ROTARY INVERTED PENDULUM USING LQR CONTROLLER

PROJECT REPORT

EKLAVYA MENTORSHIP PROGRAM

AT

SOCIETY OF ROBOTICS AND AUTOMATION,
VEERMATA JIJABAI TECHNOLOGICAL INSTITUTE,
MUMBAI-19

ACKNOWLEDGEMENT

We extend our heartfelt gratitude to our mentors, Mahesh Tupe and Janhavi Deshpande, for their unwavering support, invaluable guidance, and patience throughout the entire project. Their expertise and dedication were instrumental in our journey.

We would also like to express our sincere thanks to all the members of SRA VJTI for their continuous support and encouragement. Their collaborative spirit and the opportunity provided by the Eklavya project have been pivotal in shaping our work and enabling us to pursue our project with dedication and enthusiasm.

Roshan Adal
roshandal2004@gmail.com

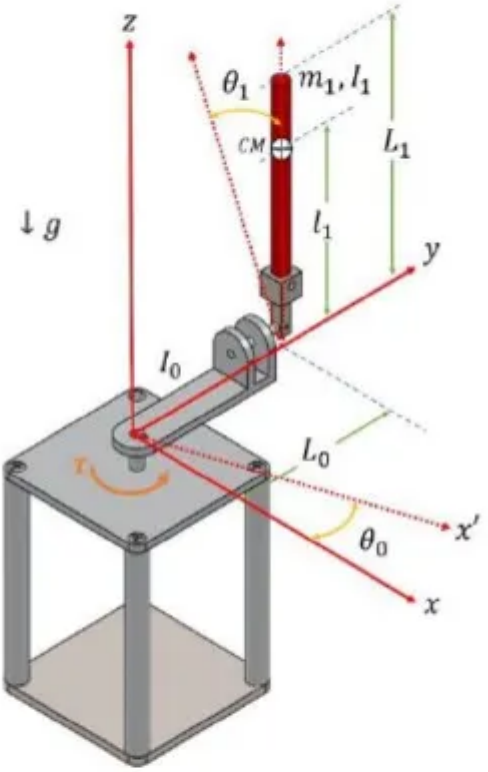
Shankari Anandakrishnan
shankari.ak0208@gmail.com

TABLE OF CONTENTS

INTRODUCTION

GOAL

The aim of this project is implementation of LQR(linear Quadratic Regulator) control and apply it to the problem where we attempt to balance the pendulum in an inverted position mounted on a rotating base in both MATLAB simulation and hardware.



Our primary objective in this project is the implementation of LQR (Linear Quadratic Regulator) control. We will apply this control strategy to address the challenge of maintaining balance for a pendulum mounted on a rotating base, with the pendulum positioned in an inverted state.

Our project will encompass two key aspects:

MATLAB Simulation:
We employed MATLAB to create a simulation environment that mirrors the real-world problem. This simulation will enable us to develop our LQR control strategy before applying it to the hardware setup.

Hardware Application:
We constructed the necessary hardware to replicate the pendulum on a rotating base. Here, we implemented our LQR control to achieve the challenging task of balancing the inverted pendulum.

Through this project, we aim to demonstrate the practicality and effectiveness of LQR control in maintaining stability and balance in a dynamic system, which has applications in various fields, including robotics, control systems, and automation.

CONTROL SYSTEMS

A control system is a system that is used to control the behavior of a device or process. It is made up of three main components: a sensor, a controller, and an actuator. The sensor detects a physical quantity such as position and converts it into an electrical signal. The controller processes this signal and generates an output signal that is used to control the actuator. The actuator is a device that translates the output signal from the controller into a physical action such as turning a motor on or off, or adjusting the speed of a motor.
In our case, we used the **Linear Quadratic Regulator.**

The Linear Quadratic Regulator (LQR) is a method that provides optimally controlled feedback gains to enable the closed-loop stable and high performance design of systems. In LQR, the feedback matrix K is calculated by minimizing the cost function :

$$J = \frac{1}{2}\int_0 \left(x^{\mathrm{T}} Q x + u^{\mathrm{T}} R u\right) \mathrm{d}t$$

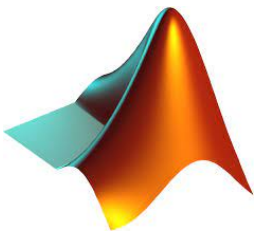
Where Q is a diagonal matrix that refers to the penalty required to be provided when each variable deviates from the intended position and the R matrix imposes a penalty on the input to the system.

