Cairo University - Faculty Of Engineering
Computer Engineering Department
Control Engineering - Spring 2025

Two-Ttank System

Name	ID
Ahmed Hamdy	9220032
Abd El-Rahman Mostafa	9220475
Mohammed Khater	9220713

Delivered to

Prof. Ragia Badr

Dr. Meena Elia

Eng. Hassan El-Menier

Part 1: Dynamic Equations and Block Diagram

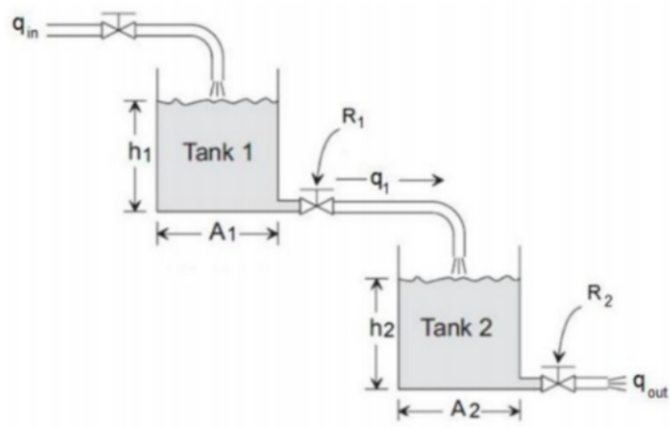


Figure 1: Two-tank System

Dynamic Equations

For Tank 1:

$$q_{\text{in}}(t) - q_1(t) = A_1 \frac{dh_1}{dt} \quad \longrightarrow \quad Q_{\text{in}}(S) - Q_1(S) = A_1 S H_1(S) \tag{1}$$

$$q_1(t) = \frac{h_1(t) - h_2(t)}{R_1} \quad \longrightarrow \quad Q_1(S) = \frac{H_1(S) - H_2(S)}{R_1} \tag{2}$$

Substituting (2) into (1):

$$\frac{dh_1}{dt} = \frac{1}{A_1}\left(q_{\text{in}}(t) - \frac{h_1(t) - h_2(t)}{R_1}\right) \quad \longrightarrow \quad H_1(S) = \frac{1}{SA_1}\left(Q_{in}(S) - \frac{H_1(S) - H_2(S)}{R_1}\right) \tag{3}$$

For Tank 2:

$$q_1(t) - q_2(t) = A_2 \frac{dh_2}{dt} \quad \longrightarrow \quad Q_1(S) - Q_2(S) = A_2 S H_2(S) \tag{4}$$

Output flow:

$$q_2(t) = \frac{h_2(t)}{R_2} \quad \longrightarrow \quad Q_2(S) = \frac{H_2(S)}{R_2} \tag{5}$$

Substituting (5) into (4):

$$\frac{dh_2}{dt} = \frac{1}{A_2}\left(\frac{h_1(t) - h_2(t)}{R_1} - \frac{h_2(t)}{R_2}\right) \quad \longrightarrow \quad H_2(S) = \frac{1}{SA_2}\left(\frac{H_1(S) - H_2(S)}{R_2} - \frac{H_2(S)}{R_2}\right) \tag{6}$$

$$Q_2(S) = Q_{in}(S) - A_1SH_1(S) - A_2SH_2(S)$$

$$Q_2(S) = Q_{in}(S) - A_1S[Q_1(S)R_1 + H_2(S)] - A_2S[R_2Q_2(S)]$$

$$Q_2(S) = Q_{in}(S) - A_1SQ_2(S)R_1 - A_1A_2S^2H_2(S)R_1 - A_1SR_2Q_2(S) - A_2SR_2Q_2(S)$$

$$Q_2(S) = Q_{in}(S) - A_1SQ_2(S)R_1 - A_1A_2S^2[R_1Q_2(S)]R_1 - A_1SR_2Q_2(S) - A_2SR_2Q_2(S)$$

$$Q_2(S)\big[1 + A_1SR_1 + A_1A_2S^2R_1R_2 + A_1SR_2 + A_2SR_2\big] = Q_{in}(S)$$

