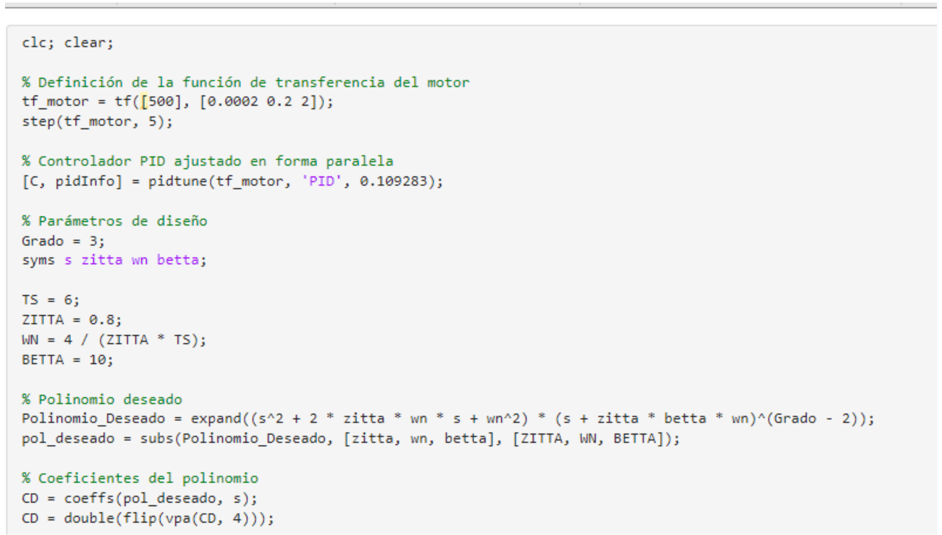
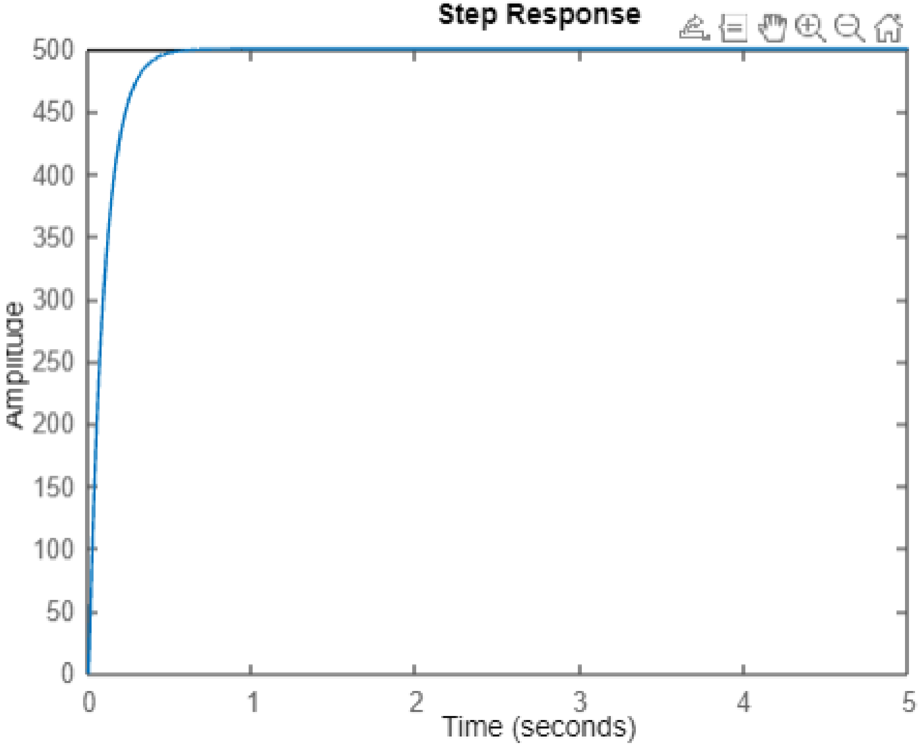
Ganancias para motor y ft motor

