Inverted Pendulum Balancing using LQR Control

A Design Lab Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Anindya Biswas (210108004)

J. Spandhan Srirag (210102040)

under the guidance of

Dr. Chayan Bhawal

to the

DEPARTMENT OF ELECTRONICS AND ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

CERTIFICATE

*This is to certify that the work contained in this project entitled "**Inverted Pendulum Balancing using LQR Control**" is a bonafide work of **Anindya Biswas (Roll No. 210108004)** and **J. Spandhan Srirag (Roll No. 210102040)**, carried out in the Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

Supervisor: **Dr. Chayan Bhawal**

Assistant Professor,

Department of Electronics and Electrical Engineering,

Indian Institute of Technology Guwahati, Assam.

May, 2024

Guwahati.

Acknowledgements

We wish to express our deep appreciation to Dr. Chayan Bhawal for his valuable guidance, assistance, patience and encouragement while developing this project.

We also want to express our sincere gratitude to the Electronics Club, IIT Guwahati for constantly supporting us with their valuable resources and giving us a working environment.

Contents

List of Figures

Chapter 1

Introduction

1.1 Problem

An inverted pendulum is a fascinating mechanical system (as shown in fig. 1.1) where unlike the conventional pendulum, pendulum rod oscillates in upright position. The inverted pendulum is inherently unstable. Without any control mechanism, it would simply fall over.

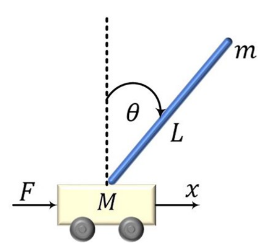

Fig. 1.1: Simple Inverted Pendulum Balancing System

To keep the inverted pendulum stable in vertically up position, we rely on feedback control systems. These systems continuously monitor the pendulum's angle and adjust the position of the pivot point horizontally when the pendulum starts to tilt. By doing so, they maintain balance and prevent the pendulum from toppling over.

This system, also referred to as the Broom-balancer, tackles a challenge akin to balancing a broomstick. Its practical application lies in controlling the orientation of missiles or rockets during their initial launch stages. Imagine a satellite booster rocket, resembling a colossal broomstick in terms of aerodynamics, requiring precise balance through engine rotation to align with its thrust vector. For our project, we simplify the problem from three dimensions to two, focusing solely on pendulum motion within a plane and restricting cart movement to the horizontal axis.

1.2 Objective

The primary objective of this project is to develop and build a balancing system for an inverted pendulum that meets specified performance criteria. The design process encompasses both mechanical and control system aspects. In mechanical system design, tasks involve structuring the system, selecting and designing components, and identifying suitable sensors for measuring linear and angular displacements and velocities. Control system design entails analyzing feedback control systems, establishing mathematical models of the dynamic system, deriving motion differential equations, linearizing equations around a given operating point, scrutinizing feedback control system characteristics, and crafting feedback controllers using state feedback control methods and linear optimal control techniques.

Chapter 2

Design of the System

2.1 Basic Components of the System

A typical feedback control system is a closed loop system comprising of: (1) A plant: a system that needs to be controlled; (2) a sensor network: systems that provides information about the plant and sends it back to a controller; and (3) a controller: the central processing unit of the control system which compares the measurements from the sensors and adjusts the input to the plant such that the output of the plant is maintained at the desired value.

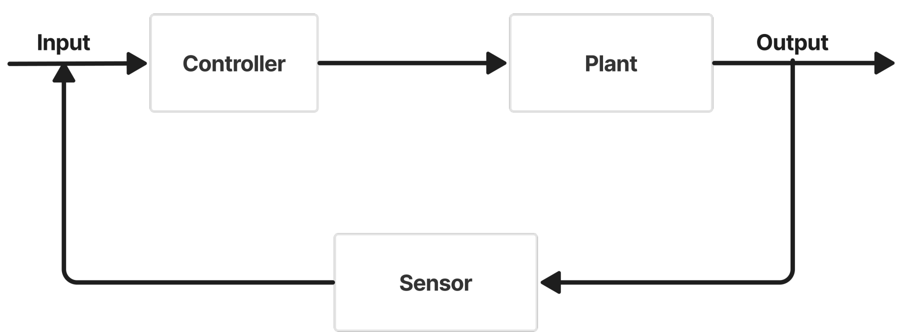

Fig. 2.1: Block Diagram of Closed-loop Control System

Figure 2.1 shows a typical closed loop system. Conforming to the block diagram in Figure 2.1, the inverted pendulum set-up designed in this work admits the following basic blocks: (1) Plant: the inverted pendulum and the platform to which the pendulum is to be attached; (2) a sensor network: this comprises of the sensors to detect the angular position, angular velocity of the pendulum and also the linear position and linear velocity of the cart; , and (3) a controller: this is an Arduino based controller which takes the readings from the sensors, processes them to provide the actuating signals to the cart. The block diagram of the designed system is shown in fig. 2.2 .

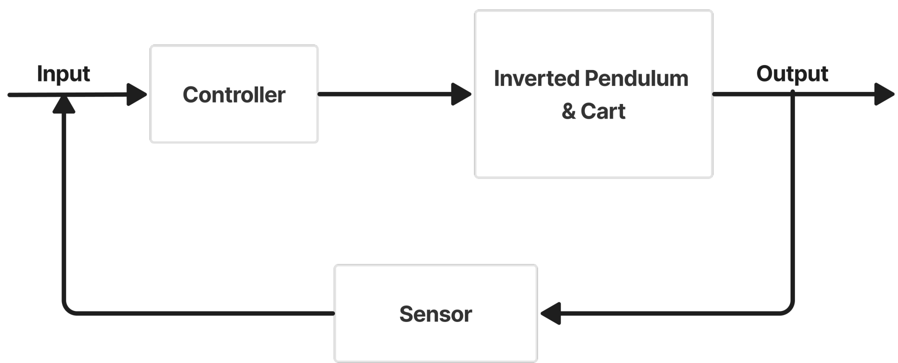

Fig. 2.2: Components of Inverted Pendulum Balancing System

2.2 Decided Model

There are many factors to consider before we chose our design. We had to consider our budget, the availibilty of components, the complexity to build the system, cost of the sensors etc. After considering all those we finally decided to proceed with the model as shown in fig. 2.3.

