1,3))/n;
        e(3) = 0.25 * (C(1,2) - C(2,1))/n;

        q = [e;n];
```

```
function eul = q2eul(q)

    n = q(4);
    e = q(1:3);

    q = [n, e(1), e(2), e(3)];

    phi = atan2(2*(q(1)*q(2) + q(3)*q(4)), 1 - 2*(q(2)^2 + q(3)^2));
    theta = asin(2*(q(1)*q(3) - q(4)*q(2)));
    psi = atan2(2*(q(1)*q(4) + q(2)*q(3)), 1-2*(q(3)^2 + q(4)^2));

    eul = [phi; theta; psi];
```

```
function [Vp,A] = OrbitProp(R,V)

    mu = 398600;

    rad = norm(R);
    rx = R(1);
    ry = R(2);
    rz = R(3);

    ax = -mu*rx/rad^3;
    ay = -mu*ry/rad^3;
    az = -mu*rz/rad^3;

    Vp = [V(1);V(2);V(3)];
    A = [ax;ay;az];

end
```



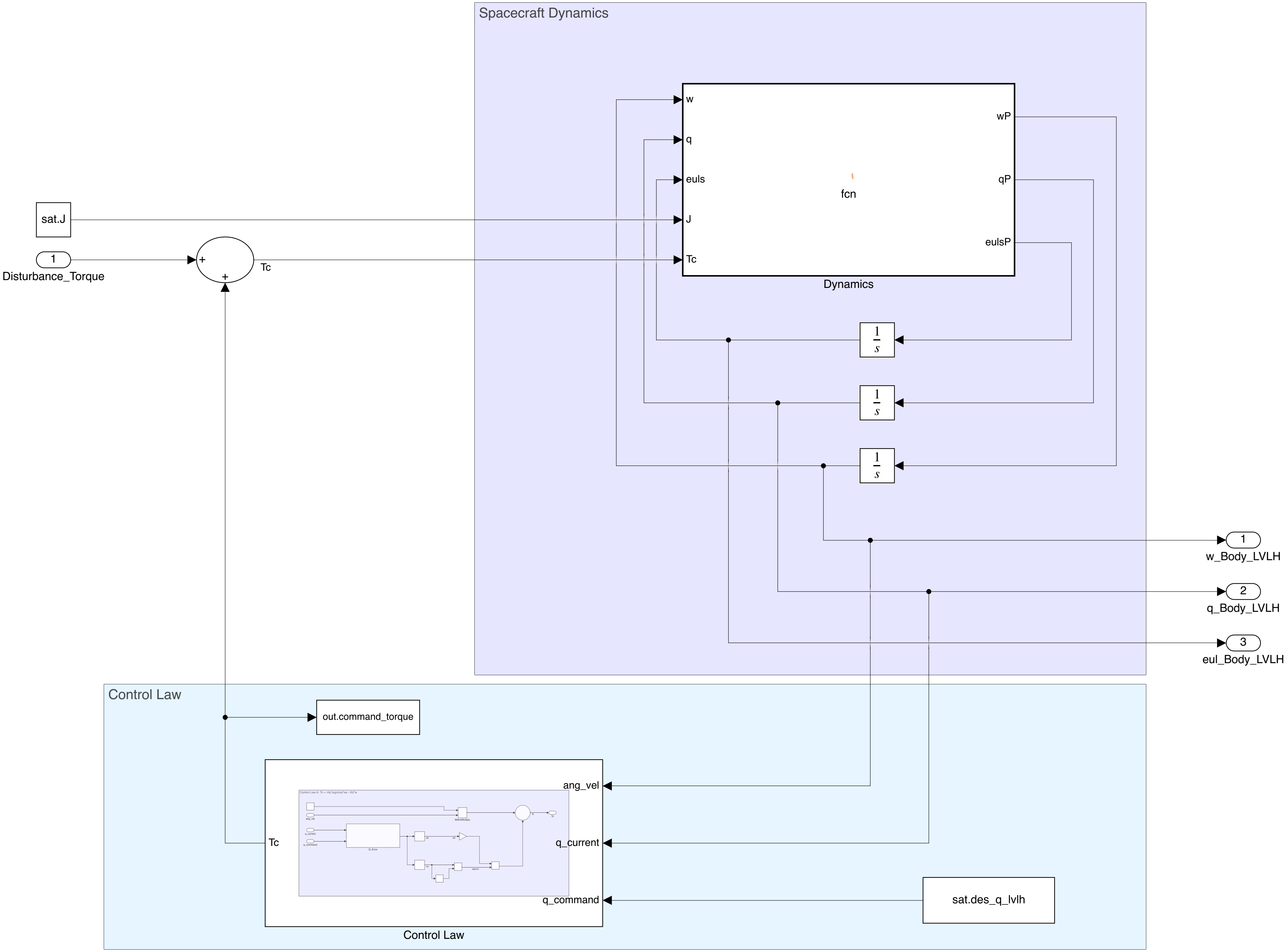
