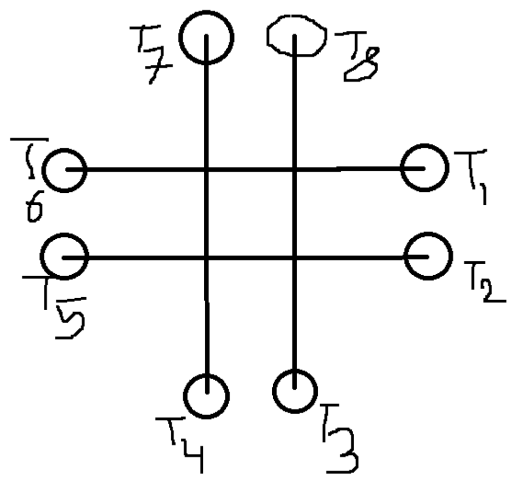
Problem 1:



U1 = -(T1+ T2+T3+T4+T5+T6+T7+T8)

U2 = L/2(T6+T5-T1-T2)

U3 = L/2(T7+T8-T4—T3)

U4 = d(-T1-T2+T3+T4-T5-T6+T7+T8)

U1 =

MATLAB Function

Octacopter ▸ Channel Mixer ▸ MATLAB Function

```
function y = fcn(u)
%#codegen
A = [-1 -1 -1 -1 -1 -1 -1 -1;...
     -1 -1  0  0  1  1  0  0;...
      0  0 -1 -1  0  0  1  1;...
     -1 -1  1  1 -1 -1  1  1];

A_inv = pinv(A)
y = A_inv*u;

```

```
a = [-1 -1 -1 -1 -1 -1 -1 -1;...
     -1 -1  0  0  1  1  0  0;...
      0  0 -1 -1  0  0  1  1;...
     -1 -1  1  1 -1 -1  1  1];


x = pinv(a);

```

Command Window

```
>> x

x =

   -0.1250   -0.2500         0   -0.1250
   -0.1250   -0.2500         0   -0.1250
   -0.1250    0.0000   -0.2500    0.1250
   -0.1250    0.0000   -0.2500    0.1250
   -0.1250    0.2500         0   -0.1250
   -0.1250    0.2500         0   -0.1250
   -0.1250    0.0000    0.2500    0.1250
   -0.1250    0.0000    0.2500    0.1250
```

Problem 2:

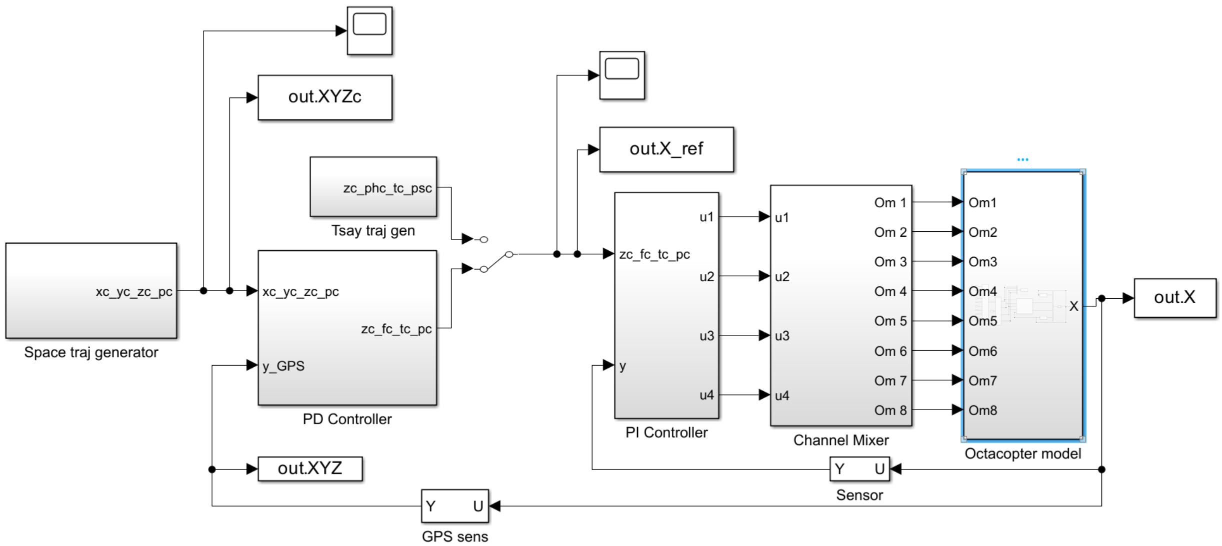


Fig: Full Octocopter Model

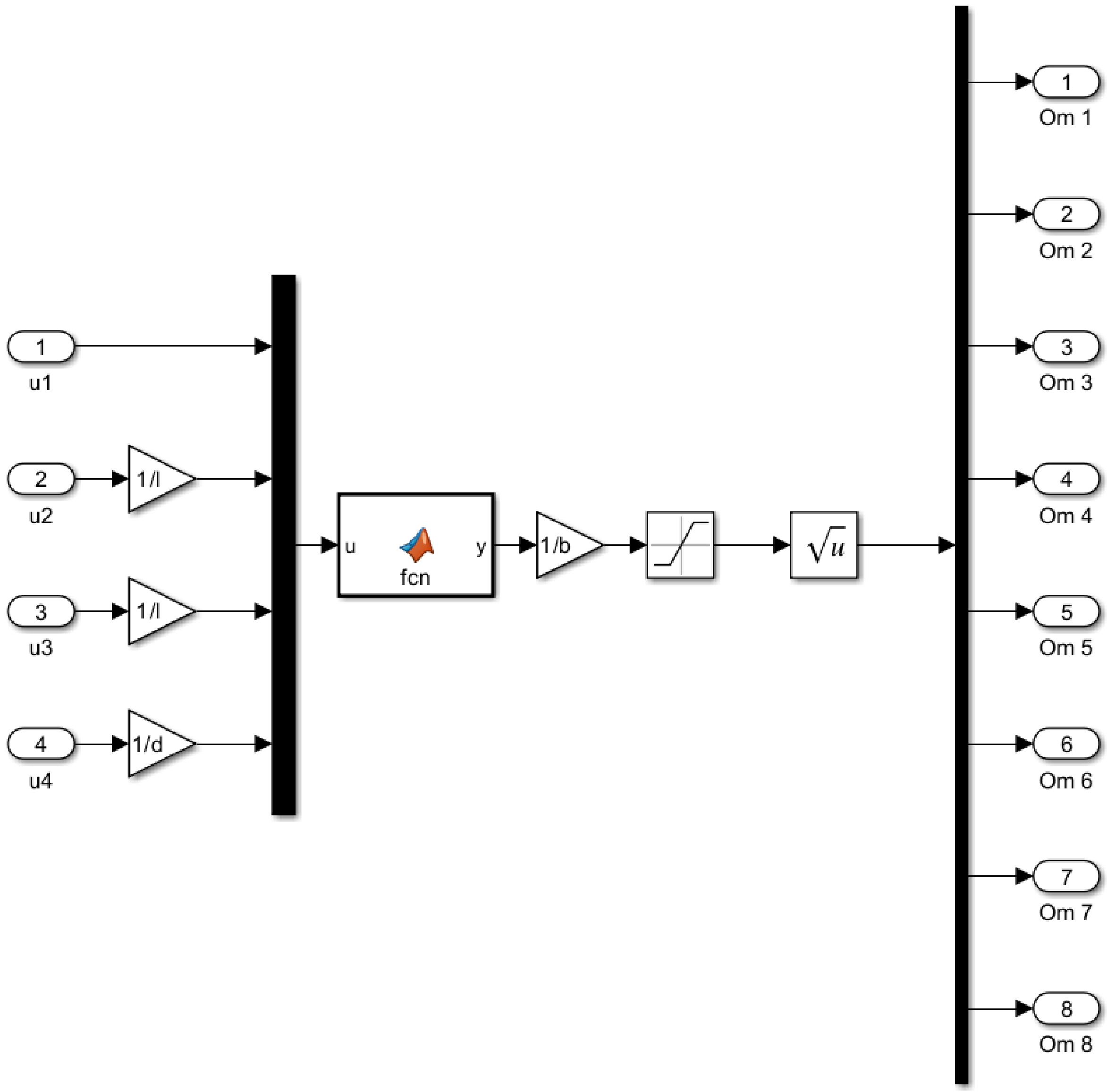


Fig: Channel Mixer

```
function y = fcn(u)
%#codegen
% A = [-1 -1 -1 -1 -1 -1 -1 -1;...
%       -1 -1  0  0  1  1  0  0;...
%        0  0 -1 -1  0  0  1  1;...
%       -1 -1  1  1 -1 -1  1  1];
A = [-0.1250   -0.2500         0   -0.1250;...
     -0.1250   -0.2500         0   -0.1250;...
     -0.1250    0.0000   -0.2500    0.1250;...
     -0.1250    0.0000   -0.2500    0.1250;...
     -0.1250    0.2500         0   -0.1250;...
     -0.1250    0.2500         0   -0.1250;...
     -0.1250    0.0000    0.2500    0.1250;...
     -0.1250    0.0000    0.2500    0.1250];

% A_inv = pinv(A);
% y = A_inv*u;

y = A*u;
```