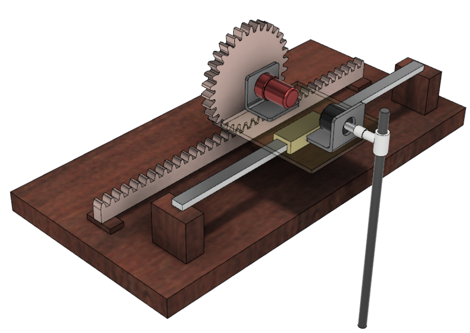

Fig. 2.3: Orthographic View of the Model

2.2.1 Cart and Pendulum

As shown in fig. 2.3, the cart (yellow colored) is mounted on the linear slide rail which is fixed on the wooden platform. The cart can be moved along the rail by applying the torque on the pinion gear that is attached with the DC motor shaft. Between the cart and rail, bearing balls and lubricants are used to reduce friction. On the top of the cart, a rectangular transparent acrylic sheet is placed to attach other components to the cart. Two L-clamps are attached to the acrylic sheet, one for holding optical rotary encoder (black colored), the other for holding the DC motor (red colored). The pendulum rod is attached to the rotary encoder using a clamp (white colored). The rotary encoder is used to measure the angular position of the pendulum.

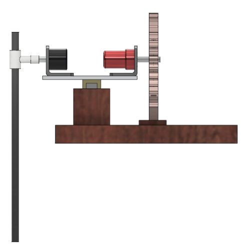

Fig. 2.4: Right Side View

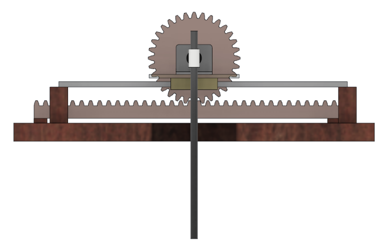

Fig. 2.5: Front View

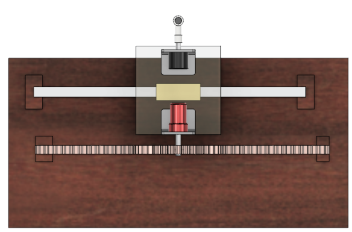

Fig. 2.6: Top View

2.2.2 DC Motor

In the system, the DC motor is fixed on an L-clamp, and a 3D-printed pinion gear is attached to the shaft of the motor. The pinion gear runs on the rack gear that is attached to the wooden platform.

The DC motor we have selected for this purpose is RHINO HEAVY DUTY PLANETARY GEARED MOTOR. The specifications of the DC motor which were chosen in the design are as follows:

Operating voltage = 12 V;

Motor Speed at Output Shaft = 103 RPM;

Stall Torque = 75 Kgcm;

Rated Torque = 25 Kgcm;

Gear Ratio = 1:174;

Weight = 300 g.

Fig. 2.7: Rhino 100RPM motor

2.2.3 Sensors

An incremental optical rotary encoder has been used to measure the angular position of the pendulum. Hall effect magnetic encoder has been used to measure the angular position of the motor shaft and eventually the linear position of the cart.

Specifications of Rotary Optical Encoder:

Model No. – 3806-OPTI-600-AB-OC

Encoder Type – Incremental

Operating Voltage (V_{DC}) – 5 $\sim$ 24

Fig. 2.8: Incremental Optical Rotary Encoder

Current Consumption (mA) – ≤ 40

Maximum Speed – 5000 RPM

Pulse Per Revolution (PPR) – 600

Counts Per Revolution (CPR) – 2400

Fig. 2.9: Hall Effect Magnetic Encoder

Specification of the Hall Effect Magnetic Encoder:

Model No. – SCX3530

Input Supply voltage(V) – $4.5 \sim 18$

Supply Current(A) – 8

Output Current(A) – 0.1

The angular velocity of the pendulum and linear velocity of the cart has been calculated from those measured quantities.

2.3 Mechanical Parts

We used the 3D printing facility through Electronics Club, IITG to print the mechanical parts such as the the pinion gear that runs on the rack (linear) gear. We 3D printed the clamps to attach motor and rotary optical encoder to the platform. Also, the joint used to join the pendulum and the shaft of the encoder was 3D printed. The rack gear is made of transparent acrylic sheets, which were cut using laser cutter facility available in the New SAC.