Block Diagram Representation

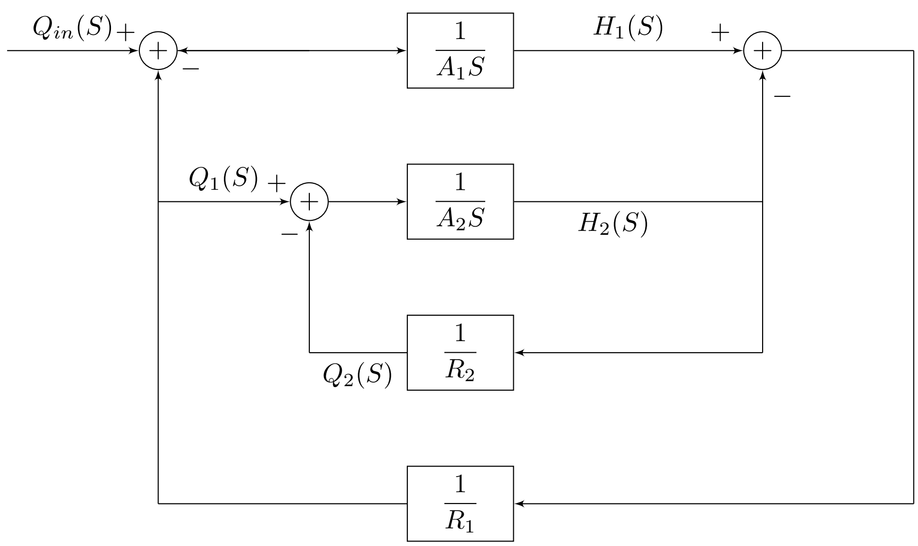


Figure 2: Block diagram of the system

Simulation

Obtaining Transfer Functions using MATLAB

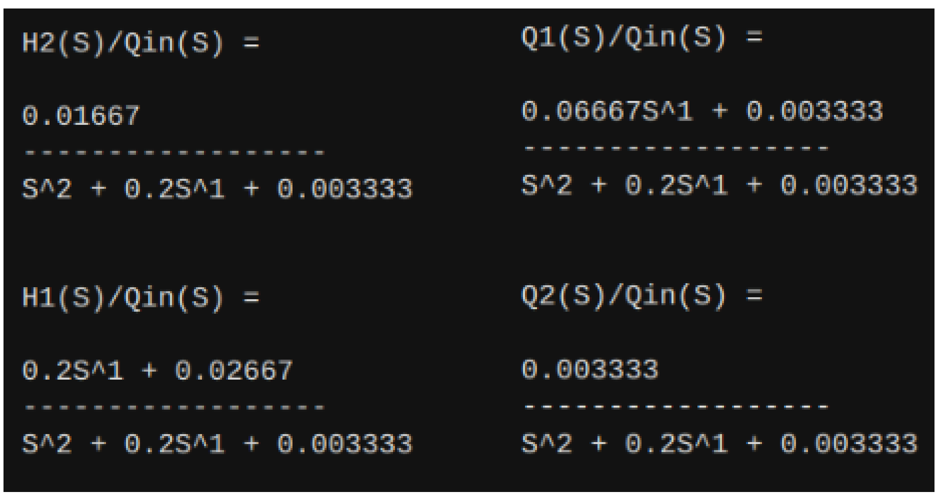
System Parameters: $A_1 = 5m^2$, $A_2 = 4m^2$, $R_1 = 3s/m^2$, $R_2 = 5s/m^2$
Using Matlab, we get the following results:

$$\frac{H_2(S)}{Q_{in}(S)} = \frac{5}{300S^2 + 60S + 1}$$

$$\frac{H_1(S)}{Q_{in}(S)} = \frac{60S + 8}{300S^2 + 60S + 1}$$

$$\frac{Q_1(S)}{Q_{in}(S)} = \frac{20S + 1}{300S^2 + 60S + 1}$$

$$\frac{Q_2(S)}{Q_{in}(S)} = \frac{1}{300S^2 + 60S + 1}$$



```
H2(S)/Qin(S) =

0.01667
------------------
S^2 + 0.2S^1 + 0.003333


H1(S)/Qin(S) =

0.2S^1 + 0.02667
------------------
S^2 + 0.2S^1 + 0.003333

Q1(S)/Qin(S) =

0.06667S^1 + 0.003333
------------------
S^2 + 0.2S^1 + 0.003333


Q2(S)/Qin(S) =

0.003333
------------------
S^2 + 0.2S^1 + 0.003333
```

Figure 3: Transfer Functions

Studying Stability

Studying the stability of the system by analyzing the denominator of TF: $300S^2 + 60S + 1 = 0$
We find that the system has 2 poles:

$$P_0 = -0.18165, \quad P_1 = -0.01835$$

We can see that both of them lie on the left half of the plane, which means that the system is stable. We also checked it using the isstable() function in MATLAB.