Fig: Function Inside Channel Mixer

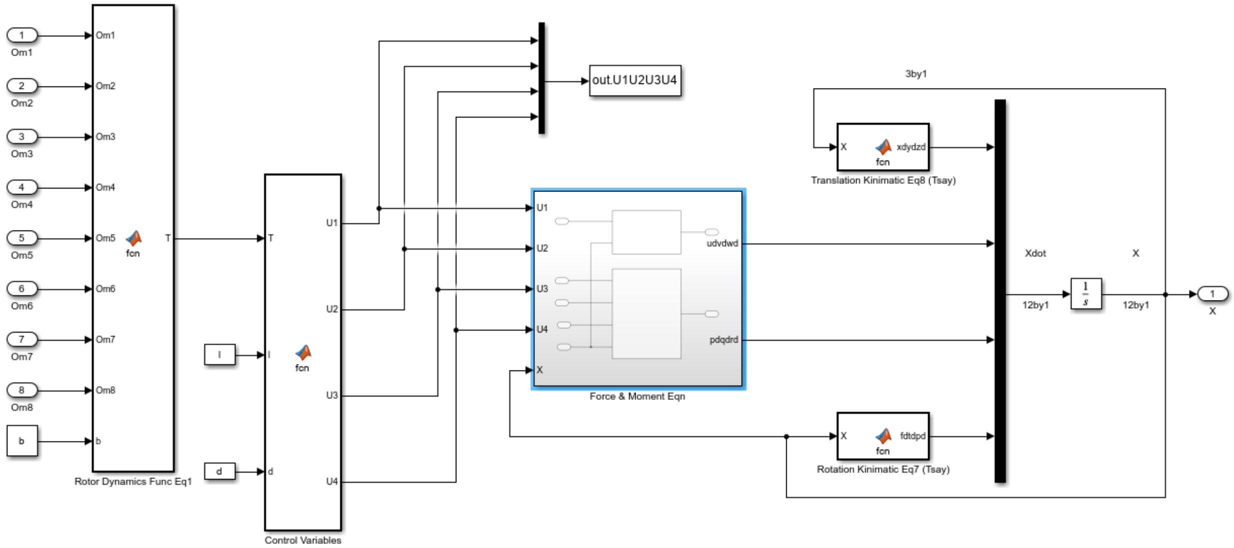


Fig: Octocopter Model

Octacopter ▸ Octacopter model ▸ Rotor Dynamics Func Eq1

```
function T = fcn(Om1, Om2, Om3, Om4, Om5, Om6, Om7, Om8,b)


Om=[Om1,Om2,Om3,Om4,Om5,Om6,Om7,Om8];
T=[0,0,0,0,0,0,0,0];

i = 1;
while i<=8
    T(i) = Om(i)^2*b;
    i = i+1;

% T1 = b*Om1^2;
% T2 = b*Om2^2;
% T3 = b*Om3^2;
% T4 = b*Om4^2;

end

```

Fig: Rotor Dynamics Function Equation (1-Tsay)

Control Variables

Octacopter ▸ Octacopter model ▸ Control Variables

```
function [U1,U2,U3,U4] = fcn(T, l, d)

T1 = T(1);
T2 = T(2);
T3 = T(3);
T4 = T(4);
T5 = T(5);
T6 = T(6);
T7 = T(7);
T8 = T(8);

U1 = -( T1 + T2 + T3 + T4 + T5 + T6 + T7 + T8);
U2 = l/2*(T6+T5-T1-T2);
U3 = l/2*(T7+T8-T4-T3);
U4 = d*(-T1 - T2 + T3 + T4 -T5 -T6 +T7 + T8);

end

```

Fig: Control Variables Function Equation (2-Tsay)

