Attack-Resilient Thermal Control System

Oluwafemi Ajeigbe
EE 502 Final Project

***Abstract*—This paper investigates the design of a Kalman filter and LQR controller for a system comprising two interconnected rooms, each with one input, focusing on the system's attack resilience to disturbances. The study proposes a state-space model for the system and outlines the steps involved in developing the Kalman filter and LQR controller. The Kalman filter estimates the system's state by processing noisy measurements, while the LQR controller calculates an optimal control input that minimizes a cost function. The study aims to design a control system that regulates the temperature in both rooms, considering that the temperature in each room affects the other. The control system will have one input for each room, and we will use a Kalman filter and an LQR controller to regulate the temperature of both rooms. The paper further discusses the simulation of the system. It evaluates the controller's performance in terms of its attack resilience to disturbances.**

***Index Terms*—State Space Model, Thermal Control, Kalman Filter, Heating System.**

I. Introduction

Controlling multi-room environments is challenging in various applications, including heating, ventilation, and air conditioning (HVAC) systems, building automation, and energy management. In such systems, the temperature in each room depends on various factors such as occupancy, outdoor weather conditions, and the temperature in other rooms. Controlling the temperature in each room is vital to maintaining a comfortable and energy-efficient indoor environment. The system is modeled using a state-space representation, where the state vector includes the temperatures of both rooms, and the input vector includes the temperature setpoints for both rooms. The Kalman filter is a powerful tool for estimating the state of a system based on noisy measurements. In this project, we will design a Kalman filter to estimate the temperatures of both rooms based on temperature measurements with some measurement noise. The estimated state will then be fed to the LQR controller, which computes the optimal control input that minimizes a cost function. The LQR controller is expected to provide an optimal control strategy that results in a comfortable and energy-efficient indoor environment. The simulation of the system conducted using MATLAB. The performance of the Kalman filter and LQR controller will be evaluated using various metrics, including the steady-state error, settling time, and overshoot. The results of this project will provide insights into the design and control of multi-room environments and contribute to the development of more efficient and reliable control strategies for HVAC systems and building automation. In the following sections, we describe the dynamic modeling, state space representation, designing the Kalman filter, designing the LQR controller, simulations, and evaluating the performance.

II. Dynamic Modeling

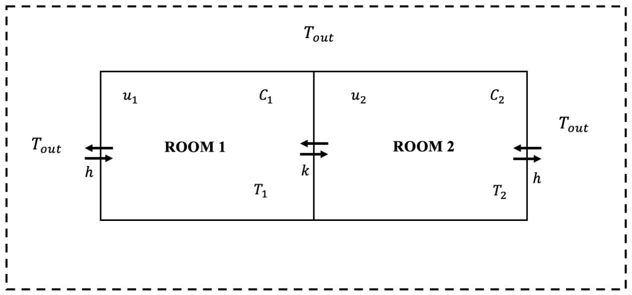


Fig. 1. Dynamic model for two room temperature

For the dynamic model described in this project, assuming that two rooms are connected, the heat can be transferred between them. Also, heat flows between the rooms and the ambient outside temperature. We model the temperature of each room as a function of time, taking into account the heat input from the input source (heater), the heat capacity of each room, and the heat flow between them. Let T_1 be the temperature of room 1, T_2 be the temperature of room 2, and let u_1 and u_2 be the heat inputs to room 1 and room 2, respectively. We write the differential equations for the temperature of each room as follows:

$$\dot{T}_1 = \frac{1}{C_1}\Big(u_1 - hA(T_1 - T_{out})\Big) - \frac{k}{C_1}\Big(T_1 - T_2\Big) \quad (1)$$

$$\dot{T}_2 = \frac{1}{C_2}\Big(u_2 - hA(T_2 - T_{out})\Big) - \frac{k}{C_2}\Big(T_2 - T_1\Big) \quad (2)$$

where C_1 and C_2 are the heat capacities of room 1 and room 2, respectively, h is the heat transfer coefficient between each room and the outside environment, A is the surface area of each room, T_{out} is the constant outside temperature, and k is the heat transfer coefficient between the two rooms. The term $hA(T - T_{out})$ represents the heat flow between the room and the outside environment due to the temperature difference, while the term $\frac{k}{C_1}(T_1-T_2)$ and $\frac{k}{C_2}(T_2-T_1)$ represent the heat flow between the two rooms due to the temperature difference.