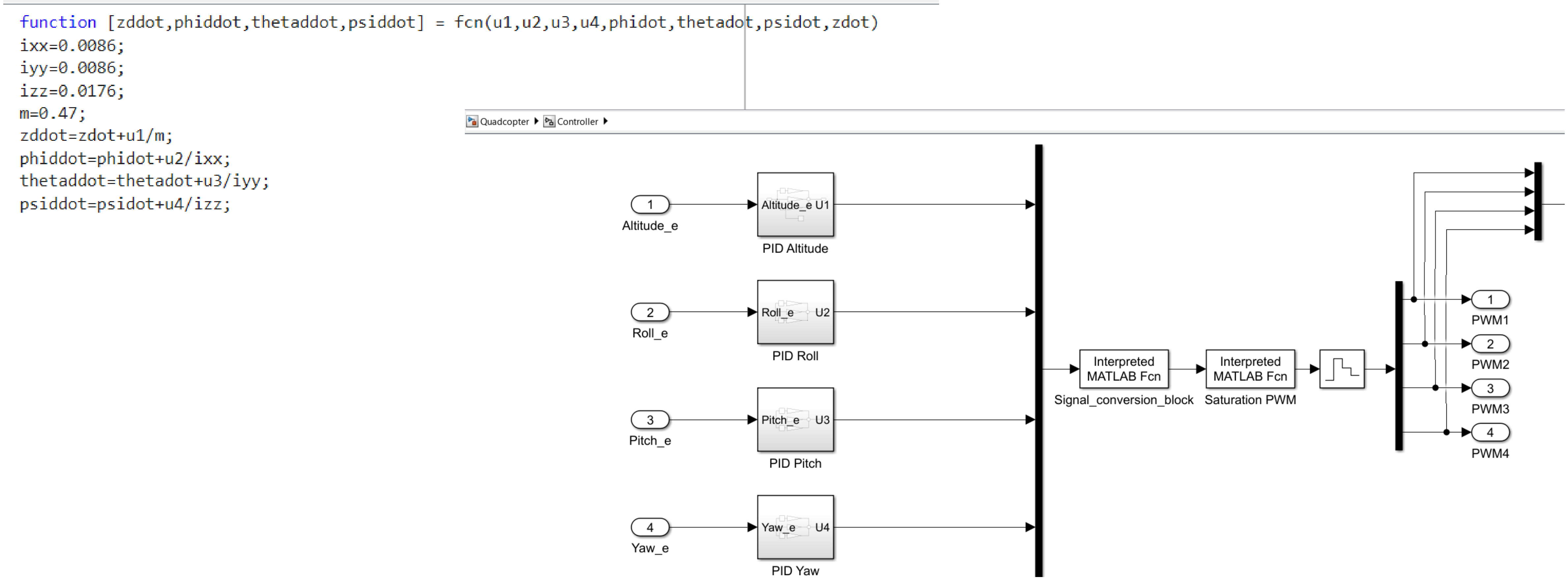


```
clc; clear;

% Definición de la función de transferencia del motor
tf_motor = tf([500], [0.0002 0.2 2]);
step(tf_motor, 5);

% Controlador PID ajustado en forma paralela
[C, pidInfo] = pidtune(tf_motor, 'PID', 0.109283);

% Parámetros de diseño
Grado = 3;
syms s zitta wn betta;

TS = 6;
ZITTA = 0.8;
WN = 4 / (ZITTA * TS);
BETTA = 10;

% Polinomio deseado
Polinomio_Deseado = expand((s^2 + 2 * zitta * wn * s + wn^2) * (s + zitta * betta * wn)^(Grado - 2));
pol_deseado = subs(Polinomio_Deseado, [zitta, wn, betta], [ZITTA, WN, BETTA]);

% Coeficientes del polinomio
CD = coeffs(pol_deseado, s);
CD = double(flip(vpa(CD, 4)));
```



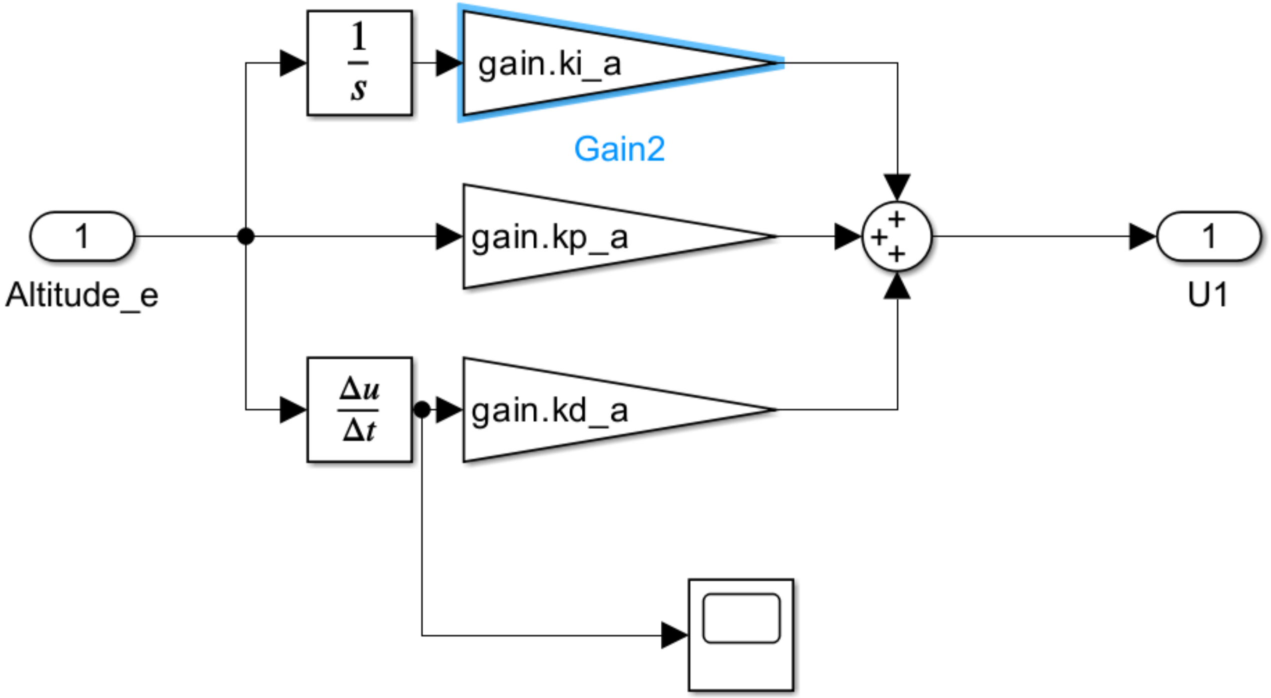
```
Kd = 8.0000

Kp = 9.5833

Ki = 4.6296
```

