Faculty of Engineering, Ain Shams University
Credit Hours Engineering Programs
Mechatronics & Automation Program

MCT 411 Hybrid Control

Project Report

Name	ID
Mohamed hatem	18P3449
Thomas Medhat Mounir Botros	18P8912

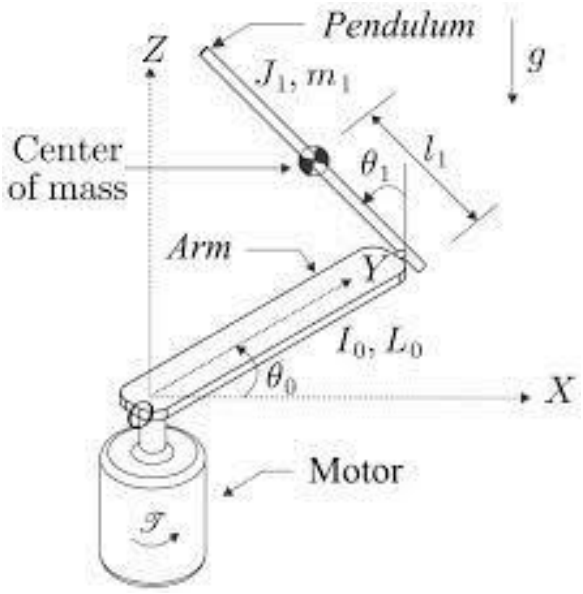
Table of Contents

Abstract :

The aim of this project is to control the Furuta pendulum.

The Furuta pendulum, or rotational inverted pendulum, consists of a driven arm which rotates in the horizontal plane and a pendulum attached to that arm which is free to rotate in the vertical plane.

It provides a compact yet impressive platform for control demonstrations and draws the attention of the control community as a platform for the development of nonlinear control laws. Despite the popularity of the platform, there are very few papers which employ the correct dynamics and only one that derives the full system dynamics.



Introduction:

A control system manages, commands, directs, or regulates the behavior of other devices or systems using control loop.
Control systems are present in many systems around us and are used for various applications.
Control systems vary in complexity and each has its own properties.
To find the suitable controller is a task itself. So we tried different controllers with our model to choose the best controller to fit.

Controller Design:

LQR (Linear Quadratic Regulator):

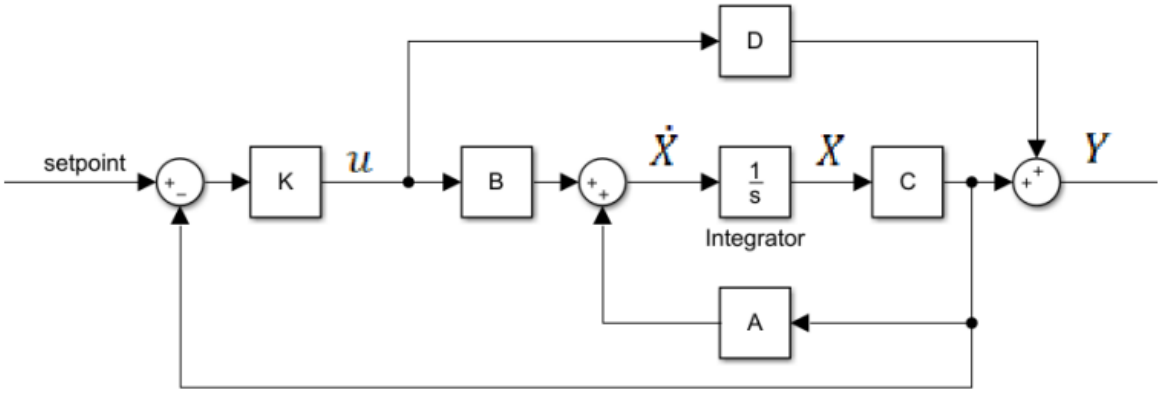
LQR is Full State Feedback. FSF controller provides control to a dynamic system described in state-space. A state-space of a system has the form:

$$\begin{cases} \dot{X} = AX + Bu \\ Y = CX + Du \end{cases}$$

Where A defines the dynamics of the system, B defines the influence of the input on the system, C defines the output of the system contributed by its state and D defines the contribution of the input to the output of the system.
LQR provides a solution such system given a cost described by a quadratic function.
LQR is based on optimal control.

Theory:

The controller behavior is altered by using a mathematical algorithm that minimizes a cost function with weighting factors provided by a human. The cost function is often defined as a sum of deviation of some variables from their desired values. The algorithm generates setting that minimizes undesired deviations. By utilizing LQR, one can get a feedback gain K.



The settings of a (regulating) controller governing either a machine or process (like an airplane or chemical reactor) are found by using a mathematical algorithm that minimizes a cost function with weighting factors supplied by a human.
The cost function is often defined as a sum of the deviations of key measurements, like altitude or process temperature, from their desired values.
The algorithm thus finds those controller settings that minimize undesired deviations.
The magnitude of the control action itself may also be included in the cost function.
Where x is the state of the system and u is the control input. Given a quadratic cost function for the system, defined as :

$$J = x^T(t_1)F(t_1)x(t_1) + \int_{t_0}^{t_1} \left(x^T Q x + u^T R u + 2x^T N u\right) dt$$

If the reference is assumed to be zero, the input to a full-state feedback system is and the state equation becomes where A_f=A-BK. The goal of the LQR control strategy is to have the state variables to approach zero from some initial condition in minimum amount of time and energy of the control, i.e., we want to minimize a performance index defined as:

$$J = \int_0^\infty [\bar{x}^T Q \bar{x} + \bar{u}^T R \bar{u}]\, dt = \int_0^\infty \bar{x}^T (Q + K^T R K)\bar{x}\, dt$$

Where Q and R are the values which we change in order to achieve the desired control on the system. Q matrix defines the weights on the states while R matrix defines the weights on the control input in the cost function

The parameters Q and R can be used as design parameters to penalize the state variables and the control signals. The larger these values are, the more you penalize these signals. Basically, choosing a large value for R means you try to stabilize the system with less (weighted) energy. This is usually called expensive control strategy. On the other hand, choosing a small value for R means you don't want to penalize the control signal (cheap control strategy). Similarly, if you choose a large value for Q means you try to stabilize the system with the least possible changes in the states and large Q implies less concern about the changes in the states. Since there is a trade-off between the two, you may want to keep Q as I (identity matrix) and only alter R. You can choose a large R, if there is a limit on the control output signal (for instance, if large control signals introduce sensor noise or cause actuator's saturation) , and choose a small R if having a large control signal is not a problem for your system.

PID Controller:

PID controller is a widely used controller due to its simplicity and high effectiveness in most situations. It has three main components that allow this controller to work and they are as its name suggests:

- Proportional Controller
- Integral Controller
- Derivative Controller

Theory:

The Proportional Controller:

Produces an output proportional to the input signal. It does so by simply multiplying the input signal with a gain Kp. The theory behind this controller is to provide an output in direct relation to input hence follow it somehow.

The Integral Controller:

Produces an output proportional to the integral of the input signal. This is achieved by multiplying the input signal by 1/s in the s-domain to get the integral of the signal. It is then multiplied by the integral gain Ki. The theory behind this controller is to generate an output related to the integral of the input hence eliminating steady state error. This is done since even small errors provide big enough outputs when integrated long enough.

The Derivative Controller:

Produces an output proportional to the derivative of the input signal. This could be done in theory by multiplying the input signal by s in the s-domain then multiplying it by the derivative gain Kd.

The theory behind this controller is to provide outputs directly related to the input signal. Hence the controller is better at following the input by studying how rapidly it is changing and providing an output according to it.

However, high frequency noise has a huge effect on the derivative controller. Therefore, the derivative controller is designed to incorporate a low pass filter to filter out higher frequency noise and disturbances.

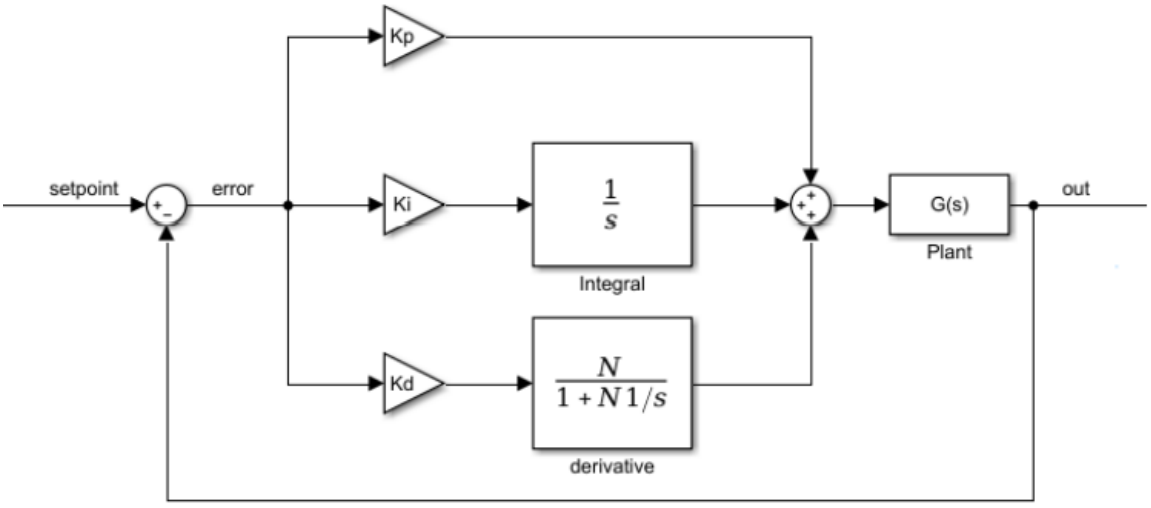
This is done by multiplying by $\frac{N}{1+N\frac{1}{s}}$ instead of simply multiplying by s (in the s domain) where N is the filter coefficient.

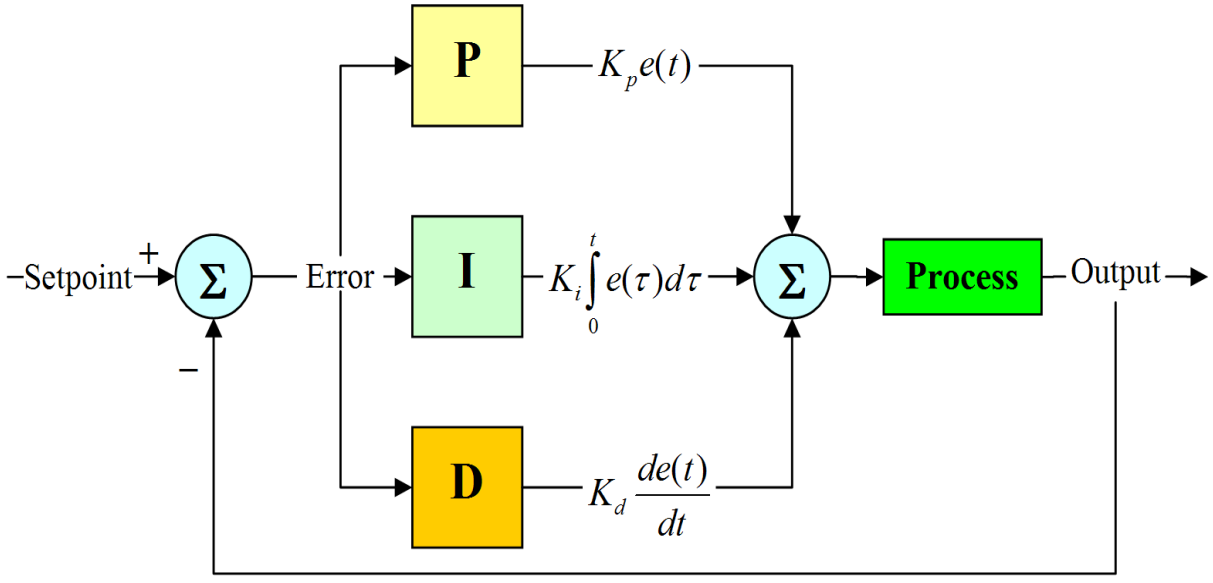
The PID controller uses the 3 previously mentioned controllers to give its output by summing them all together.