Disturbance Torque
t
n
distTorque
Td
orbit.n
dist_torque_calc
1
out.dist_torque
Body_LVLH
Disturbance_Torque
w_Body_LVLH
q_Body_LVLH
eul_Body_LVLH
Body_LVLH
1
w_Body_LVLH
2
q_Body_LVLH
3
eul_Body_LVLH
Body_ECI Conversion
w_Body_LVLH
q_Body_LVLH
w_LVLH_ECI
q_LVLH_ECI
w_Body_ECI
q_Body_ECI
eul_Body_ECI
Body_ECI
1
w_LVLH_ECI
2
q_LVLH_ECI
4
w_Body_ECI
5
q_Body_ECI
6
eul_Body_ECI


Angular Velocity
1
w_Body_LVLH
q
quat2c
C
Matrix Multiply
MatrixMultiply
3
w_LVLH_ECI
1
w_Body_ECI
Quaternion
2
q_Body_LVLH
q
quatmult
qp
p
4
q_LVLH_ECI
2
q_Body_ECI
Euler Angles
q
quat_eul
eul
3
eul_Body_ECI

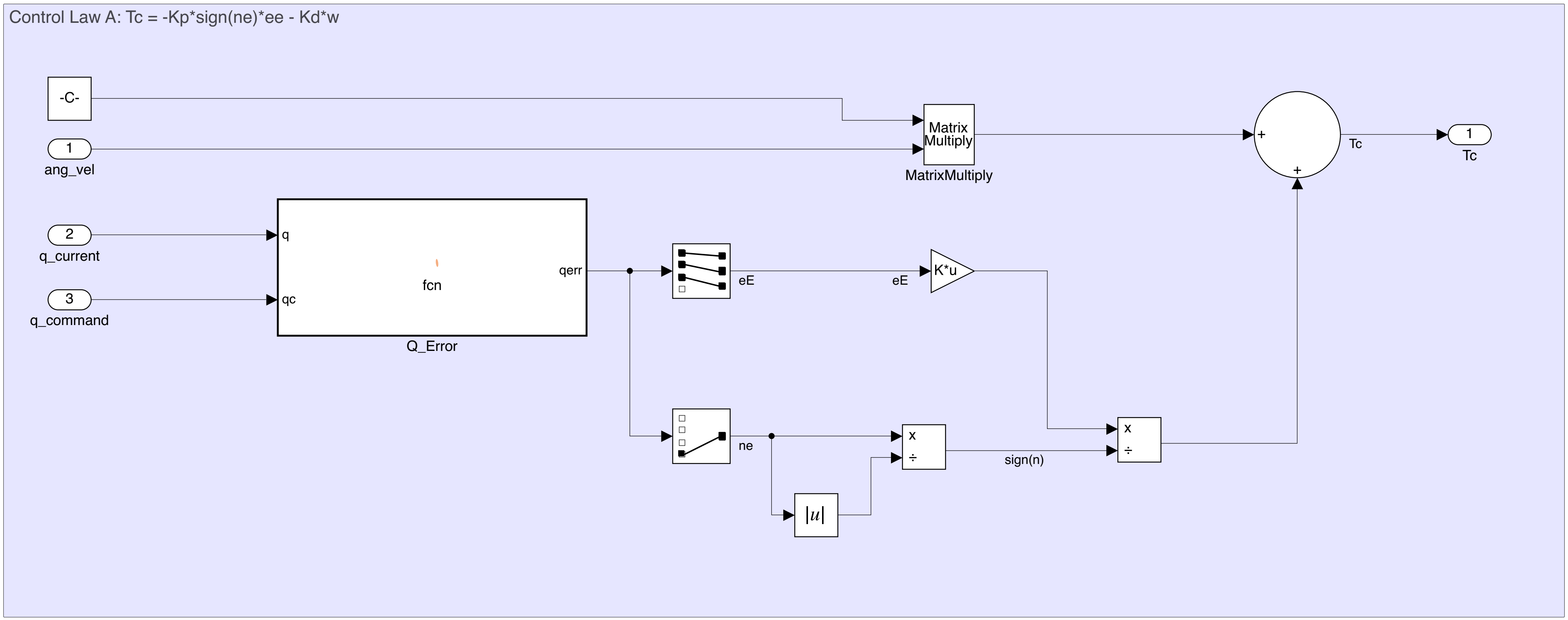
```
function qp = quatmult(q, p)