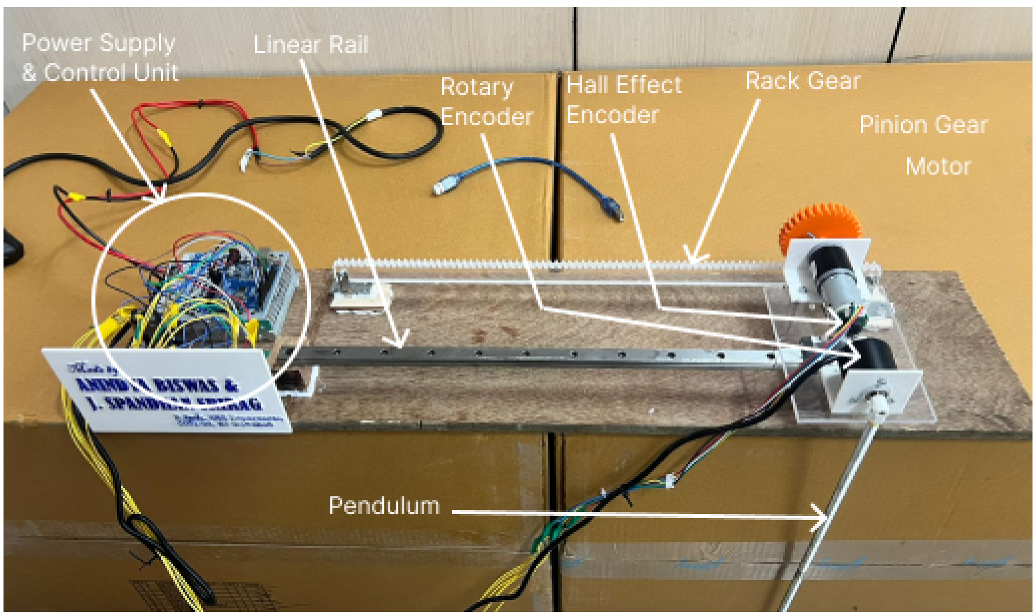

Fig. 2.10: The Physical Setup

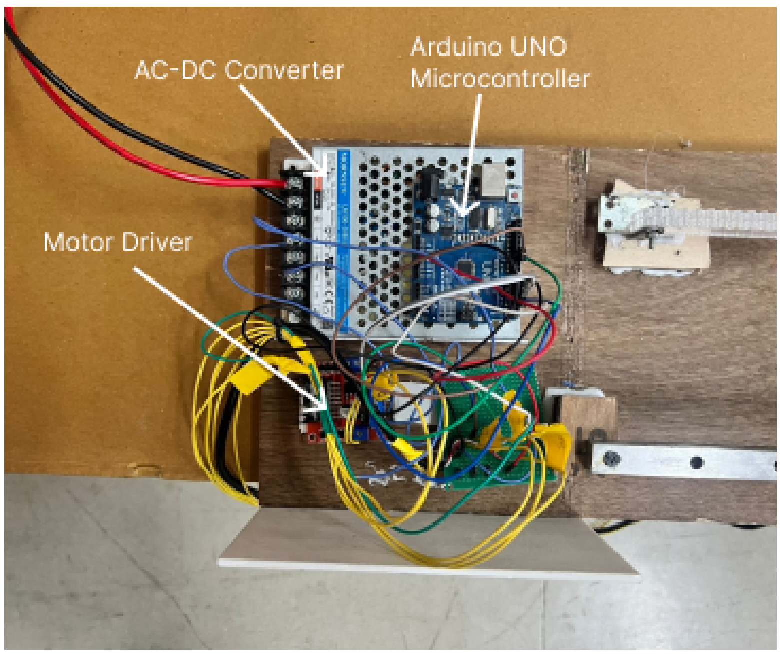

Fig. 2.11: Power Supply & Control Unit

Chapter 3

Control System Analysis and Design

In this chapter we present the dynamic equations governing a typical inverted pendulum set-up.

3.1 Dynamic Equations

In order to design a controller to stabilize a inverted pendulum, we need to determine the dynamical equations governing the system. We reprodue the equations from [2] next. The system in fig. 3.1 houses a cart and an inverted pendulum attached to the cart. We define the following variables for the slider:

1. Mass of the cart: M.

2. Mass of the pendulum: m.

3. Length of the pendulum: $2 \times l$.

4. Force applied to the cart to keep the pendulum upright: f.

5. Position of the cart: x.

6. Angular positio of the pendulum: θ.

The free-body diagram of the slider is shown in fig. 3.2.

We define the following variables for the cart:

1. Horizontal force exerted by the cart on the pendulum: H_f.

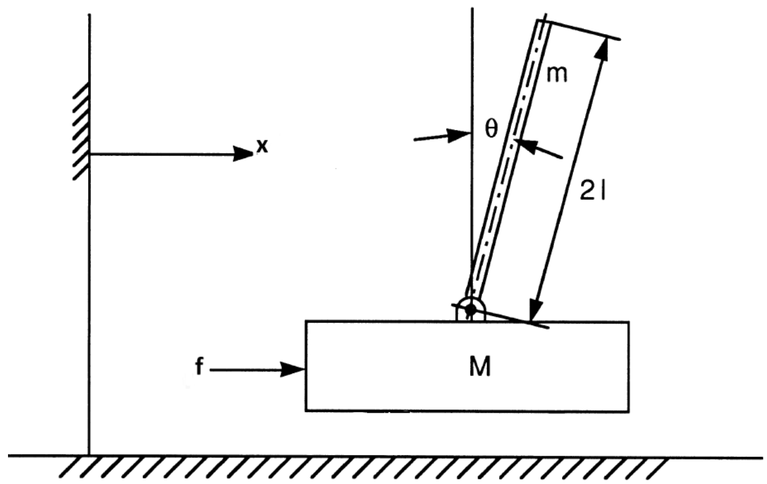

Fig. 3.1: Inverted pendulum with slider

2. Vertical force exerted by the cart on the pendulum: V_f.

3. Viscous friction coefficient for rotary motion of the pendulum: b_2.

4. Viscous friction coefficient for linear motion of the cart b_1.

5. Acceleration due to gravity: g.

6. Moment inertia of pendulum with respect to the center of gravity: I.

The free-body diagram of the pendulum is shown in fig. 3.3.

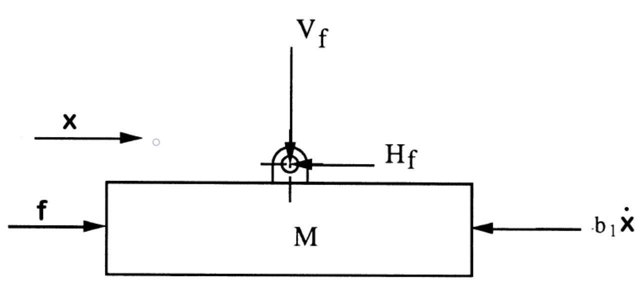

Fig. 3.2: Free body diagram of cart(slider)

By applying Newton's second law to the horizontal motion of the cart, and to the horizontal, vertical and rotary motions of the pendulum, we have the following equations.

$$M\ddot{x} = f - H_f - b_1\dot{x} \tag{3.1}$$

$$V_f - mg = m\frac{\mathrm{d}^2(lcos\theta)}{\mathrm{d}t^2} \tag{3.2}$$

$$H_f = m\frac{\mathrm{d}^2(x + lsin\theta)}{\mathrm{d}t^2} \tag{3.3}$$

$$I\ddot{\theta} = (V_f sin\theta - H_f cos\theta)l - b_2\dot{\theta} \tag{3.4}$$

where, $I = \frac{1}{3}ml^2$

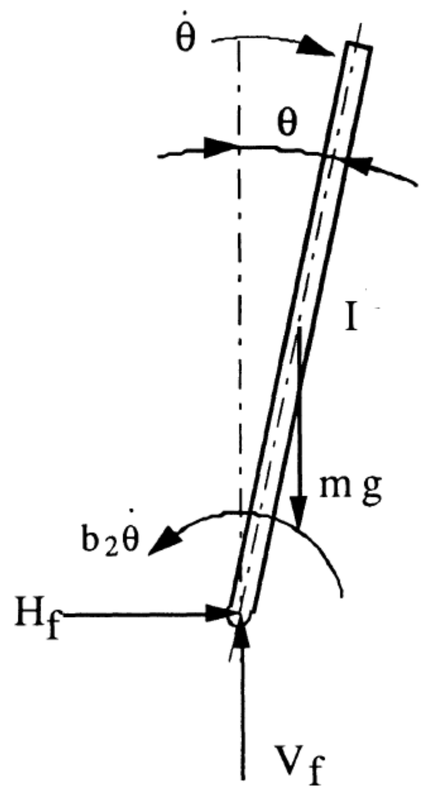

Fig. 3.3: Free body diagram of pendulum

The viscous friction coefficients b_1 and b_2 are neglected because in the physical model, all the components that experience friction are properly lubricated. Now, substituting the values of V_f and H_f in eq. 3.1 and 3.4 we get the following equations.