Controladores H, Roll, Pitch, Yaw

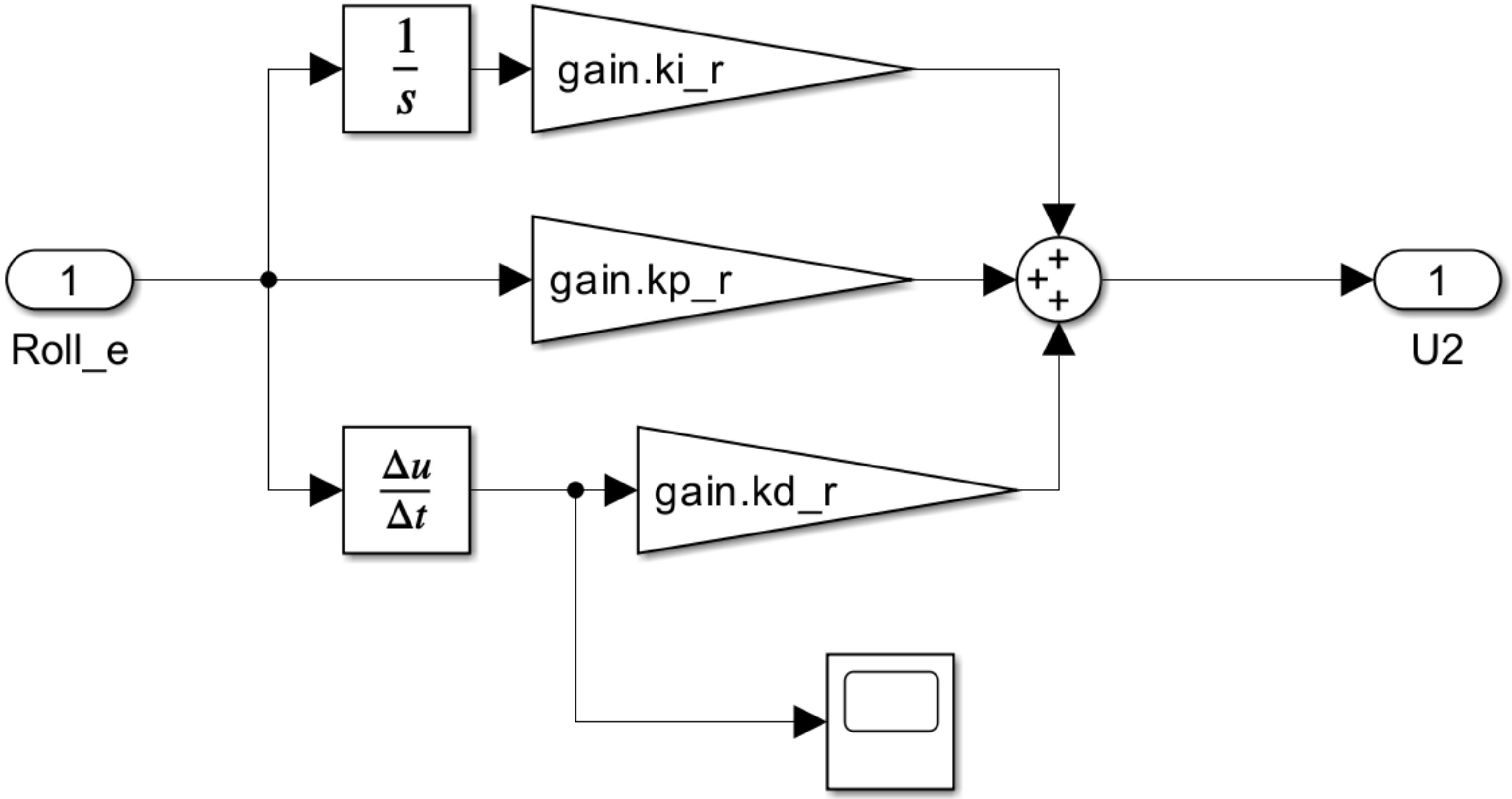


```
function [zddot,phiddot,thetaddot,psiddot] = fcn(u1,u2,u3,u4,phidot,thetadot,psidot,zdot)
ixx=0.0086;
iyy=0.0086;
izz=0.0176;
m=0.47;
zddot=zdot+u1/m;
phiddot=phidot+u2/ixx;
thetaddot=thetadot+u3/iyy;
psiddot=psidot+u4/izz;
```

