# Βοηθητικό Υλικό

Κώδικας σε MATLAB® by MathWorks®

Αλίνα Εκτάμι, PhD

Ο παρών κώδικας υλοποιεί ένα μοντέλο μη γραμμικής δυναμικής ενός μη επανδρωμένου ιπτάμενου οχήματος (UAV) και εφαρμόζει ελεγκτή τύπου PID για την επίτευξη ελέγχου θέσης και στάσης. Πραγματοποιείται προσομοίωση πτήσης με εναλλαγές φάσεων απογείωσης, αιώρησης και προσγείωσης, ενώ παράγονται γραφήματα που παρουσιάζουν την εξέλιξη των μεταβλητών κατάστασης, των ταχυτήτων, των γωνιών και των ροπών ελέγχου.

Το έγγραφο προορίζεται ως βοηθητικό υλικό για το Μάθημα 8.

## Κώδικας MATLAB®

```
% Script for PID simulations

close all; clear; clc;

g = 9.81;
m = @(t) 5.89;
% m = @(t) 5.89 - 0.5*heaviside(t - 80) + 0.5*heaviside(t - 120);
% Inertia
Ix = 0.19788093606;
Iy = 0.36886847425;
Iz = 0.17265001195;
% Drag coefficient
b_drag = 0.962;
% Gains
Kp_alt = 12;     Kd_alt = 5;
Kp_x = 2.0;      Kd_x = 1.2;
Kp_y = 2.0;      Kd_y = 1.2;
Kp_a = diag([6.0 6.0 2.5]);
Kd_a = diag([2.0 2.0 1.0]);
Ki_a = diag([0.3 0.3 0.0]);    % No integral on yaw
```

```
21
22 flag_reg = 1;
23
24 t_A = 2;  % t_A: takeoff init.
25 t_B = 35; % t_B transition init
26 t_C = 70; % t_C: landing init
27 t_D = 85; % t_D: payload acquisition and new takeoff init
28 t_E = 110;% t_E: transition init
29 t_F = 150;% t_F: final landing init
30
31 if flag_reg == 1
32     % Regulation trajectory (takeoff-hover-land)
33     x_des = @(t) 10*heaviside(t-t_B) - 10*heaviside(t-t_E);
34     y_des = @(t) 10*heaviside(t-t_B) - 10*heaviside(t-t_E);
35     z_des = @(t) 15*heaviside(t-t_A) - 15*heaviside(t-t_C) + ...
36                  15*heaviside(t-t_D) - 15*heaviside(t-t_F);
37     % reference vector same length as expected by the ODE
38     x_ref = @(t) [x_des(t); y_des(t); z_des(t); 0; 0; 0; 0; 0; 0; 0;
          0; 0];
39 else
40     % Example sinusoidal trajectory for tracking
41     x_des = @(t) 5*cos(0.2*t);
42     y_des = @(t) 5*sin(0.2*t);
43     z_des = @(t) 10 + 2*sin(0.1*t);
44     xd_des = @(t) -5*0.2*sin(0.2*t);
45     yd_des = @(t)  5*0.2*cos(0.2*t);
46     zd_des = @(t)  0.2*cos(0.1*t);
47     x_ref = @(t) [x_des(t); y_des(t); z_des(t); xd_des(t); yd_des(t);
           zd_des(t); 0; 0; 0; 0; 0; 0];
48 end
49
50 x0 = zeros(15,1);
51 tspan = [0 200];
52 x0 = [0; 0 ; 0; 0; 0; 0; 0; 0; 0; 0; 0; 0; 0; 0; 0;];
53
54 % Definion of the function handle for quadcopter dynamics:
55 f = @(t,x) UAV_PID_ode_Final(t, x, m, g,  Ix, Iy, Iz, b_drag, x_ref,
      Kp_alt,...
56     Kd_alt, Kp_x, Kd_x, Kp_y, Kd_y, Kp_a, Kd_a, Ki_a);
57 opts = odeset('RelTol', 10^-10, 'AbsTol', 10^-9);