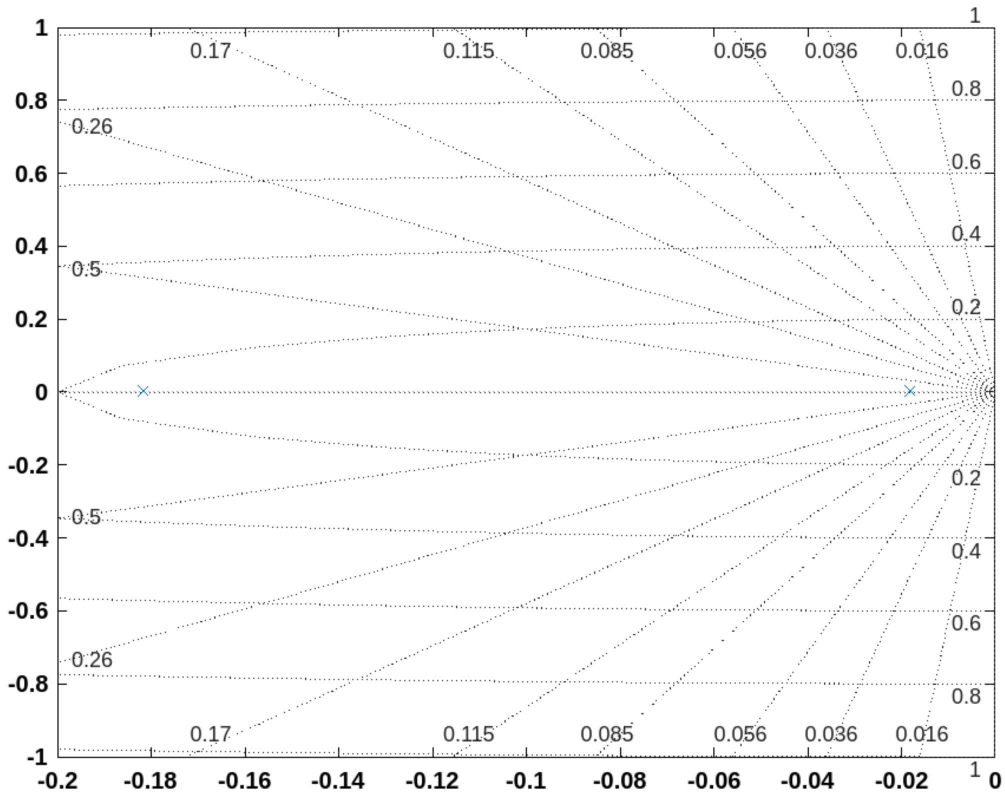


Figure 4: Pole-Zero map of the system

Simulating with $Q_{in} = 1m/s$

After simulating we get the following results

```
Steady-state values:
h1 = 7.9992 m
h2 = 4.9994 m
Q1 = 0.9999 m^3/s
Q2 = 0.9999 m^3/s
```