        function wx = skewSymmetric(w)
            wx = [0, -1*w(3), w(2);
                 w(3), 0, -1*w(1);
                 -1*w(2), w(1), 0];
        end

        qn = q(4);
        qe = q(1:3);

        pn = p(4);
        pe = p(1:3);

        n = pn * qn - pe'*qe;
        e = pn * qe + qn*pe + skewSymmetric(pe)*qe;

        qp = [e(1);e(2);e(3);n];

end
```

```
function eul = quat_eul(q)

    n = q(4);
    ex = q(1);
    ey = q(2);
    ez = q(3);

    a = 2*(n*ey - ez*ex);
    if a > 1
        a = 1;
    elseif a < -1
        a = -1;
    end

    phi = atan2(2*(n*ex + ey*ez), 1 - 2*(ex^2 + ey^2));
    theta = asin(a);
    psi = atan2(2*(n*ez + ex*ey), 1 - 2*(ey^2 + ez^2));

    eul = [phi;theta;psi];

end
```

```
function C = quat2c(q)

C = quat2rotm([q(4);q(1:3)]');
```



Spacecraft Dynamics
w
q
euls
J
Tc
wP
qP
eulsP
fcn
Dynamics
sat.J
1
Disturbance_Torque
+
+
Tc
1/s
1/s
1/s
1
w_Body_LVLH
2
q_Body_LVLH
3
eul_Body_LVLH
Control Law
out.command_torque
ang_vel
q_current
q_command
Tc
Control Law
sat.des_q_lvlh

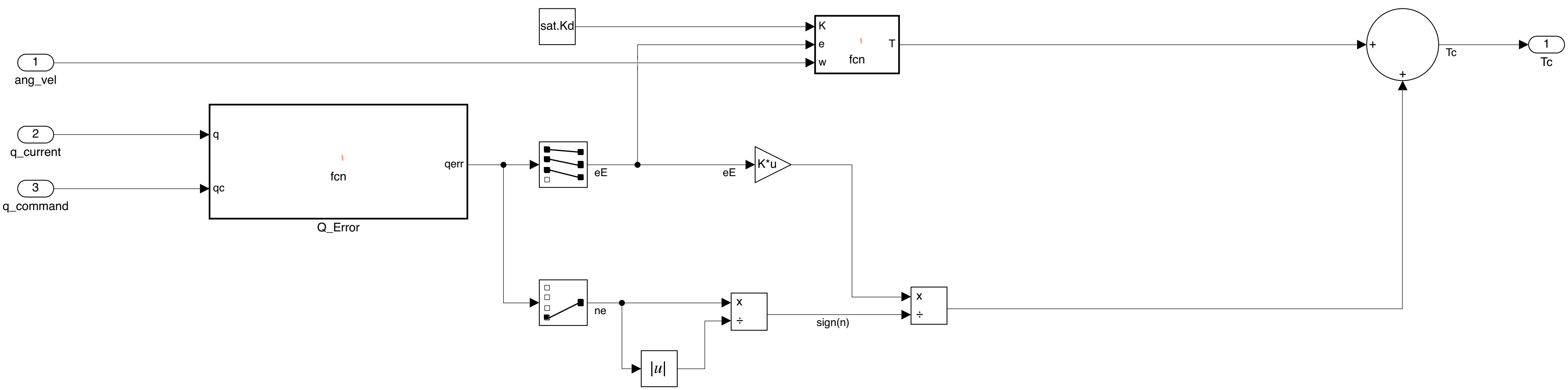


Control Law A: Tc = -Kp*sign(ne)*ee - Kd*w
-C-
1
ang_vel
2
q_current
3
q_command
q
qc
fcn
qerr
Q_Error
Matrix Multiply
MatrixMultiply
eE
eE
K*u
ne
x
÷
|u|
sign(n)
x
÷
+
+
Tc
1
Tc