$$I\ddot{\theta} = l(mgsin\theta - m\ddot{x}cos\theta - ml\ddot{\theta}) \tag{3.5}$$

$$f = (M+m)\ddot{x} + ml\ddot{\theta}cos\theta - ml\dot{\theta}^2 sin\theta \tag{3.6}$$

The non-linear equations in eq. (3.5) and eq. (3.6) are linearized first with the assumption that θ and $\dot{\theta}$ will be small. Since the objective of controlling the system is to keep the pendulum upright, it is reasonable to assume that θ and $\dot{\theta}$ will remain close to zero. With this assumption, the equations can be linearized by retaining only those terms that are linear in θ and $\dot{\theta}$ and neglecting higher order terms because they will be insignificantly small when θ and $\dot{\theta}$ are close to zero. So after simplifications, we get the following equations from eq. 3.5 and 3.6.

$$I\ddot{\theta} = lmg\theta - lm\ddot{x} - ml^2\ddot{\theta} \tag{3.7}$$

$$f = (M+m)\ddot{x} + ml\ddot{\theta} \tag{3.8}$$

3.2 DC Motor Characteristics and State Space

We need to obtain the characteristics[1] of the DC motor being used so that we can calculate the voltage required to exert a certain force on the slider. We start by modeling the DC motor as shown in the circuit below.

In the circuit above we see various parameters. These are:

V_m: The input armature voltage of the DC motor; (V)

i: The armature current of the DC motor; (A)

R_m: The armature resistance of the DC motor; (Ω)

L_m: The armature inductance of the DC motor; (H)

T_m: The output torque of the DC motor; (Nm)

θ_m: The angular position of the motor shaft; (rad)

Motor parameters are obtained as below by fitting the practical dataset to the simulated dataset

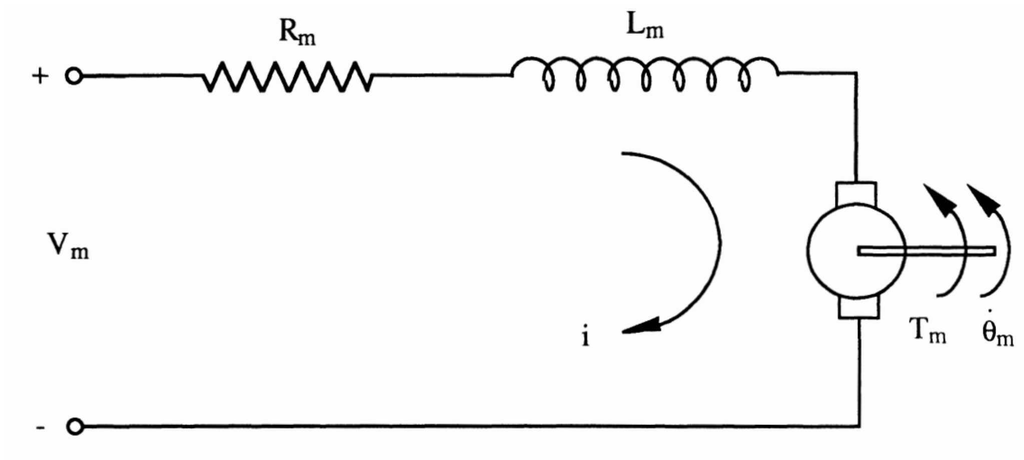

Fig. 3.4: Circuit of DC Motor

using MATLAB Parameter Estimation utility.

The torque constant, $K_T = 0.0459\,\mathrm{N\cdot m/A}$;

The e.m.f. constant, $K_b = 0.23\,\mathrm{V\cdot s/rad}$;

The armature resistance, $R = 2\,\Omega$;

The armature inductance, $L = 2\,\mathrm{mH}$;

The Moment of inertia of motor shaft, $I = 0.05\,\mathrm{kg\cdot m^2}$.

$$T_m = K_T i - b\dot{\theta}_m \tag{3.9}$$

(where K_T is the torque constant and b is the viscous friction coefficient of the motor.)

Now by applying KVL on the above circuit, we get.

$$V_m = L_m \frac{\mathrm{d}i}{\mathrm{d}t} + iR_m + K_e\dot{\theta}_m \tag{3.10}$$

(where K_e is emf constant of motor)

Dividing by R_m on both sides.

$$\frac{V_m}{R_m} = \frac{L_m}{R_m}\frac{\mathrm{d}i}{\mathrm{d}t} + i + \frac{K_e\dot{\theta}_m}{R_m} \tag{3.11}$$

As the value of $\frac{L_m}{R_m}$ is very small we will approximate it to zero. Now we get an expression for i, by substituting in eq. 3.9 we get.

$$T_m = \frac{K_T V_m}{R_m} - \frac{K_T K_e}{R_m}\dot{\theta}_m - b\dot{\theta}_m \tag{3.12}$$

By equating motor's torque to force applied we get.

$$T_m - fr = I_m \ddot{\theta_m} \tag{3.13}$$

(where I_m is moment of inertia of motor shaft)

$$x = \theta_m r \tag{3.14}$$

From eq. 3.8 and 3.13 we get

$$(m+M)r\ddot{x} + mlr\ddot{\theta} = T_m - I_m\frac{\ddot{x}}{r} \tag{3.15}$$

Substituting result of eq. 3.12 in eq. 3.15 we get

$$((m+M)r^2 + I_m)\frac{\ddot{x}}{r} + mlr\ddot{\theta} = T_m \tag{3.16}$$

Substituting result of eq.3.12 we get

$$((m+M)r^2 + I_m)\frac{\ddot{x}}{r} + mlr\ddot{\theta} = \frac{K_T V_m}{R_m} - \frac{K_T K_e}{R_m}\frac{\dot{x}}{r} - b\frac{\dot{x}}{r} \tag{3.17}$$

