Contents


```
% In the name of Allah the beneficent the merciful
% Code by NVE Team, Sharif University, Tehran, Iran
% Date : 1395/12/24
```

Cleaning

```
clc; clear; close all;
```

Variables

```
syms t
syms X Y Z
syms Xd Yd Zd
syms Xdd Ydd Zdd
syms wx wy wz
syms wxd wyd wzd
syms gamma beta alpha
syms gammad betad alphad
syms gammadd betadd alphadd
syms ax ay az
```

Rotation Matrix

```
R1 = [cos(alpha) sin(alpha) 0
     -sin(alpha) cos(alpha) 0
      0            0        1];

R2 = [cos(beta) 0 -sin(beta)
      0         1          0
      sin(beta) 0  cos(beta)];

R3 = [1           0          0
      0  cos(gamma) sin(gamma)
      0 -sin(gamma) cos(gamma)];

R = R3*R2*R1; % R*X_global = X_local, A = R.'(TAO 2000);
```

Mass Matrix

```
m = 150; % mass of the engine or powertrain. M = [m*eye(3) zeros(3) ; zeros(3) I];
I = [5.82 -0.82 0.19 ; -0.82 3.41 -0.21 ; 0.19 -0.21 5.50];     % Inertia matrix
```

Stiffness Matrix

Mount Positions r = [ax ay az]';

```
r1_xyz = [-93 417.4 172.0]'*1e-3;
r2_xyz = [-56 -433.1 116.5]'*1e-3;
r3_xyz = [262 36.9 -68]'*1e-3;

r1_XYZ = R.'*r1_xyz;
r2_XYZ = R.'*r2_xyz;
r3_XYZ = R.'*r3_xyz;

% Cross product Matrix
B = [0 -az ay; az 0 -ax; -ay ax 0];
B_1 = double(subs(B,[ax ay az],r1_xyz'));
B_2 = double(subs(B,[ax ay az],r2_xyz'));
B_3 = double(subs(B,[ax ay az],r3_xyz'));

% Mount Inclinations
o_1 = [180 10 180]'*pi/180;
o_2 = [0 0 0]'*pi/180;
o_3 = [180 0 0]'*pi/180;

A_1 = double(subs(R,[alpha beta gamma],o_1.'));
A_2 = double(subs(R,[alpha beta gamma],o_2.'));
A_3 = double(subs(R,[alpha beta gamma],o_3.'));

omega_body = [wx; wy; wz];
% omega_body = alphad K  + betad j1 + gammad k2
% omega_body = alphad k1 + betad j2 + gammad k
omega_xyz = alphad*[-sin(beta); cos(beta)*sin(gamma); cos(beta)*cos(gamma)] +
betad*[0; cos(gamma); -sin(gamma)] + gammad*[1; 0; 0];

omega_rpy = [gammad; betad; alphad];
% omega_xyz = Kin_Mat * [gammad; betad; alphad];
Kin_Mat = jacobian (omega_xyz,omega_rpy);
omega_XYZ = R.'*omega_xyz;

U_G = [X; Y; Z];
VG = [Xd; Yd; Zd];

V_1_XYZ = VG + R.'*cross(omega_body,r1_xyz);
V_2_XYZ = VG + R.'*cross(omega_body,r2_xyz);
```

```
V_3_XYZ = VG + R.'*cross(omega_body,r3_xyz);

Rot = [gamma; beta; alpha];
Rot_xyz = Kin_Mat * Rot;
Rot_XYZ = R.' * Rot_xyz;

U_1 = U_G + R.'*cross(Rot_xyz,r1_xyz);
U_2 = U_G + R.'*cross(Rot_xyz,r2_xyz);
U_3 = U_G + R.'*cross(Rot_xyz,r3_xyz);

% Mount Stiffness
k_l_1 = diag([94.6 111.3 92.4])*1e3;
k_l_2 = diag([72.8 72.8 84.4])*1e3;
k_l_3 = diag([203.9 41.7 82.0])*1e3;
k_1 = A_1*k_l_1*A_1';
k_2 = A_2*k_l_2*A_2';
k_3 = A_3*k_l_3*A_3';

F_K_1 =-k_1*U_1;
F_K_2 =-k_2*U_2;
F_K_3 =-k_3*U_3;
F_K = F_K_1 + F_K_2 + F_K_3;

M_K_1 = cross(r1_xyz,F_K_1);
M_K_2 = cross(r2_xyz,F_K_2);
M_K_3 = cross(r3_xyz,F_K_3);
M_K = M_K_1 + M_K_2 + M_K_3;
```

Damping Matrix

Mount Damping Coefficients

```
c_l_1 = diag([94.6 111.3 92.4]);
c_l_2 = diag([72.8 72.8 84.4]);
c_l_3 = diag([203.9 41.7 82.0]);
c_1 = A_1*c_l_1*A_1';
c_2 = A_2*c_l_2*A_2';
c_3 = A_3*c_l_3*A_3';

F_C_1 =-c_1*V_1_XYZ;
F_C_2 =-c_2*V_2_XYZ;
F_C_3 =-c_3*V_3_XYZ;
F_C = F_C_1 + F_C_2 + F_C_3;

M_C_1 = cross(r1_xyz,F_C_1);
M_C_2 = cross(r2_xyz,F_C_2);
```

```
M_C_3 = cross(r3_xyz,F_C_3);
M_C = M_C_1 + M_C_2 + M_C_3;
```

Final Equations

```
r = [X; Y; Z];
rd = [Xd; Yd; Zd];
rdd = [Xdd; Ydd; Zdd];

Rot_acc_xyz = [wxd; wyd; wzd];
I_KM = inv(Kin_Mat);

H_xyz = I*omega_body;
dH_xyz_dt = diff(H_xyz,t) + jacobian(H_xyz,Rot.')*I_KM*omega_body +
jacobian(H_xyz,omega_body)*Rot_acc_xyz + cross(omega_body,H_xyz);
dH_XYZ_dt = R.'* dH_xyz_dt;

M1 = m*eye(3);
M = [M1 zeros(3); zeros(3) I];
Eq_Force = M1 * rdd - F_C - F_K;
% Bias_1 = -F_C - F_K;
Bias_1 =  - F_K;
% Bias_2 = -M_C - M_K + cross(omega_body, H_xyz);
Bias_2 = - M_K + cross(omega_body, H_xyz);
B = [Bias_1; Bias_2];
B = simplify(B);
```