We can represent the system using the following state-space model:

$$\dot{x}(t) = Ax(t) + Bu(t) + w(t)$$
$$y(t) = Cx(t) + v(t)$$

where $x(t)$ is the state vector, $u(t)$ is the input vector, $y(t)$ is the output vector, $w(t)$ and $v(t)$ are the process and measurement noise vectors, respectively. The matrices A, B, and C are:

$$A = \begin{bmatrix} -\frac{hA_s+k}{C_1} & \frac{k}{C_1} \\ \frac{k}{C_2} & -\frac{hA_s2+k}{C_2} \end{bmatrix}, \quad B = \begin{bmatrix} \frac{1}{C_1} & 0 \\ 0 & \frac{1}{C_2} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The state vector $x(t)$ contains the temperatures of both rooms:

$$x(t) = \begin{bmatrix} T_1(t) & T_2(t) \end{bmatrix}'$$

The input vector $u(t)$ contains the heat inputs to each room:

$$u(t) = \begin{bmatrix} u_1(t) & u_2(t) \end{bmatrix}'$$

And the output vector $y(t)$ contains the temperatures of both rooms:

$$y(t) = \begin{bmatrix} T_1(t) & T_2(t) \end{bmatrix}'$$

The vector $w(t)$ represents the process noise, which is assumed to be a zero-mean white Gaussian process with covariance matrix QN. Similarly, the vector $v(t)$ represents the measurement noise, which is also assumed to be a zero-mean white Gaussian process with covariance matrix RN. We further assume that $w(t)$ and $v(t)$ are uncorrelated.

Therefore, we can write:

$$w(t) = \begin{bmatrix} w_1(t) \ w_2(t) \end{bmatrix} \sim \mathcal{N}(0, QN)$$

and

$$v(t) = \begin{bmatrix} v_1(t) \ v_2(t) \end{bmatrix} \sim \mathcal{N}(0, RN)$$

III. LINEAR QUADRATIC REGULATOR

The next step in the design process is to find the vector of state-feedback control gains K, assuming that we have access (i.e., can measure) to all four state variables. LQR is a widely used control method in optimal control problems. The LQR method is used to control complex systems that need high performance. A system described by linear differential equations can be shown in the steady-state form given in (3) and (4). A is system matrix, B is input matrix, C is the output matrix and D is the feed-forward matrix. x is an $n \times 1$ state vector, y is the output vector and u is an $m \times 1$ input(control) vector. A conventional LQR problem is to find the Q and R matrix, which minimizes the cost function (performance index) based on the input "u". Performance index J is defined as given in (6). The control energy is represented by $u(t)^T Ru(t)$, while the transient energy is expressed as $x(t)^T Qx(t)$. Q is an $n \times n$ symmetric positive semi-definite matrix that specifies the weight associated with the states. The R is an $m \times m$ symmetric positive definite matrix that specifies the weight associated with the inputs. In Matlab, we can use the ***lqr*** command, which returns the optimal controller gain assuming a linear plant, quadratic cost function, and reference equal to zero.

$$\dot{x}(t) = Ax(t) + Bu(t), \quad t \geq 0, \quad x(0) = x_0 \tag{3}$$
$$y(t) = Cx(t) + Du(t), \quad t \geq 0 \tag{4}$$
$$u(t) = -Kx(t) \tag{5}$$
$$J = \int_0^\infty x^T(t)Qx(t) + u^T(t)Ru(t)\mathrm{d}t \tag{6}$$

