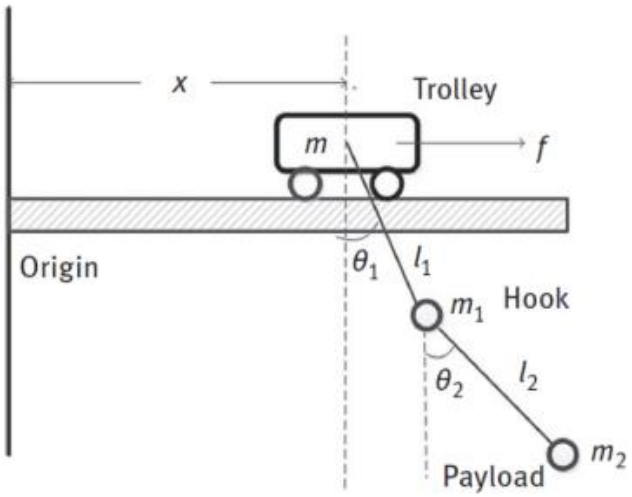
Name: *Pham Ngoc Huy*

Student ID: *20424004*

Questions: (70 pts)

1. Use the book, design the model of the crane as **double pendulum** and find the **dynamic equations**. Change the equations as the **state-space model** [10 pts]

2-Dimension Crane:



<u>Model of the crane as double pendulum – dynamic equations:</u>

- <u>Concerning the force f applied to the trolley:</u>

$$(m_0 + m_1 + m_2)\ddot{x} + (m_1 + m_2)l_1\ddot{\theta}_1 cos\theta_1 - (m_1 + m_2)l_1\dot{\theta_1}^2 cos\theta_1 - m_2 l_2\dot{\theta_2}^2 sin\theta_2 = f$$

- <u>Concerning no inputs applied to the hook:</u>

$$(m_1 + m_2)l_1\ddot{x}cos\theta_1 + (m_1 + m_2)l_2^2\ddot{\theta}_2 \cos(\theta_1 - \theta_2) + m_2 l_1 l_2\dot{\theta}_2^2 \sin(\theta_1 - \theta_2) + (m_1 + m_2)g l_1 sin\theta_1 = 0$$

- <u>Concerning no inputs applied to the payload:</u>

$$m_2 l_2\ddot{x}cos\theta_2 + m_2 l_1 l_2\ddot{\theta}_1 \cos(\theta_1 - \theta_2) + m_2 l_2^2\ddot{\theta}_2 - m_2 l_1 l_2\dot{\theta}_1 \sin(\theta_1 - \theta_2) + m_2 g l_2 sin\theta_2 = 0$$

Where:

m_0: trolley mass (kg)

m_1: hook mass (kg)

m_2: payload mass (kg)

x: trolley position

l_1: cable length between trolley and hook (m)

l_2: cable length between hook and payload (m)

f: the crane drive force can move along x-direction
g: the gravitational acceleration (m/s^2)

θ_1: hook angle with respect to the vertical line (rad)

θ_2: payload angle with respect to the vertical line (rad)

State-space model:

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = f_1(x) + b_1(x)u$$
$$\dot{x}_3 = x_4$$
$$\dot{x}_4 = f_2(x) + b_2(x)u$$
$$\dot{x}_5 = x_6$$
$$\dot{x}_6 = f_3(x) + b_3(x)u$$

x is defined by $[x_1, x_2, x_3, x_4, x_5, x_6]^T$

x_1 = x

$x_3 = \theta_1$

$x_5 = \theta_2$

x_2: is the trolley velocity

x_4: is the angular velocity of the hook

x_6: is the angular velocity of the payload

$u = f$: denote as a control input

$f_i(x)$ *and* $b_i(x)$ *with* $(i = 1,2,3)$: are nonlinear functions of the vector x, formulated by:

- $f_i = \frac{\Gamma_i}{\Delta}$
- $b_i = \frac{T_i}{\Delta}$

Where:

$$\Delta = (m_1 + m_2)m_2 l_1^2 l_2^2[(m_0 + m_1 + m_2) - (m_1 + m_2)\cos^2(x_3)] - m_2^2 l_1^2 l_2^2[(m_1 + m_2)\cos^2(x_5) + (m_0 + m_1 + m_2)\cos^2(x_3 - x_5) - 2(m_1 + m_2)\cos(x_3)\cos(x_5)\cos(x_3 - x_5))$$

$$\Gamma_1 = [(m_1 + m_2)m_2 l_1^2 l_2^2 - m_2^2 l_1^2 l_2^2 \cos^2(x_3 - x_5)][(m_1 + m_2)l_1 x_4^2 \sin(x_3) + m_2 l_2 x_6^2 \sin(x_5)] + [(m_1 + m_2)m_2 l_1 l_2^2 \cos(x_3) - m_2^2 l_1 l_2^2 \cos(x_5)\cos(x_3 - x_5)][m_2 l_1 l_2 x_6^2 \sin(x_3 - x_5)) + (m_1 + m_2)g l_1 \sin(x_3)] + [(m_1 + m_2)m_2 l_1^2 l_2 \cos(x_5) - m_2 l_1^2 l_2 \cos(x_3)\cos(x_3 - x_5)][-m_2 l_1 l_2 x_4^2 \sin(x_3 - x_5) + m_2 g l_2 \sin(x_5)]$$

$$T_1 = (m_1 + m_2)m_2 l_1^2 l_2^2 - m_2^2 l_1^2 l_2^2 \cos^2(x_3 - x_5)$$

$$\Gamma_2 = [m_2^2 l_1 l_2^2 \cos(x_5)\cos(x_3 - x_5) - (m_1 + m_2)m_2 l_1 l_2^2 \cos(x_3)][(m_1 + m_2)l_1 x_4^2 \sin(x_3) + m_2 l_2 x_6^2 \sin(x_5)] + [m_2^2 l_2^2 \cos^2(x_5) - (m_0 + m_1 + m_2)m_2 l_2^2][m_2 l_1 l_2 x_6^2 \sin(x_3 - x_5) + (m_1 + m_2)g l_1 \sin(x_3)] + [(m_0 + m_1 + m_2)m_2 l_1 l_2 \cos(x_3 - x_5) - (m_1 + m_2)m_2 l_1 l_2 \cos(x_3)\cos(x_5)][-m_2 l_1 l_2 x_4^2 \sin(x_3 - x_5) + m_2 g l_2 \sin(x_5)]$$

$$T_2 = m_2^2 l_1^2 l_2^2 \cos(x_5)\, cost(x_3 - x_5) - (m_1 + m_2)m_2 l_1 l_2^2 \cos(x_3)$$

$$\Gamma_3 = \big[\{(m_1 + m_2)m_2 l_1^2 l_2[\cos(x_3)\cos(x_3 - x_5) - \cos(x_5)]\}[(m_1 + m_2)l_1 x_4^2 \sin(x_3) + m_2 l_2 x_6^2 \sin(x_5)] + [(m_0 + m_1 + m_2)m_2 l_1 l_2 \cos(x_3 - x_5) - (m_1 + m_2)m_2 l_1 l_2 \cos(x_3)\cos(x_5)]\big[[m_2 l_1 l_2 x_6^2 \sin(x_3 - x_5)) + (m_1 + m_2)g l_1 si\, n(x_3)\big] + [(m_1 + m_2)^2 l_1^2 \cos^2(x_3) - (m_0 + m_1 + m_2)(m_1 + m_2)l_1^2] * [-m_2 l_1 l_2 x_4^2 \sin(x_3 - x_5) + m_2 g l_2 \sin(x_5)]\big]$$

$$\mathrm{T}_3 = (m_1 + m_2)m_2 l_1^2 l_2 cos(x_3)\cos(x_3 - x_5) - (m_1 + m_2)m_2 l_1^2 l_2 cos(x_5)$$

2. Design of adaptive optimal PID control for the crane model. Discuss the simulation results [60 pts].

PID control:

$$u = K_p e + K_i \int e\, dt + K_d \frac{de}{dt}$$

Where:

$$e \equiv x_d - x: is\ the\ control\ error$$

$$K_p, K_i, K_d: are\ the\ positive\ bounded\ control\ gains$$

Adaptive optimal PID:

I propose an intelligent controller based on the ordinary PID controller. The main idea is that I will collect the current control errors to make proper changes on the control gains. After the error gets into a predefined region, the gains will be decreased to release the control system.

$$\begin{cases} \dot{K_p} = \beta_p(e^2 sgn(e^2 - e_0)) \\ \dot{K_i} = \beta_i(e^{\frac{1}{3}}(\int e\, dt)sgn(e^2 - e_0)) \\ \dot{K_d} = \beta_d(e^{\frac{5}{3}}\dot{e}sgn(e^2 - e_0)) \end{cases}$$

Where:

$$sgn(e^2 - e_0) = \begin{cases} 0 & (e^2 - e_{0)} = 0 \\ 1 & (e^2 - e_0) > 0 \\ -1 & (e^2 - e_0) < 0 \end{cases}$$

$$\beta_p, \beta_d, \beta_i : are\ positive\ learning\ gains$$

$$e_0: is\ a\ predefined\ steady - state\ error$$

Note:

The exponent values are chosen mainly for their observational effect, rather than strict theoretical derivation. Each exponent is selected to suit the nature of its corresponding gain term:

- K_p: The squared error ensures positivity and stronger response to large deviations, regardless of error sign.
- K_i : A fractional exponent dampens the effect of error, allowing slower, more stable adaptation and reducing the risk of wind-up.
- K_d: A moderate exponent balances sensitivity to rapid error changes while avoiding overreaction to noise.

In this adaptation rule, I increase the proportional gain K_p when the control error exceeds the expected region. When the accumulated error shares the same sign as the current error, this indicates the system requires additional energy to eliminate the offset control error, so I increase the integral control gain K_i (and decrease it in the opposite case). Furthermore, I decrease the derivative control gain K_d when the error and its time derivative show opposite signs, indicating the closed-loop system is converging. Notably, the gains adapt at different rates depending on how far the current state is from the target region.

The Crane Model System:

Mass Matrix (M)

The mass matrix M captures the inertial properties of the system, including how masses are coupled through geometric relationships. The diagonal elements represent the effective inertias of the cart and both pendulums, while off-diagonal elements represent inertial coupling between different parts of the system. These coupling terms depend on the pendulum angles and create the complex dynamics we observe.

$$M(q) = \begin{bmatrix} m + m_1 + m_2 & (m_1 + m_2)l_1 cos\theta_1 & m_2 l_2 cos\theta_2 \\ (m_1 + m_2)l_1 cos\theta_1 & (m_1 + m_2)l_1^2 & m_2 l_1 l_2 cos(\theta_1 - \theta_2) \\ m_2 l_2 cos\theta_2 & m_2 l_1 l_2 cos(\theta_1 - \theta_2) & m_2 l_2^2 \end{bmatrix}$$

Coriolis Matrix (C)

The Coriolis matrix accounts for centrifugal and Coriolis forces that emerge from the system's motion. These forces depend on both position (angles) and velocities. They're particularly important in multi-body systems like double pendulums, as they represent how energy transfers between different modes of motion. The sine terms indicate the nonlinear nature of these forces.

$$C(q, \dot{q}) = \begin{bmatrix} 0 & -(m_1 + m_2)l_1 \dot{\theta}_1 sin\theta_1 & -m_2 l_2 \dot{\theta}_2 sin\theta_2 \\ 0 & 0 & m_2 l_1 l_2 \dot{\theta}_2 sin(\theta_1 - \theta_2) \\ 0 & -m_2 l_1 l_2 \dot{\theta}_1 sin(\theta_1 - \theta_2) & 0 \end{bmatrix}$$

Gravity Vector (G)

The gravity vector represents the gravitational forces acting on the system. Note that only the pendulums are directly affected by gravity (the cart moves horizontally), which is why the first

element is zero. The gravitational forces depend on the *sine* of the pendulum angles, creating a restoring force toward the downward equilibrium position.

$$G(q) = \begin{bmatrix} 0 & (m_1 + m_2)gl_1 sin\theta_1 & m_2 gl_2 sin\theta_2 \end{bmatrix}^T$$