The PID controller is tuned by providing certain gains (Kp, Ki, Kd) that allow the system to react according to certain constraints.

Hence the PID controller has the following open loop transfer function:

$$Kp + Ki\frac{1}{s} + kd\frac{N}{1 + N\frac{1}{s}}$$





Pole Placement:

Pole placement is a Full state Feedback controller. FSF controller provides control to a dynamic system described in state-space. A state-space of a system has the form:

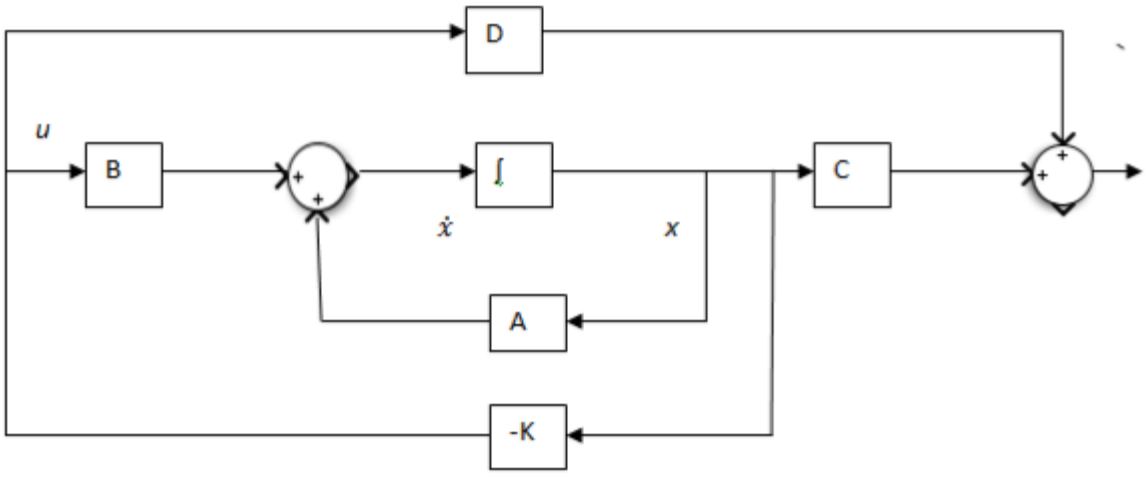
$$\begin{cases} \dot{X} = AX + Bu \\ Y = CX + Du \end{cases}$$

Where A defines the dynamics of the system, B defines the influence of the input on the system, C defines the output of the system contributed by its state and D defines the contribution of the input to the output of the system.

Pole placement is a method of calculating the optimum gain matrix used to assign closed-loop poles to specified locations, thereby ensuring system stability. Closed-loop pole locations have a direct impact on time response characteristics such as rise time, settling time, and transient oscillations.

Placing poles is desirable because the location of the poles corresponds directly to the eigenvalues of the system, which control the characteristics of the response of the system. The system must be considered controllable in order to implement this method.

The disadvantage of the pole placement method is that location of the closed loop poles on the complex plain is quiet difficult, and requires high level of skills in establishing relationship between poles and dynamic performances of the closed loop control system.

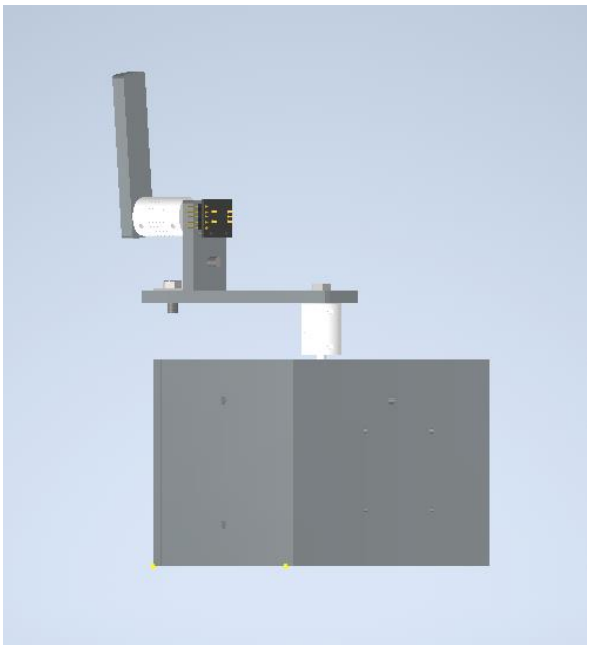


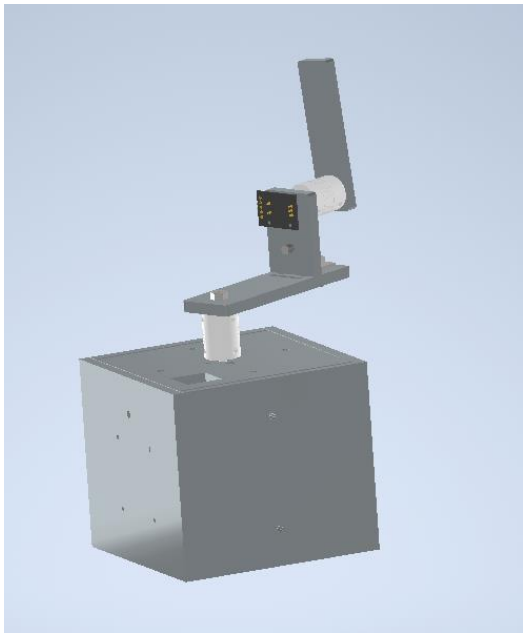
$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{B}u = \mathbf{Ax} + \mathbf{B}(-\mathbf{Kx} + r) = (\mathbf{A} - \mathbf{BK})\mathbf{x} + \mathbf{B}r$$

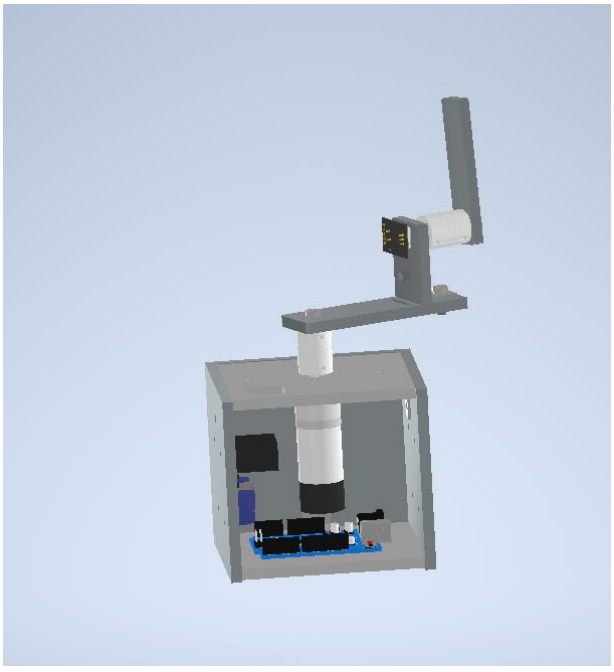
$$y = \mathbf{Cx}$$

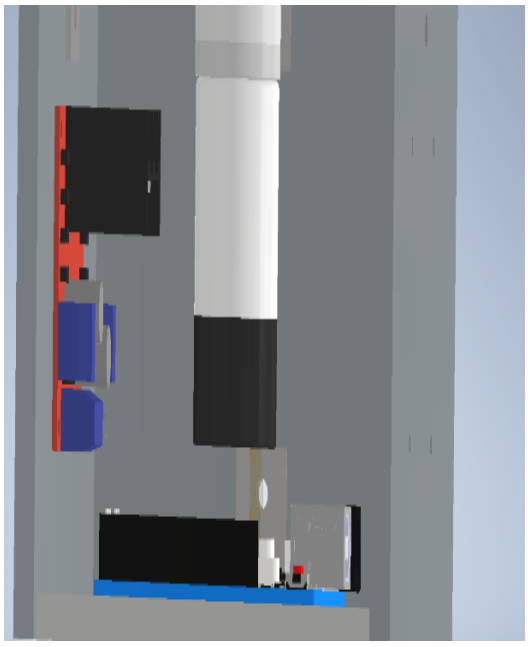
Mechanical Design :

CAD:

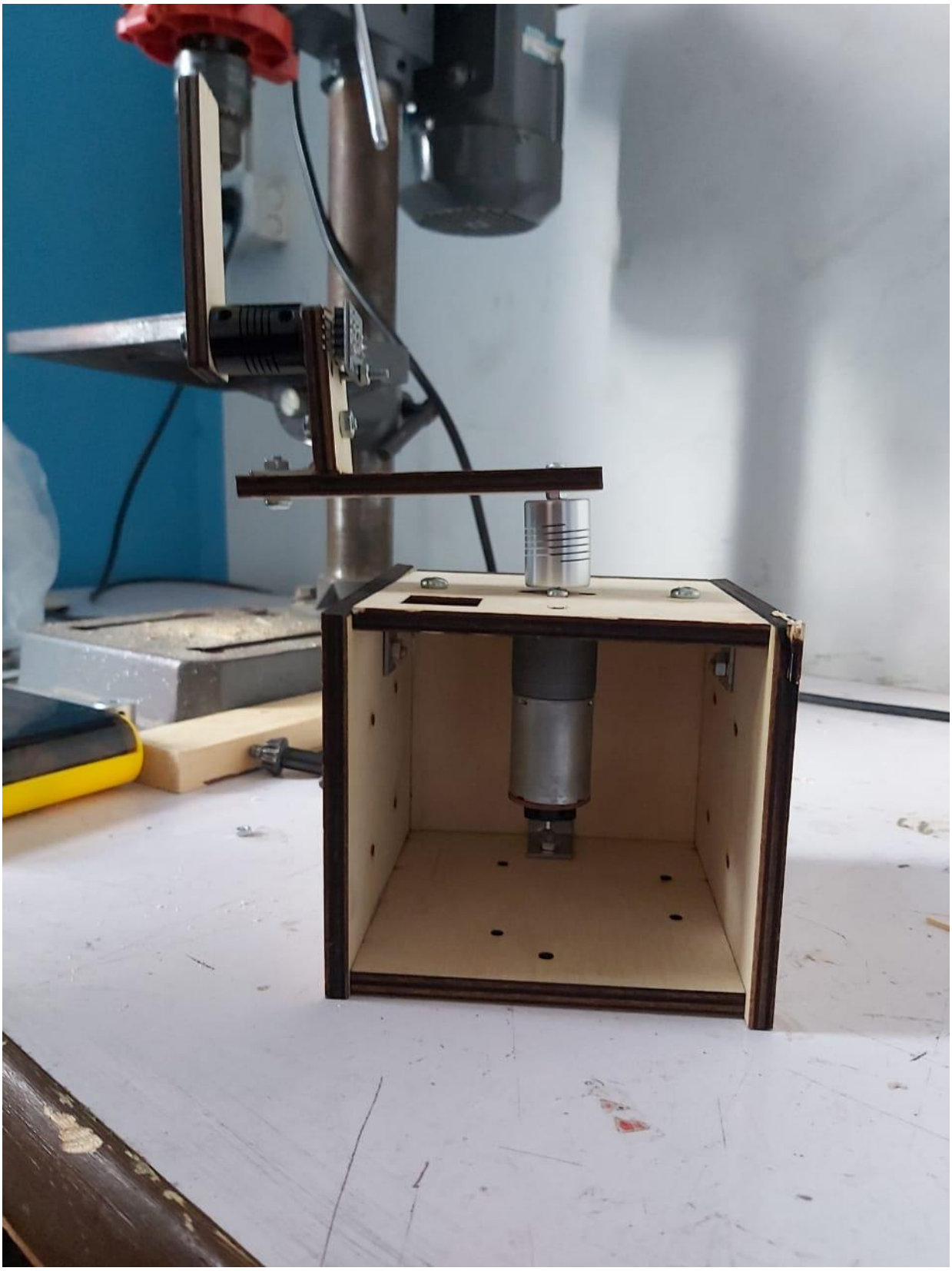








Real Life:



Model:

A rotary inverted pendulum is system consisting of a motor (coupled to a shaft encoder) rotating an arm. The rotating arm is connected to another shaft encoder having its shaft parallel to the arm. The second encoder (Multiturn Potentiometer) now acts as a pivot point of a pendulum connected to it. The pendulum is either a homogeneous rod or a light rod connected to an end block.