Figure 5: Steady State Values for System Outputs

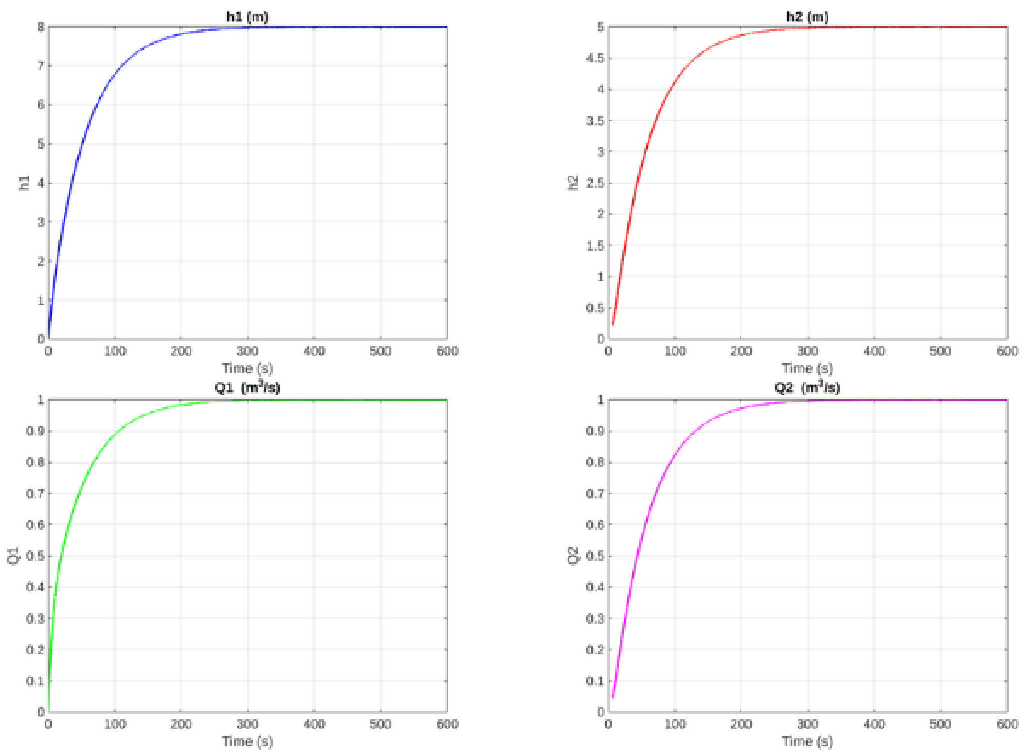


Figure 6: Time Responses for System Outputs

Theoretically

$$h_1(t \approx \infty) = \lim_{S \to 0} SH_1(S) \approx 8\ m$$

$$h_2(t \approx \infty) = \lim_{S \to 0} SH_2(S) \approx 5\ m$$

$$q_1(t \approx \infty) = \lim_{S \to 0} SQ_1(S) \approx 1\ m^3/s$$

$$q_2(t \approx \infty) = \lim_{S \to 0} SQ_2(S) \approx 1\ m^3/s$$

Part 2: Feedback Modification and Stability Check

Representation of the feedback modification block diagram

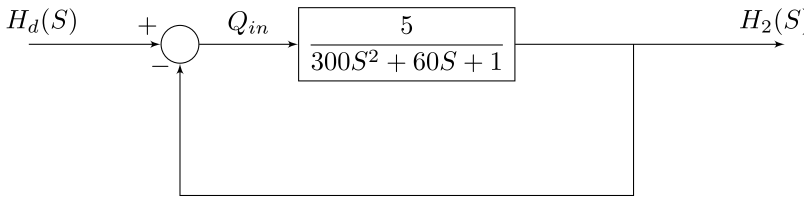


Figure 7: Block Diagram of the Feedback-Modified System

Simulating with $h_d = 5m$

After simulating we get the following results

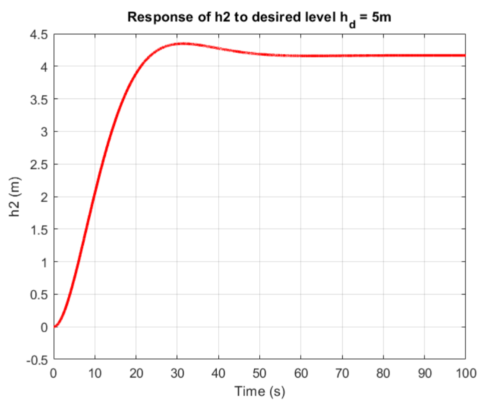


Figure 8: Steady-State Response for Feedback Stabilization

Theoretically

Natural frequency

$$\omega_n^2 = \frac{1}{50} \implies \omega_n = \frac{1}{5\sqrt{2}}$$

Damping ratio

$$2\zeta\omega_n = \frac{1}{5} \implies \zeta = \frac{1}{\sqrt{2}}$$

Phase angle

$$\zeta = \cos\psi = \frac{1}{\sqrt{2}} \implies \psi = \frac{\pi}{4}$$

Steady-state gain

$$M = \lim_{s\to 0} sH_2(s) = \frac{5}{6}h_d \approx 4.1667$$

Resonant frequency

$$\omega = \omega_n \sin\psi = \frac{1}{10}$$

Rise time

$$t_r = \frac{\pi - \psi}{\omega} = \frac{15}{2}\pi \approx 23.562\,\text{seconds}$$

Peak time

$$t_p = \frac{\pi}{\omega} = 10\pi \approx 31.416\,\text{seconds}$$

Peak overshoot

$$M_p = M\left(1 + e^{-\dfrac{\pi}{\tan\psi}}\right) = M\left(1 + e^{-\pi}\right) \approx 4.3467\,\text{meters}$$