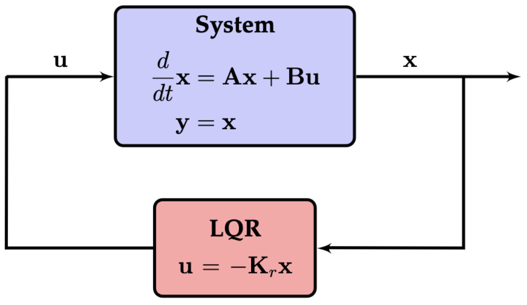

System
$\frac{d}{dt}\mathbf{x} = \mathbf{Ax} + \mathbf{Bu}$
$\mathbf{y} = \mathbf{x}$
u
x
LQR
$\mathbf{u} = -\mathbf{K}_r\mathbf{x}$

Fig. 2. Schematic of the linear quadratic regulator (LQR) for optimal full-state feedback

From linear system theory knowledge, we know that we must first verify that the system is controllable before we can design a controller. Satisfaction with this property means that we can drive the system's state anywhere we like in a finite time (under the physical constraints of the system). For the system to be completely state controllable, the controllability matrix must have rank n where the rank of a matrix is the number of linearly independent rows (or columns). The controllability matrix of the system takes the form shown below. The number n corresponds to the number of state variables of the system. In our case, $n = 2$. Adding additional terms to the controllability matrix with higher powers of the matrix A does not increase the rank of the controllability matrix, as these additional terms will be linear combinations of the earlier terms.

$$C = \begin{bmatrix} B & AB & A^2B & \dots & A^{n-1}B \end{bmatrix} \tag{7}$$

Since our controllability matrix is 2×2, the rank of the matrix must be 2. We use the MATLAB command *ctrb* to generate the controllability matrix and the MATLAB command *rank* to test the rank of the matrix. [Appendix]

A. STEPS

Given the A and B matrices of a plant, designing a LQR controller consists of the following steps:

- Initialise state weighting matrix Q and control force weighting matrix R.
- Solve the algebraic riccati equation, given in (8) to obtain Riccati coefficient P
- Compute control input gain K.
- Compute optimal control action by $u(t) = -Kx(t)$
- Obtain the quadratic performance index J as in (17)

$$AP + PA^T - PB^T R^{-1} BP + Q = 0 \tag{8}$$

$$K = R^{-1} B^T S \tag{9}$$

After we verify that our system is controllable and thus we should be able to design a controller that achieves the given requirements. Specifically, we used the linear quadratic regulation method for determining our state-feedback control gain matrix K. The MATLAB function lqr allows us to choose two parameters, R and Q, which will balance the relative importance of the control effort (u) and error (deviation from 0), respectively, in the cost function that we are trying to optimize. The simplest case is to assume $R = eye(2)$, and $Q = C'C$ using Bryson's Rule to define the initial weighted matrices Q and R. The cost function corresponding to R, and Q places equal importance on the control and the state variables, which are outputs (the temperature of both rooms). Essentially, the lqr method allows for the control of both outputs. The controller can be tuned by changing the nonzero elements in the Q matrix to achieve the desired response.

$$Q = C'C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{10}$$

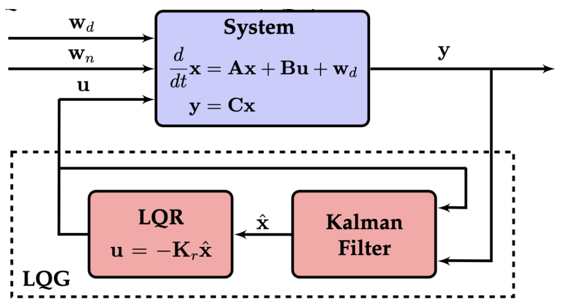
The element in the (1,1) position of Q represents the weight on the temperature of room 1, and the element in the (2,2) position represents the weight on room's two temperature. The input weighting $R = 1$. Initializing different values of Q and R to find the K matrix gave different step responses. We get the K matrix as below:

$$K = \begin{bmatrix} 0.4147 & 0.0157 \\ 0.0105 & 0.2365 \end{bmatrix} \tag{11}$$