State Space Model of the System:

Given the below simplified schematic of an inverse pendulum.
α denotes the angle the pendulum. Hence $\dot{\alpha}$ is its angular speed.
θ denotes the angle of the motor shaft. Hence $\dot{\theta}$ is its angular speed.

$$\begin{cases} \dot{X} = AX + Bu \\ Y = CX + Du \end{cases} \text{ Where:}$$

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{M_p^2 \cdot l_p^2 \cdot r \cdot g}{J_{eq} \cdot M_p \cdot l_p^2 + M_p \cdot r^2 \cdot J_p + J_{eq} \cdot J_p} & -\frac{(J_p . K_t . K_m + M_p . l_p^2 . K_t . K_m)}{R_m . (J_{eq} . J_p + J_{eq} . M_p . l_p^2 + M_p . r^2 . J_p)} & -B_{eq} \\ 0 & \frac{M_p . l_p . g (J_{eq} + M_p . r^2)}{J_{eq} \cdot M_p \cdot l_p^2 + M_p \cdot r^2 \cdot J_p + J_{eq} \cdot J_p} & -\frac{M_p . l_p . r . K_t . K_m}{R_m . (J_{eq} . J_p + J_{eq} . M_p . l_p^2 + M_p . r^2 . J_p)} & -B_p \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ -\frac{K_t . (J_p + M_p . l_p^2)}{R_m . (J_{eq} . J_p + J_{eq} . M_p . l_p^2 + M_p . r^2 . J_p)} \\ -\frac{M_p . l_p . r . K_t}{R_m . (J_{eq} . J_p + J_{eq} . M_p . l_p^2 + M_p . r^2 . J_p)} \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$X = \begin{bmatrix} \theta \\ \alpha \\ \dot{\theta} \\ \dot{\alpha} \end{bmatrix}$$

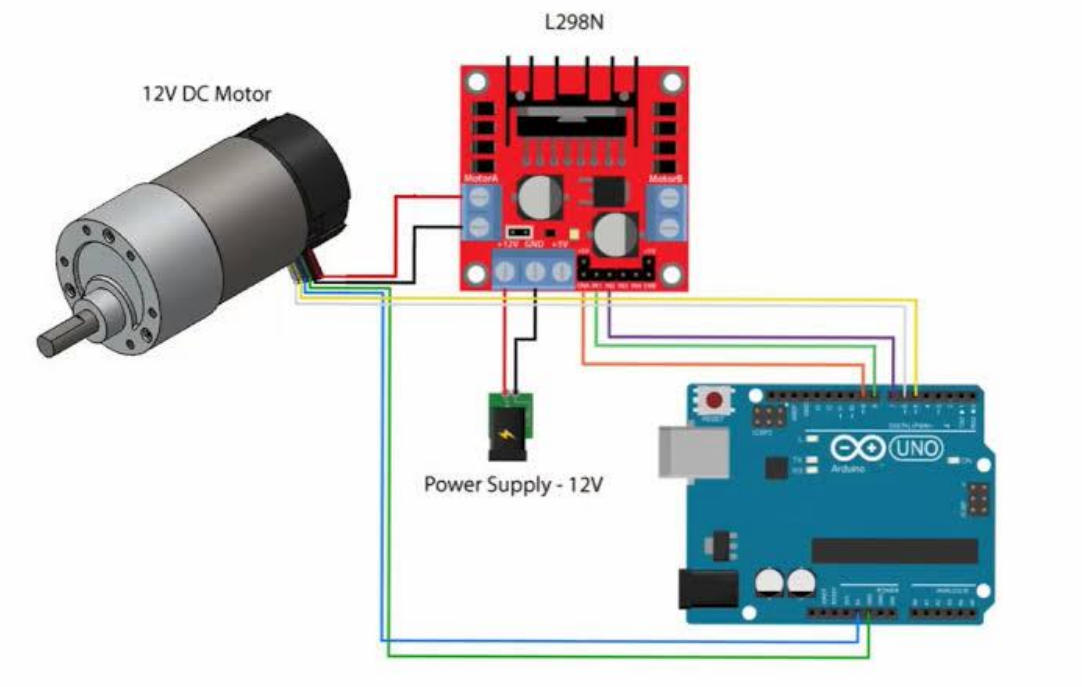
Check the appendix section **Modeling** for the mathematical walkthrough.

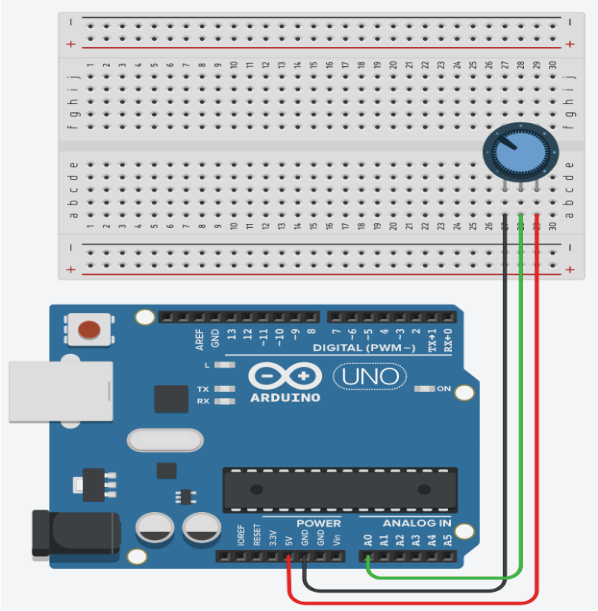
Model parameters:

Parameter	Description	Unit	Value
g	Gravitational constant	N/kg	9.81
M_p	Mass pendulum assembly	kg	0.027
l_p	Center of mass of pendulum assembly	m	0.04
r	Length from motor shaft to pendulum pivot	m	0.0826
J_p	Pendulum moment of inertia relative to pivot	$kg.m^2$	0.000698
J_{eq}	Equivalent moment of inertia acting on motor shaft	$kg.m^2$	0.000368
B_p	Viscous Damping about the pendulum pivot	N.m.s/rad	1.4
B_{eq}	Equivalent viscous damping acting on motor shaft	N.m.s/rad	0.93
K_t	DC motor current-torque constant	N.m/A	0.0333
K_m	DC motor back-emf constant	V.s/rad	0.0333
R_m	Electric resistance of motor armature	Ω	8.7

Electrical:

In this project, the controller used was the **Arduino Uno** , and the actuator used was a **DC Motor** Wired to an **H-Bridge** and the **encoder** used to measure the change of angle of the pendulum is a **Multi-Turn Potentiometer.**





MATLAB CONTROLLING THE PENDULUM:

A simulation of the pendulum was done over MATLAB and Simulink using the provided model to test the different controllers.

Manipulating an inverted pendulum utilizes three different controllers that switch automatically. The swing up controller, catch controller and balance controller.

The job of swing up controller is to swing the pendulum from its resting down state to an up state.

The job of the catch controller is to catch the flung-up pendulum by reducing the angular velocity of the pendulum. The job of the balance controller is to keep the pendulum in upright position once it is up.

The scope of this project focuses only on the balance controller where the pendulum is assumed to not deviate far from the upright position.

The initial conditions on the simulation are:

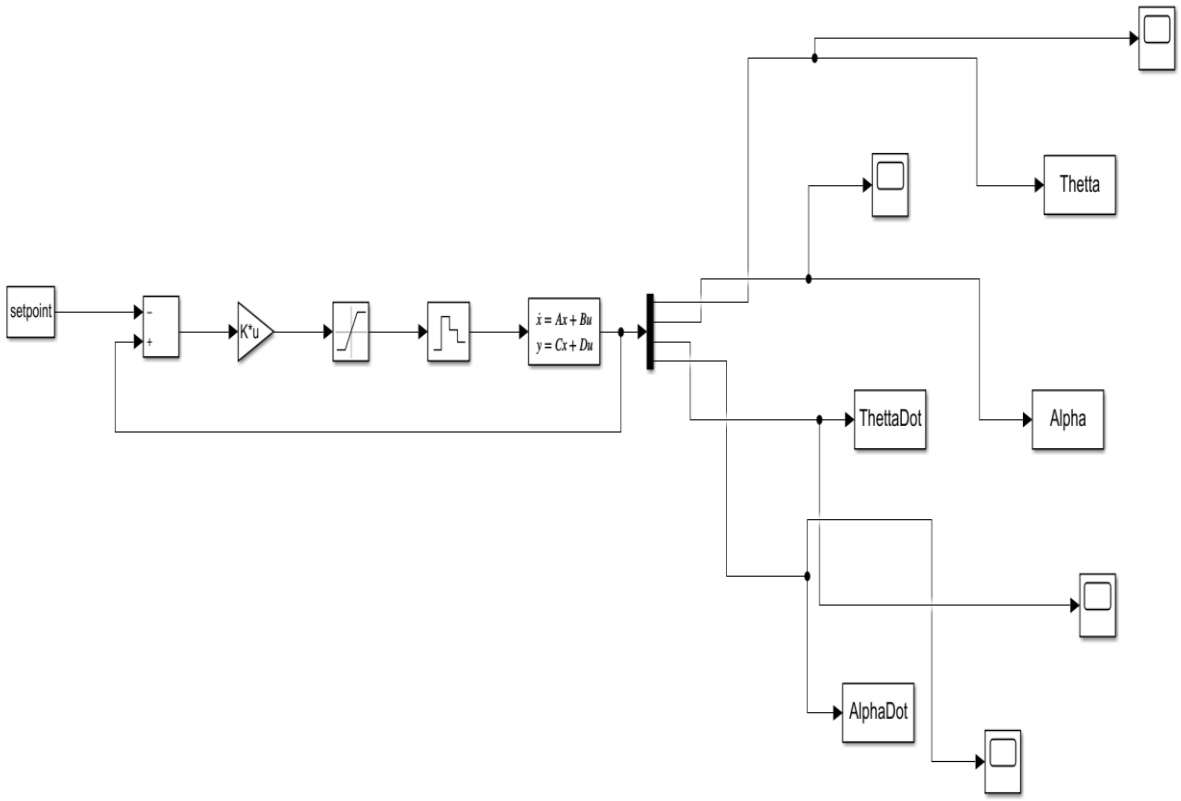
$\theta = 0 \qquad \alpha = 0.2 \qquad \dot{\theta} = 0 \qquad \dot{\alpha} = 0$

LQR:

In our FSF controller, we used LQR to provide solution to the optimal control algorithm by calculating K from the matrix Q and the scalar R.