MATLAB

MATLAB, a high-performance tool for technical computing, seamlessly combines computation, visualization, and programming in a user-friendly environment. It's aptly named "matrix laboratory" due to its exceptional capabilities in matrix-based computations.

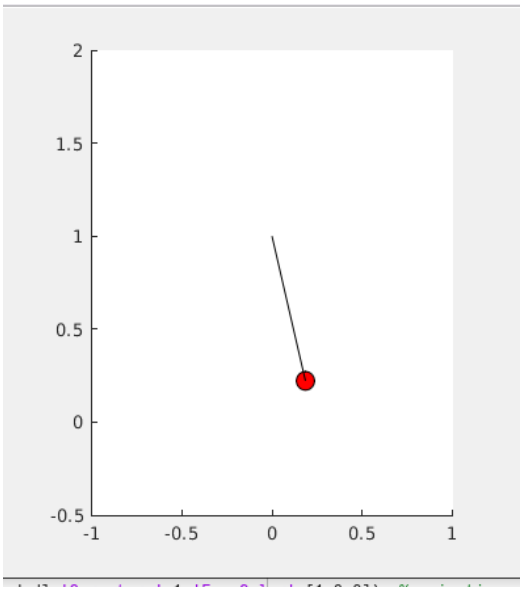
Notably, MATLAB offers specialized toolboxes, collections of functions (M-files), tailored for specific problem domains. These toolboxes extend MATLAB's capabilities, catering to diverse areas like signal processing, control systems, neural networks, and more.

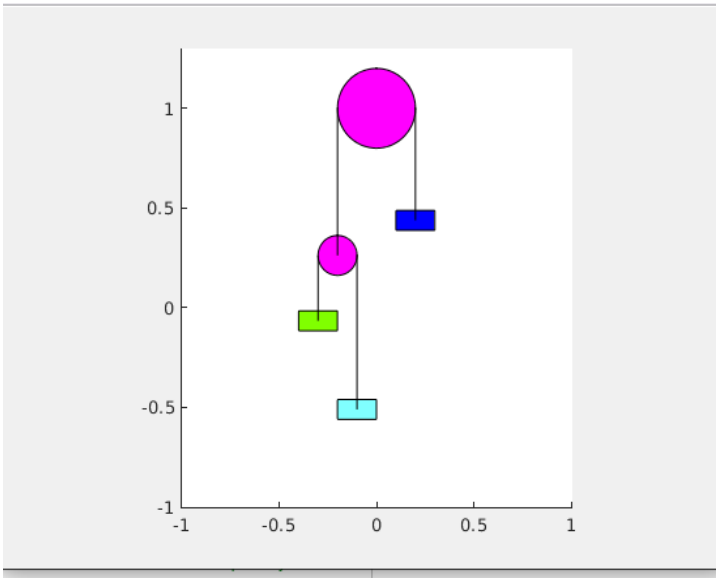
We used MATLAB to model and simulate the system using its various libraries.