IV. Kalman Filtering

The optimal LQR controller from section III relies on full-state measurements of the system. Full-state estimation is mathematically possible if the pair (A, C) is observable. The (A,C) pair in this project is observable. However, estimation effectiveness depends on the degree of observability as quantified by the observability Gramian. The Kalman filter is the most commonly used full-state estimator, as it optimally balances the competing effects of measurement noise, disturbances. Using the full-state estimate from a Kalman filter in conjunction with the optimal full-state LQR feedback law is possible. Implementing the Kalman filter involves defining the measurement equation that relates the measurements to the state vector, assuming that the measurement noise is uncorrelated with the process noise. The measurement equation for the temperature measurements of both rooms is given by:

$$y(t) = \begin{bmatrix} T_1(t) \ T_2(t) \end{bmatrix} \tag{12}$$



$\mathbf{w}_d$
$\mathbf{w}_n$
$\mathbf{u}$
System
$\frac{d}{dt}\mathbf{x} = \mathbf{Ax} + \mathbf{Bu} + \mathbf{w}_d$
$\mathbf{y} = \mathbf{Cx}$
$\mathbf{y}$
LQR
$\mathbf{u} = -\mathbf{K}_r\hat{\mathbf{x}}$
$\hat{\mathbf{x}}$
Kalman Filter
LQG

Fig. 3. Schematic illustrating the linear quadratic Gaussian (LQG) con- troller for optimal closed-loop feedback based on noisy measurements

The measurement equation in continuous time is then given by:

$$y(t) = Cx(t) + v(t)$$

The Kalman filter estimates the state vector based on the measurements and the dynamic model in continuous time. The filter has two stages: the prediction stage and the update stage. In the prediction stage, the filter predicts the state vector at any time t based on the previous estimate and the dynamic model:

$$\dot{x}(t) = Ax(t) + Bu(t) + Gw(t) \tag{13}$$
$$y = Cx + Du + Hw + v \tag{14}$$

where $x(t)$ is the state vector, $u(t)$ is the control input at time t, $w(t)$ is the process disturbance at time t, and A, B, and G are the system matrices. Usually w does not come in the output, so we set $H = 0$

In the update stage, the filter updates the predicted state estimate based on the new measurement:

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + L\big(y(t) - C\hat{x}(t) - Du(t)\big), \qquad \hat{x}(to) = x_0$$

where $\hat{x}(t)$ is the estimated state vector at time t, $y(t)$ is the measurement at time t, D is the input-to-measurement matrix, and L is the Kalman gain matrix, which is computed as:

$$L = PC^T R^{-1}$$

where P is the error covariance matrix and R is the measurement noise covariance matrix, which satisfies the algebraic Riccati equation given by:

$$A^T P + PA - PBK + Q = 0$$

To evaluate the performance of the control strategy and the Kalman filter, simulations can be conducted in MATLAB. The simulations can show that the control strategy is able to regulate the system to its desired setpoints. The Kalman filter is also able to estimate the states of the system accurately, and the estimation errors are within acceptable bounds. The simulations can also show that the control strategy and the Kalman filter are robust to measurement noise and process disturbances. Overall, the simulations demonstrate the effectiveness of the proposed control strategy and Kalman filter for state estimation and control in the presence of disturbances in a continuous-time system.

In MATLAB, the Kalman estimator can be designed using the command:

```
[K,P] = lqe(A,G,C,Q,R)
```

where A, G, C, Q, and R are the system matrices and the covariance matrices of the process and measurement noise, respectively. We chose $G = Q = \text{eye}(2), R = 100{*}\text{eye}(2), H = 0, N = 0$ for the kalman filtering simulations. After the measurement and disturbance noise covariance matrices are determined, the MATLAB function lqe was used to find the optimal Kalman gain L and the covariance matrix P. Measurement noise covariance matrix is determined out from an expected noise on each channel. The Kalman gain was found as:

$$L = \begin{bmatrix} 0.0907 & 0.0051 \\ 0.0051 & 0.0877 \end{bmatrix} \tag{15}$$

$$P = \begin{bmatrix} 9.0725 & 0.5141 \\ 0.5141 & 8.7658 \end{bmatrix} \tag{16}$$