Now if we consider x, $\dot{x}$, θ, $\dot{\theta}$ as four states from eqs. 3.17 and 3.8 we get the final state space as

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & a_1 & a_2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & a_3 & a_4 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ b_1 \\ 0 \\ b_2 \end{bmatrix} V_m \tag{3.18}$$

Where

$a_1 = \frac{-(I+ml^2)(\frac{K_T K_e}{R_m}+b)}{\Delta};$

$a_2 = \frac{-m^2 l^2 g r^2}{\Delta};$

$a_3 = \frac{(\frac{K_T K_e}{R_m}+b)ml}{\Delta};$

$a_4 = \frac{((m+M)r^2+I_m)mlg}{\Delta};$

$b_1 = \frac{(I+ml^2)K_T r}{\Delta R_m};$

$b_2 = \frac{-mrlK_T}{\Delta R_m};$

$\Delta = ((m+M)r^2 + I_m)I + Mmr^2l^2 + mI_ml^2.$

This can be written in the form

$$\dot{X} = AX + BU \tag{3.19}$$

Where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & a_1 & a_2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & a_3 & a_4 & 0 \end{bmatrix} \tag{3.20}$$

$$B = \begin{bmatrix} 0 \\ b_1 \\ 0 \\ b_2 \end{bmatrix} \tag{3.21}$$

$$U = V_m \tag{3.22}$$

Substituting the values

$M = 0.545\,\mathrm{Kg};$

$m = 0.133\,\mathrm{Kg};$

$l = 0.2125\,\mathrm{m};$

$I = 0.002\,\mathrm{Kgm^2};$

$I_m = 0.407\,73\,\mathrm{Kgm^2};$

$b = 0.424\,26\,\mathrm{Kgs}^{-1}$;

$L_m = 6.257\,\mathrm{H}$;

$R_m = 136.51\,\Omega$;

$K_T = 386.05\,\mathrm{N \cdot m/A}$;

$K_e = 0.6392\,\mathrm{V \cdot s/rad}$;

$r = 0.032\,\mathrm{m}$;

$g = 9.8\,\mathrm{m/s^2}$;

We get

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -5.4661 & -0.0025 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 19.2920 & 34.5969 & 0 \end{bmatrix} \tag{3.23}$$

$$B = \begin{bmatrix} 0 \\ 0.2216 \\ 0 \\ -0.7822 \end{bmatrix} \tag{3.24}$$

3.3 Controllability and Observability

Before designing a controller we need to first check the controllability and observability of the above system.

For linear systems if there exists an input U that can drive system from initial state to zero state in finite time then the particular state is said to be controllable. If all initial states are controllable, the system is said to be completely state controllable.

Compute controllability matrix.

$$\Gamma_c = \begin{bmatrix} B & AB & A^2B & A^3B \end{bmatrix} \tag{3.25}$$

$$\Gamma_c = \begin{bmatrix} 0 & 0.2216 & -1.2114 & 6.6237 \\ 0.2216 & -1.2114 & 6.6237 & -36.2162 \\ 0 & -0.7822 & 4.2757 & -50.4334 \\ -0.7822 & 4.2757 & -50.4334 & 275.7094 \end{bmatrix} \tag{3.26}$$

$$rank(\Gamma_c) = 4 \tag{3.27}$$

From eq. 3.27 we can say that the system is completely state controllable.

For the construction of the states from the output it should satisfy the observability condition.

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.28}$$

$$\Gamma_o = \begin{bmatrix} C^T & A^T C^T & (A^T)^2 C^T & (A^T)^3 C^T \end{bmatrix} \tag{3.29}$$

$$\Gamma_o = 0.001 \begin{bmatrix} 0.0010 & 0 & 0 & 0 \\ 0 & 0.0010 & 0 & 0 \\ 0 & 0 & 0.0010 & 0 \\ 0 & 0 & 0 & 0.0010 \\ 0 & 0.0010 & 0 & 0 \\ 0 & -0.0055 & -0.0000 & 0 \\ 0 & 0 & 0 & 0.0010 \\ 0 & 0.0193 & 0.0346 & 0 \\ 0 & -0.0055 & -0.0000 & 0 \\ 0 & 0.0299 & 0.0000 & -0.0000 \\ 0 & 0.0193 & 0.0346 & 0 \\ 0 & -0.1055 & -0.0000 & 0.0346 \\ 0 & 0.0299 & 0.0000 & -0.0000 \\ 0 & -0.1634 & -0.0002 & 0.0000 \\ 0 & -0.1055 & -0.0000 & 0.0346 \\ 0 & 1.2438 & 1.1972 & -0.0000 \end{bmatrix} \tag{3.30}$$

$$rank(\Gamma_o) = 4 \tag{3.31}$$

From eq. 3.31 we can say that the system is completely observable. As the system is observable as well as controllable, the system can be made stable by feedback through an observer.

3.4 Controller Design

We have used the optimal control method for our controller design. The controller is achieved by working with quadratic performance indices and minimizing a specified performance criterion. This gives us best controller according to the cost assigned to each state and the cost assigned to control input.

The controller is called as LQR (Linear Quadratic Regulator) controller[3]. For a linear system $\dot{X} = AX + BU$ the optimal controller is obtained by minimizing the cost function J of the form

$$J = \frac{1}{2}\int_0^\infty (X^TQX + U^TRU)\,dt \tag{3.32}$$

The controller is given by

$$U = -KX \tag{3.33}$$

$$K = R^{-1}B^TP \tag{3.34}$$

In the above equations

P : symmetric and positive definite matrix;

Q : symmetric and positive semidefinite matrix;

R : symmetric and positive definite matrix.

The weighting matrices Q and R are usually diagonal. The terms X^TQX and U^TRU of the integrand are quadratic forms which measure, respectively, the performance and the cost of control. So choosing Q and R is very important.