Control Implementation

The control force f is calculated by the adaptive PID controller and applied only to the cart

$$T = \begin{bmatrix} f \\ 0 \\ 0 \end{bmatrix}$$

Integration Method

The equation uses Euler method to advance the system state. First, accelerations are calculated by inverting the mass matrix, then velocities are updated, and finally positions are updated

$$\ddot{q} = M^{-1}(T - G - C\dot{q})$$

$$\dot{q_{next}} = \dot{q} + dt * \dot{q}$$

$$q_{next} = q + dt * \dot{q_{next}}$$

$$s = q_{next}(1)$$

$$v = \dot{q}_{next}(1)$$

$$\theta_1 = q_{next}(2)$$

$$\dot{\theta_1} = \dot{q}_{next}(2)$$

$$\theta_2 = q_{next}(3)$$

$$\dot{\theta_2} = \dot{q}_{next}(3)$$

$$t = t + dt$$

Figure 1: System States Response (Cart Position, Cart Velocity, Pendulum Angles, and Angular Velocities)

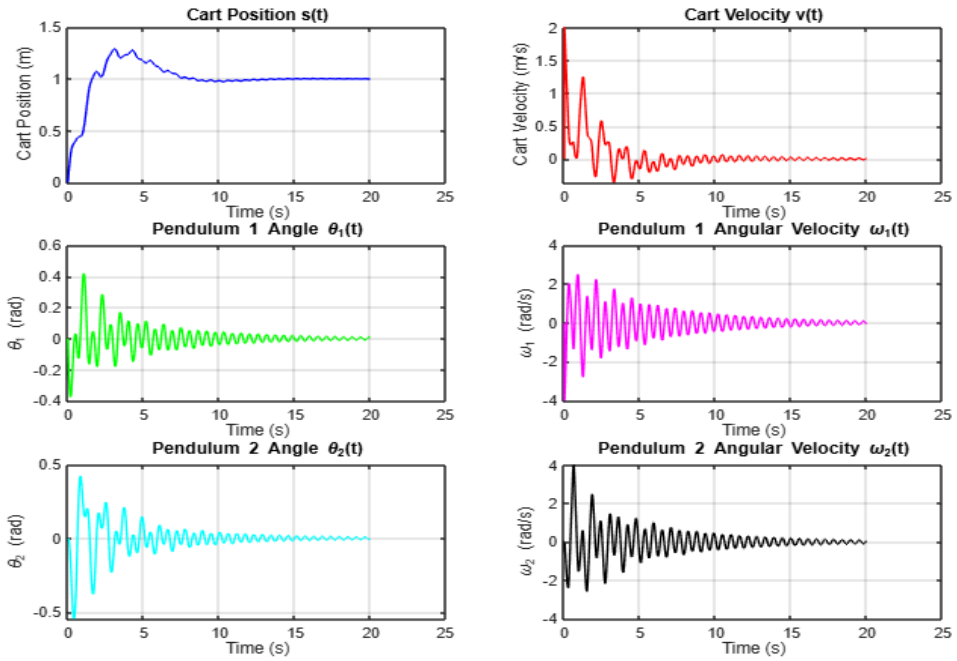
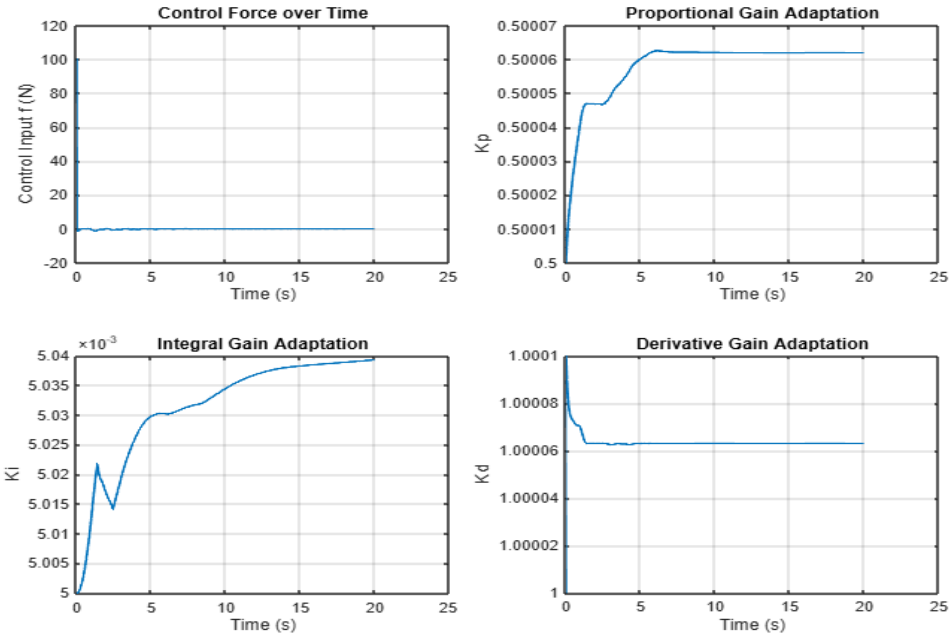