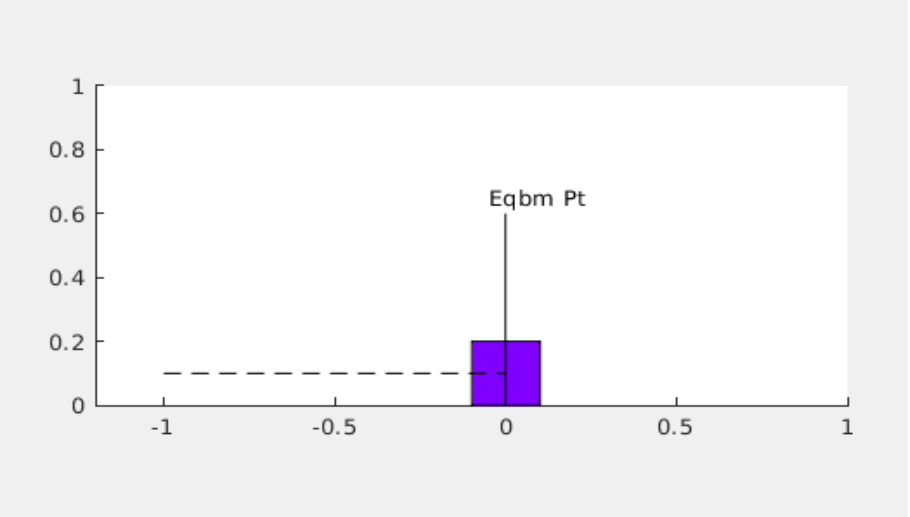


SIMULATION:

To gain familiarity with MATLAB's diverse features, our initial exploration involved creating simulations of physical systems. We started by simulating an ideal pendulum, derived its equations, and constructed a graphical representation of the system using MATLAB. Additionally, we simulated an ideal spring-mass system on a frictionless horizontal surface, visualizing its behavior through graphical output. We even took on the challenge of modeling a complex pulley system.







Building on this foundation, we ventured into our project simulation of an inverted pendulum, incorporating three dimensions. We harnessed MATLAB's extensive toolbox of functions and features to bring this dynamic system to life, enabling us to explore the intricacies of such complex scenarios.

Euler - Lagrangian function :

It states that the state equations of a system can be obtained by solving :

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) - \frac{\partial L}{\partial \theta_1} = 0$$

where L is the Lagrangian function given by the difference of the Kinetic and Potential energies of the system.
On further solving we get the following equations :

$$I_0\ddot{\theta}_0 + {L_0}^2 m_1 \ddot{\theta}_0 + \left\{{l_1}^2 m_1 \left(\ddot{\theta}_0 \sin^2\theta_1 + 2\dot{\theta}_0\dot{\theta}_1 \sin\theta_1 \cos\theta_1)\right\} + \{L_0 l_1 m_1 (\ddot{\theta}_1 \cos\theta_1 - {\dot{\theta}_1}^2 \sin\theta_1\right)\} = \tau$$

$$I_1\ddot{\theta}_1 + {l_1}^2 m_1 \ddot{\theta}_1 + L_0 l_1 m_1 \ddot{\theta}_0 \cos\theta_1 - {l_1}^2 m_1 {\dot{\theta}_0}^2 \sin\theta_1 \cos\theta_1 - m_1 g l_1 \sin\theta_1 = 0$$

The system state is considered as :

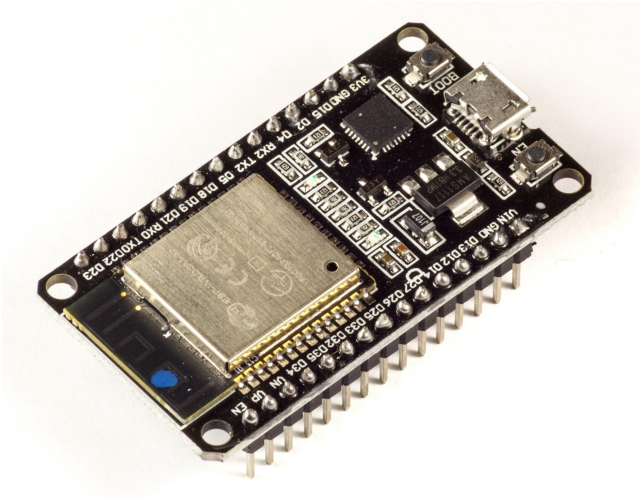
$$\dot{x} = Ax + Bu$$

Where A and B are the jacobians.