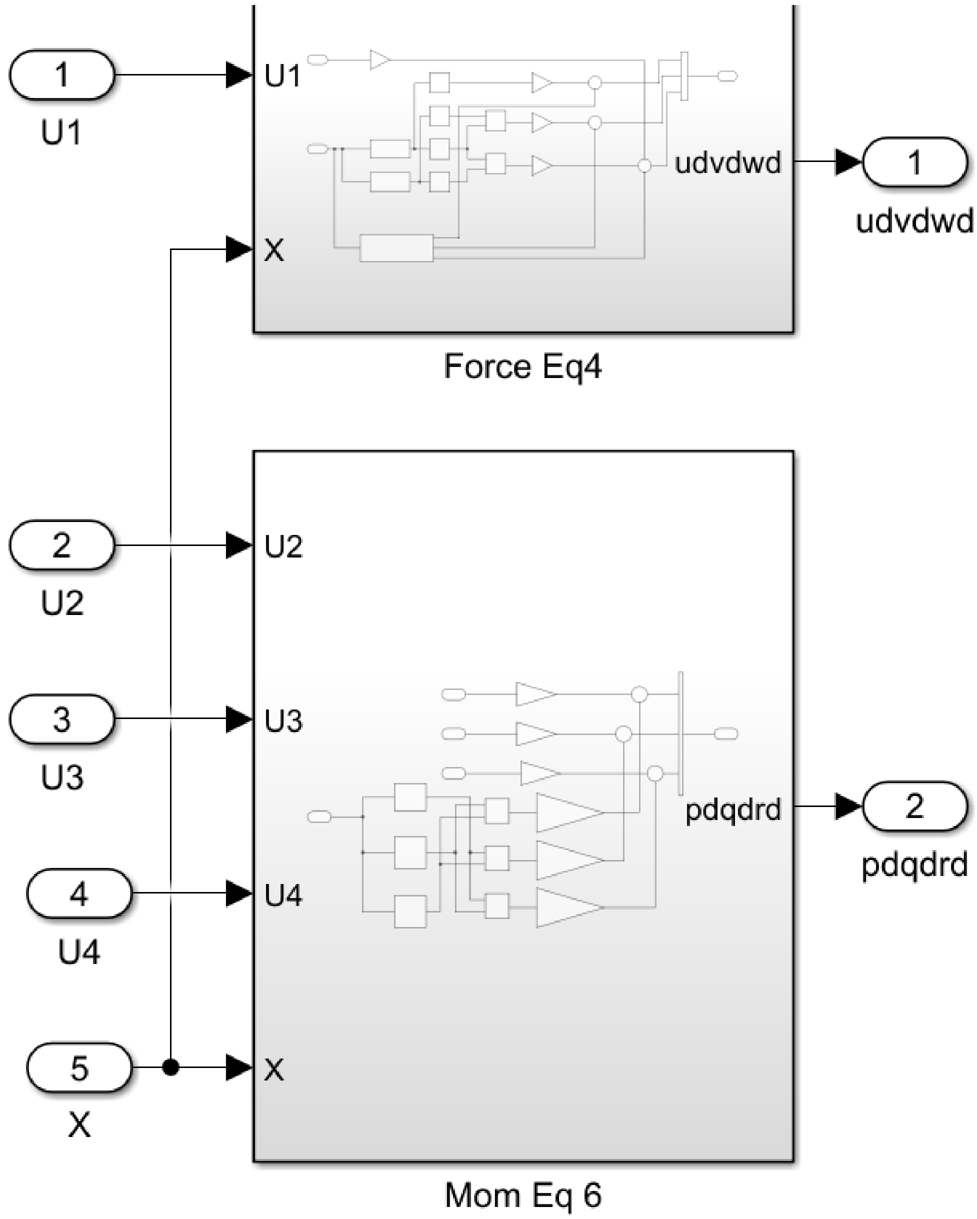


Fig: Force and Moment Equation

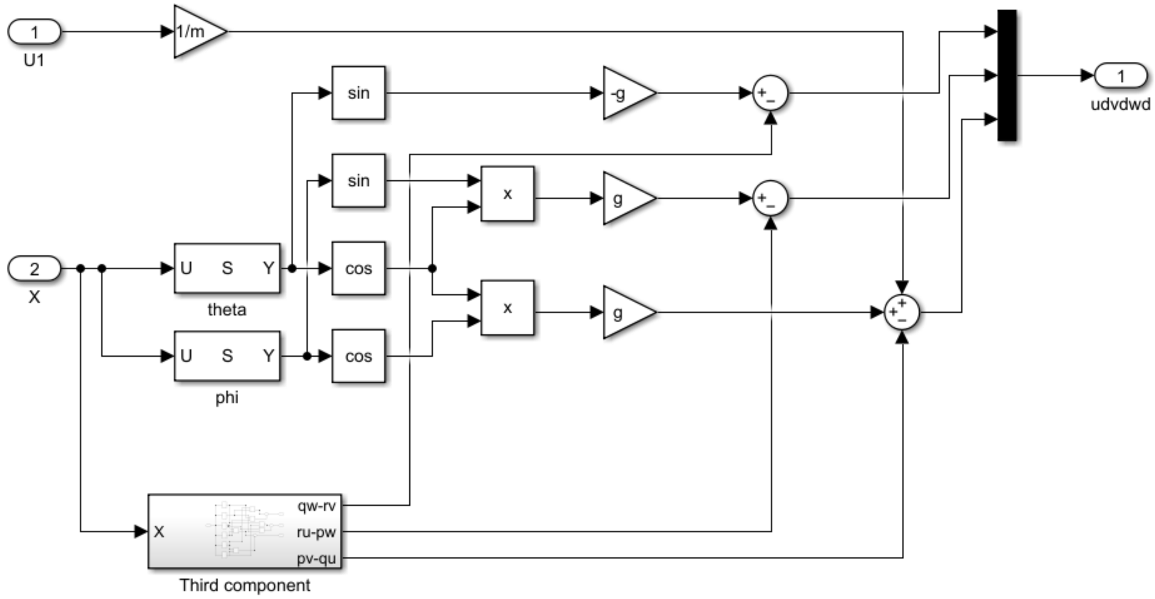


Fig: Force Equation (Tsay-4)

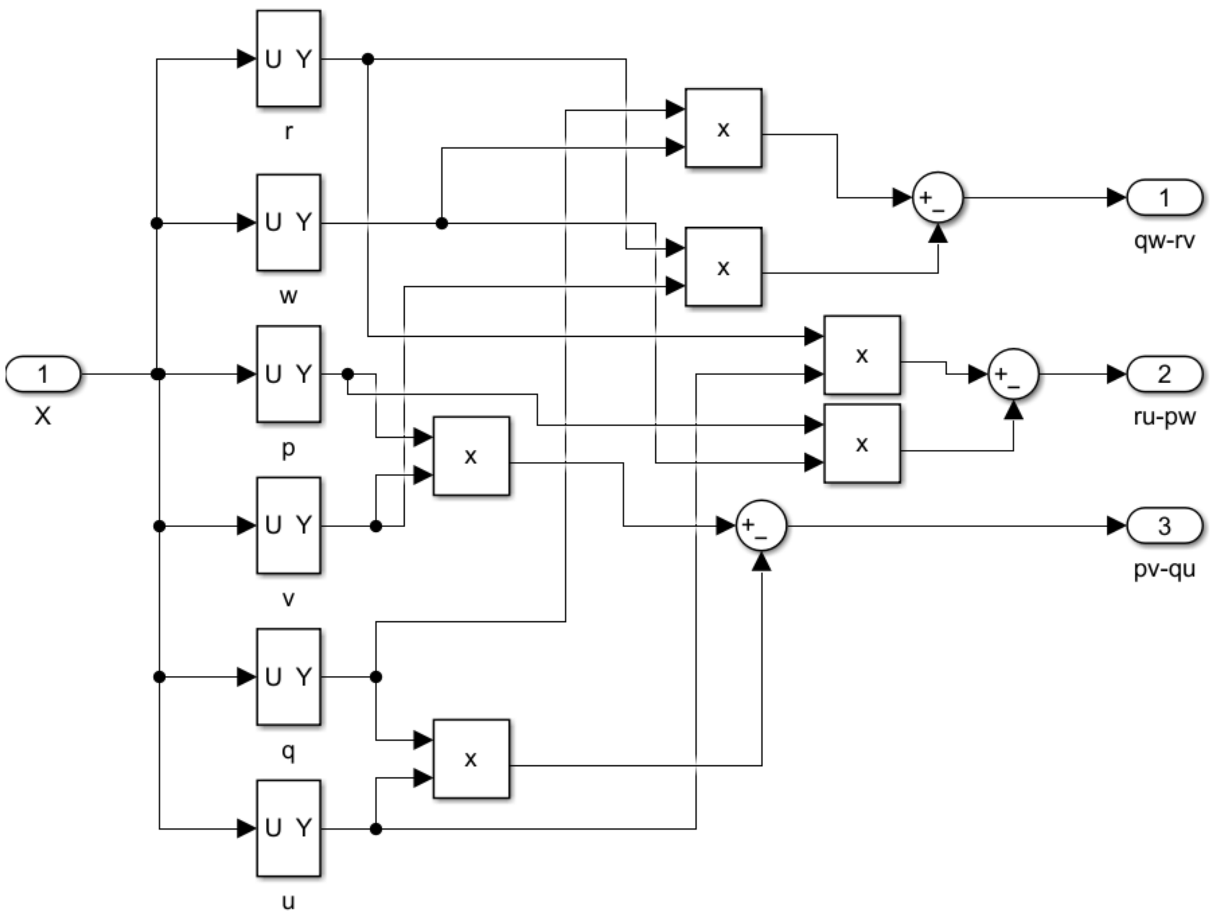


Fig: Third Component of Force equation (Tsay)