V. Result

The simulation results demonstrate the effectiveness of the proposed control strategy in regulating the temperatures of both rooms. The control strategy successfully mitigates the influence of external disturbances, such as attacks on the sensors or input, and maintains the desired room temperatures. The Kalman filter plays a crucial role in accurately estimating room temperatures and restoring them to the desired values as shown in Figure 5. Additionally, the control strategy exhibits robustness to changes in the heat sources' input and adapts well to external disturbances.

A. State Space Simulations

The plot in figure 4 was generated using the $ode45$ solver in MATLAB. The ode45 solver simulates the system over a specified time span using the initial state of the system. It numerically integrates the system dynamics defined by the system to determine the temperature of each room at each time point. Figure 4 shows the simulation results of a two-room model, where the temperature of each room is plotted against time. The blue line represents the temperature of room 1, while the red line represents the temperature of room 2. Initially, both rooms start at 20 degrees Celsius, with a heat input of 100W and 150W to room 1 and room 2, respectively. As time progresses, the temperature of both rooms increases due to the heat input. Around 2000 seconds, the temperature of room 1 starts to stabilize at around 24 degrees Celsius, while the temperature of room 2 continues to increase. This behavior is due to the difference in the two rooms' heat capacity and surface area. At around 3000 seconds, the temperature of room 2 also starts to stabilize at around 28 degrees Celsius, and the system reaches a steady state.

B. LQG

To improve the robustness of the thermal control system, we designed a Linear Quadratic Gaussian (LQG) controller using the two-room model as a basis. The LQG controller was designed to minimize the temperature difference between the two rooms while accounting for disturbances and noise in the system. The simulation results of the LQG controller are shown in Figure 5, where the temperature of each room is plotted against time. The LQG controller was able to maintain the temperature of both rooms within an acceptable range despite disturbances and noise in the system. Initially, both rooms start at 20 degrees Celsius, with a heat input of 100W and 150W to room 1 and room 2, respectively. As time progresses, the temperature of both rooms increases due to the heat input, but the LQG controller is able to maintain the temperature difference between the two rooms within a narrow range. Around 301.7 seconds, the temperature of room 1 stabilizes at around 53.3 degrees Celsius, while the temperature of room 2 continues to increase. However, the LQG controller is able to quickly detect this temperature difference and adjust the heat input to room 2, bringing the temperature back within the acceptable range. At around 500 seconds, the temperature of room 2 also stabilizes at around 78.6 degrees Celsius, and the system reaches a steady state. We notice that the Kalman filter tracks the states perfectly. Overall, the simulation results shown in figure 5 demonstrate the effectiveness of the LQG controller in getting the rooms to the desired temperature in a shorter period as compared to the states $[T_1, T_2]$ dymanical behavior shown in figure 4. Also the LQG was able to maintain the temperature of a thermal control system within an acceptable range despite disturbances and noise.

VI. Conclusions

The proposed MIMO control system for thermal control of two rooms with one input heat source in each room is expected to provide a robust solution to controlling the temperature in buildings. Implementing a Kalman filter control to estimate the states of the room temperatures and restore the temperatures in the event of an attack on the input is expected to enhance the control system's effectiveness. The simulation results are