The system was tuned by manipulating the values of Q and R by trial and error using Q = I and R=1 as base values

$$Q = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad R = 1$$

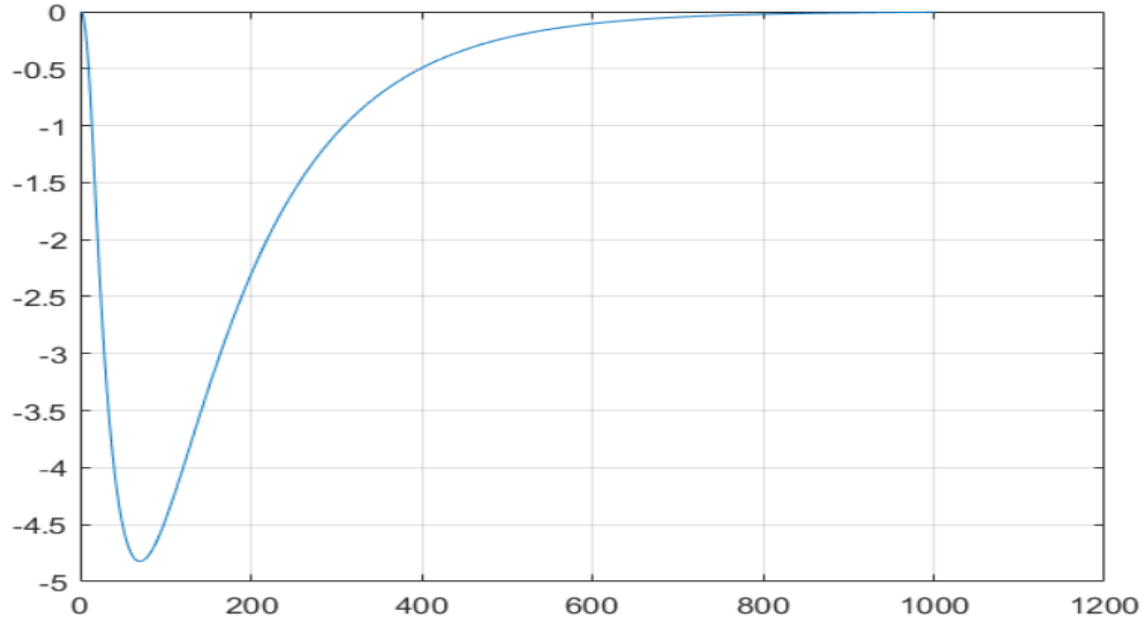


The presence of a saturation block to prevent the controller from bypassing the rated voltage of the motor. By setting the setpoint to zero.

Results:

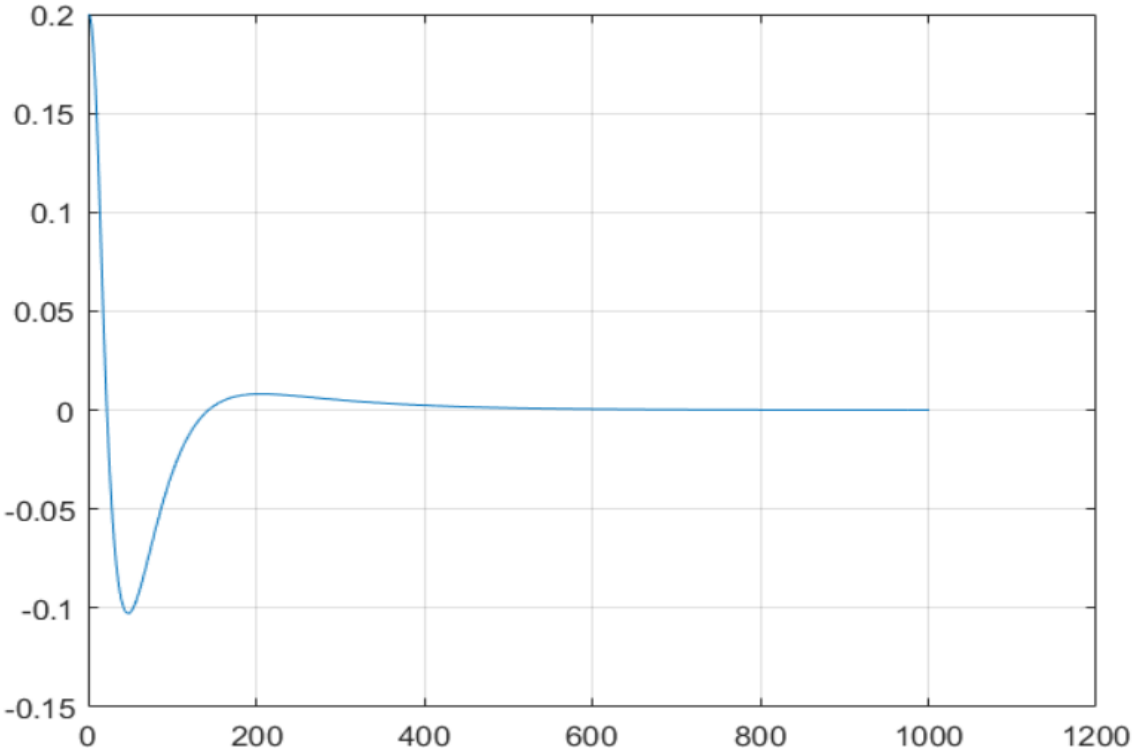
Theta

Which represents the Angle of motor



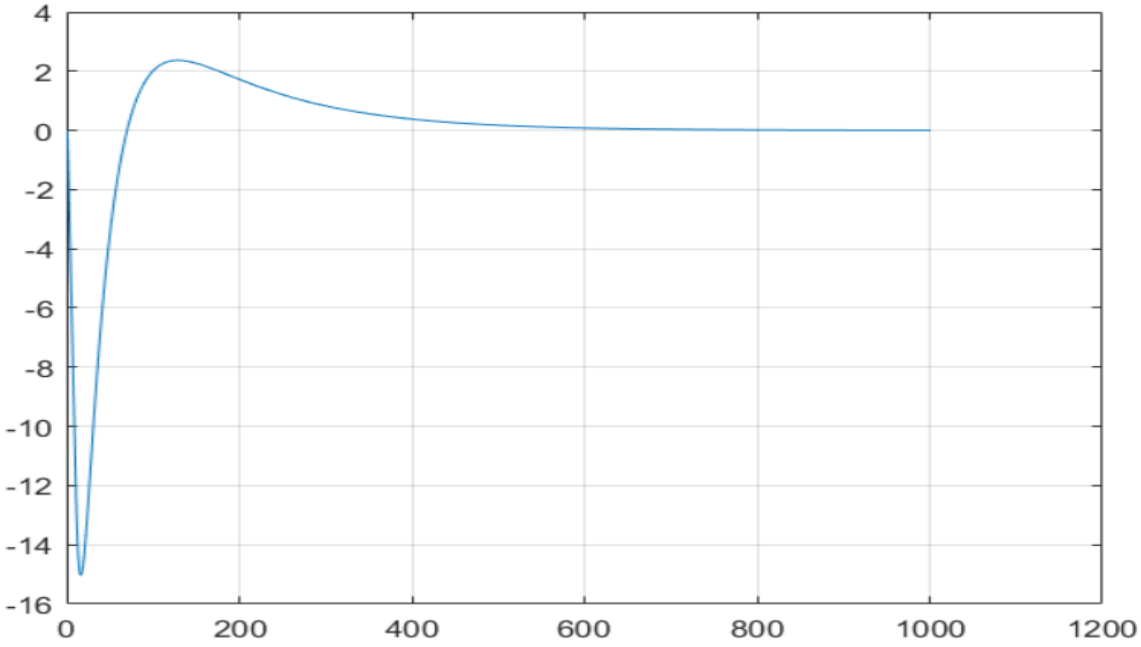
Alpha

Which represents the angle of pendulum.



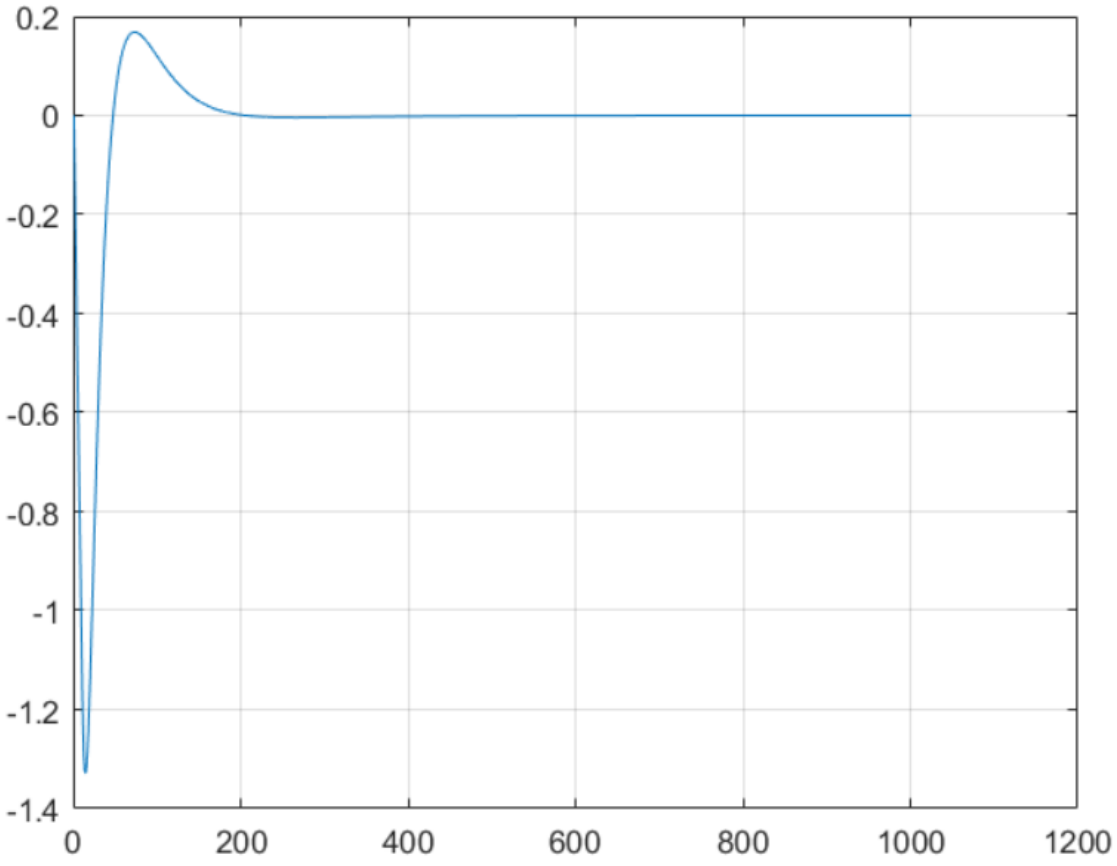
Theta dot

Which represents the speed of rotor.



Alpha dot

Which represents the speed of pendulum.