Figure 2: Control Input and Adaptive PID Gain Evolution (Control Force, Proportional, Integral, and Derivative Gain Adaptation)



Discussion:

The simulation results demonstrate an effective adaptive PID controller for a double pendulum crane system. The controller successfully positions the cart at the 1m target with minimal oscillations, as shown in *Figure 1*, where all pendulum movements stabilize within approximately 15 seconds. *Figure 2* reveals the adaptation mechanism: K_p quickly increases to improve response time, K_i gradually rises to eliminate steady-state error, and K_d makes minor adjustments to dampen oscillations. This parameter evolution directly correlates with the system behavior observed in *Figure 1*. The initial K_p increase drives rapid movement toward the setpoint (with slight overshoot), while $K_i's$ progressive growth ensures precise final positioning. The learning rates and sign-based adaptation law (using the $e^2 - e_0$ threshold) create an intelligent tuning mechanism that becomes less aggressive as the system approaches stability, preventing parameter drift while maintaining responsive control. The control force remains within appropriate bounds throughout the operation as displayed in *Figure 2*, demonstrating an efficiently tuned system that balances speed of response with stability. To achieve these results illustrated in *Figure 1* and *Figure 2*, careful parameter tuning was necessary, including adjustments to the initial PID gains (K_p, K_i, K_d) and the learning rate parameters that govern how quickly each gain adapts to system errors.