According to the tolerable error limits of all the states and motor capacity through various iterations in Matlab matrices Q and R are chosen as

$$Q = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 199 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 199 \end{bmatrix} \tag{3.35}$$

$$R = 10000 \tag{3.36}$$

These Q and R values are used to compute K matrix using Matlab code. The K Obtained is

$$K = \begin{bmatrix} -0.0838 & -45.3809 & -156.9463 & -26.6846 \end{bmatrix} \tag{3.37}$$

Chapter 4

Experimental Results

At first, we have simulated our designed model on MATLAB and checked for different performance criteria. Then we performed the experiment on our physical model.

4.1 Matlab Simulation

For the designed controller in the above section, simulation is performed giving an initial deflection. Initial values of the states are :

$\mathrm{x} = 0.00\,\mathrm{m}$

$\dot{x} = 0.00\,\mathrm{m}$

$\theta = 0.05\,\mathrm{rad}$

$\dot{\theta} = 0.1\,\mathrm{rad/s}$

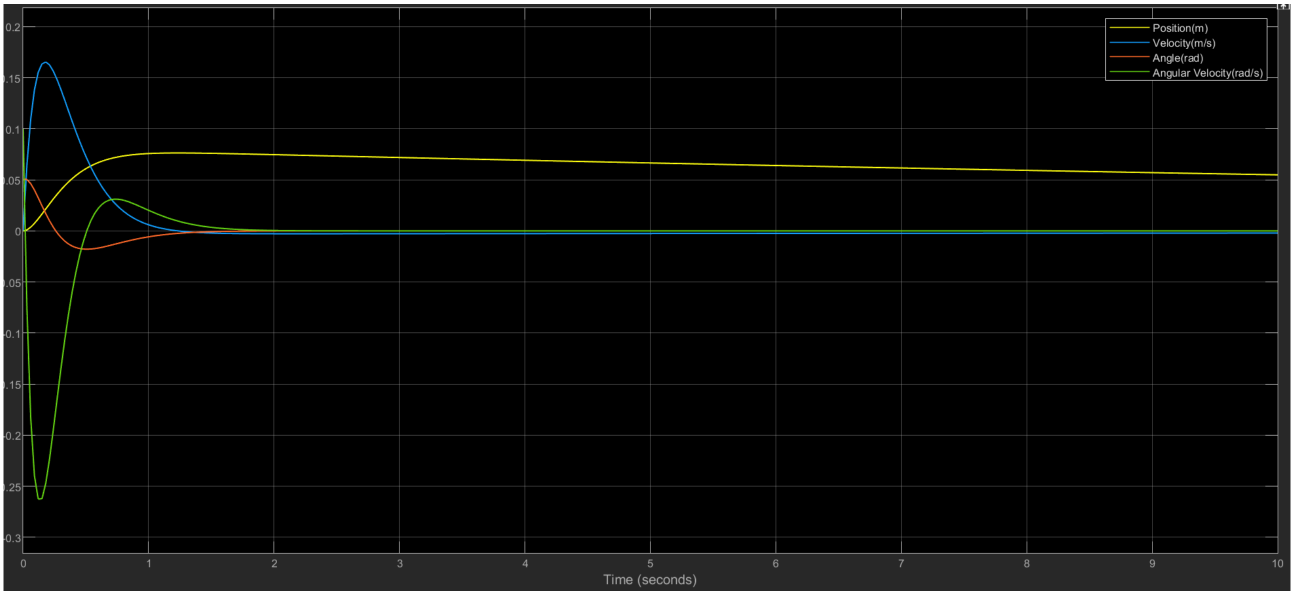

Fig. 4.1: Simulated Response of the System

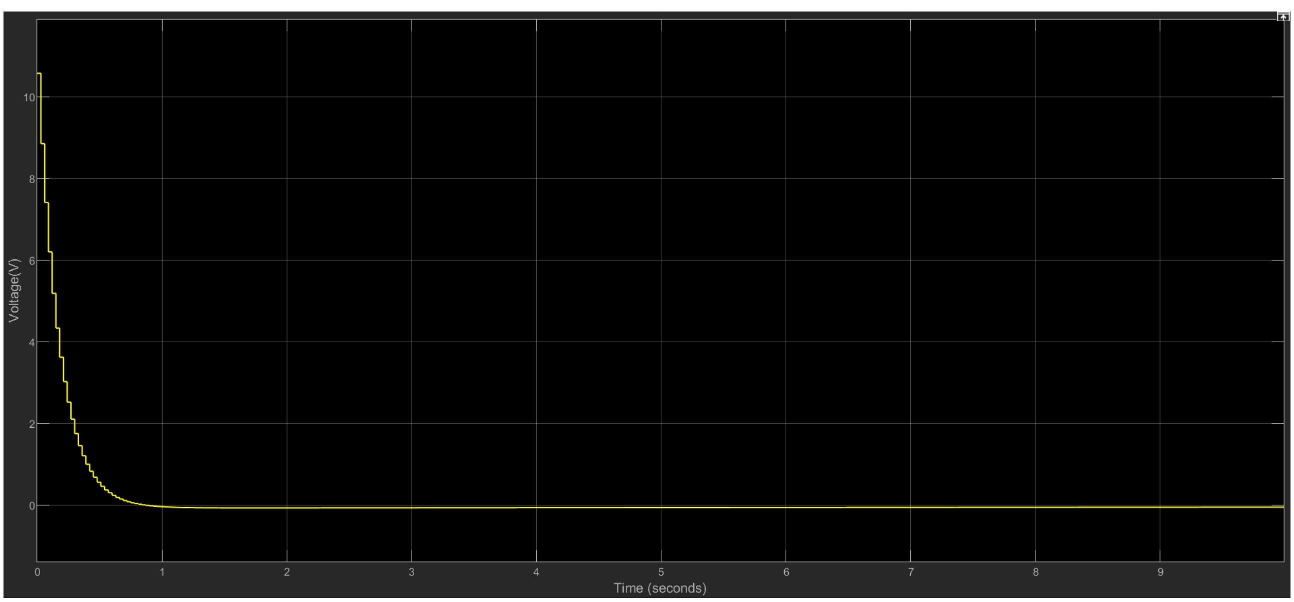

Fig. 4.2: Simulated Input Voltage

4.2 Response from the Experiemnt

Below is the data obtained from the experiment.