Settling time

$$t_s = \frac{4}{\zeta\omega_n} = 40\,\text{seconds} \quad (2\%\ \text{criterion})$$

Steady-state error

$$e_{ss} = 5 - 4.1667 = 0.8333\,\text{meters}$$

adding a proportional controller

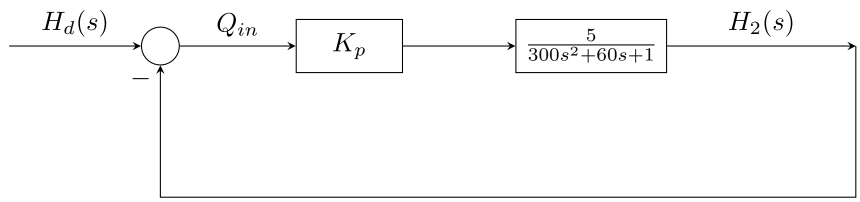
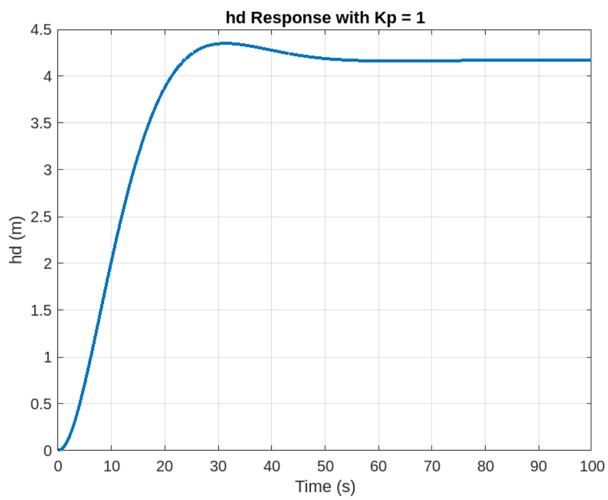


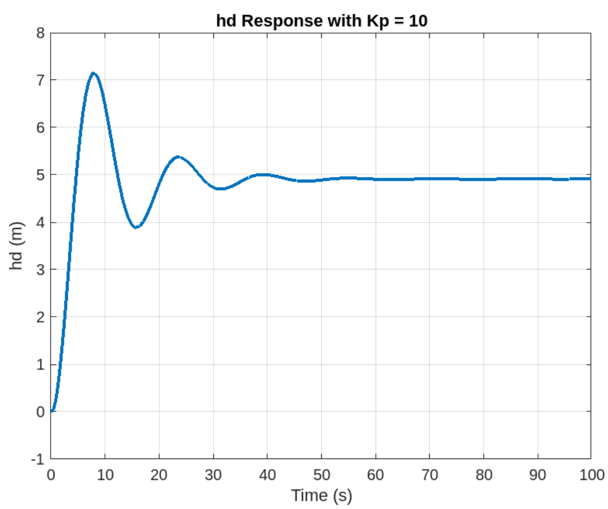
Figure 9: Block diagram of a unity feedback control system with a proportional controller K_p.



(a) Figure 9: proportional controller with kp = 1

```
--- Kp = 1 ---
Rise Time: NaN s
Peak Time: 31.4131 s
Overshoot: 0.00 %
Settling Time: NaN s
Steady-State Error: 0.8331 m
```

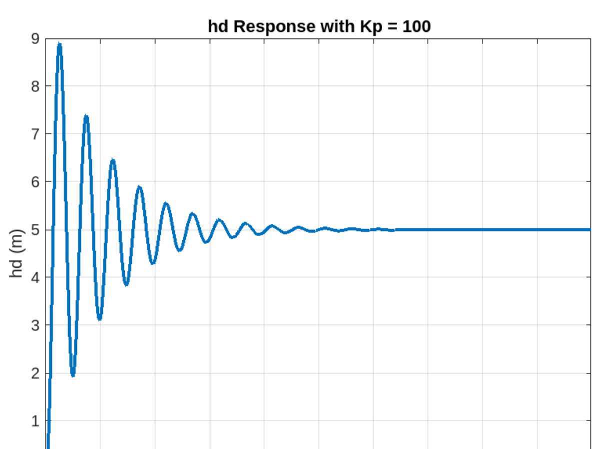
(b) Figure 10: Transient Response Output with kp = 1



(c) Figure 11: proportional controller with kp = 10

```
--- Kp = 10 ---
Rise Time: 3.0871 s
Peak Time: 7.8508 s
Overshoot: 42.74 %
Settling Time: 66.6018 s
Steady-State Error: 0.0979 m
```

(d) Figure 12: Transient Response Output with kp = 10




```
--- Kp = 100 ---
Rise Time: 0.8400 s
Peak Time: 2.4402 s
Overshoot: 78.01 %
```

Comment on Previous Results

Increasing K_p decreases the steady-state error (ESS), but also reduces the system's stability and increases the transient response parameters.