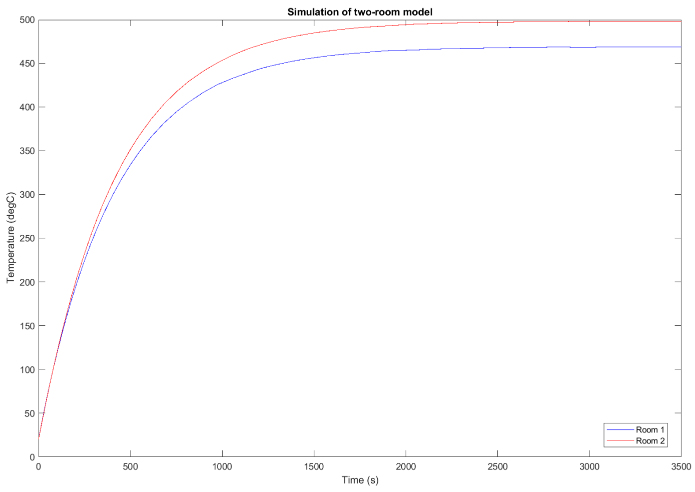


Fig. 4. Dynamics of Room Temperatures in a Two-Room Model with Heat Transfer and External Forcing

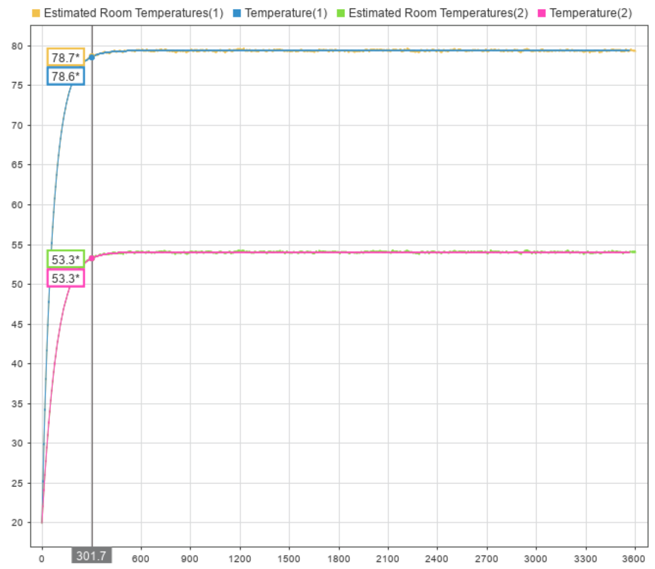


Fig. 5. LQG Control

expected to show the performance of the proposed control system.

VII. Future Work

A future goal is to enhance the existing design to handle larger rooms and support diverse input designs. This could involve denoting the subsystems as nodes and their interconnections as edges to gain insights into managing the rooms as a complex network. Additionally, the future direction involves modeling the rooms to account for nonlinearity and devising nonlinear controls to govern the system.

VIII. References

[1] Ogata, K. (2010). Modern control engineering. Prentice Hall.

[2] Simon, D. (2006). Optimal state estimation: Kalman, H infinity, and nonlinear approaches. John Wiley & Sons.

[3] Lewis, F. L., & Syrmos, V. L. (2014). Optimal control. John Wiley & Sons.

[4] Paridari, K., O'Mahony, N., Mady, A. E. D., Chabukswar, R., Boubekeur, M., & Sandberg, H. (2017). A framework for attack-resilient industrial control systems: Attack detection and controller reconfiguration. Proceedings of the IEEE, 106(1), 113-128.

$$\dot{T}_1 = \frac{1}{C_1}\left(u_1 - hA(T_1 - T_{out})\right) - \frac{k}{C_1}\left(T_1 - T_2\right)$$
$$\dot{T}_2 = \frac{1}{C_2}\left(u_2 - hA(T_2 - T_{out})\right) - \frac{k}{C_2}\left(T_2 - T_1\right) - \frac{k}{C_2}\left(T_2 - T_3\right)$$
$$\dot{T}_3 = -\frac{k}{C_3}\left(T_3 - T_2\right) - \frac{k}{C_3}\left(T_3 - T_4\right)$$
$$\dot{T}_4 = -\frac{k}{C_4}\left(T_4 - T_3\right) - \frac{k}{C_4}\left(T_4 - T_5\right)$$
$$\dot{T}_5 = -\frac{k}{C_5}\left(T_5 - T_4\right) - \frac{k}{C_5}\left(T_5 - T_6\right)$$
$$\dot{T}_6 = -\frac{k}{C_6}\left(T_6 - T_5\right) - \frac{k}{C_6}\left(T_6 - T_7\right)$$
$$\dot{T}_7 = -\frac{k}{C_7}\left(T_7 - T_6\right) + \frac{1}{C_7}\left(u_2 - hA(T_7 - T_{out})\right)$$

