**Assignment #3 Yufeng Sun**

3-1-3 Euler angles: (radians) [-0.5236 0.6981 0.3491]

direct cosine matrix:

0.9448 -0.2429 0.2198

0.0637 0.7944 0.6040

-0.3214 -0.5567 0.7660

1.a: principle rotation elements: (phi or phi-2\*pi, degree), axis)

choose phi in [0,180] for less maneuver: 41.1813 (or -318.8187)

[0.8814 -0.4110 -0.2329]

1.b: Euler parameter vector: (beta0, beta1, beta2, beta3)

(0.9361 0.3100 -0.1445 -0.0819)

1.c: Rodrigues parameter vector: (rough1, rough2, rough3)

(0.3311 -0.1544 -0.0875)

1.d: modified Rodrigues parameter vector: (sigma1, sigma2, sigma3)

(0.3311 -0.1544 -0.0875)

%--------------------------------------------------------------------------

% AEEM6036 Assignment 3-1

% Description:

% give a 3-1-3 euler angles, find

% 1) principle rotation elements

% 2) euler parameters

% 3) classical rodrigues parameters

% 4) modified rodrigues parameters

% Author: Yufeng Sun

% Date: Oct 10, 2020

%--------------------------------------------------------------------------

eas = (pi/180)\*[-30 40 20];

disp('3-1-3 euler angles: (radians)');

disp(eas);

dcm = DCM313(eas);

disp('direct cosine matrix:');

disp(dcm);

[phi, axis] = DCM2PRE(dcm);

disp('principle rotation elements: (phi or phi-2\*pi, degree), axis)');

disp('choose the one in [0,180] for less maneuver');

disp(phi\*180/pi);

disp(phi\*180/pi-360);

disp(axis);

epv = PRE2EPV(phi,axis);

disp('euler parameter vector: (beta0,beta1,beta2,beta3)');

disp(epv);

rps = EPV2RPS(epv);

disp('rodrigues parameter vector: (rough1,rough2,rough3)');

disp(rps);

mrps = RPS2MRPS(epv);

disp('modified rodrigues parameter vector: (sigma1,sigma2,sigma3)');

disp(rps);

% modified rodrigues parameters from euler parameters

function mrps = RPS2MRPS(epv)

mrps = zeros(3,1);

mrps(:,1) = epv(2:4,1)/(1+epv(1,1));

end

% rodrigues parameters from euler parameter

function rps = EPV2RPS(epv)

rps = zeros(3,1);

rps(:,1) = epv(2:4,1)/epv(1,1);

end

% euler parameter vector from principle rotation elements

function epv = PRE2EPV(phi,axis)

epv = zeros(4,1);

epv(1,1) = cos(phi/2);

epv(2:4,1) = sin(phi/2)\*axis;

end

% priciple rotation elements from direct cosine angles

function [phi, axis] = DCM2PRE(dcm)

phi = acos(0.5\*(trace(dcm)-1));

e1 = dcm(2,3)-dcm(3,2);

e2 = dcm(3,1)-dcm(1,3);

e3 = dcm(1,2)-dcm(2,1);

axis = (1/(2\*sin(phi)))\*[e1 e2 e3]';

end

% Direct Cosine Matrix

function mat = DCM313(eas)

mat = R3(eas(3))\*R1(eas(2))\*R3(eas(1));

end

% rotation matrix about axis 1

function mat = R1(ea)

mat = [1 0 0;

0 cos(ea) sin(ea);

0 -sin(ea) cos(ea)];

end

% rotation matrix about axis 3

function mat = R3(ea)

mat = [cos(ea) sin(ea) 0;

-sin(ea) cos(ea) 0;

0 0 1];

end

2.

direct cosine matrix of spacecraft in the inertial frame:

-0.2198 -0.2620 -0.9397

-0.7609 -0.5568 0.3333

-0.6105 0.7882 -0.0769

direct cosine matrix of space station in the inertial frame:

-0.4006 -0.8607 -0.3142

-0.5427 0.4992 -0.6755

0.7383 -0.1001 -0.6670

direct cosine matrix of space station in the space station frame:

-0.4006 -0.8607 -0.3142

-0.5427 0.4992 -0.6755

0.7383 -0.1001 -0.6670

Principal rotation elements: (phi (degree), axis)

118.6745, [-0.8577 -0.5131 -0.0319]

%--------------------------------------------------------------------------

% AEEM6036 Assignment 3-2

% Description:

% evaluate attitude of spacecraft relative to space station given the

% transforms of spacecraft and space sation relative to an inertial frame

% Author: Yufeng Sun

% Date: Oct 10, 2020

%--------------------------------------------------------------------------

% DCM of spacecraft in the fixed inertial frame

eas = (pi/180)\*[230 70 103];

C1 = DCM321(eas);

disp('direct cosine matrix of spacecraft in the inertial frame:');

disp(C1);

% DCM of space station in the fixed inertial frame

epv = [0.328474 -0.437966 0.801059 -0.242062];

C2 = EP2DCM(epv);

disp('direct cosine matrix of space station in the inertial frame:');

disp(C2);

% DCM of spacecraft in the space station frame

C = C1\*C2^(-1);

disp('direct cosine matrix of space station in the space station frame:');

disp(C2);

[phi,axis] = PRE(C);

disp('principle rotation elements: (phi (degree), axis)');

disp(phi\*180/pi);

disp(axis);

% priciple rotation elements from direct cosine angles

function [phi, axis] = PRE(dcm)

phi = acos(0.5\*(trace(dcm)-1));

e1 = dcm(2,3)-dcm(3,2);

e2 = dcm(3,1)-dcm(1,3);

e3 = dcm(1,2)-dcm(2,1);

axis = (1/(2\*sin(phi)))\*[e1 e2 e3]';

end

% Direct Cosine Matrix from Euler parameter

function mat = EP2DCM(epv)

b0 = epv(1);

b1 = epv(2);

b2 = epv(3);

b3 = epv(4);

mat = [b0^2+b1^2-b2^2-b3^2 2\*(b1\*b2+b0\*b3) 2\*(b1\*b3-b0\*b2);

2\*(b1\*b2-b0\*b3) b0^2-b1^2+b2^2-b3^2 2\*(b2\*b3+b0\*b1);

2\*(b1\*b3+b0\*b2) 2\*(b2\*b3-b0\*b1) b0^2-b1^2-b2^2+b3^2];

end

% Direct Cosine Matrix from euler angles

function mat = DCM321(eas)

mat = R1(eas(3))\*R2(eas(2))\*R3(eas(1));

end

% rotation matrix about axis 1

function mat = R1(ea)

mat = [1 0 0;

0 cos(ea) sin(ea);

0 -sin(ea) cos(ea)];

end

% rotation matrix about axis 2

function mat = R2(ea)

mat = [cos(ea) 0 -sin(ea);

0 1 0;

sin(ea) 0 cos(ea)];

end

% rotation matrix about axis 3

function mat = R3(ea)

mat = [cos(ea) sin(ea) 0;

-sin(ea) cos(ea) 0;

0 0 1];

end

3.a



3.b Euler parameters constraint equals to 1 and is slightly decreasing over simulation time.



3.c



%--------------------------------------------------------------------------

% AEEM6036 Assignment 3-3

% Description:

% Give a initial eular angle of a vehicle and the angular velocity in body

% frame, intergrate the euler parameter over a simulation time 1 minutes

% Author: Yufeng Sun

% Date: Oct 10, 2020

%--------------------------------------------------------------------------

init\_eas = (pi/180)\*[40 30 80]';

[t,eps,epsc,eas] = Simulation(60,init\_eas);

print\_eps(t,eps);

print\_eps\_constraint(t,epsc);

print\_eas(t,(180/pi)\*eas);