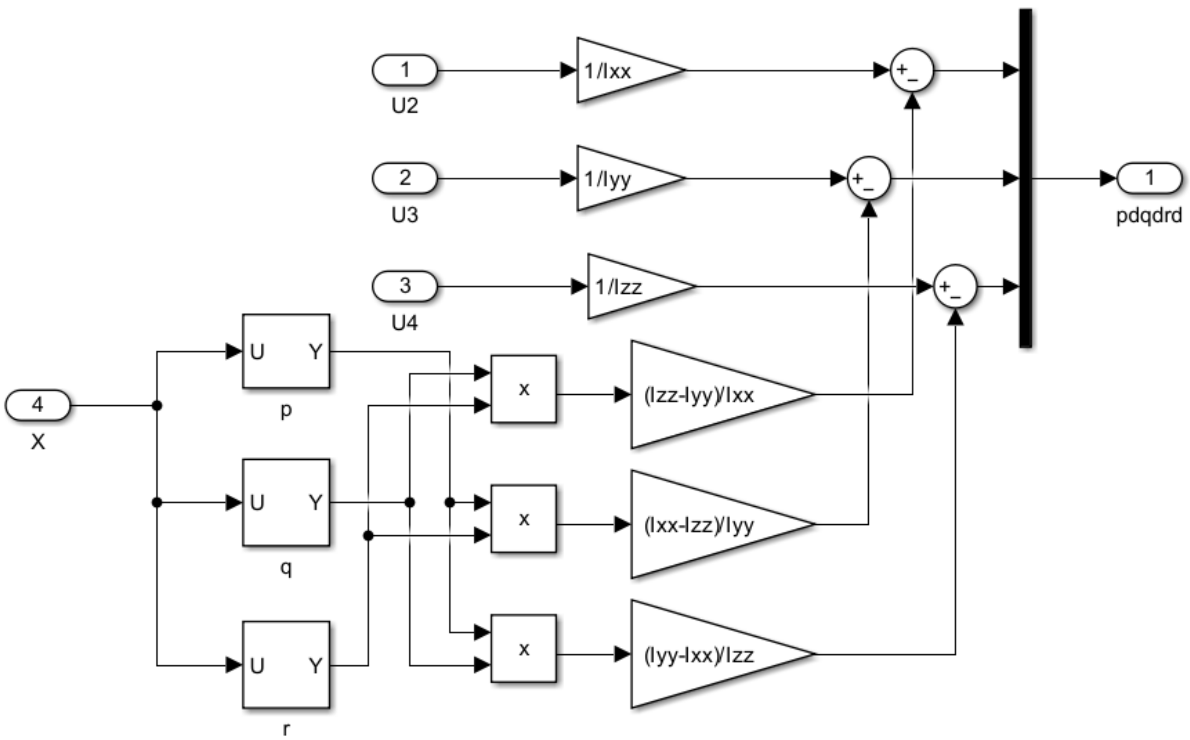
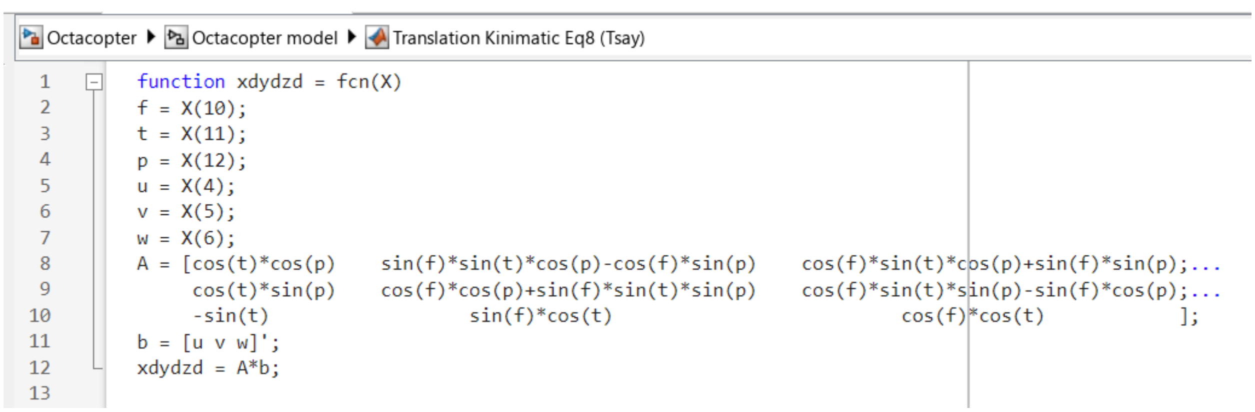


Fig: Moment Equation (Tsay -6)



Octacopter ▸ Octacopter model ▸ Translation Kinimatic Eq8 (Tsay)

```
function xdydzd = fcn(X)
f = X(10);
t = X(11);
p = X(12);
u = X(4);
v = X(5);
w = X(6);
A = [cos(t)*cos(p)    sin(f)*sin(t)*cos(p)-cos(f)*sin(p)    cos(f)*sin(t)*cos(p)+sin(f)*sin(p);...
     cos(t)*sin(p)    cos(f)*cos(p)+sin(f)*sin(t)*sin(p)    cos(f)*sin(t)*sin(p)-sin(f)*cos(p);...
     -sin(t)                  sin(f)*cos(t)                          cos(f)*cos(t)            ];
b = [u v w]';
xdydzd = A*b;
```

Fig: Kinematic Translation Function Equation (Tsay-8)

Octacopter ▶ Octacopter model ▶ Rotation Kinimatic Eq7 (Tsay)

```
function fdtdpd = fcn(X)
phi = X(10);
theta = X(11);
psi = X(12);
p = X(7);
q = X(8);
r = X(9);
A = [1     tan(theta)*sin(phi)      tan(theta)*cos(phi);...
     0          cos(phi)             -sin(phi);...
     0     sec(theta)*sin(phi)     sec(theta)*cos(phi)];
b = [p q r]';
fdtdpd = A*b;
```

Fig: Rotation Kinematic Equation function (7-Tsay)

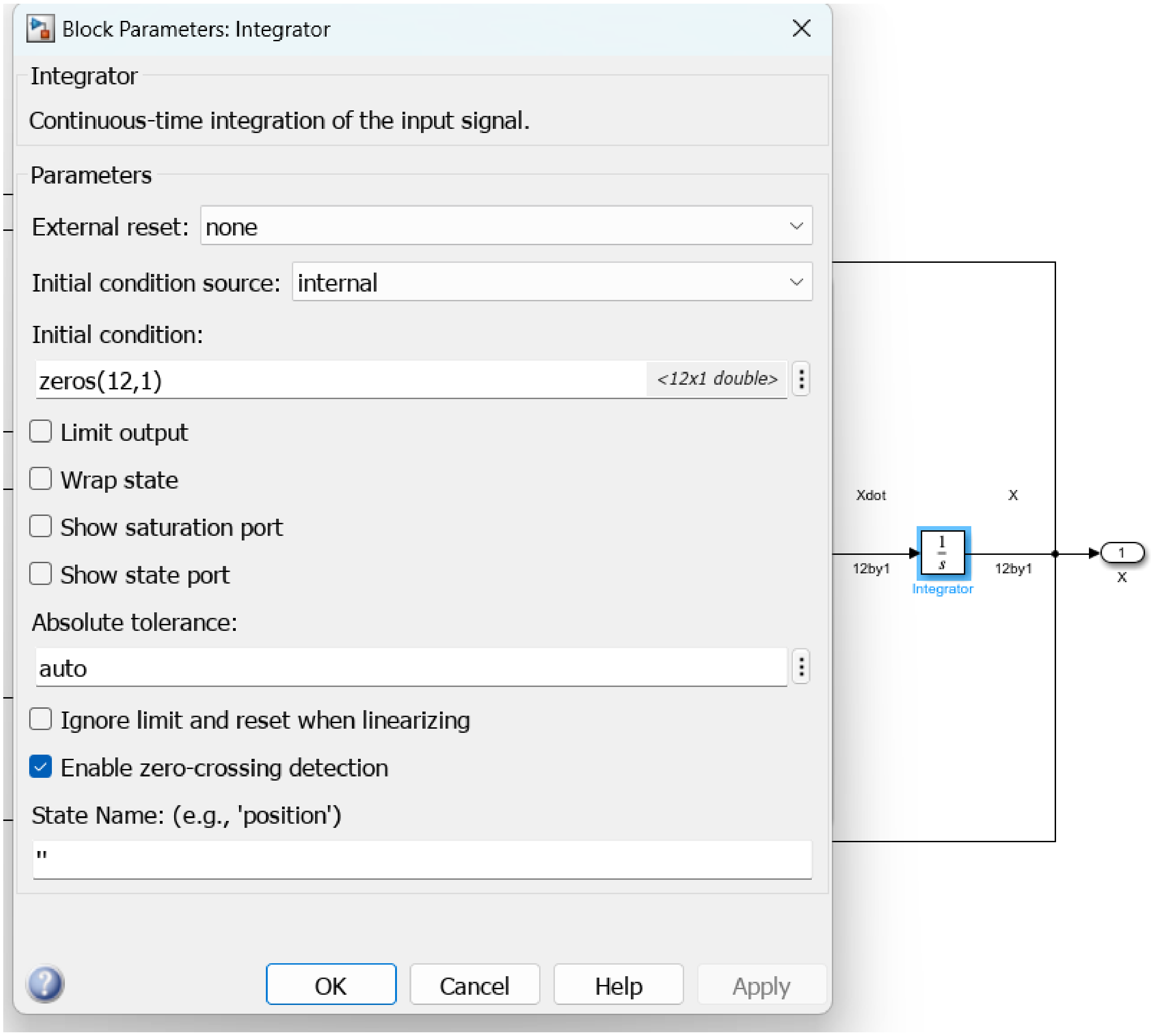


Fig: Integrator Initial Conditions set to “zeros(12-1)