Question 9: If the actual height of the second tank walls is 6 m, is it possible to obtain a steady-state error less than 0.01 using a proportional-only controller? Why?

Answer:

For a proportional controller, the steady-state error (ESS) is given by:

$$\text{ESS} = \frac{6}{1 + 5K_p}$$

To achieve a steady-state error less than 0.01, we set:

$$\frac{6}{1 + 5K_p} < 0.01$$

Solving for K_p:

$$\frac{6}{0.01} < 1 + 5K_p \Rightarrow 600 < 1 + 5K_p \Rightarrow 599 < 5K_p$$

$$K_p > \frac{599}{5} = 119.8$$

So, we need $K_p > 119.8$ to achieve ESS ¡ 0.01.

Now, let's analyze the stability of the system. The closed-loop transfer function is:

$$\frac{5K_p}{300s^2 + 60s + (1 + 5K_p)}$$

From this, we can extract:

$$\omega_n^2 = \frac{1 + 5K_p}{300}$$

Let's use $K_p = 120$ as a practical value (just above 119.8). Then:

$$\omega_n^2 = \frac{1 + 5 \times 120}{300} = \frac{601}{300} \approx 2.003 \Rightarrow \omega_n \approx \sqrt{2.003} \approx 1.414$$

The damping ratio ζ is:

$$\zeta = \frac{60}{2 \cdot 300 \cdot \omega_n} = \frac{60}{600 \cdot 1.414} \approx \frac{60}{848.4} \approx 0.071$$

This shows that increasing K_p decreases ESS but also decreases ζ, meaning the system becomes less stable with more oscillatory transient response.

Conclusion:

Yes, it is theoretically possible to achieve ESS ¡ 0.01 using only a proportional controller by choosing $K_p > 119.8$, but this significantly reduces the damping ratio $\zeta \approx 0.071$, leading to a highly oscillatory and potentially unstable response.

Question 10: Suggest a suitable controller to eliminate the steady-state error. Then, simulate the system using your proposed controller.

Answer:

To eliminate the steady-state error, we propose using a Proportional-Integral (PI) controller:

$$C(s) = K_p \left(1 + \frac{1}{T_i s}\right)$$

This increases the system type and ensures zero steady-state error. However, we must also verify system stability. The closed-loop transfer function becomes:

$$\frac{5K_p(s + \frac{1}{T_i})}{s(300s^2 + 60s + 1) + 5K_p(s + \frac{1}{T_i})}$$

Simplifying the denominator:

$$s(300s^2 + 60s + 1) + 5K_p \left(s + \frac{1}{T_i}\right) = 300s^3 + 60s^2 + s + 5K_p s + \frac{5K_p}{T_i}$$

So the closed-loop transfer function becomes:

$$\frac{5K_p(s + \frac{1}{T_i})}{300s^3 + 60s^2 + (1 + 5K_p)s + \frac{5K_p}{T_i}}$$

To ensure stability of this third-order system, the **Routh-Hurwitz** criteria must be satisfied. One important condition is:

$$60(1 + 5K_p) > 300 \cdot \left(\frac{5K_p}{T_i}\right)$$

This ensures that the middle term product is greater than the product of the outer terms.

For the simulation, we choose:

$$K_p = 5, \quad T_i = 15$$

Simulation Results:

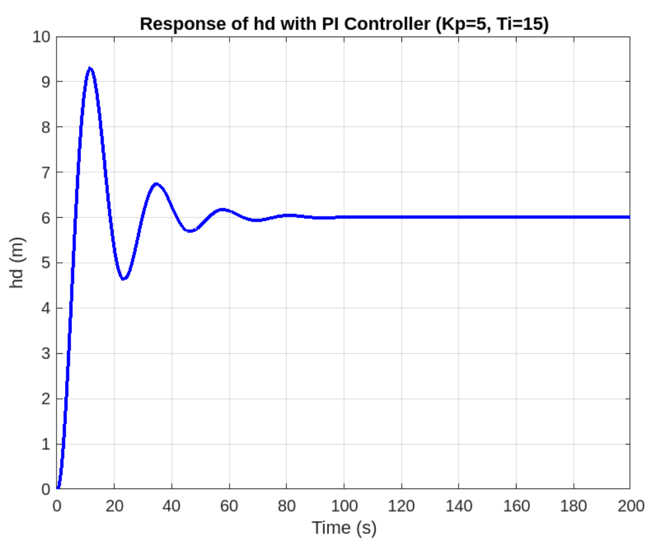


Figure 15: System Output with PI Controller

```
--- PI Controller Results (Kp = 5, Ti = 15) ---
Rise Time: 3.6638 s
Peak Time: 11.4611 s
Overshoot: 85.80 %
Settling Time: NaN s
Steady-State Error: 0.0000 m
```