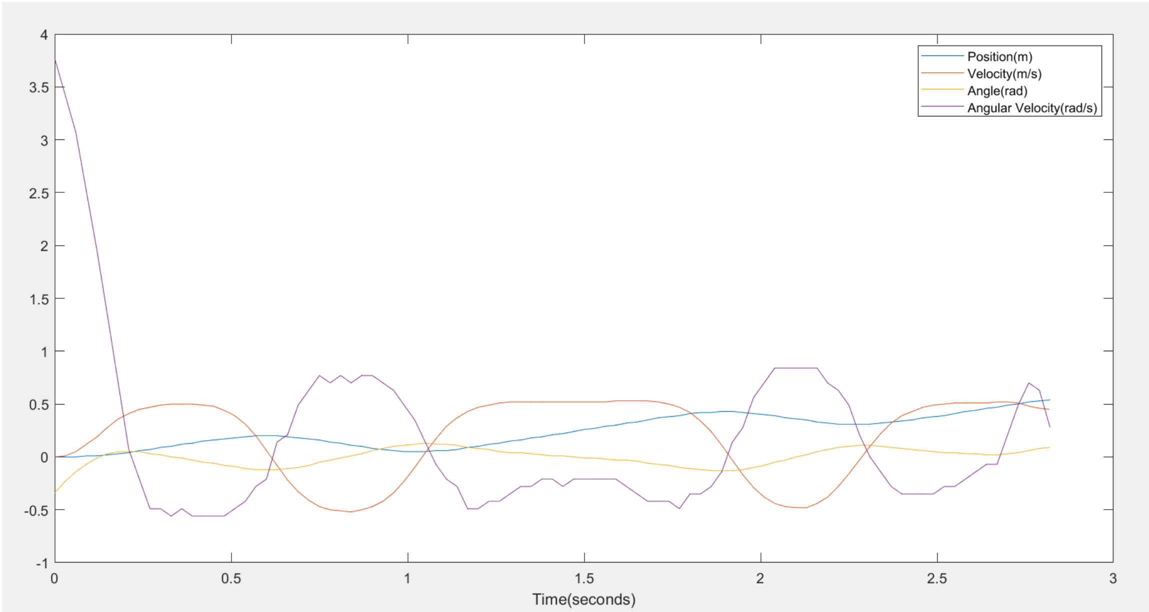

Fig. 4.3: Response of the Physical system

We can see that the system started at the initial condition of $\theta = -0.45$ at $t = 0$, the system gave feedback and θ reached 0 in around 0.2 sec. After that, the system started damping oscillations around zero degrees.

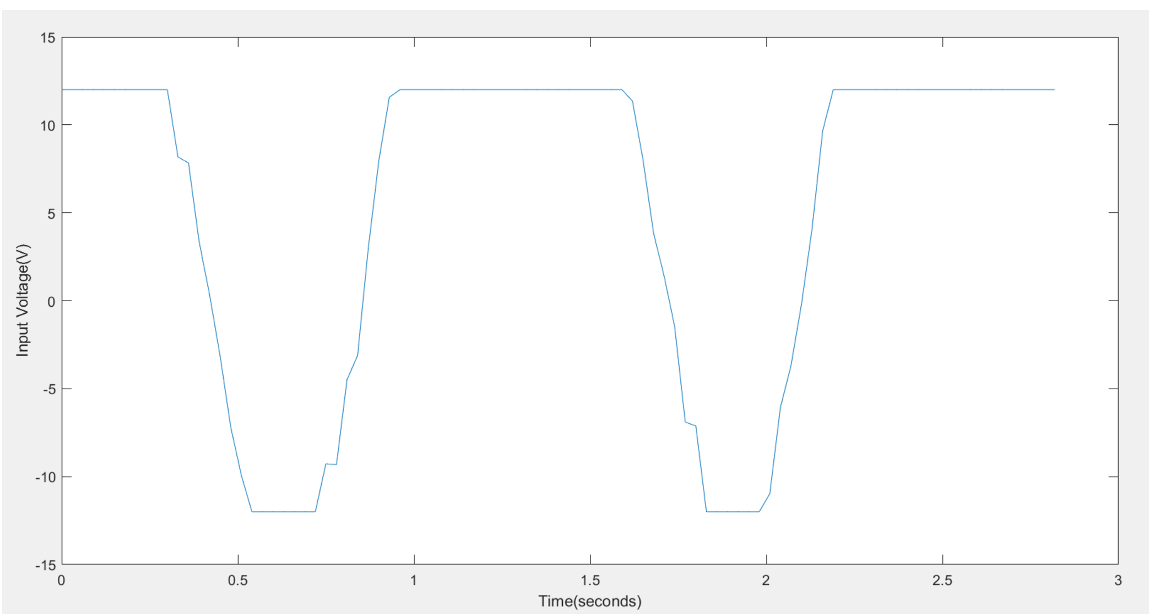

Fig. 4.4: Input Voltage to the System

If we observe carefully we can see that the voltage input given to the motor matches with the change in states and we can see that the voltage is switching between positive and negative values which signifies oscillations.

Chapter 5

Challenges faced and Overcoming them

We initially used Johnson 60 RPM DC motor for the system. We did the primary calculations for the speed of the cart provided by this motor but it was falling short of the required speed. Then we switched to Johnson 300 RPM DC motor but this time it fell short of the required torque. Then we finally switched to RHINO 100 RPM motor which provided us with decent torque and speed.

As we were using LQR for the controller we needed to retain the state space for feedback. To retain the state space we need all the characteristics and transfer function of the DC motor. For this, we took the practical readings of the motor's angular velocity and used this data to find the best fit for all the values on MATLAB.

Initially, we were using accelerometer for the position and velocity measurement of the cart. However, this sensor was causing too much error in the measurements. So we have switched to Hall effect encoder which provided us with better accuracy.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

The designed system showed a good response. It was able to balance the pendulum for around five seconds. After that the slider hit the end of the railing. This was expected as the state x took more time to reach stable state than other states. Moreover, the rail chosen is not long enough and it might be one of the reasons for poor performance as we suspect the system would have been able to reach fully stable state if it had been given more space. Utilization of more precise sensors may help improve the response of the system.

6.2 Future Work

We plan to improve the system if provided with proper support and resources. We will try to make the system with a longer rail so that the slider gets enough spatial freedom to stabilize the pendulum. More advanced sensors will be used to make the system more accurate and robust. We also aim to implement swing-up feature in the future.