%--------------------------------------------------------------------------

% print the euler prameters

function print\_eps(t,eps)

hold on

figure(1)

title('Euler Parameters');

xlabel('time (s)');

ylabel('value');

plot(t,eps(:,1));

plot(t,eps(:,2));

plot(t,eps(:,3));

plot(t,eps(:,4));

legend('\beta\_0','\beta\_1','\beta\_2','\beta\_3');

hold off

end

%--------------------------------------------------------------------------

% print the euler prameters constraint

function print\_eps\_constraint(t,epsc)

hold on

figure(2)

title('Euler Parameters Constraint');

xlabel('time (s)');

ylabel('value');

plot(t,epsc);

hold off

end

%--------------------------------------------------------------------------

% print the euler angles

function print\_eas(t,eas)

hold on

figure(3)

title('Euler Angles');

xlabel('time (s)');

ylabel('angle (degrees)');

plot(t,eas(1,:));

plot(t,eas(2,:));

plot(t,eas(3,:));

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

hold off

end

% simulation with ode45 in time span [0,T] with

% an initial value of euler angles

function [t,eps,epsc,eas] = Simulation(T,init\_eas)

init\_eps = EAS2EPV(init\_eas);

[t,eps] = ode45(@EulerParamKDE,[0,T],init\_eps);

epsc = zeros(1,length(t));

eas = zeros(3,length(t));

for i=1:length(t)

epsc(1,i) = eps(i,1)^2+eps(i,2)^2+eps(i,3)^2+eps(i,4)^2;

eas(1:3,i) = EPS2EAS(eps(i,1:4));

end

end

% compute the euler angle velocities with the

% angular velocity define in body frame

function dydt = EulerParamKDE(t,y)

omega = (pi/180)\*20\*[sin(0.1\*t);0.01;cos(0.1\*t)];

b0 = y(1);

b1 = y(2);

b2 = y(3);

b3 = y(4);

mat = [-b1 -b2 -b3;

b0 -b3 b2;

b3 b0 -b1;

-b2 b1 b0];

dydt = 0.5\*mat\*omega;

end

% Euler Angles from Euler Parameters

function eas = EPS2EAS(epv)

eas = [0;0;0];

dcm = EP2DCM(epv);

eas(1) = atan2(dcm(1,2),dcm(1,1));

eas(2) = asin(dcm(1,3));

eas(3) = atan2(dcm(2,3),dcm(3,3));

end

% Direct Cosine Matrix from Euler parameter

function mat = EP2DCM(epv)

b0 = epv(1);

b1 = epv(2);

b2 = epv(3);

b3 = epv(4);

mat = [b0^2+b1^2-b2^2-b3^2 2\*(b1\*b2+b0\*b3) 2\*(b1\*b3-b0\*b2);

2\*(b1\*b2-b0\*b3) b0^2-b1^2+b2^2-b3^2 2\*(b2\*b3+b0\*b1);

2\*(b1\*b3+b0\*b2) 2\*(b2\*b3-b0\*b1) b0^2-b1^2-b2^2+b3^2];

end

% euler parameters from euler angles

function epv = EAS2EPV(eas)

dcm = DCM321(eas);

[phi, axis] = DCM2PRE(dcm);

epv = PRE2EPV(phi,axis);

end

% euler parameter vector from principle rotation elements

function epv = PRE2EPV(phi,axis)

epv = zeros(4,1);

epv(1,1) = cos(phi/2);

epv(2:4,1) = sin(phi/2)\*axis;

end

% priciple rotation elements from direct cosine angles

function [phi, axis] = DCM2PRE(dcm)

phi = acos(0.5\*(trace(dcm)-1));

e1 = dcm(2,3)-dcm(3,2);

e2 = dcm(3,1)-dcm(1,3);

e3 = dcm(1,2)-dcm(2,1);

axis = (1/(2\*sin(phi)))\*[e1 e2 e3]';

end

% Direct Cosine Matrix from euler angles

function mat = DCM321(eas)

mat = R1(eas(3))\*R2(eas(2))\*R3(eas(1));

end

% rotation matrix about axis 1

function mat = R1(ea)

mat = [1 0 0;

0 cos(ea) sin(ea);

0 -sin(ea) cos(ea)];

end

% rotation matrix about axis 2

function mat = R2(ea)

mat = [cos(ea) 0 -sin(ea);

0 1 0;

sin(ea) 0 cos(ea)];

end

% rotation matrix about axis 3

function mat = R3(ea)

mat = [cos(ea) sin(ea) 0;

-sin(ea) cos(ea) 0;

0 0 1];

end