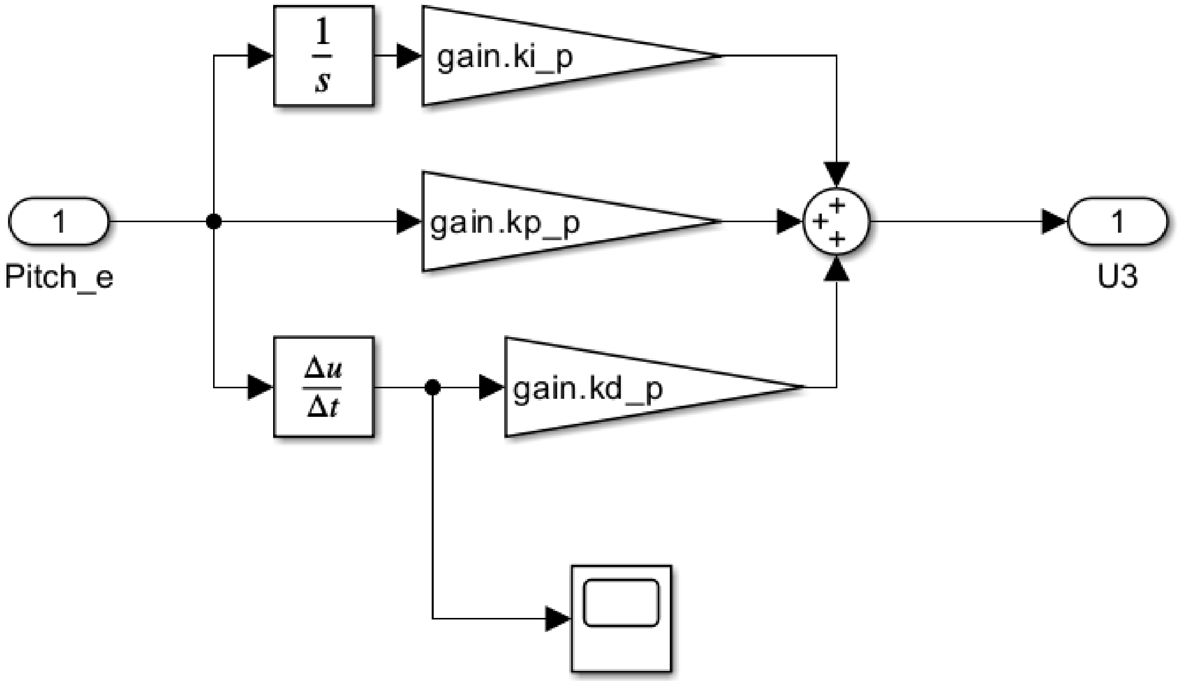
PID ALTITUD



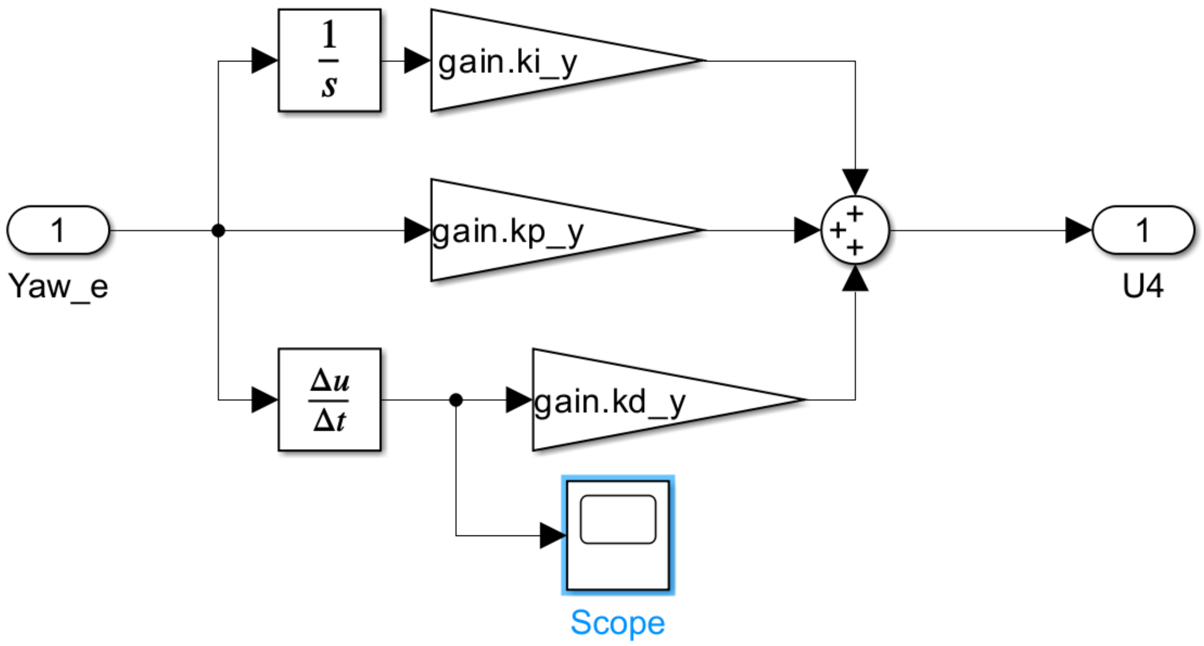
PID ROLL


1
Roll_e
$\frac{1}{s}$
gain.ki_r
gain.kp_r
$\frac{\Delta u}{\Delta t}$