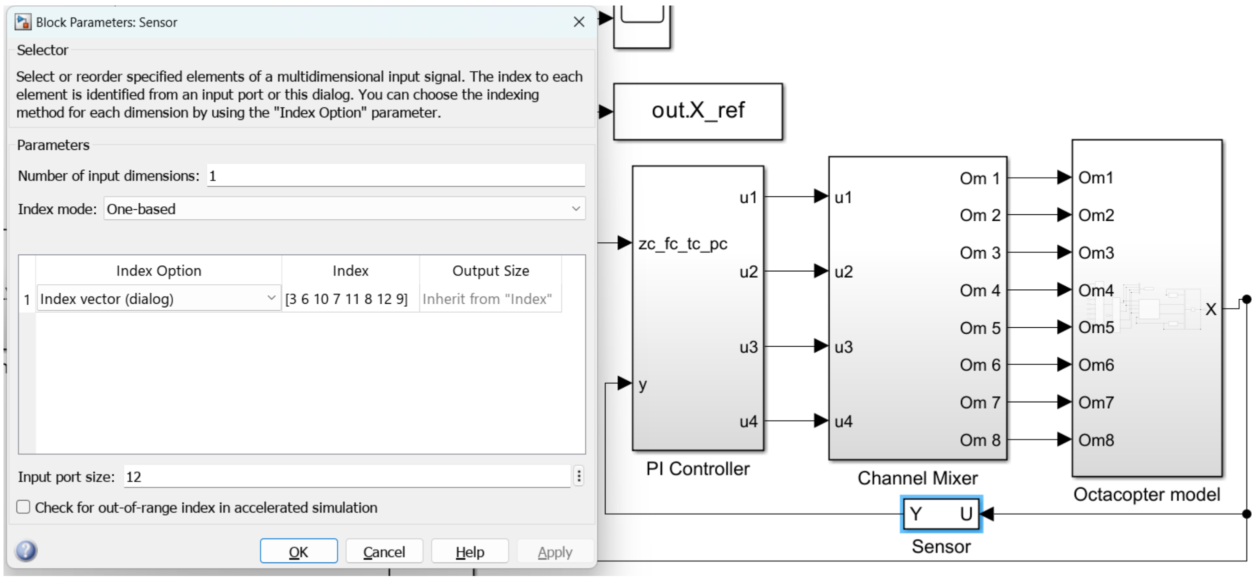


Fig: Sensor Parameters for PI Controller

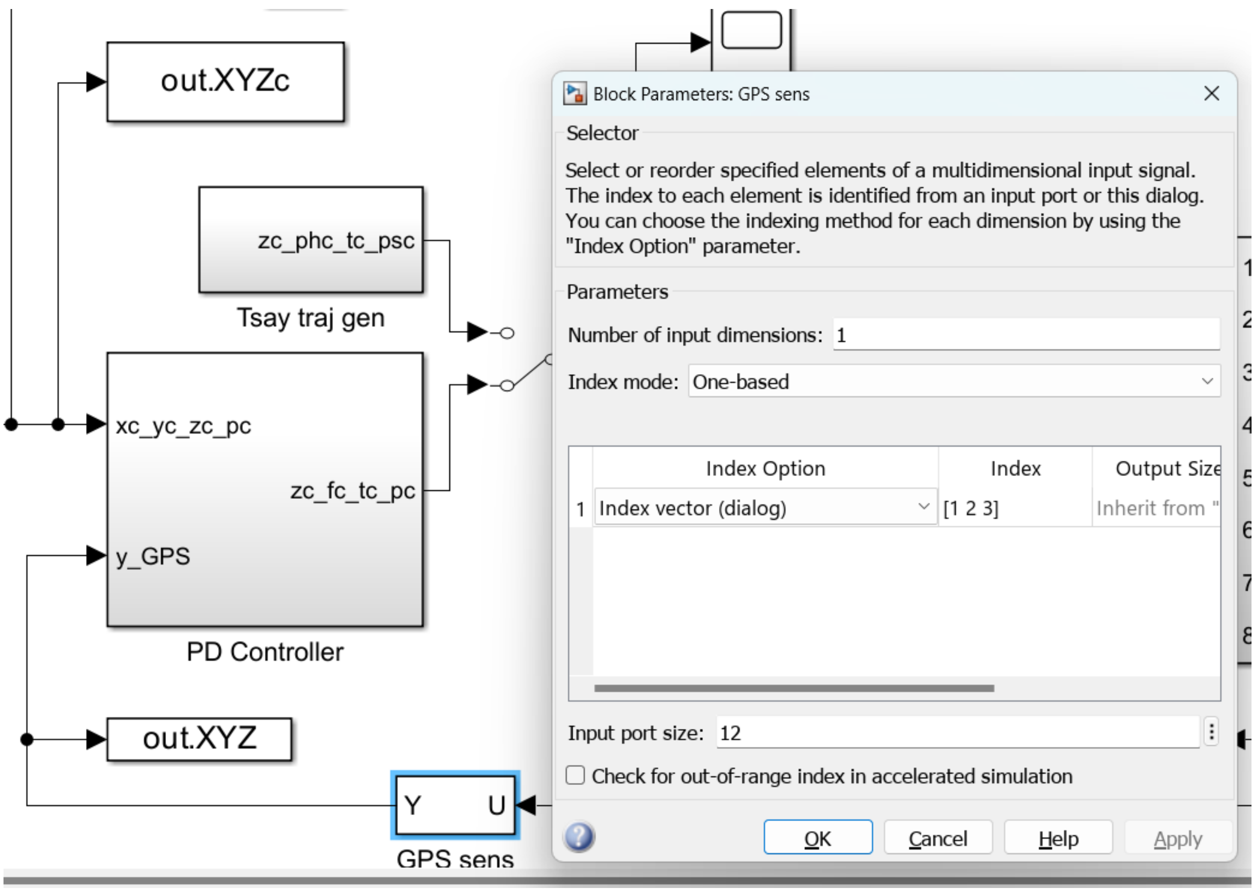


Fig: GPS sensor for PD Controller

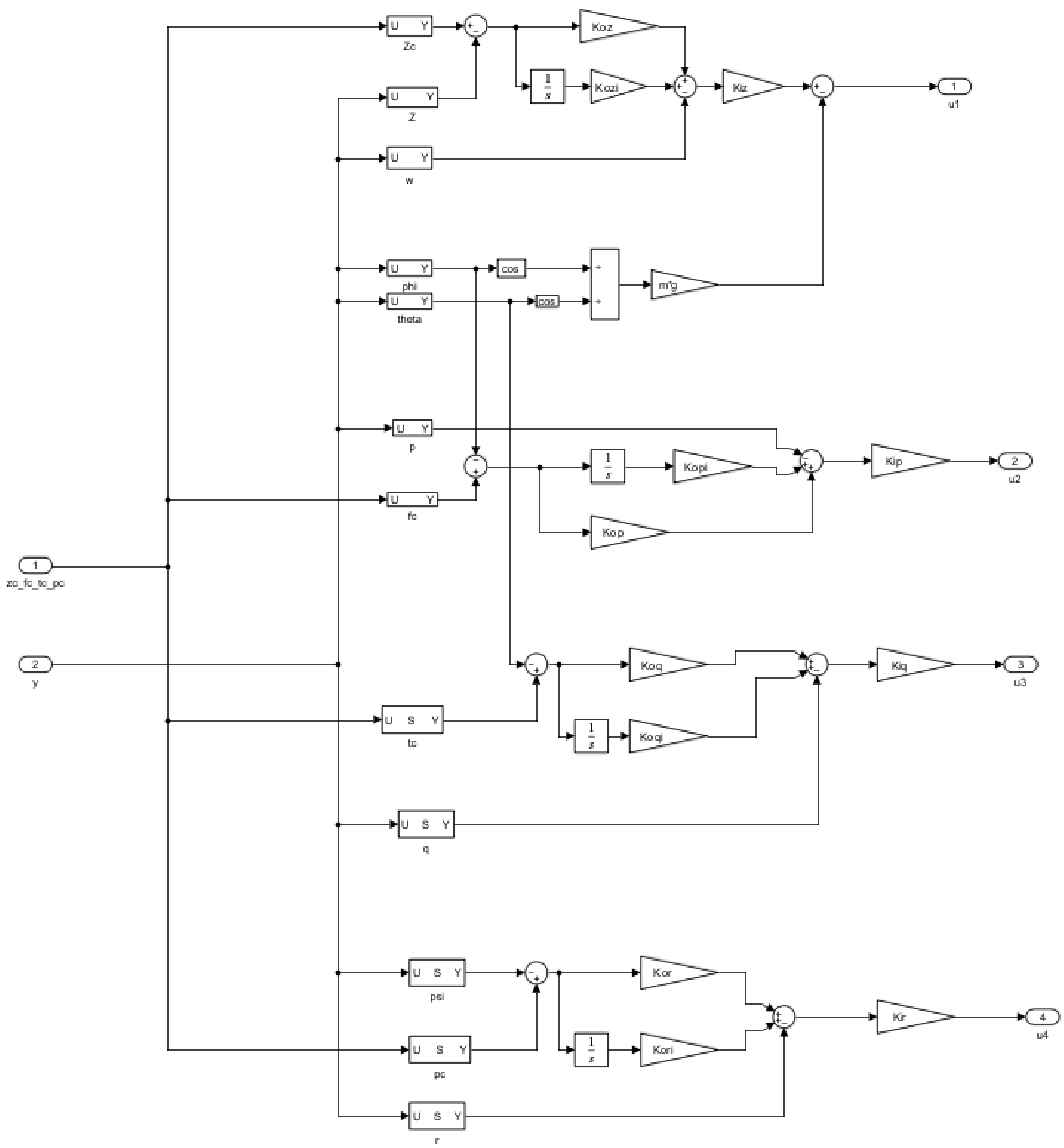


Fig: PI Controller

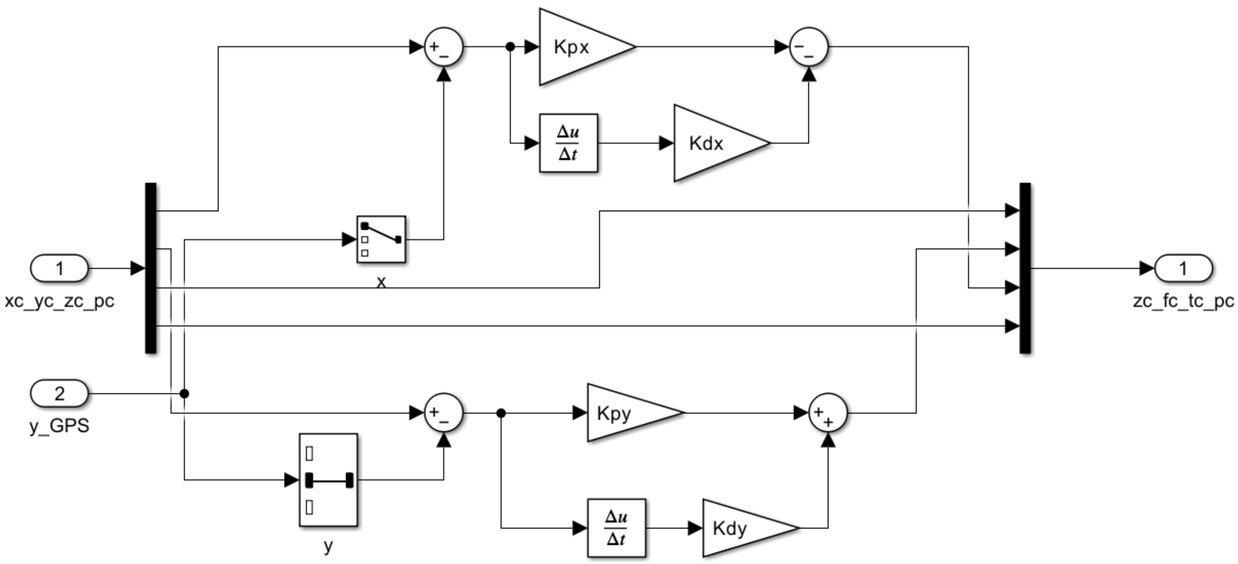


Fig: PD Controller

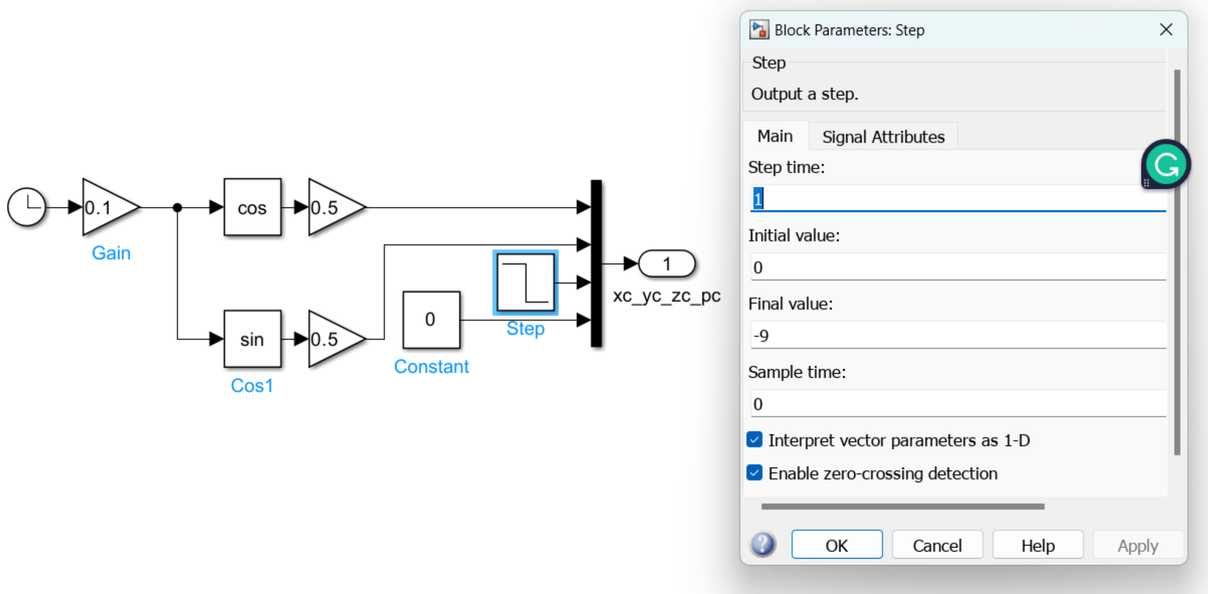


Fig: Space Trajectory Generator

Here, we ask the octocopter to go Z direction upto 9 meters. And then, make a round with given parametric equation of circle.