```
function qerr = fcn(q, qc)

%     function wx = skewSymmetric(w)
%         wx = [0, -1*w(3), w(2);
%               w(3), 0, -1*w(1);
%               -1*w(2), w(1), 0];
%     end

    function qp = quatmult(q, p)

        function wx = skewSymmetric(w)
            wx = [0, -1*w(3), w(2);
                  w(3), 0, -1*w(1);
                  -1*w(2), w(1), 0];
        end

        qn = q(4);
        qe = q(1:3);

        pn = p(4);
        pe = p(1:3);

        n = pn * qn - pe'*qe;
        e = pn * qe + qn*pe + skewSymmetric(pe)*qe;

        qp = [e(1);e(2);e(3);n];

    end

qc(1:3) = -1*qc(1:3);
qerr = quatmult(qc, q);

end
```

```
function [wP, qP, eulsP] = fcn(w, q, euls, J, Tc)

    function wx = skewSymmetric(w)
        wx = [0, -1*w(3), w(2);
             w(3), 0, -1*w(1);
             -1*w(2), w(1), 0];
    end

    T = [Tc(1);Tc(2);Tc(3)];

    wP = J\(T - skewSymmetric(w)*J*w);

    e = q(1:3);
    n = q(4);
    eP = 0.5*(n*eye(3) + skewSymmetric(e))*w;
    nP = -0.5*e'*w;
    qP = [eP;nP];

    phi = euls(1);
    theta = euls(2);
    psi = euls(3);
    eulsP = 1/(cos(theta)) * [cos(theta), sin(phi)*sin(theta), cos(phi)*sin(theta);
                            0, cos(phi)*cos(theta), -1*sin(phi)*cos(theta);
                            0, sin(phi), cos(phi)] * w;


end
```

```
function Td  = distTorque(t, n)

T = sin(3*n*t)*[0;0.5;0]*10^(-3);
Td = T;
end
```



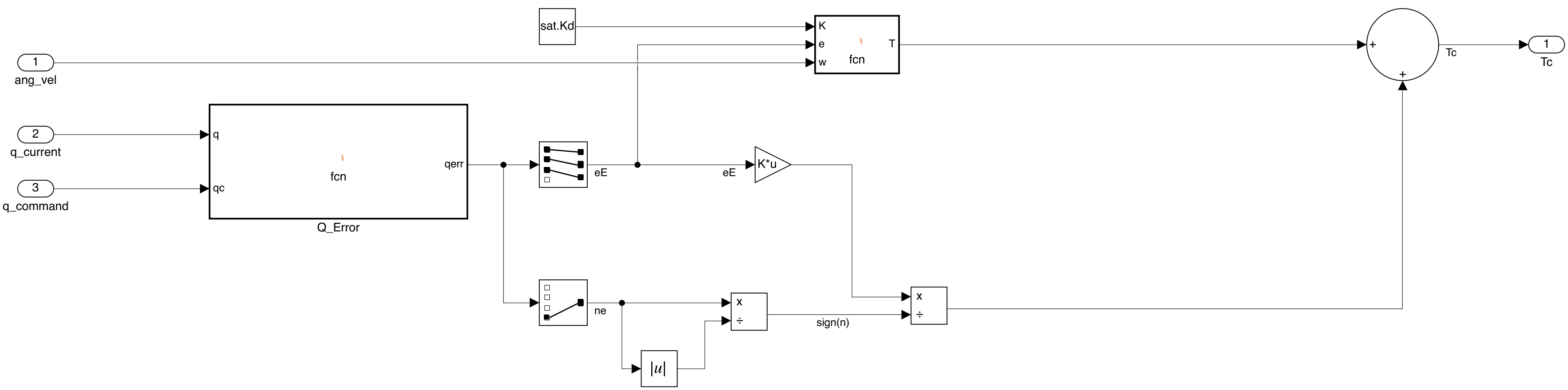
sat.Kd
K
e
w
fcn
T
1
ang_vel
2
q_current
3
q_command
q
qc
fcn
qerr
Q_Error
eE
eE
K*u
ne
x
÷
|u|
sign(n)
x
÷
+
+
Tc
1
Tc