Figure 16: ESS value and transient parameters values

Appendix A

Appendix

A.1 Code

```
A1 = 5; A2 = 4; R1 = 3; R2 = 5;
% State-space matrices

% Assume our state is [h1, h2] , input is Qin, outputs are Q2, Q1, H1, H2

% State Matrix
A = [-1/(A1*R1), 1/(A1*R1);         % dh1/dt equation
    1/(A2*R1), -(1/(A2*R1) + 1/(A2*R2))];  % dh2/dt equation

% Input Matrix
B = [1/A1; 0];                      % Input only affects dh1/dt

% Output Matrix
C = [0, 1/R2;                       % q_out output
    1/R1, -1/R1;                   % q1 output
    1, 0;                          % h1 output
    0, 1];                         % h2 output

% Direct Feedthrough Matrix
D = zeros(4,1);

sys = ss(A,B,C,D,'InputName','Qin','OutputName',{'Q2','Q1','H1','H2'});

transfer_functions = {
    tf(sys(4)),  % H2/Qin
    tf(sys(3)),  % H1/Qin
    tf(sys(2)),  % Q1/Qin
    tf(sys(1))   % Q2/Qin
    };

names = {'H2(S)/Qin(S)', 'H1(S)/Qin(S)', 'Q1(S)/Qin(S)', 'Q2(S)/Qin(S)'};

for k = 1:length(transfer_functions)
    tf_current = transfer_functions{k};
    [num, den] = tfdata(tf_current, 'v');

    fprintf('\n\n\n%s = \n\n', names{k});

    % Print numerator
    for i = 1:length(num)
        power = length(num)-i;
        if num(i) == 0
            continue;
        end
        if power > 0
            if num(i) == 1
                fprintf('S^%d + ', power);
            else
                fprintf('%.4gS^%d + ', num(i), power);
```

```
            end
        else
            fprintf('%.4g', num(i));
        end
    end

    % Print denominator
    fprintf('\n------------------\n');
    for i = 1:length(den)
        power = length(den)-i;
        if den(i) == 0
            continue;
        end
        if power > 0
            if den(i) == 1
                fprintf('S^%d + ', power);
            else
                fprintf('%.4gS^%d + ', den(i), power);
            end
        else
            fprintf('%.4g', den(i));
        end
    end

end

% Stability analysis
P = pole(tf(sys(4)));
fprintf('\n\n\nPoles of H2/Qin: P0 = %.4f, P1 = %.4f\n', P(1), P(2));
is_stable = isstable(tf(sys(4)));
fprintf('Is stable: %d\n', is_stable);

figure;
pzmap(tf(sys(4)));
title('Pole-Zero Map of H2/Qin');
grid on;

% Simulate response to a step input (1 m^3/s)
t = linspace(0, 500, 10000);  % 10,000 samples over 500 seconds
u = ones(size(t));            % Step input of 1 m^3/s

[y, t_out, x] = lsim(sys, u, t);

% Plot h1
figure;
plot(t_out, y(:,3), 'b', 'LineWidth', 1.5);
grid on;
title('h1 (m)');
xlabel('Time (s)');
ylabel('h1');

% Plot h2
figure;
plot(t_out, y(:,4), 'r', 'LineWidth', 1.5);
grid on;
title('h2 (m)');
xlabel('Time (s)');
ylabel('h2');

% Plot Q1
figure;
plot(t_out, y(:,2), 'g', 'LineWidth', 1.5);
grid on;
title('Q1 (m^3/s)');
xlabel('Time (s)');
ylabel('Q1');

% Plot Q2
figure;
plot(t_out, y(:,1), 'm', 'LineWidth', 1.5);
grid on;
```

```
title('Q2 (m^3/s)');
xlabel('Time (s)');
ylabel('Q2');

% Calculate steady-state values
steady_state_values = y(end, :);
fprintf('\nSteady-state values:\n');
fprintf('h1 = %.4f m\n', steady_state_values(3));
fprintf('h2 = %.4f m\n', steady_state_values(4));
fprintf('Q1 = %.4f m^3/s\n', steady_state_values(2));
fprintf('Q2 = %.4f m^3/s\n', steady_state_values(1));

% Modify the system to have a feedback with a reference signal h_d
sys_cl = feedback(sys(4,:),1);

% Simulate response to a step input (h_d = 5 meters)
t = linspace(0,100,10000);  % 10,000 samples over 100 seconds

hd = 5 * ones(size(t));

[h2_response,t_out] = lsim(sys_cl,hd,t);

% Plot h2 response
figure;
plot(t_out, h2_response, 'r', 'LineWidth', 2);
grid on;
title('Response of h2 to desired level h_d = 5m');
xlabel('Time (s)');
ylabel('h2 (m)');

info = stepinfo(h2_response, t_out, 5);  % 5 is the desired final value

fprintf('Rise time: %.4f seconds\n', info.RiseTime);
fprintf('Peak time: %.4f seconds\n', info.PeakTime);
fprintf('Maximum overshoot: %.2f%%\n', info.Overshoot);
fprintf('Settling time: %.4f seconds\n', info.SettlingTime);

% Steady-state error
ess = abs(5 - h2_response(end));
fprintf('Steady-state error (ess): %.4f meters\n', ess);

% adding the controller to the system

Kp_values = [1, 10, 100];
hd = 5 * ones(size(t));  % Desired height

for i = 1:length(Kp_values)
    Kp = Kp_values(i);

    % Closed-loop transfer function with proportional controller
    sys_cl = feedback(Kp * tf(sys(4)), 1);

    % Simulate response
    [h2_response, t_out] = lsim(sys_cl, hd, t);

    % Plot response
    figure;
    plot(t_out, h2_response, 'LineWidth', 2);
    grid on;
    title(sprintf('hd Response with Kp = %d', Kp));
    xlabel('Time (s)');
    ylabel('hd (m)');

    % Step response characteristics
    info = stepinfo(h2_response, t_out, 5);  % Final value = 5
    ess = abs(5 - h2_response(end));         % Steady-state error

    fprintf('\n\n--- Kp = %d ---\n', Kp);
    fprintf('Rise Time: %.4f s\n', info.RiseTime);
    fprintf('Peak Time: %.4f s\n', info.PeakTime);
    fprintf('Overshoot: %.2f %%\n', info.Overshoot);
```

```
    fprintf('Settling Time: %.4f s\n', info.SettlingTime);
    fprintf('Steady-State Error: %.4f m\n', ess);
end



% Define PI controller parameters
Kp = 5;
Ti = 15;

% Define PI controller transfer function: C(s) = Kp * (1 + 1/(Ti*s))
s = tf('s');
C = Kp * (1 + 1/(Ti*s));  % Alternatively: C = (Kp*Ti*s + Kp)/(Ti*s)

% Get plant G(s) = H2(s)/Qin(s)
G = tf(sys(4));  % From Qin to h2

% Closed-loop system: feedback of C(s)*G(s)
sys_cl_PI = feedback(C * G, 1);

% Simulate step response to desired h_d = 5m
t = linspace(0, 200, 10000);
hd = 6 * ones(size(t));

[h2_PI, t_out] = lsim(sys_cl_PI, hd, t);

% Plot response
figure;
plot(t_out, h2_PI, 'b', 'LineWidth', 2);
grid on;
title('Response of hd with PI Controller (Kp=5, Ti=15)');
xlabel('Time (s)');
ylabel('hd (m)');

% Analyze response characteristics
info = stepinfo(h2_PI, t_out, 5);
ess = abs(6 - h2_PI(end));

fprintf('\n--- PI Controller Results (Kp = 5, Ti = 15) ---\n');
fprintf('Rise Time: %.4f s\n', info.RiseTime);
fprintf('Peak Time: %.4f s\n', info.PeakTime);
fprintf('Overshoot: %.2f %%\n', info.Overshoot);
fprintf('Settling Time: %.4f s\n', info.SettlingTime);
fprintf('Steady-State Error: %.4f m\n', ess);
```

Listing A.1: project code