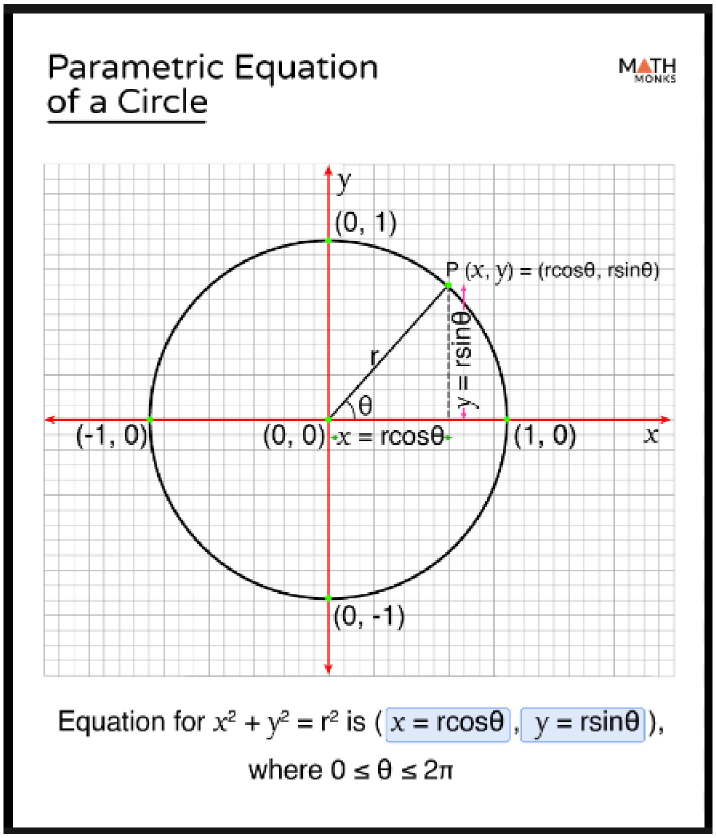


Fig: Parametric Equation of a circle

Here, in this case we have,

$X_c = Acos(\omega t)$ [m] i.e., in meters

$Y_c = Asin(\omega t)$ [m] i.e., in meters

$$A(amplitude) = 0.5\, m, \omega = 0.1 \frac{rad}{s},$$

$$t = set\ as\ per\ you\ like, in\ our\ case, we\ set\ time\ to\ 250\ seconds\ where\ we\ ca\ provide\ it\ directly\ by\ a\ clock$$