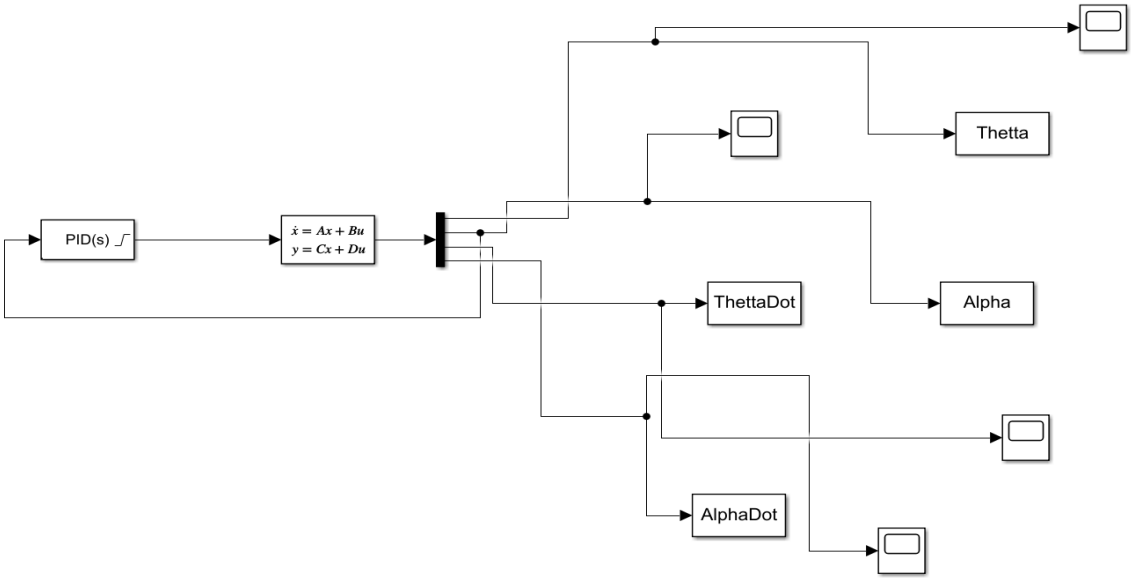


When running this model that represents the LQR Controller with an initial condition, and parameters that was already calculated in an .m file, the following outputs are observed.

Check the appendix section **LQR code** for the code walkthrough.

PID:

A PID controller is single input single output controller. The block diagram shows the PID controller using α as an input and tries to stabilize it at zero.

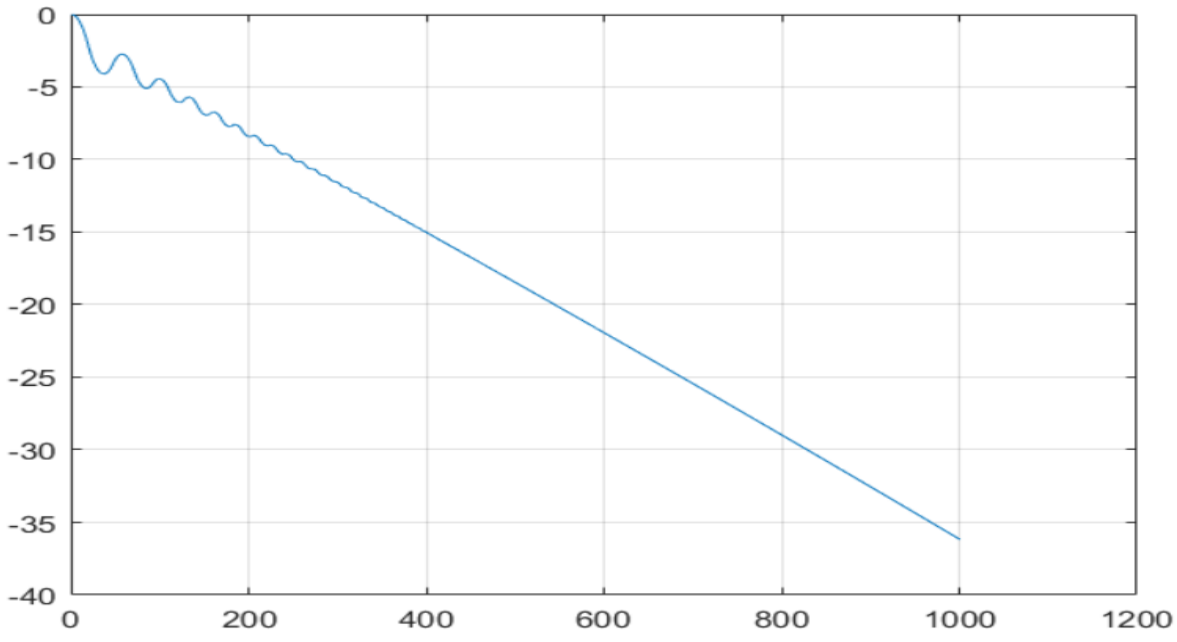


The presence of a saturation block to prevent the controller from bypassing the rated voltage of the motor. By setting the setpoint to zero. Saturation part is inside PID block.

Results:

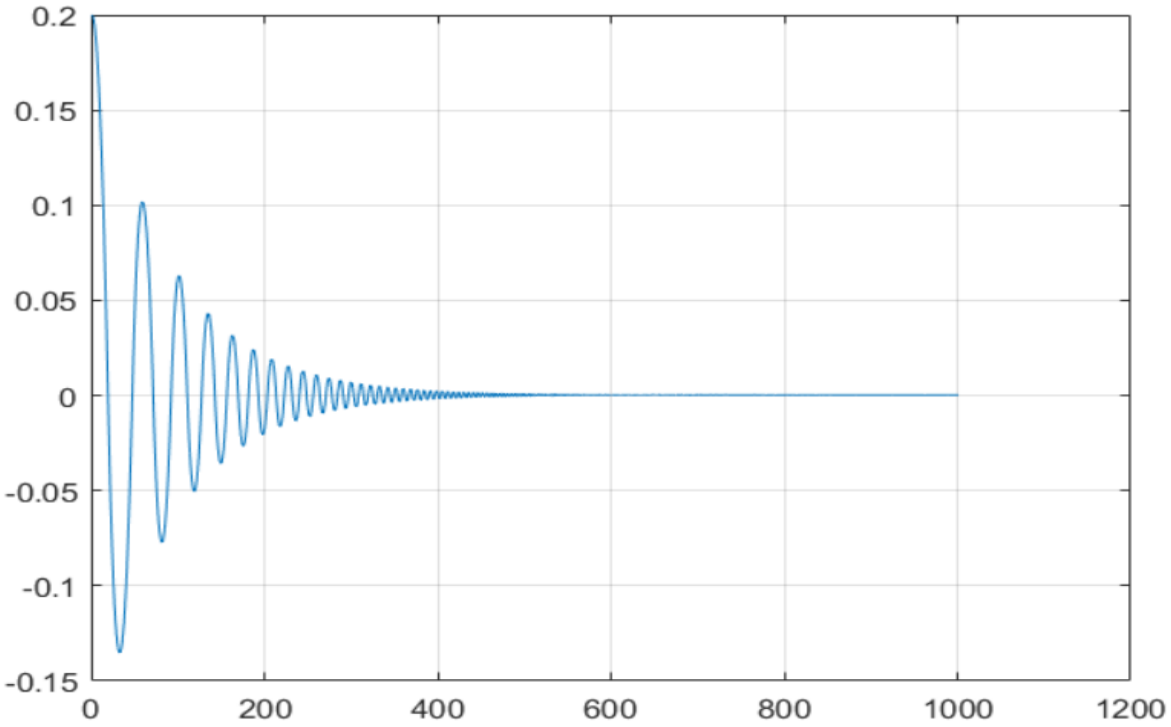
Theta

Which represents the Angle of motor



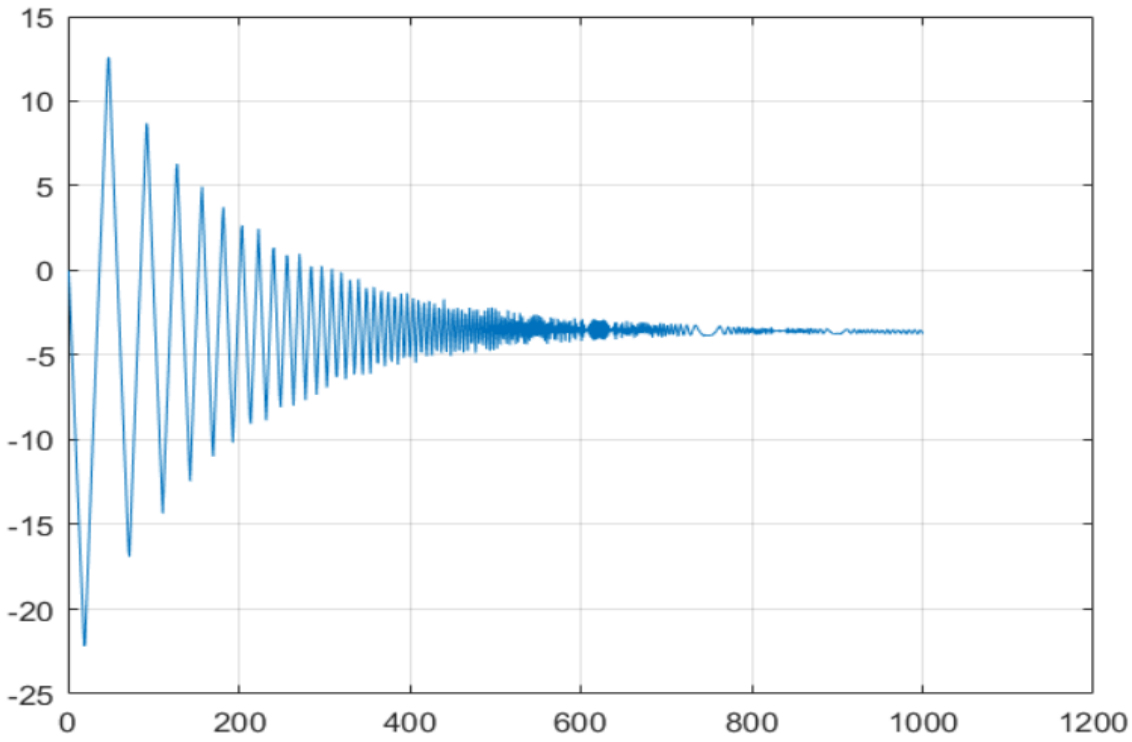
Alpha

Which represents the angle of pendulum.



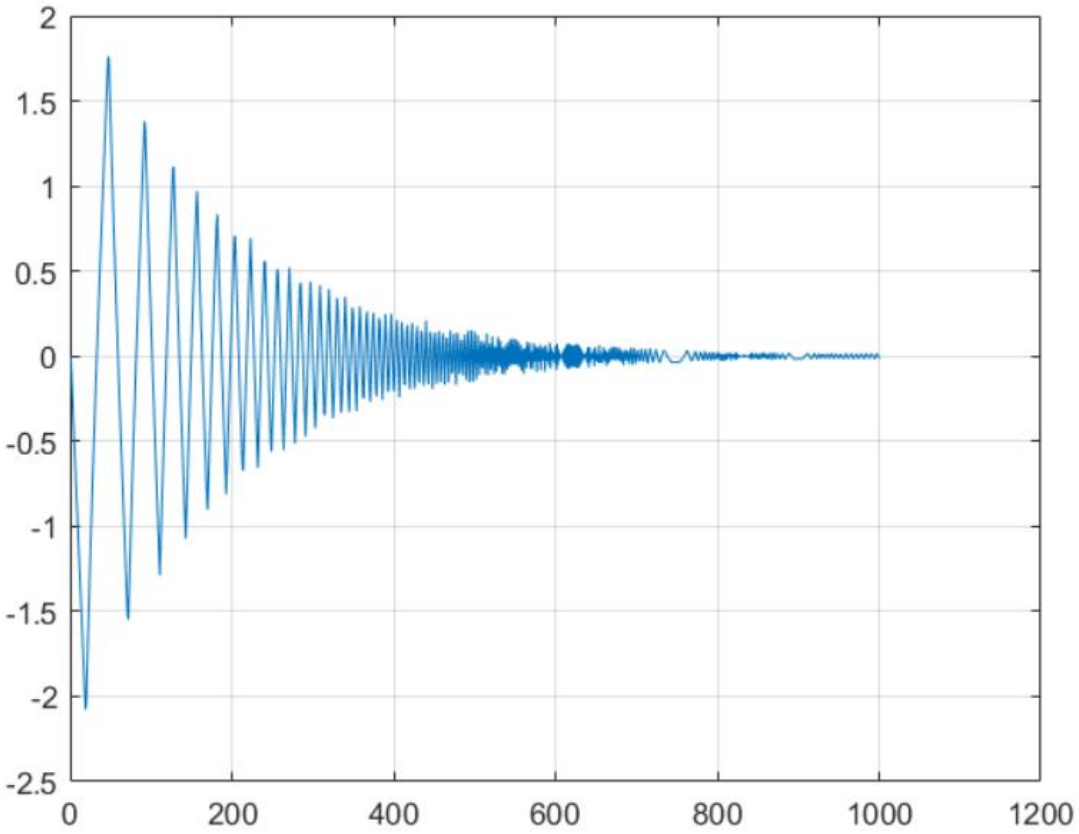
Theta dot

Which represents the speed of rotor.