gain.kd_r
+
+
+
1
U2

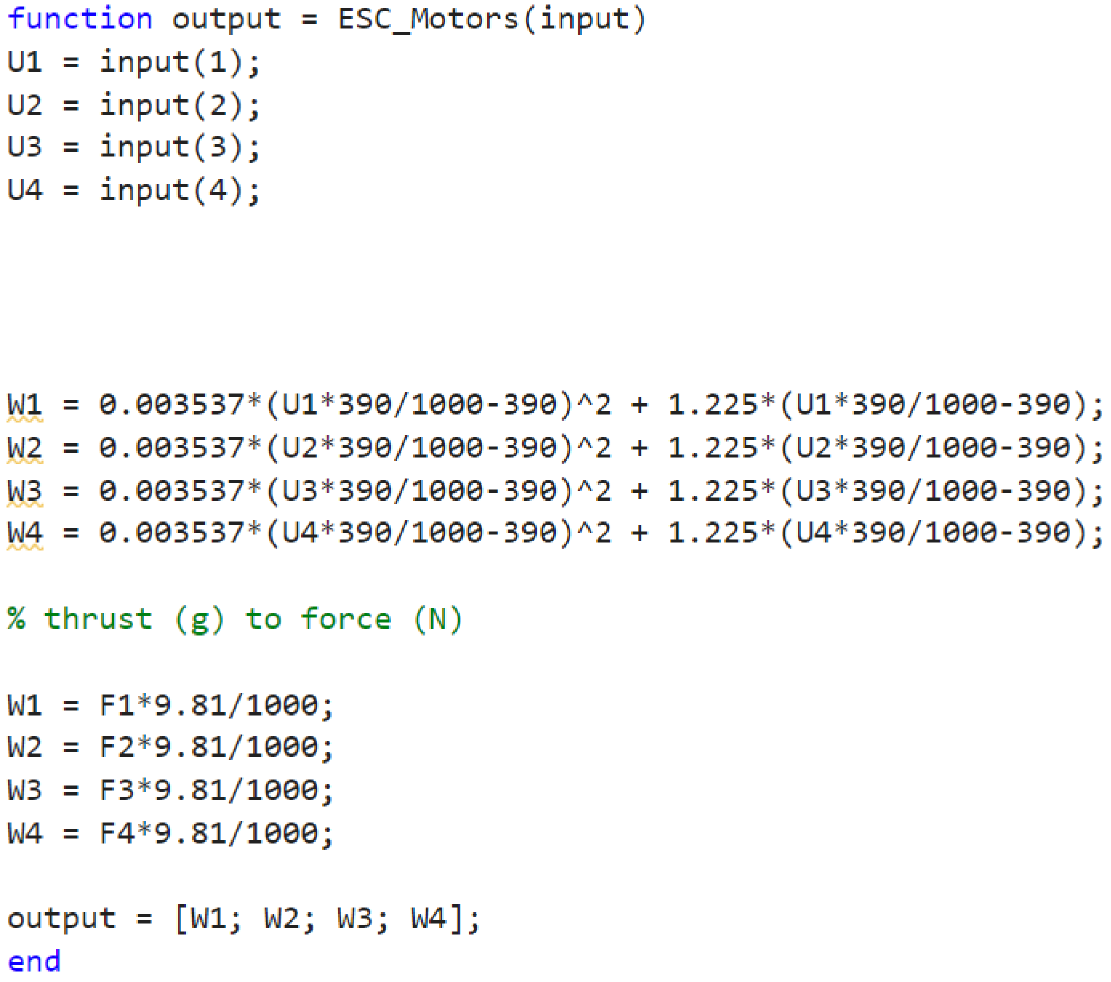
PID PITCH


$\frac{1}{s}$
gain.ki_p
1
Pitch_e
gain.kp_p
+
+
+
1
U3
$\frac{\Delta u}{\Delta t}$
gain.kd_p