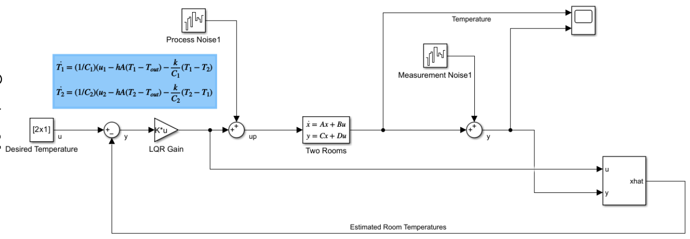
IX. Appendix

A. MATLAB CODE

```
clc;
close all;
clear;

% Define system parameters
C1 = 100; % Heat capacity of room 1 (J/K)
C2 = 150; % Heat capacity of room 2 (J/K)
h = 0.05; % Heat transfer coefficient btw room and
A_s = 20; % Surface area of room 1 (m^2)
A_s2 = 40; % Surface area of room 2 (m^2)
k = 0.7; % Heat transfer coefficient btw the rooms
Tout = 20; % Outside temperature (degC)


%System State Space Representation
A = [-h*A_s/C1, k/C1;
k/C2, -h*A_s2/C2];
B = [1/C1, 0; 0, 1/C2];
C = [1 0; 0 1];
D = 0*eye(2);


% Calculate the observability matrix
CO = ctrb(A,B);
```

```
% Check if the system is controllable
if rank(CO) == size(A,1)
disp('System is controllable!');
else
disp('System is not controllable!');
end

% Calculate the observability matrix
O = obsv(A, C);

% Check the rank of the observability matrix
if rank(O) == size(A,1)
disp('System is observable');
else
disp('System is not observable');
end


%Define cost function
Q1 = diag([1,1]);
Q2 = 2*(C'*C);
R1 = eye(2);
R2 = eye(2);

% LQR controller
[K1,S1,P1] = lqr(A,B,Q1,R1);
[K2,S2,P2] = lqr(A,B,Q2,R2);


% Kalman filter
G = eye(2); % Process noise covariance matrix
Q = 1*eye(2); % Measurement noise covariance matrix
R = 100*eye(2); % Error covariance matrix
[L,P] = lqe(A,G,C,Q,R); % Kalman gain

% Define simulation parameters
tspan = [0 3500]; % Simulation time span (s)
initial_state = [20; 20]; % Initial temperature of each room (degC)
u1 = 100; % Heat input to room 1 (W)
u2 = 150; % Heat input to room 2 (W)

% Define the system dynamics as a function o
f = @(t, S) [(-h*A_s/C1)*(S(1)-Tout) + k/C1*
k/C2*S(1) + (-h*A_s2/C2)*(S(2)-Tout) + u2/C2

% Simulate the system using ODE45 solver
[t, S] = ode45(f, tspan, initial_state);


% Plot the results
figure;
plot(t, S(:,1), 'b-', t, S(:,2), 'r-');
xlabel('Time (s)');
ylabel('Temperature (degC)');
legend('Room 1', 'Room 2', 'Location', 'southeast');
title('Simulation of two-room model');
```

B. SIMULINK



Process Noise1
$\dot{T_1} = (1/C_1)(u_1 - hA(T_1 - T_{out}) - \frac{k}{C_1}(T_1 - T_2)$
$\dot{T_2} = (1/C_2)(u_2 - hA(T_2 - T_{out}) - \frac{k}{C_2}(T_2 - T_1)$
[2x1]
u
Desired Temperature
y
K*u
LQR Gain
up
$\dot{x} = Ax + Bu$
$y = Cx + Du$
Two Rooms
Temperature
Measurement Noise1
y
u
y
xhat
Estimated Room Temperatures

Fig. 6. Simulink