Alpha dot

Which represents the speed of pendulum.



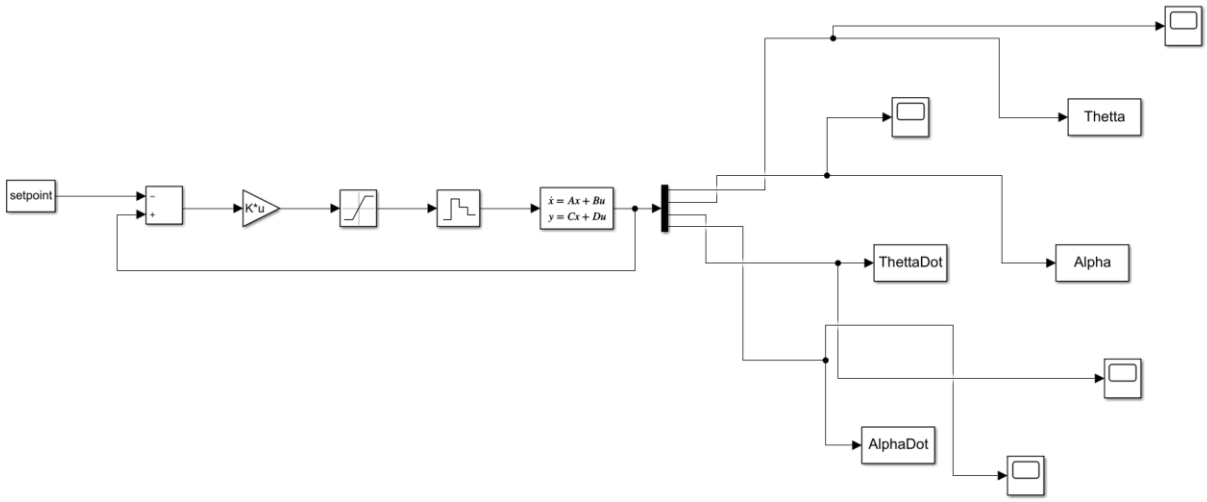
When running this model that represents the PID Controller with an initial condition, and parameters that was already calculated in an .m file, the following outputs are observed.

Check the appendix section **PID code** for the code walkthrough.

Though the PID controller was able to stabilize α at zero, θ̇ is nonzero and hence θ is constantly changing even after steadying α. Though we can't fix θ at zero easily without sacrificing the stability of our existing controller, we can however stop it from moving by eliminating the steady state error of θ̇ to zero.

Pole Placement:

Pole placement is also Full state Feedback

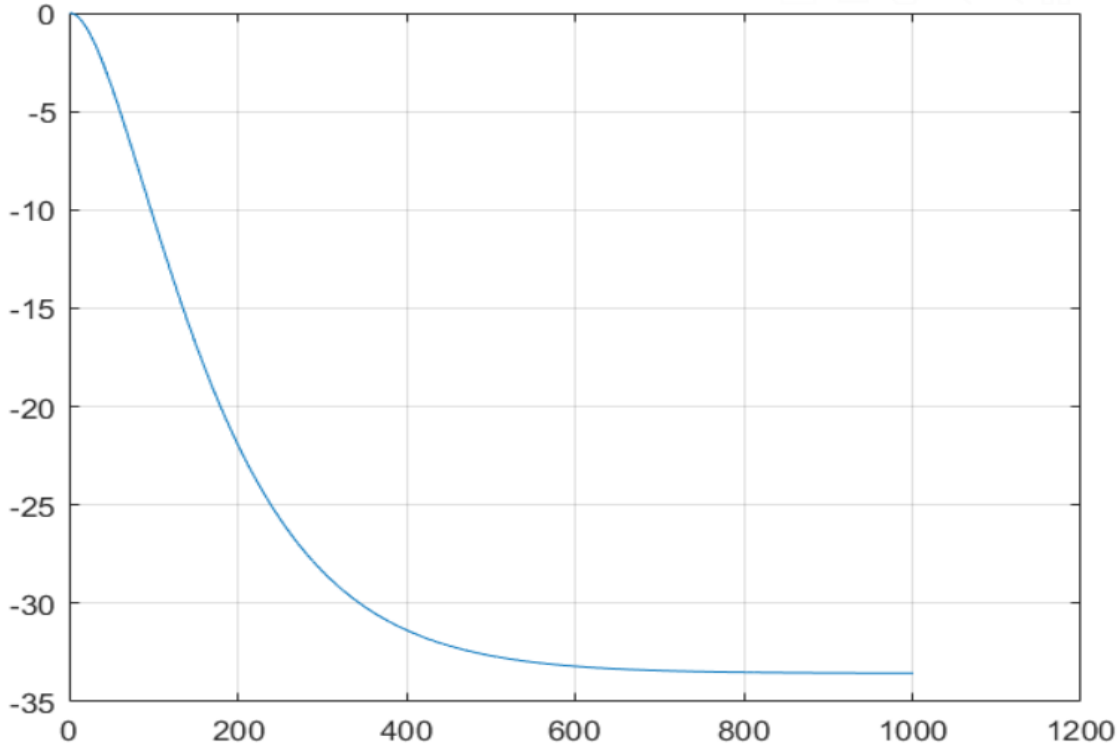


The presence of a saturation block to prevent the controller from bypassing the rated voltage of the motor. By setting the setpoint to zero.

Results:

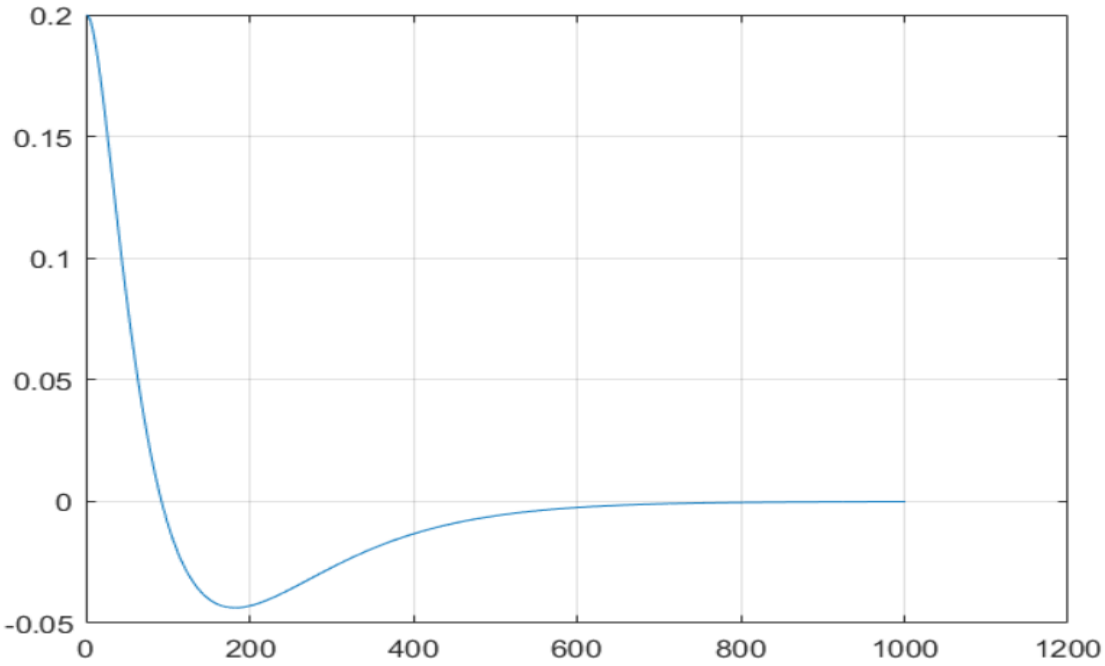
Theta

Which represents the Angle of motor



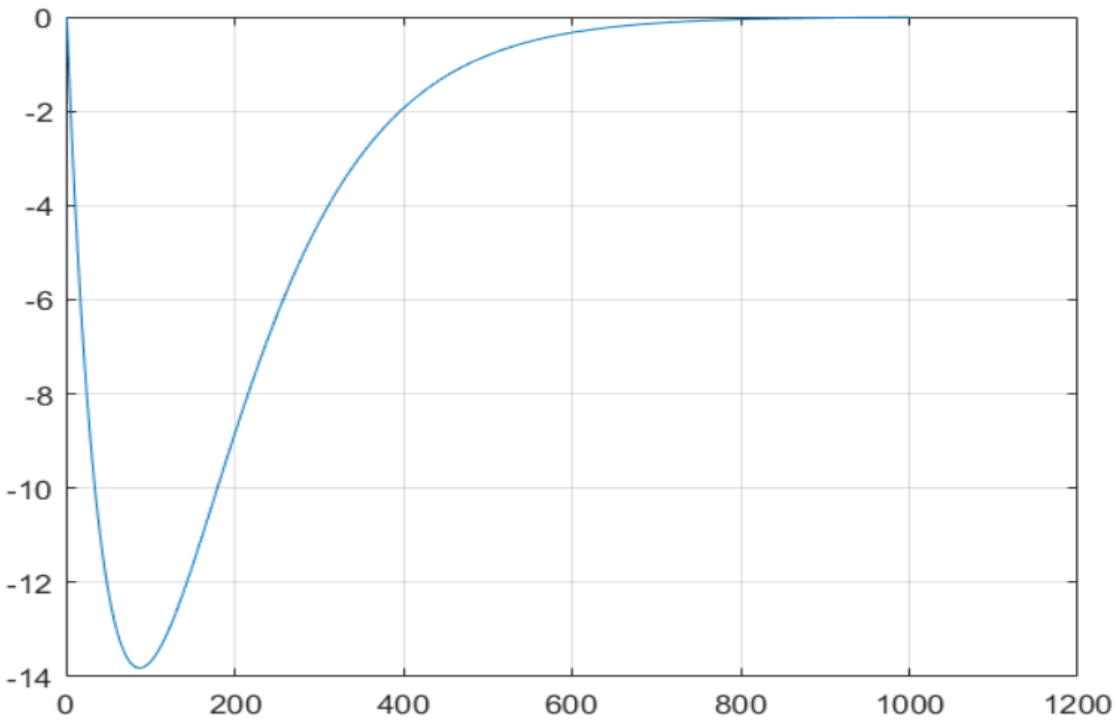
Alpha

Which represents the angle of pendulum.