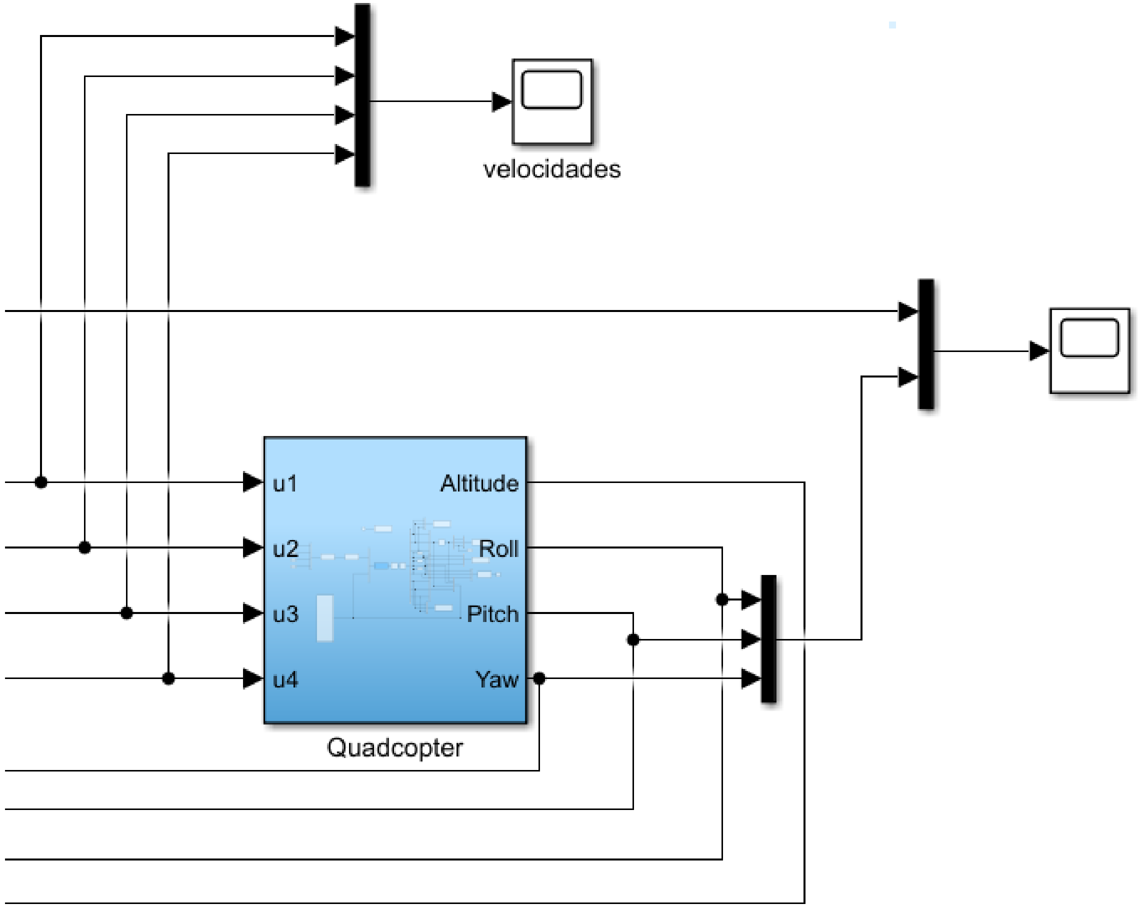
PID YAW


1
Yaw_e
$\frac{1}{s}$
gain.ki_y
gain.kp_y
$\frac{\Delta u}{\Delta t}$
gain.kd_y
+
+
+
1
U4
Scope

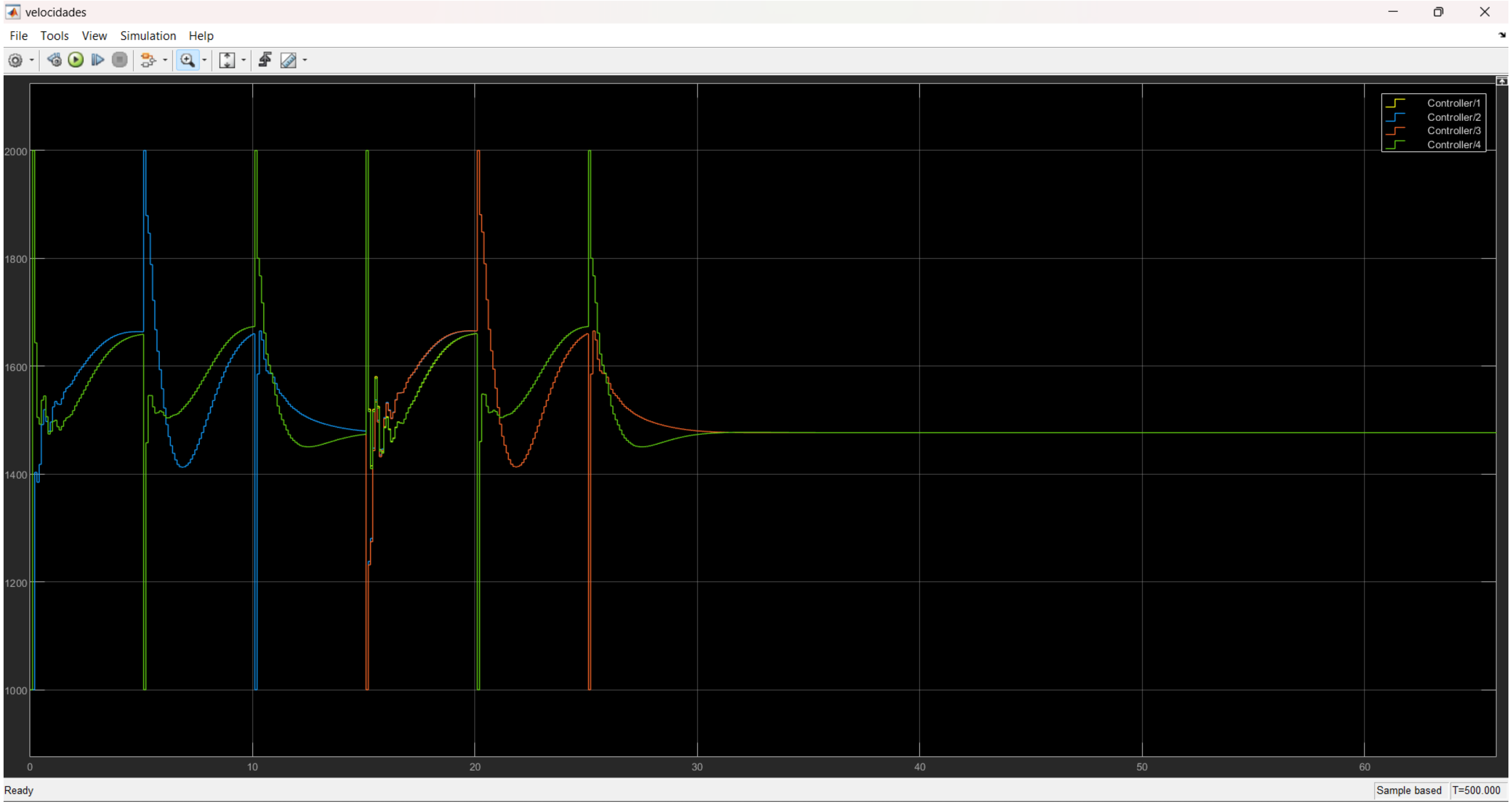
ECUACIONES TORQUES U1,U2,U3,U4 PARA CONTROL VELOCIDAD MOTOR



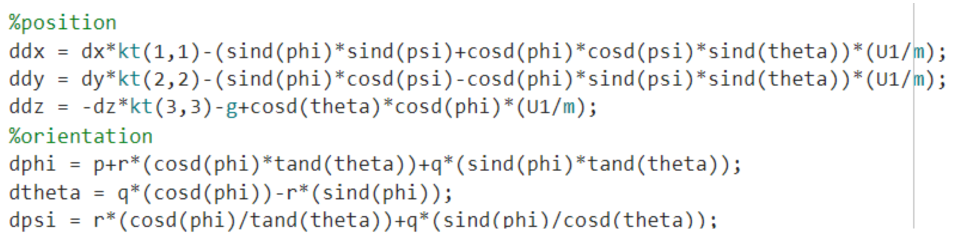
```
function output = ESC_Motors(input)
U1 = input(1);
U2 = input(2);
U3 = input(3);
U4 = input(4);




W1 = 0.003537*(U1*390/1000-390)^2 + 1.225*(U1*390/1000-390);
W2 = 0.003537*(U2*390/1000-390)^2 + 1.225*(U2*390/1000-390);
W3 = 0.003537*(U3*390/1000-390)^2 + 1.225*(U3*390/1000-390);
W4 = 0.003537*(U4*390/1000-390)^2 + 1.225*(U4*390/1000-390);

% thrust (g) to force (N)

W1 = F1*9.81/1000;
W2 = F2*9.81/1000;
W3 = F3*9.81/1000;
W4 = F4*9.81/1000;

output = [W1; W2; W3; W4];
end
```