MATLAB CODES:

```
clear all;

close all;


% Crane parameters

len1 = 0.5;

len2 = 0.5;

mass = 0.5;

mass1 = 0.25;

mass2 = 0.5;

g = 9.81;


% Initial values

s = 0;

v = 0;

th1 = 0;
```

```
th2 = 0;
w1 = 0;
w2 = 0;
f = 0;
t = 0;

dt = 0.01;
xend = 1;

% PID initial parameters
Kp = 0.5;
Ki = 0.005;
Kd = 1;

% Adaptation rates
beta_p = 0.0001;
beta_i = 0.00005;
beta_d = 0.0001;

% Storage of states
sdata = s;
vdata = v;
th1data = th1;
th2data = th2;
w1data = w1;
w2data = w2;
fdata = 0;
tdata = t;
```

```
% Storage for PID parameters
Kp_data = Kp;
Ki_data = Ki;
Kd_data = Kd;

% Simulation loop
for i = 1:1:2000
   q = [s; th1; th2];
   dq = [v; w1; w2];

   M = [mass+mass1+mass2, (mass1+mass2)*len1*cos(th1), mass2*len2*cos(th2);
      (mass1+mass2)*len1*cos(th1), (mass1+mass2)*len1^2, mass2*len1*len2*cos(th1-th2);
      mass2*len2*cos(th2), mass2*len1*len2*cos(th1-th2), mass2*len2^2];

   C = [0, -1*(mass1+mass2)*len1*w1*sin(th1), -1*mass2*len2*w2*sin(th2);
      0, 0, mass2*len1*len2*w2*sin(th1-th2);
      0, -1*mass2*len1*len2*w1*sin(th1-th2), 0];

   G = [0; (mass1+mass2)*g*len1*sin(th1); mass2*g*len2*sin(th2)];

   [f, Kp, Ki, Kd] = adaptivePIDcontroller(Kp, Kd, Ki, xend, s, dt, beta_p, beta_d, beta_i);

   T = [f; 0; 0];

   ddq = pinv(M) * (T - G - C*dq);

   dq_next = dq + dt * ddq;
   q_next = q + dt * dq_next;
```

```
    s = q_next(1);
    v = dq_next(1);
    th1 = q_next(2);
    w1 = dq_next(2);
    th2 = q_next(3);
    w2 = dq_next(3);
    t = t + dt;

    tdata = [tdata, t];
    fdata = [fdata, f];
    vdata = [vdata, v];
    th1data = [th1data, th1];
    th2data = [th2data, th2];
    w1data = [w1data, w1];
    w2data = [w2data, w2];
    sdata = [sdata, s];

    Kp_data = [Kp_data, Kp];
    Ki_data = [Ki_data, Ki];
    Kd_data = [Kd_data, Kd];
end

figure;
subplot(3,2,1);
plot(tdata, sdata, 'b');
xlabel('Time (s)');
ylabel('Cart Position (m)');
title('Cart Position s(t)');
grid on;
```

```
subplot(3,2,2);
plot(tdata, vdata, 'r');
xlabel('Time (s)');
ylabel('Cart Velocity (m/s)');
title('Cart Velocity v(t)');
grid on;

subplot(3,2,3);
plot(tdata, th1data, 'g');
xlabel('Time (s)');
ylabel('\theta_1 (rad)');
title('Pendulum 1 Angle \theta_1(t)');
grid on;

subplot(3,2,4);
plot(tdata, w1data, 'm');
xlabel('Time (s)');
ylabel('\omega_1 (rad/s)');
title('Pendulum 1 Angular Velocity \omega_1(t)');
grid on;

subplot(3,2,5);
plot(tdata, th2data, 'c');
xlabel('Time (s)');
ylabel('\theta_2 (rad)');
title('Pendulum 2 Angle \theta_2(t)');
grid on;
```

```
subplot(3,2,6);
plot(tdata, w2data, 'k');
xlabel('Time (s)');
ylabel('\omega_2 (rad/s)');
title('Pendulum 2 Angular Velocity \omega_2(t)');
grid on;

figure;
subplot(2,2,1);
plot(tdata, fdata, 'LineWidth', 1.5);
xlabel('Time (s)');
ylabel('Control Input f (N)');
title('Control Force over Time');
grid on;

subplot(2,2,2);
plot(tdata, Kp_data, 'LineWidth', 1.5);
xlabel('Time (s)');
ylabel('Kp');
title('Proportional Gain Adaptation');
grid on;

subplot(2,2,3);
plot(tdata, Ki_data, 'LineWidth', 1.5);
xlabel('Time (s)');
ylabel('Ki');
title('Integral Gain Adaptation');
grid on;
```

```
subplot(2,2,4);
plot(tdata, Kd_data, 'LineWidth', 1.5);
xlabel('Time (s)');
ylabel('Kd');
title('Derivative Gain Adaptation');
grid on;

function [f, Kp, Ki, Kd] = adaptivePIDcontroller(Kp, Kd, Ki, xend, s, dt, beta_p, beta_d, beta_i)
  persistent e_prev e_int

  if isempty(e_prev)
    e_prev = 0;
    e_int = 0;
  end

  e = xend - s;
  e_int = e_int + e * dt;
  e_der = (e-e_prev)/dt;

  e_prev = e;

  f = Kp * e + Ki * e_int + Kd * e_der;

  e0 = 0.01;

  e_squared = e^2;
  if (e_squared - e0) == 0
    sgn_term = 0;
  elseif (e_squared - e0) > 0
```

```
    sgn_term = 1;
  elseif (e_squared - e0) < 0
    sgn_term = -1;
  end

  Kp = Kp + beta_p * (e_squared * sgn_term) * dt;
  Ki = Ki + beta_i * (e^(1/3) * e_int * sgn_term)* dt;
  Kd = Kd + beta_d * (e^(5/3) * e_der * sgn_term)*dt;


end
```

Please send your works to my email: phu.dx@vgu.edu.vn include:

A word's file with answer of all questions. Insert ALL MATLAB CODES into the word's file + figures (use format of matlab .fig).

IF YOUR RESULTS ARE COPIED FROM YOUR FRIENDS, THIS GRADE WILL BE CANCELLED FOR ALL QUESTIONS.