References

[1] G. F. Franklin, J. D. Powell, and A. Emami-Naeini. *Feedback Control of Dynamic Systems*. Pearson, 2014.

[2] Haqreu. Chewing the linear quadratic regulator to control the inverted pendulum, 2016. Available at https://habr.com/en/articles/301276/ Accessed on April 12, 2024.

[3] R. Tedrake. Linear quadratic regulators (lqr), 2016. Available at https://underactuated.mit.edu/lqr.html Accessed on April 12, 2024.

Appendices

The video of the running system and the corresponding codes are also available at github.com/AtomicAnind
Pendulum-Balance/tree/main

Appendix A: MATLAB code for analyzing the system

```
%measured quantities in SI units
m=0.133;%pendulum old mass
%m=0.070;%new pendu mass
M=0.545;%cart mass %0.120
l=0.425/2; %pendulum old length is 2l
%l=0.236/2;%new pendulum length is 2l
I=(m*l^2)/3;%pendulum moi about com
b=0.42426;
Im=0.40773;
Kt=386.05;
Ke=0.6392;
Lm=6.257;
Rm=136.51;
r=0.032;
g=9.8;

d=(((M+m)*r^2+Im)*I)+(M*m*(r^2)*(l^2))+(m*Im*(l^2));
%this is D in calculation,
```

```
%determinant of that matrix
a1=(-(I+m*(l^2))*(((Kt*Ke)/Rm)+b))/d;
a2=(-(m^2)*(l^2)*g*(r^2))/d;
a3=((((Kt*Ke)/Rm)+b)*m*l)/d;
a4=(((M+m)*r^2+Im)*m*l*g)/d;
A=[0 1 0 0; 0 a1 a2 0;0 0 0 1;0 a3 a4 0];

B=[0; ((I+m*l^2)*Kt*r)/(d*Rm); 0; (-m*l*r*Kt)/(d*Rm)];

C=[1 0 0 0];

D=0;% this is D of state-space

%% LQR ka parameters
Q=[100 0 0 0; 0 199 0 0; 0 0 100 0; 0 0 0 199];

R=10000;
%R=1;
K=lqr(A,B,Q,R);




A_=A-B*K;
%% Build System
Sys= ss(A,B,C,D);

%% Discretization
Ts=0.03;
Sys_d=c2d(Sys,Ts);
```

```
Ad=Sys_d.a;
Bd=Sys_d.b;
Cd=Sys_d.c;
Dd=Sys_d.d;
N=zeros(length(Bd),1);



%% initial condition
x0=[0; 0; 0.05; 0.1];


[K_d,~,~]=dlqr(Ad,Bd,Q,R,N);
```

Appendix B: Arduino UNO Code

```
float M=0.497,m=0.164,r=0.032;

double t,pret=0,deltaT=30000;
float vol=0;
float V=0,X=0,Xprev=0,theta_p=0,X_tem=0;
float k_1=-0.0838,k_2=-45.3809,k_3=-156.9463,k_4=-26.6846;
float theta=0,theta_prev=0,omega=0,theta_tem=0;
double pi=3.1415926535;
volatile long counter1 = 0,counter2=0;
float th[5],x[5];
#define motor 6
#define in1 4
#define in2 5
```

```
void setup() {
   Serial.begin(9600);
   pinMode(2, INPUT_PULLUP);
   pinMode(3, INPUT_PULLUP);
   pinMode(10, INPUT_PULLUP);
   pinMode(11, INPUT_PULLUP);
   pinMode(in1, INPUT);
   pinMode(in2, INPUT);
   attachInterrupt(0, ai0, RISING);
   attachInterrupt(1, ai1, RISING);
   pinMode(motor,OUTPUT);

}

void loop() {
  t=micros();

theta_tem=((counter1*pi)/300)-pi;
//theta=(counter1*pi)/300;
theta_p=(counter2*2*pi*1.49)/(12*174);
//Serial.println(theta_p);
X_tem=theta_p*r;
//Serial.println(t-pret);

  if(((t-pret)<=(deltaT+10000))&&((t-pret)>=(deltaT-10000)))
  {

 if(pret==0)
```

```
{
  for(int i=0;i<=4;i++)
  {
    x[i]=X_tem;
    th[i]=theta_tem;
  }
}

x[4]=x[3];
x[3]=x[2];
x[2]=x[1];
x[1]=x[0];
x[0]=X_tem;
th[4]=th[3];
th[3]=th[2];
th[2]=th[1];
th[1]=th[0];
th[0]=theta_tem;
theta=(th[0]+th[1]+th[2]+th[3]+th[4])/5;
X=(x[0]+x[1]+x[2]+x[3]+x[4])/5;

  omega=(theta-theta_prev)/(deltaT/1000000);
  theta_prev=theta;
  //X=theta_p*r;
  V=(X-Xprev)/(deltaT/1000000);
  Xprev=X;
  //if((theta<(pi/9)&&theta>(-pi/9))&&((X>-0.2)&&(X<0.2)))
  if(theta<(pi/9)&&theta>(-pi/9))
  {
```

```
  vol=(-(k_1*X+k_2*V+k_3*theta+k_4*omega));
  }
  else
  {
    vol=0;
  }
//vol=(-(k_1*X+k_2*V+k_3*theta+k_4*omega));
  if(vol>=12)
  {
    vol=12;
  }
  else if(vol<=-12)
  {
    vol=-12;
  }
  //analogWrite(motor,((255*vol)/12));
  if(vol>=0){
    digitalWrite(in1,LOW);
    digitalWrite(in2,HIGH);
    analogWrite(motor,((255*vol)/12));
  }
  else
  {
    digitalWrite(in1,HIGH);
    digitalWrite(in2,LOW);
    analogWrite(motor,(-(255*vol)/12));
  }
  Serial.print(X);
  Serial.print(",");
```

```
  Serial.print(V);
  Serial.print(",");
  Serial.print(theta);
  Serial.print(",");
  Serial.print(omega);
  Serial.print(",");
  Serial.println(vol);

  pret=t;
 }



//theta=(counter*pi)/600;

//accel.getEvent(&event);
}

void ai0() {
  if(digitalRead(10)==LOW) {
  counter1++;
  }else{
  counter1--;
  }
  }

  void ai1() {
  if(digitalRead(11)==LOW) {
  counter2--;
```

```
}else{
counter2++;
}
}
```