```
function T = fcn(K, e, w)

T = -1*K*(1 + e'*e)*w;
```

```
function qerr = fcn(q, qc)

%     function wx = skewSymmetric(w)
%         wx = [0, -1*w(3), w(2);
%               w(3), 0, -1*w(1);
%               -1*w(2), w(1), 0];
%     end

    function qp = quatmult(q, p)

        function wx = skewSymmetric(w)
            wx = [0, -1*w(3), w(2);
                  w(3), 0, -1*w(1);
                  -1*w(2), w(1), 0];
        end

        qn = q(4);
        qe = q(1:3);

        pn = p(4);
        pe = p(1:3);

        n = pn * qn - pe'*qe;
        e = pn * qe + qn*pe + skewSymmetric(pe)*qe;

        qp = [e(1);e(2);e(3);n];

    end

qc(1:3) = -1*qc(1:3);
qerr = quatmult(qc, q);

end
```

```
function [wP, qP, eulsP] = fcn(w, q, euls, J, Tc)

    function wx = skewSymmetric(w)
        wx = [0, -1*w(3), w(2);
             w(3), 0, -1*w(1);
             -1*w(2), w(1), 0];
    end

    T = [Tc(1);Tc(2);Tc(3)];

    wP = J\(T - skewSymmetric(w)*J*w);

    e = q(1:3);
    n = q(4);
    eP = 0.5*(n*eye(3) + skewSymmetric(e))*w;
    nP = -0.5*e'*w;
    qP = [eP;nP];

    phi = euls(1);
    theta = euls(2);
    psi = euls(3);
    eulsP = 1/(cos(theta)) * [cos(theta), sin(phi)*sin(theta), cos(phi)*sin(theta);
                                0, cos(phi)*cos(theta), -1*sin(phi)*cos(theta);
                                0, sin(phi), cos(phi)] * w;


end
```

```
function Td  = distTorque(t, n)

T = sin(3*n*t)*[0;0.5;0]*10^(-3);
Td = T;
end
```


1
ang_vel
2
q_current
3
q_command
q
qc
fcn
qerr
Q_Error
sat.Kd
K
e
w
fcn
T
eE
eE
K*u
ne
x
÷
|u|
sign(n)
x
÷
+
+
Tc
1
Tc

```
function T = fcn(K, e, w)

T = -1*K*(1 + e'*e)*w;
```

```
function qerr = fcn(q, qc)

%     function wx = skewSymmetric(w)
%         wx = [0, -1*w(3), w(2);
%               w(3), 0, -1*w(1);
%               -1*w(2), w(1), 0];
%     end

    function qp = quatmult(q, p)

        function wx = skewSymmetric(w)
            wx = [0, -1*w(3), w(2);
                  w(3), 0, -1*w(1);
                  -1*w(2), w(1), 0];
        end

        qn = q(4);
        qe = q(1:3);

        pn = p(4);
        pe = p(1:3);

        n = pn * qn - pe'*qe;
        e = pn * qe + qn*pe + skewSymmetric(pe)*qe;

        qp = [e(1);e(2);e(3);n];

    end

qc(1:3) = -1*qc(1:3);
qerr = quatmult(qc, q);

end
```

```
function [wP, qP, eulsP] = fcn(w, q, euls, J, Tc)

    function wx = skewSymmetric(w)
        wx = [0, -1*w(3), w(2);
              w(3), 0, -1*w(1);
              -1*w(2), w(1), 0];
    end

    T = [Tc(1);Tc(2);Tc(3)];

    wP = J\(T - skewSymmetric(w)*J*w);

    e = q(1:3);
    n = q(4);
    eP = 0.5*(n*eye(3) + skewSymmetric(e))*w;
    nP = -0.5*e'*w;
    qP = [eP;nP];

    phi = euls(1);
    theta = euls(2);
    psi = euls(3);
    eulsP = 1/(cos(theta)) * [cos(theta), sin(phi)*sin(theta), cos(phi)*sin(theta);
                                0, cos(phi)*cos(theta), -1*sin(phi)*cos(theta);
                                0, sin(phi), cos(phi)] * w;


end
```

```
function Td  = distTorque(t, n)

T = sin(3*n*t)*[0;0.5;0]*10^(-3);
Td = T;
end
```