58
```

```
59 [t_PID, x_PID] = ode15s(f, tspan, x0, opts);
60 u_PID(size(t_PID),4) = 0;
61
62 for i=1:1:size(t_PID)
63     temp = x_ref(t_PID(i));
64     e_pos = temp(1:3) - x_PID(i,1:3)';
65     f_tot = Kp_alt * e_pos(3) - Kd_alt * x_PID(i,6);
66     phi_d = Kp_y * e_pos(2) - Kd_y * x_PID(i,5);
67     theta_d = Kp_x * e_pos(1) - Kd_x * x_PID(i,4);
68     phi_d = 0;
69     theta_d = 0;
70     e_a = [phi_d;theta_d;0] -  x_PID(i,7:9)';
71     temp =  Kp_a * e_a - Kd_a * x_PID(i,10:12)' + Ki_a * x_PID(i
           ,13:15)';
72     u_PID(i,:) = [f_tot,temp'];
73 end
```

```
% Plots:

% Translation:
    figure(1)
    subplot(3,1,1);
    plot(t_PID,x_PID(:,1), 'LineWidth', 2 ,'LineStyle','-','Color','b
        ');
    legend('x', 'x_{d}', 'x_{PID}');
    subplot(3,1,2);
    plot(t_PID, x_PID(:,2), 'LineWidth', 2, 'LineStyle','-','Color','
        g');
    legend('y','y_{d}', 'y_{PID}')
    subplot(3,1,3);
    plot(t_PID, x_PID(:,3), 'LineWidth', 2, 'LineStyle','-', 'Color',
        'r');
    legend('z','z_d', 'z_{PID}');

% Control Inputs:
    figure(2)
    subplot(4,1,1);
    plot(t_PID, u_PID(:,1), 'LineWidth', 2, 'Color','b');
    legend('Total␣Thrust', 'Upper␣Bound', 'Lower␣Bound', 'Tot␣Thrust␣
        PID');
    subplot(4,1,2);
    plot(t_PID, u_PID(:,2), 'LineWidth', 2, 'Color','g');
    legend('x-torque', 'Upper␣Bound', 'Lower␣Bound', 'x-torque␣PID');
    subplot(4,1,3);
    plot(t_PID, u_PID(:,3), 'LineWidth', 2, 'Color','r');
    legend('y-torque', 'Upper␣Bound', 'Lower␣Bound', 'y-torque␣PID');
    subplot(4,1,4);
    plot(t_PID, u_PID(:,4), 'LineWidth', 2, 'Color','m');
    legend('z-torque', 'Upper␣Bound', 'Lower␣Bound', 'z-torque␣PID');

    figure(3)

    subplot(3,1,1);
    plot(t_PID, x_PID(:,4), 'LineWidth', 2,'Color','b');
    title('x-velocity␣PID')
    subplot(3,1,2);
    plot(t_PID, x_PID(:,5), 'LineWidth', 2,'Color','g');
    title('y-velocity␣PID')
```

```
38      subplot(3,1,3);
39      plot(t_PID, x_PID(:,6), 'LineWidth', 2,'Color','r');
40      title('z-velocity␣PID')
41
42      figure(4)
43      subplot(3,1,1);
44      plot(t_PID, x_PID(:,7), 'LineWidth', 2,'Color','b');
45      xlabel('Time␣[s]');
46      ylabel('\phi␣[rad]');
47      legend('Roll');
48      title('Roll␣Evolution␣PID');
49      subplot(3,1,2);
50      plot(t_PID, x_PID(:,8), 'LineWidth', 2,'Color','g');
51      xlabel('Time␣[s]');
52      ylabel('\theta␣[rad]');
53      legend('Pitch');
54      title('Pitch␣Evolution␣PID');
55      subplot(3,1,3);
56      plot(t_PID, x_PID(:,9), 'LineWidth', 2,'Color','r');