Matlab Script:

```
clear,
clc,
close all
sim_time = 250;% Simulation time [s]
dt = 0.01;% Simulation time step length [s]

g = 9.81;%Gravity [m/s^2]
m = 4.34;% Quadrotor mass
b = 1.2953*1e-5;% Thrust coefficient [Ns^2]
l = 0.315;% Rotor arm length [m]
d = 0.008;% Reaction torque coefficient [m]
Ixx = 0.0820;% Moment of inertia along x axis [kg m^2]
Iyy = 0.0845;% Moment of inertia along y axis [kg m^2]
Izz = 0.1377;% Moment of inertia along z axis [kg m^2]

%Controller Gains
%Preleminary values.

Kpx = 0.3;
Kdx = 0.50;
Kpy = 0;
Kdy = 0.15;

Kiz = 15;
Koz = 1;
Kozi = 0;

Kip = 2;
Kop = 5;
Kopi = 0;

Kiq = 2;
Koq = 5;
Koqi = 0;

Kir = 2.2;
Kor = 1;
Kori = 0;

sim_res = sim('Octacopter.slx');
time = sim_res.tout;% Retrieve time vector

%Unpack state variables
x = sim_res.X(:,1);
y = sim_res.X(:,2);
z = sim_res.X(:,3);
u = sim_res.X(:,4);
v = sim_res.X(:,5);
w = sim_res.X(:,6);
```

```
p = sim_res.X(:,7);
q = sim_res.X(:,8);
r = sim_res.X(:,9);
phi = sim_res.X(:,10);
theta = sim_res.X(:,11);
psi = sim_res.X(:,12);


%Unpack commanded variables
z_c = sim_res.X_ref(:,1);
Phi_c = sim_res.X_ref(:,2);
Theta_c = sim_res.X_ref(:,3);
Psi_c = sim_res.X_ref(:,4);
%
%Unpack space trajectories
X = sim_res.XYZ(:,1);
Y = sim_res.XYZ(:,2);
Z = sim_res.XYZ(:,3);

Xc = sim_res.XYZc(:,1);
Yc = sim_res.XYZc(:,2);
Zc = sim_res.XYZc(:,3);

%Generate plots
plot(time,[z_c,z]),grid,xlabel('time [s]'),legend('z_d[m]','z_a [m]'),title("Altitude
Maneuver")
figure(2),
plot(time,[Phi_c,phi]*180/pi),grid,xlabel('time[s]'),legend('Phi_d [deg]','Phi_a
[deg]'),title("Roll Maneuver")
figure(3),
plot(time,[Theta_c,theta]*180/pi),grid,xlabel('time[s]'),legend('Theta_d
[deg]','Theta_a [deg]'),title("Pitch Maneuver")
figure(4),
plot(time,[Psi_c,psi]*180/pi),grid,xlabel('time[s]'),legend('Psi_d [deg]','Psi_a
[deg]'),title("Yaw Maneuver")
figure(5),
plot(time,[Xc X]),grid,xlabel('time [s]'),...
legend('X_c [m]', 'X [m]'),title('X - Component ')
figure(6),
plot(time,[Yc Y]),grid,xlabel('time [s]'),...
legend('Y_c [m]', 'Y [m]'),title('Y - Component ')
figure(7),
plot(time,[Zc Z]),grid,xlabel('time [s]'),...
legend('Z_c [m]', 'Z [m]'),title('Z - Component ')
figure(8),
plot3([X Xc], [Y Yc] , [Z Zc] ),grid,legend('Drone path','Commanded path'),title('3D
Maneuver')
```

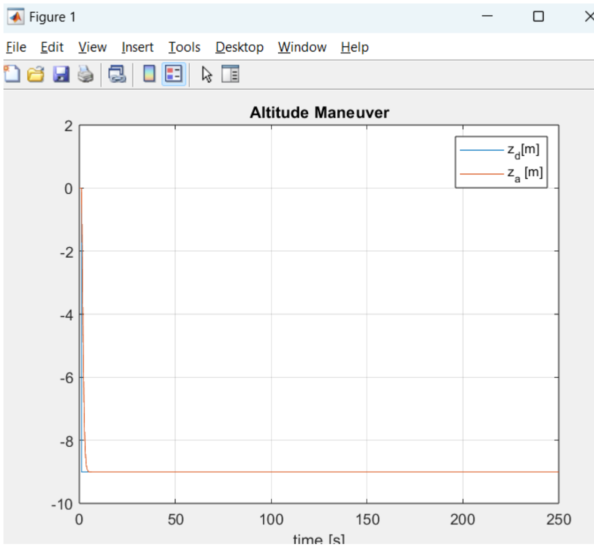


Figure 1
File Edit View Insert Tools Desktop Window Help
Altitude Maneuver
z_d[m]
z_a [m]
2
0
-2
-4
-6
-8
-10
0 50 100 150 200 250
time [s]

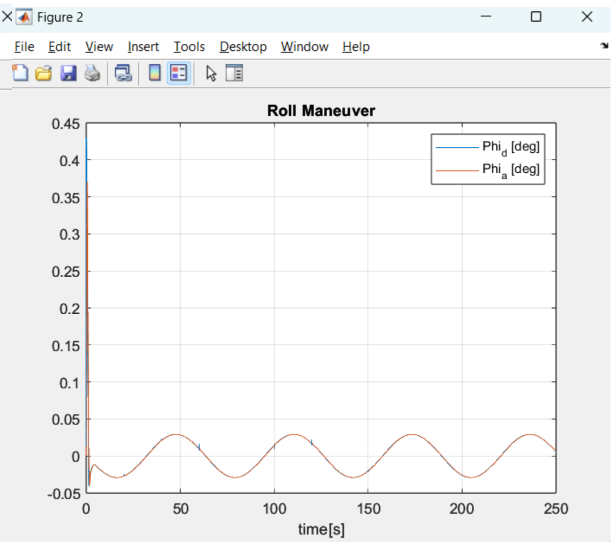


Figure 2
File Edit View Insert Tools Desktop Window Help
Roll Maneuver
Phi_d [deg]
Phi_a [deg]
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
0 50 100 150 200 250
time[s]

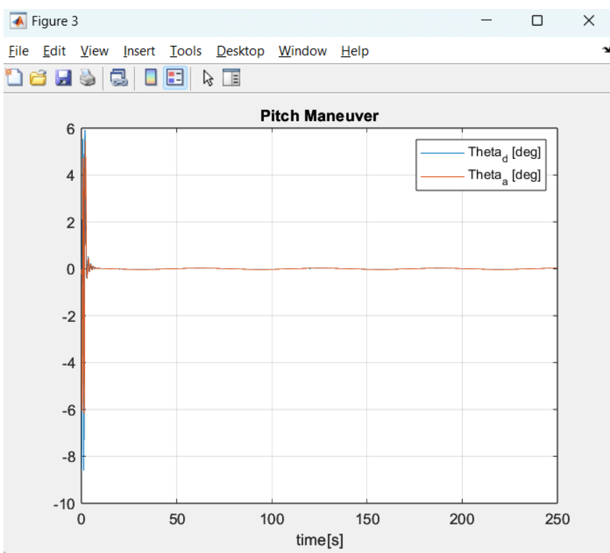


Figure 3
File Edit View Insert Tools Desktop Window Help
Pitch Maneuver
Theta_d [deg]
Theta_a [deg]
6
4
2
0
-2
-4
-6
-8
-10
0 50 100 150 200 250
time[s]