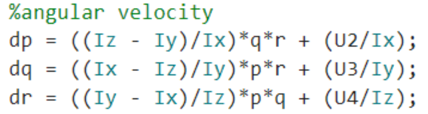
VELOCIDADES



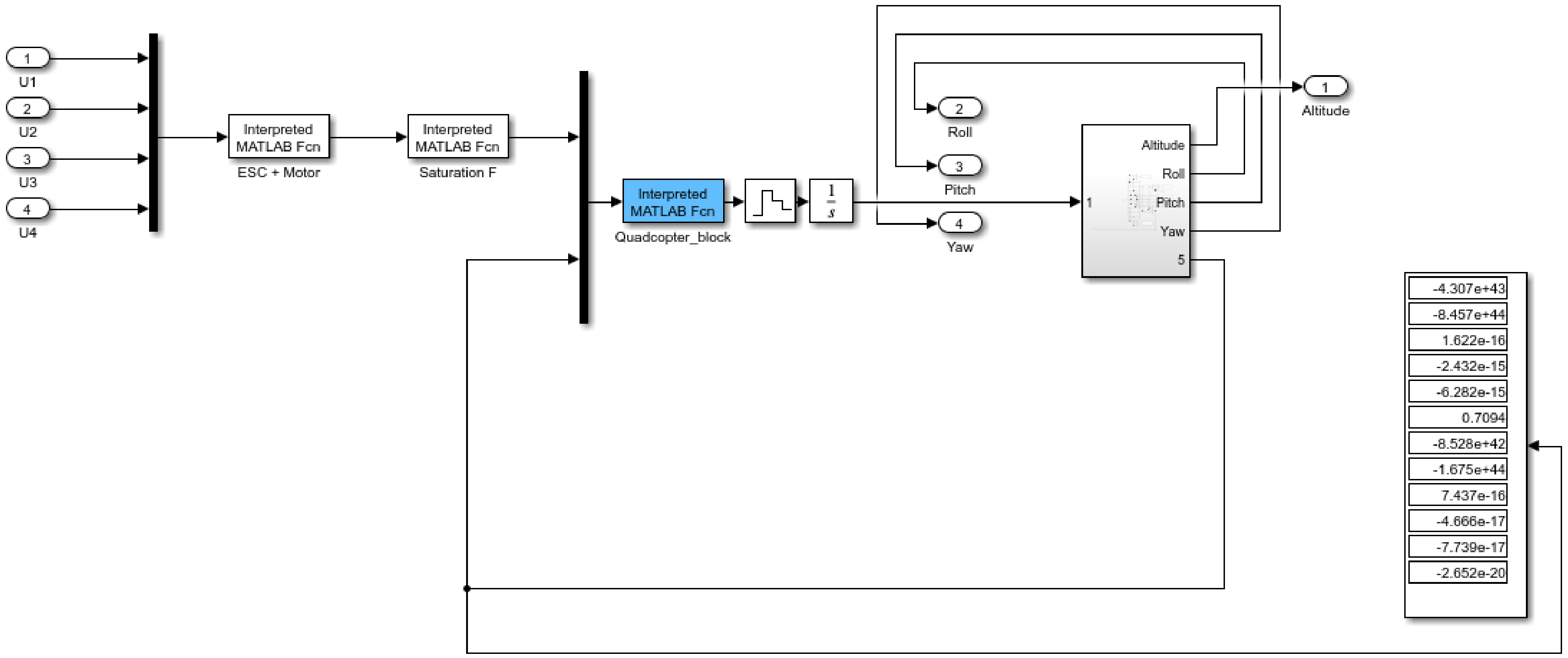
DINAMICA EN CUANTO A LA POSICION Y ORIENTACION (x ,yz ,phi, theta, psi ,dx, dy, p,q,r)



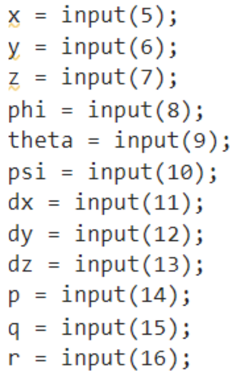
```
%position
ddx = dx*kt(1,1)-(sind(phi)*sind(psi)+cosd(phi)*cosd(psi)*sind(theta))*(U1/m);
ddy = dy*kt(2,2)-(sind(phi)*cosd(psi)-cosd(phi)*sind(psi)*sind(theta))*(U1/m);
ddz = -dz*kt(3,3)-g+cosd(theta)*cosd(phi)*(U1/m);
%orientation
dphi = p+r*(cosd(phi)*tand(theta))+q*(sind(phi)*tand(theta));
dtheta = q*(cosd(phi))-r*(sind(phi));
dpsi = r*(cosd(phi)/tand(theta))+q*(sind(phi)/cosd(theta));
```



```
%angular velocity
dp = ((Iz - Iy)/Ix)*q*r + (U2/Ix);
dq = ((Ix - Iz)/Iy)*p*r + (U3/Iy);
dr = ((Iy - Ix)/Iz)*p*q + (U4/Iz);
```