57      xlabel('Time␣[s]');
58      ylabel('\psi␣[rad]');
59      legend('Yaw');
60      title('Yaw␣Evolution␣PID');
61
62      figure(5)
63      subplot(3,1,1);
64      plot(t_PID, x_PID(:,10), 'LineWidth', 2,'Color','b');
65      xlabel('Time␣[s]');
66      ylabel('$\dot{\phi}$␣[rad/s]','Interpreter','latex');
67      legend('Roll␣Rate');
68      title('Roll␣Rate␣Evolution␣PID');
69      subplot(3,1,2);
70      plot(t_PID, x_PID(:,11), 'LineWidth', 2,'Color','g');
71      xlabel('Time␣[s]');
72      ylabel('$\dot{\theta}$␣[rad/s]','Interpreter','latex');
73      legend('Pitch␣Rate');
74      title('Pitch␣Rate␣Evolution␣PID');
75      subplot(3,1,3);
76      plot(t_PID, x_PID(:,12), 'LineWidth', 2,'Color','r');
77      xlabel('Time␣[s]');
78      ylabel('$\dot{\psi}$␣[rad/s]','Interpreter','latex');
```

```
79      legend('Yaw␣Rate');
80      title('Yaw␣Rate␣Evolution␣PID');
81
82      if (flag_reg  == 1)
83      else
84          figure(6)
85          plot(x_PID(:,1),x_PID(:,2),'Color','g','LineWidth',2,'
                LineStyle',':')
86          legend('Ref','MPC','PID')
87          title('XY␣tracking␣accuracy')
88      end
```

```
% The ODE Function
function xdot = UAV_PID_ode_Final(t, x, m, g,  Ix, Iy, Iz, b_drag,
   x_ref, Kp_alt,...
    Kd_alt, Kp_x,Kd_x, Kp_y, Kd_y, Kp_a, Kd_a, Ki_a)

mass_t = m(t);
R_w = [cos(x(9))*cos(x(8)), -sin(x(9))*cos(x(7))+...
       cos(x(9))*sin(x(8))*sin(x(7)), sin(x(9))*sin(x(7))+...
       cos(x(9))*sin(x(8))*cos(x(7));
       sin(x(9))*cos(x(8)), cos(x(9))*cos(x(7))+...
       sin(x(9))*sin(x(8))*sin(x(7)),...
       -cos(x(9))*sin(x(7))+sin(x(9))*sin(x(8))*cos(x(9));
       -sin(x(8)), cos(x(8))*sin(x(7)), cos(x(8))*cos(x(7))];

temp = x_ref(t);
e_pos = temp(1:6) - x(1:6);
f_tot = Kp_alt * e_pos(3) + Kd_alt * e_pos(6);
phi_d = Kp_y * e_pos(2) + Kd_y * e_pos(5);
theta_d = Kp_x * e_pos(1) + Kd_x * e_pos(4);
e_a = [phi_d;theta_d;0] - x(7:9);
u_a =  Kp_a * e_a - Kd_a * x(10:12) + Ki_a * x(13:15);

Gravity = [0, 0, -mass_t*g]';
F_d = -b_drag * x(4:6);
F_translational =  R_w * [0; 0; f_tot + mass_t*g] + Gravity + F_d;
acc = (1/mass_t)  *  F_translational;

Q_BW = [1, sin(x(9)) * tan(x(8)), cos(x(7)) * tan(x(8));
        0, cos(x(9)), -sin(x(7));
        0, sin(x(9))/cos(x(8)), cos(x(9))/cos(x(8))];
Q_WB = Q_BW';
OMEGA1 = Q_WB * [x(10); x(11); x(12)];
a1 = [Ix, 0, 0; 0, Iy, 0; 0, 0, Iz] * OMEGA1;
OMEGA2 = cross(OMEGA1,a1);
tau = [Ix^-1, 0, 0; 0, Iy^-1, 0; 0, 0, Iz^-1] * (u_a - OMEGA2);

xdot = [x(4:6); acc; x(10:12); tau; e_a];
end
```