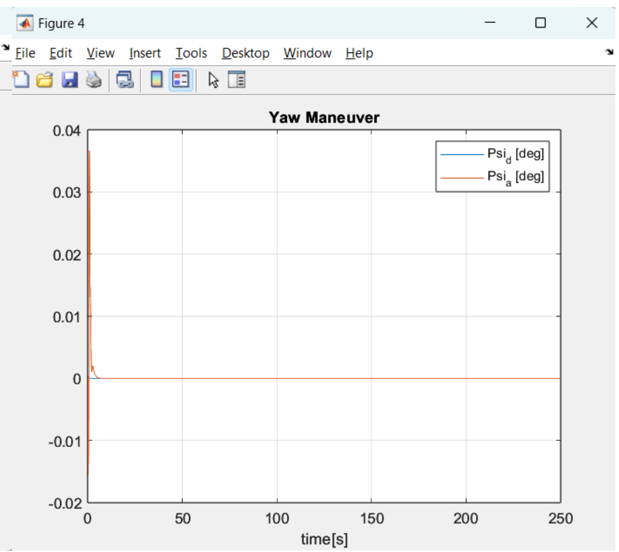


Figure 4
File Edit View Insert Tools Desktop Window Help
Yaw Maneuver
Psi_d [deg]
Psi_a [deg]
0.04
0.03
0.02
0.01
0
-0.01
-0.02
0 50 100 150 200 250
time[s]

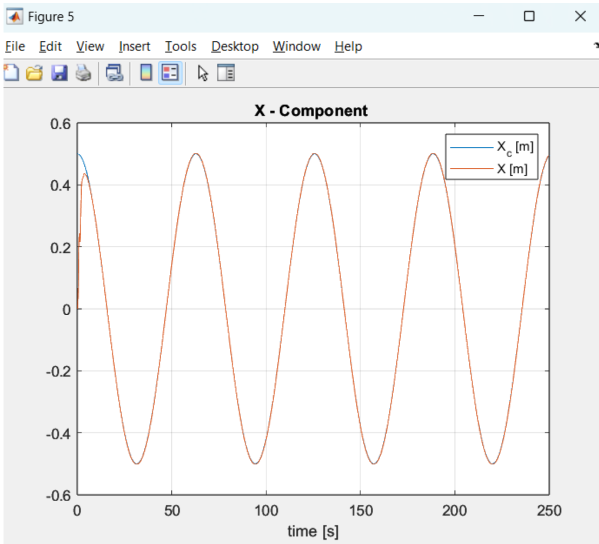


Figure 5
File Edit View Insert Tools Desktop Window Help
X - Component
Xc [m]
X [m]
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0 50 100 150 200 250
time [s]

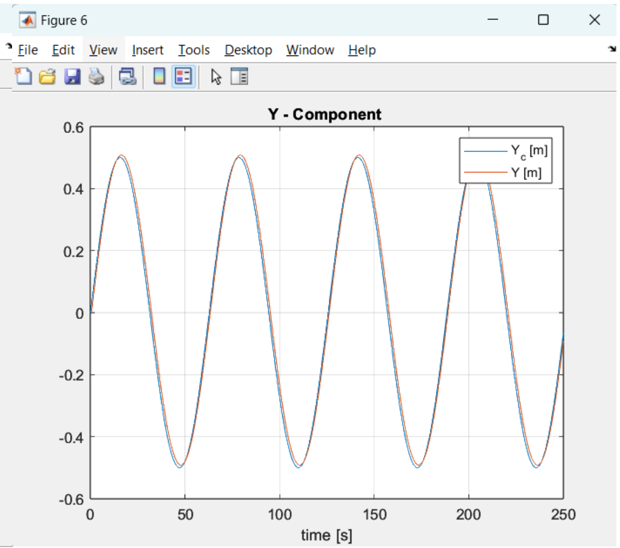


Figure 6
File Edit View Insert Tools Desktop Window Help
Y - Component
Yc [m]
Y [m]
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0 50 100 150 200 250
time [s]

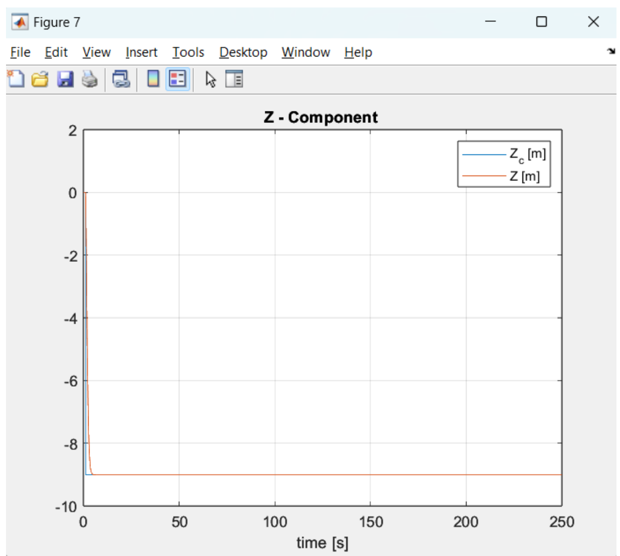


Figure 7
File Edit View Insert Tools Desktop Window Help
Z - Component
Zc [m]
Z [m]
2
0
-2
-4
-6
-8
-10
0 50 100 150 200 250
time [s]

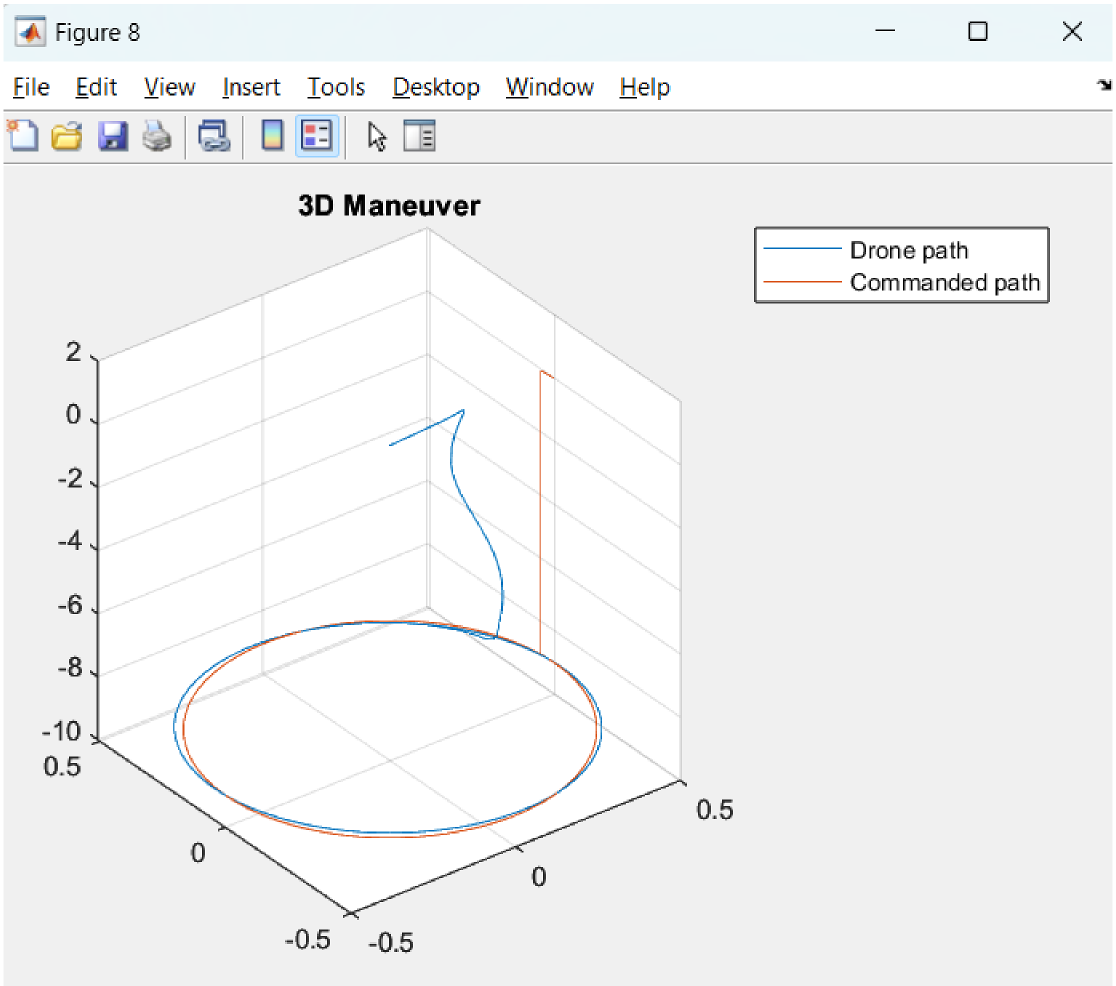


Fig: Final 3D Maneuver of the Drone.