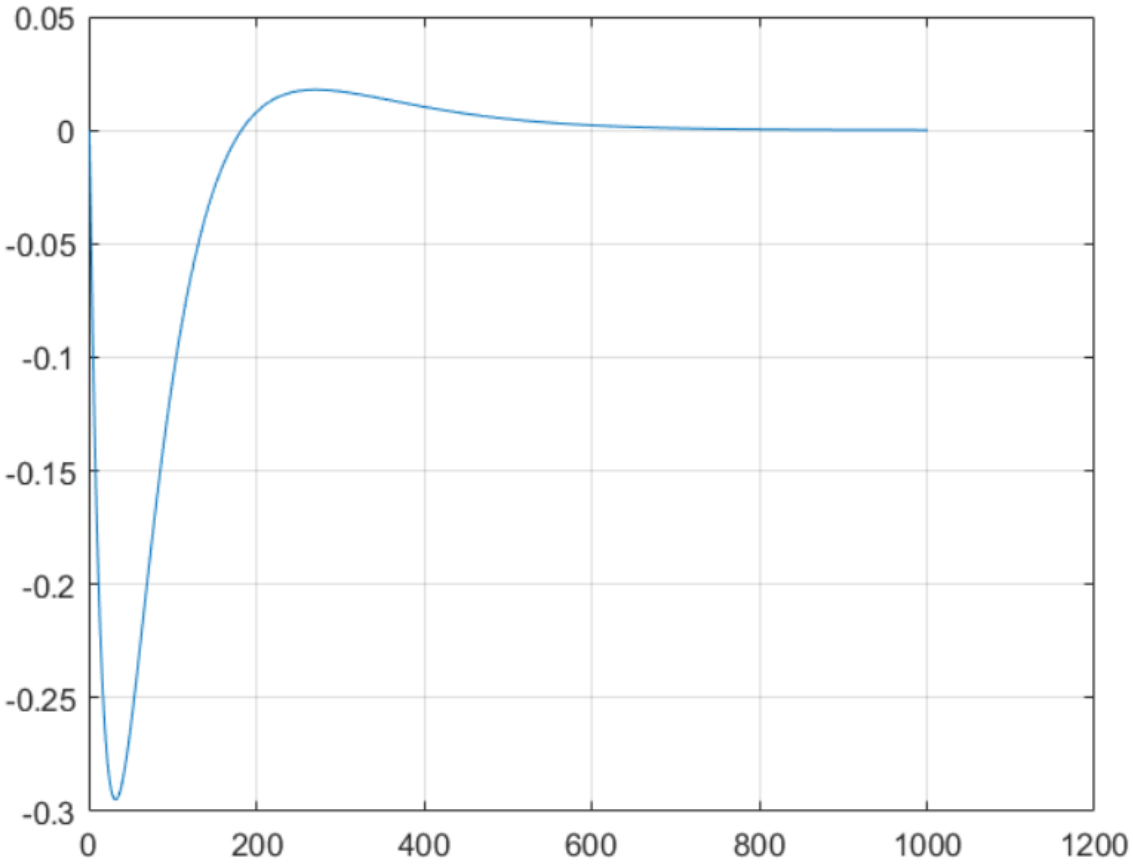
Theta dot

Which represents the speed of rotor.



Alpha dot

Which represents the speed of pendulum.



When running this model that represents the Pole Placement Controller with an initial condition, and parameters that was already calculated in an .m file, the following outputs are observed.

Check the appendix section **Pole Placement code** for the code walkthrough.

Though the Pole Placement controller was able to stabilize α at zero, θ is nonzero.

Comparison Results

The PID controller was able to precisely balance the pendulum and keep it upright. However, without introducing the second PID controller the system continues spinning even after successfully balancing the pendulum. Another PID controller has been added to eliminate spinning but we still weren't able to control θ.

The FSF controller performed well. It was able to keep the pendulum upright and the arm at position zero without compromising rise times and overshoot and LQR gives better performance than pole placement.

The full state feedback controller performed the best giving excellent reaction time and accuracy. Following it the PID controller which did not perform as well but was able to keep the pendulum upright.

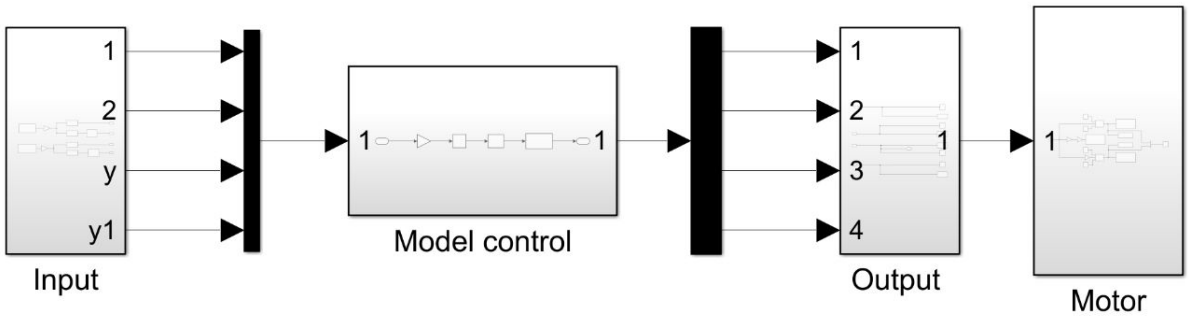
Based on the compare we choose LQR to be our controller for the task.

Hardware In the Loop :

After testing the model and making sure that it works, now the implementation of the controller in real time is needed, so changes in the model must be done to satisfy the conditions in real life, in short it is time to implement the hardware in the loop to our Simulink model

A motor and 2 encoders must be added in order to actuate the motor in real life and to read the output from the encoders .

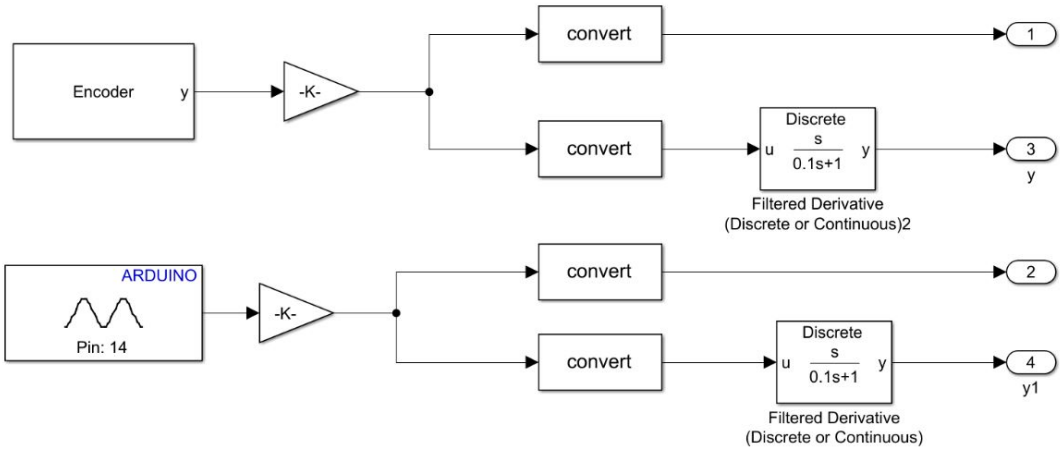
The following model shows the implementation of the hardware in the loop,



Now let's dive into each block.

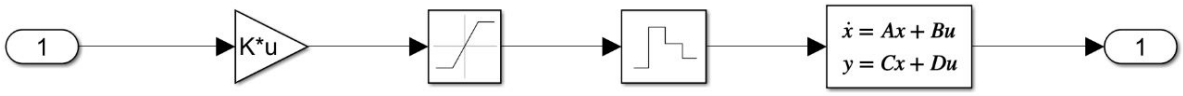
Input Block :

This block contains the encoder which will read the output of the motor and the multi-turn potentiometer.



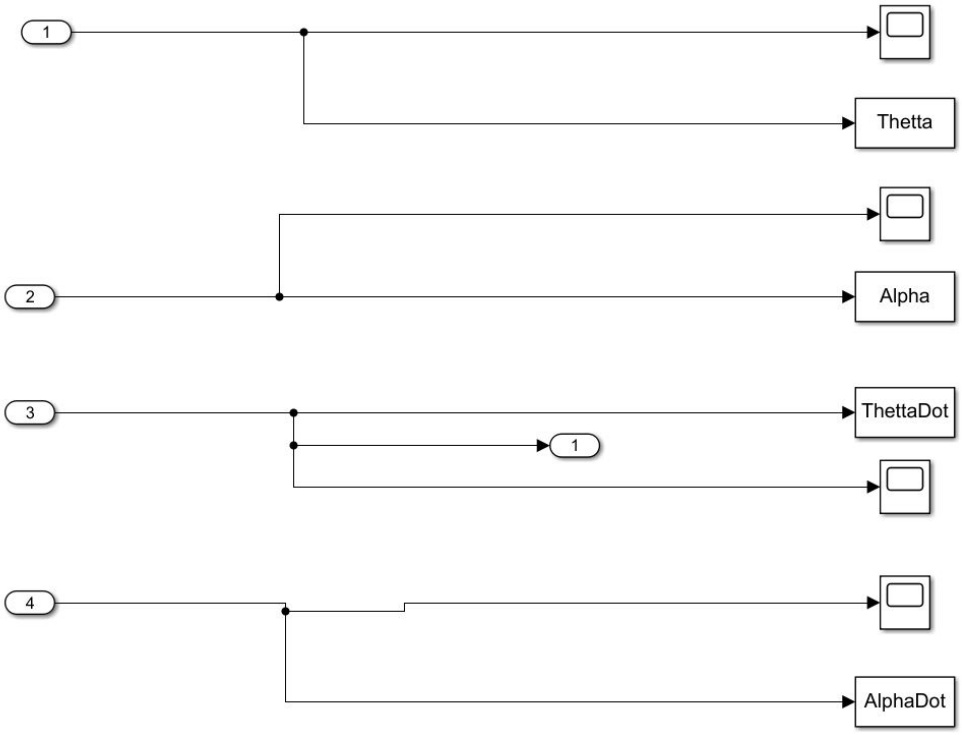
Model control Block :

This block contains the LQR controller



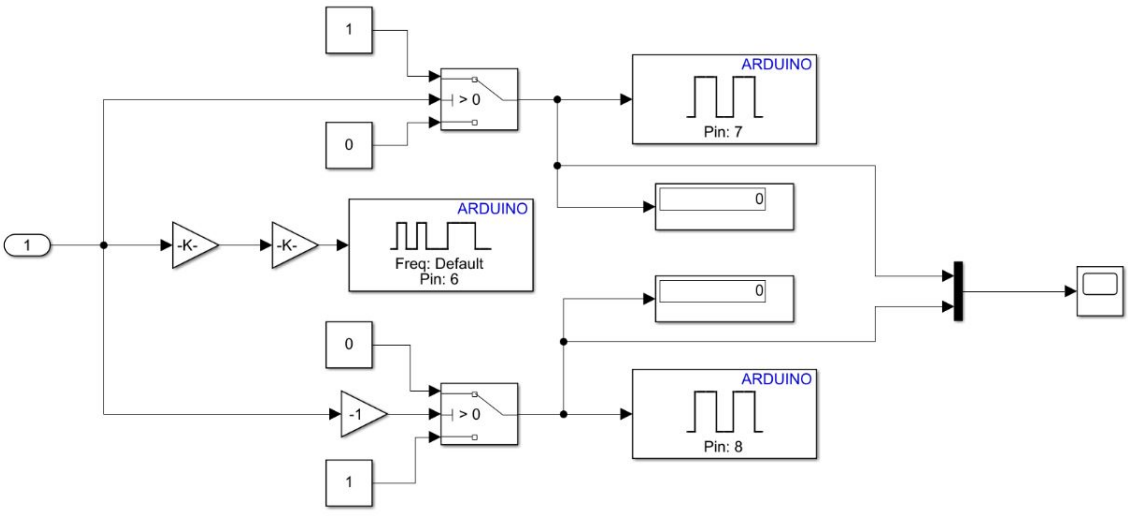
Output Block :

This block contains the output of the controller like theta and alpha.



Motor Block :

This block contains the pins that are used to control the motor in real time.