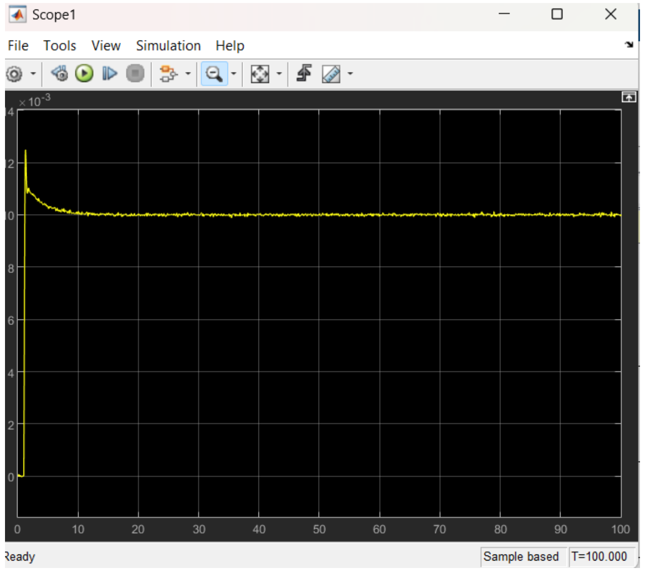


```
-4.307e+43
-8.457e+44
1.622e-16
-2.432e-15
-6.282e-15
0.7094
-8.528e+42
-1.675e+44
7.437e-16
-4.666e-17
-7.739e-17
-2.652e-20
```

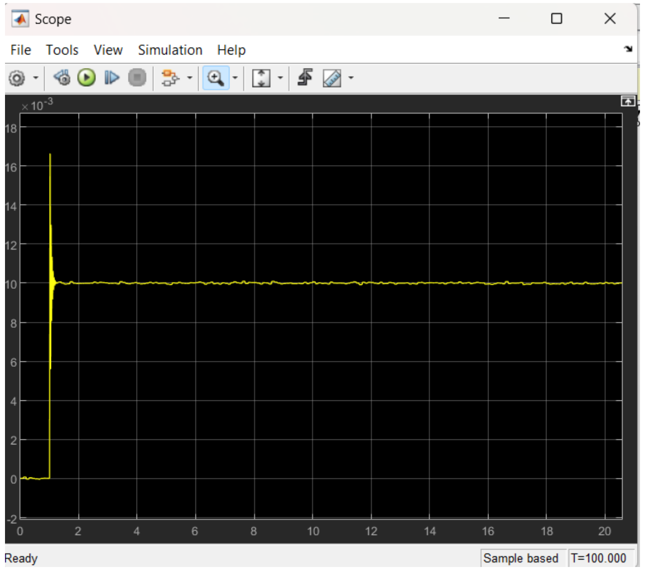


```
x = input(5);
y = input(6);
z = input(7);
phi = input(8);
theta = input(9);
psi = input(10);
dx = input(11);
dy = input(12);
dz = input(13);
p = input(14);
q = input(15);
r = input(16);
```

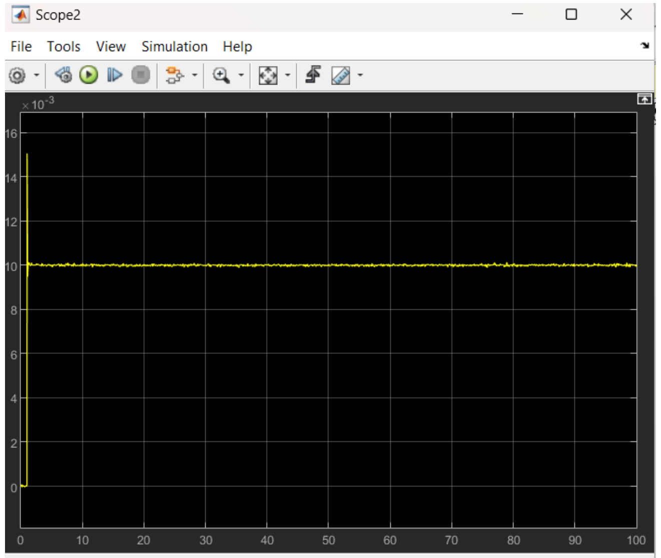
Resultados de Control



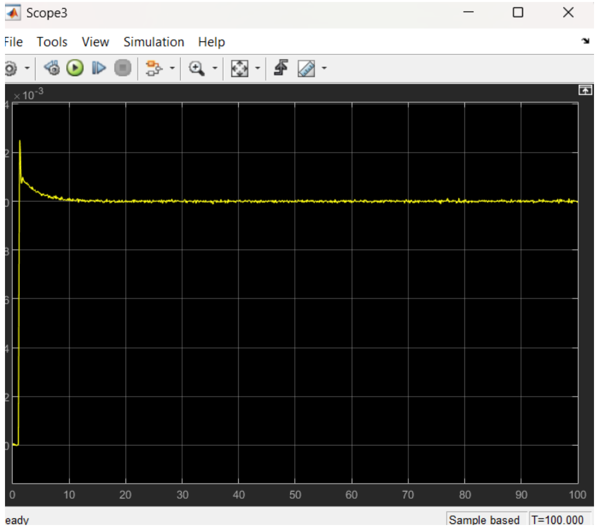
ALTURA



ROLL



PITCH



YAW