Links and Contributions :

Final Video: Final Video - Google Drive

Trials and Tests link: Trials and Testing - Google Drive

Link to full CAD: CAD - Google Drive

Link to MATLAB: MATLAB - Google Drive

Contributions :

Mohamed hatem:

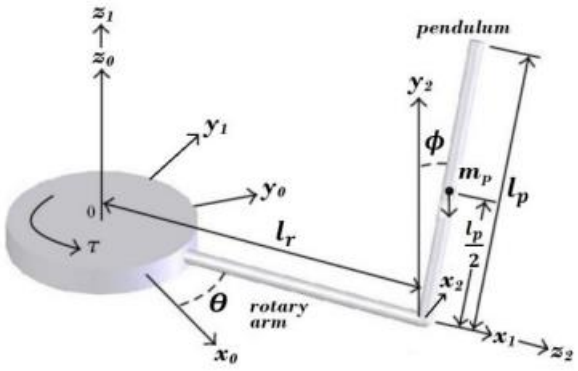
1- Wiring of the Mechanism
2- Mechanical Assembly
3- Hard ware in the loop
4- Mechanical design
5- Report Writing
6- Model linearization

Thomas Medhat:

1- CAD
2- Controller design
3- Controller comparison
4- Hardware in Loop
5- Report Writing
6- Model linearization

Appendix:

Modeling:



The Lagrange function L is defined as the difference of the kinetic energy and potential energy, and the equations describing these two energies respectively are expressed in terms of the variables in the specified coordinate systems

$$L = \left(\frac{1}{2}J_r + \frac{1}{2}m_p l_r^2\right)\dot{\theta}^2 - \frac{1}{2}m_p l_r l_p \dot{\theta}\dot{\phi}cos\phi + \left(\frac{1}{8}m_p l_p^2 + \frac{1}{2}J_p\right)\dot{\phi}^2 - \frac{1}{2}m_p l_p g cos\phi \quad (1)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} + \frac{\partial E_d}{\partial \dot{q}} = F_q$$

$$\left(J_r + m_p l_r^2\right)\ddot{\theta} - \frac{1}{2}m_p l_r l_p \ddot{\phi} cos\phi + \frac{1}{2}m_p l_r l_p \dot{\phi}^2 sin\phi = \tau \quad (3)$$

Euler-Lagrange Equation with respect to ϕ leads to

$$\left(J_p + \frac{1}{4}m_p l_p^2\right)\ddot{\phi} - \frac{1}{2}m_p l_r l_p \ddot{\theta} cos\phi - \frac{1}{2}m_p l_r l_p \dot{\theta}\dot{\phi} sin\phi = 0 \quad (4)$$

The system dynamic model is composed of equations (3) and (4). The linearized equations of motions are shown as below:

$$\left(J_r + m_p l_r^2\right)\ddot{\theta} - \frac{1}{2}m_p l_r l_p \ddot{\phi} = \tau \quad (5)$$

$$\left(J_p + \frac{1}{4}m_p l_p^2\right)\ddot{\phi} - \frac{1}{2}m_p l_r l_p \ddot{\theta} - \frac{1}{2}m_p g l_p \theta = 0 \quad (6)$$

where τ is the output torque of the DC motor, and

$$J_p = \frac{1}{12} m_p l_p^2 \quad , \quad J_r = \frac{1}{3} m_r l_r^2$$

For electrical DC motor, there exist the following relationship:

$$\tau = \frac{k_T (V_m - K_m \dot{\theta})}{R_m} \tag{7}$$

where K_T is the torque constant of the motor, and K_m is the EMF constant of the motor, V_m is the input voltage of the motor, and R_m is the armature resistance of the motor.

State Space Presentation of Rotary Inverted Pendulum System

Denoting

$$\begin{cases} c_1 = J_r + m_p l_r^2 \\ c_2 = -\frac{1}{2} m_p l_r l_p \\ c_3 = \frac{1}{4} m_p l_p^2 + J_p \\ c_4 = -\frac{1}{2} mpl_r l_p \\ c_5 = -\frac{1}{2} m_p g l_p \end{cases}$$

The state-space representation of the model can be written as:

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{(c_3 * K_T * K_m)}{(R_m * (c_1 * c_3 - c_2 * c_4)} & \frac{c_2 * c_5}{c_1 * c_3 - c_2 * c_4} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{c_4 * K_T * K_m}{R_m * (c_1 * c_3 - c_2 * c_4)} & -\frac{c_5}{c_3} & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{x}\theta \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{c_3 * K_T}{R_m * (c_1 * c_3 - c_2 * c_4)} \\ 0 \\ -\frac{c_4 * K_T}{R_m * (c_1 * c_3 - c_2 * c_4)} \end{bmatrix} u \tag{8a}$$

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = A \begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix} + Bu \tag{8b}$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \tag{9}$$

That will lead to for the four state

$$\begin{cases} \dot{X} = AX + Bu \\ Y = CX + Du \end{cases}$$ Where:

$$X = \begin{bmatrix} \theta \\ \alpha \\ \dot{\theta} \\ \dot{\alpha} \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{M_p^2 \cdot l_p^2 \cdot r \cdot g}{J_{eq} \cdot M_p \cdot l_p^2 + M_p \cdot r^2 \cdot J_p + J_{eq} \cdot J_p} & -\frac{(J_p . K_t . K_m + M_p . l_p^2 . K_t . K_m)}{R_m . (J_{eq} . J_p + J_{eq} . M_p . l_p^2 + M_p . r^2 . J_p)} & -B_{eq} \\ 0 & \frac{M_p . l_p . g (J_{eq} + M_p . r^2)}{J_{eq} \cdot M_p \cdot l_p^2 + M_p \cdot r^2 \cdot J_p + J_{eq} \cdot J_p} & -\frac{M_p . l_p . r . K_t . K_m}{R_m . (J_{eq} . J_p + J_{eq} . M_p . l_p^2 + M_p . r^2 . J_{p)}} & -B_p \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ -\frac{K_t . (J_p + M_p . l_p^2)}{R_m . (J_{eq} . J_p + J_{eq} . M_p . l_p^2 + M_p . r^2 . J_{p)}} \\ -\frac{M_p . l_p . r . K_t}{R_m . (J_{eq} . J_p + J_{eq} . M_p . l_p^2 + M_p . r^2 . J_{p)}} \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

LQR code

```
clc;
close all;
clear;
g= 9.81;        %gravity constant
Mp = 0.027;     %mass of pendulum assembly
lp = 0.04;     %center of mass of pendulum assembly  (0.153)
r=  0.0826;     %length from motor shaft to pendulum pivot
Jp= 0.000698;   %pendulum moment of inertia relative to pivot
Jeq= 0.000368;  %equivalent moment of inertia acting on the DC motor shaft
Bp= 1.4;        %Viscous damping about the pendulum pivot
Beq= 0.93;      %equivalent viscous damping acting on the DC motor shaft
Kt= 0.0333;     %DC motor current-torque constant
Km= 0.0333;     %DC motor back-emf constant
Rm= 8.7;        %Electric resistance of the DC motor armature
IC=[0 0.2 0 0]
setpoint = [0 0 0 0]
A= [[0 , 0 , 1 , 0];
    [0 , 0 , 0 , 1];
    [0 , Mp^2*lp^2*r*g/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -
(Jp*Kt*Km+Mp*lp^2*Kt*Km)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -Beq];
    [0 , Mp*lp*g*(Jeq+Mp*r^2)/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -
Mp*lp*r*(Kt*Km)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -Bp] ]
B=  [0 ; 0 ; Kt*(Jp+Mp*lp^2)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) ;
Mp*lp*Kt*r/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp)]
C= [[1 0 0 0];
    [0 1 0 0];
    [0 0 1 0];
    [0 0 0 1]]
D= [0 ; 0 ; 0 ; 0]
Q= [[3 0 0 0]
    [0 2 0 0]
    [0 0 5 0]
    [0 0 0 0]];
R=1;
[K,S,e]=lqr(A,B,Q,R)
simout= sim("LQR.slx");
plot(Thetta)
grid on
plot(ThettaDot)
grid on
plot(Alpha)
grid on
plot(AlphaDot)
grid on
close all;
```

PID code

```
clc;
close all;
clear;
g= 9.81;        %gravity constant
Mp = 0.027;     %mass of pendulum assembly
lp = 0.04;     %center of mass of pendulum assembly  (0.153)
r=  0.0826;     %length from motor shaft to pendulum pivot
Jp= 0.000698;   %pendulum moment of inertia relative to pivot
Jeq= 0.000368;  %equivalent moment of inertia acting on the DC motor shaft
Bp= 1.4;        %Viscous damping about the pendulum pivot
Beq= 0.93;      %equivalent viscous damping acting on the DC motor shaft
Kt= 0.0333;     %DC motor current-torque constant
Km= 0.0333;     %DC motor back-emf constant
Rm= 8.7;        %Electric resistance of the DC motor armature
IC=[0 0.2 0 0]
setpoint = [0 0 0 0]
A= [[0 , 0 , 1 , 0];
    [0 , 0 , 0 , 1];
    [0 , Mp^2*lp^2*r*g/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -
(Jp*Kt*Km+Mp*lp^2*Kt*Km)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -Beq];
    [0 , Mp*lp*g*(Jeq+Mp*r^2)/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -
Mp*lp*r*(Kt*Km)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -Bp] ]
B=  [0 ; 0 ; Kt*(Jp+Mp*lp^2)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) ;
Mp*lp*Kt*r/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp)]
C= [[1 0 0 0];
    [0 1 0 0];
    [0 0 1 0];
    [0 0 0 1]]
D= [0 ; 0 ; 0 ; 0]
Kp = -2118458.11238048
Ki = -41.326250198697
Kd = -14758415775.2329
N = 0.0290267992226422
simout= sim("PID.slx");
plot(Thetta)
grid on
plot(ThettaDot)
grid on
plot(Alpha)
grid on
plot(AlphaDot)
grid on
close all;
```

Pole Placement code

```
clc;
close all;
clear;
g= 9.81;        %gravity constant
Mp = 0.027;     %mass of pendulum assembly
lp = 0.04;     %center of mass of pendulum assembly  (0.153)
r=  0.0826;     %length from motor shaft to pendulum pivot
Jp= 0.000698;   %pendulum moment of inertia relative to pivot
Jeq= 0.000368;  %equivalent moment of inertia acting on the DC motor shaft
Bp= 1.4;        %Viscous damping about the pendulum pivot
Beq= 0.93;      %equivalent viscous damping acting on the DC motor shaft
Kt= 0.0333;     %DC motor current-torque constant
Km= 0.0333;     %DC motor back-emf constant
Rm= 8.7;        %Electric resistance of the DC motor armature
IC=[0 0.2 0 0]
setpoint = [0 0 0 0]
A= [[0 , 0 , 1 , 0];
    [0 , 0 , 0 , 1];
    [0 , Mp^2*lp^2*r*g/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -
(Jp*Kt*Km+Mp*lp^2*Kt*Km)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -Beq];
    [0 , Mp*lp*g*(Jeq+Mp*r^2)/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -
Mp*lp*r*(Kt*Km)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) , -Bp] ]
B=  [0 ; 0 ; Kt*(Jp+Mp*lp^2)/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp) ;
Mp*lp*Kt*r/Rm/(Jeq*Jp+Jeq*Mp*lp^2+Mp*r^2*Jp)]
C= [[1 0 0 0];
    [0 1 0 0];
    [0 0 1 0];
    [0 0 0 1]]
D= [0 ; 0 ; 0 ; 0]
rank(ctrb(A,B))
K = place(A,B,[-1,-1.23,-5.0,0])
simout= sim("Pole.slx");
plot(Thetta)
grid on
plot(ThettaDot)
grid on
plot(Alpha)
grid on
plot(AlphaDot)
grid on
close all;
```