HARDWARE IMPLEMENTATION:

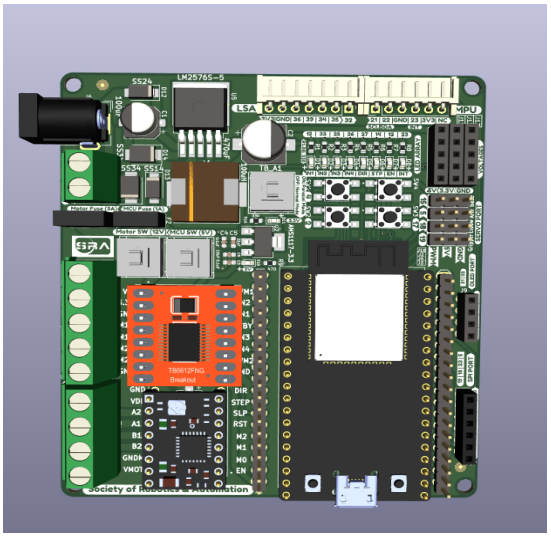
ESP32 microcontroller :

The ESP32 is a versatile System-on-a-Chip (SoC) microcontroller developed by Espressif Systems. It's widely used in applications such as wireless communication, IoT devices, home automation, robotics, and more. Key features include dual-core processors, 448 KB ROM, 520 KB SRAM, various I/O pins, communication interfaces, WiFi, Bluetooth, and LED control. Its flexibility and performance make it a top choice for developers in various applications.

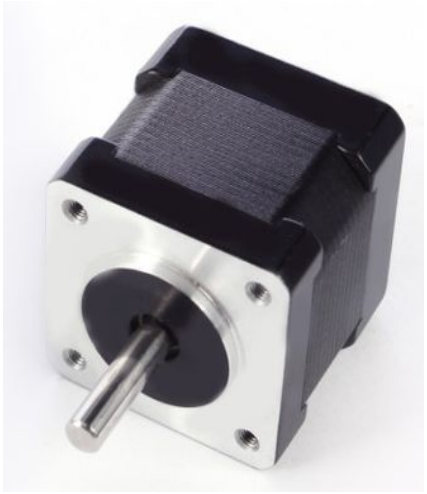


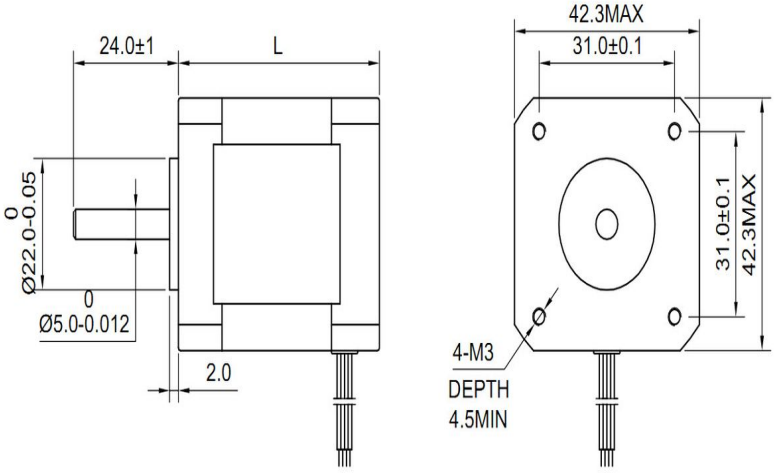
SRA board :

The SRA board is a development board built around the ESP-32 microcontroller. It is designed with various built-in peripherals, including programmable LEDs, switches, sensor ports, and protection circuits for over-current and reverse voltage. Additionally, the board features motor drivers, enhancing its functionality and versatility for a wide range of projects and applications.



NEMA-17 Stepper motor:

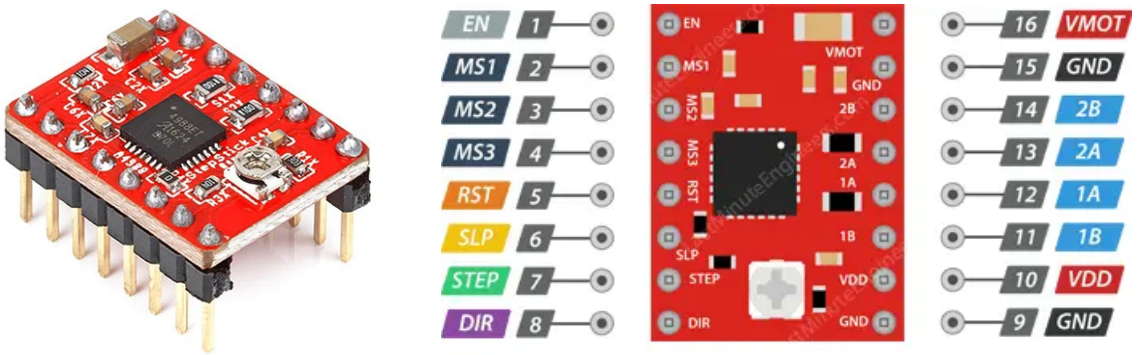




A stepper motor is an electric motor that divides a full rotation into equally spaced steps, offering precise control of position and rotation. They move in discrete steps, each representing a fixed angle of rotation, and typically operate in an open-loop control system without position feedback. Stepper motors provide substantial holding torque at low speeds, making them ideal for applications requiring position holding without external

locking mechanisms. Common uses include 3D printers, CNC machines, plotters, disk drives, camera lenses, and more.

A4988 Motor Driver ;



The A4988 stepper motor driver is a module used to control stepper motors in applications like 3D printers and CNC machines. It offers microstepping for smoother movements, current control for motor tuning, and protection features. It's compatible with various microcontrollers, has simple wiring, and is widely used in DIY projects.

AUTONICS Rotary Encoder:

A quadrature encoder is an incremental encoder with two out-of-phase output channels used in many general automation applications where sensing the direction of movement is

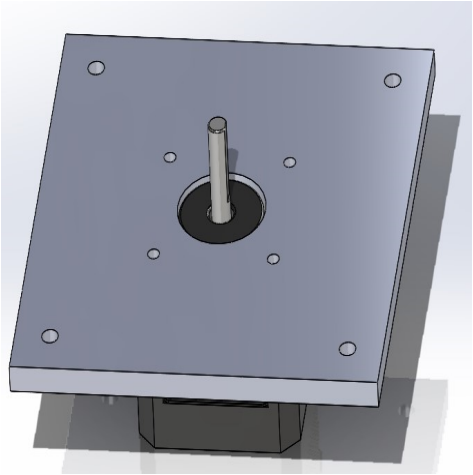
required. Each channel provides a specific number of equally spaced pulses per revolution (PPR), and the direction of motion is detected by the phase relationship of one channel leading or trailing the other channel.

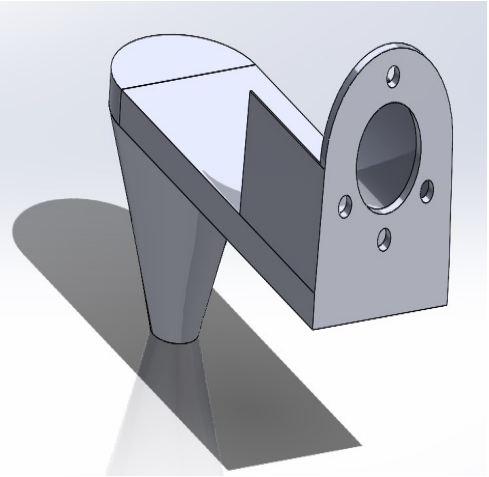
SOLIDWORKS DESIGN:

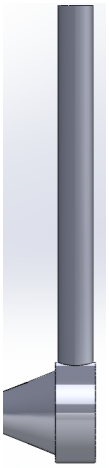
For our system, which primarily revolves around rotational components, we opted for a standard design approach. The initial focus was on creating a top plate that would seamlessly accommodate the NEMA-17 motor. We crafted the top plate with careful consideration to ensure it would be compatible with this motor model.

In addition to the top plate, our design incorporated a horizontal bar that rests on the motor's shaft. This component was meticulously designed,as it interacts with the motor. At one end of the bar, a horn was incorporated,with a slot in its base to insert the shaft of the motor serving as a pivotal point, and on the other end, we designed a case to house the rotary encoder. The encoder's case was carefully crafted to ensure a perfect fit, with its face intact with the surrounding cover.

In our design, we also accounted for the pendulum, creating a slot specifically designed to accommodate the shaft of the encoder, ensuring a seamless integration of all components in the system.







`

SETTING UP THE STEPPER MOTOR FOR THE REQUIRED FREQUENCY:

For the motor to run in such a way that it rotates the horizontal rod in a way that the pendulum stabilizes, we need to provide the necessary frequency. The frequency change is calculated based on the torque required to be provided to the system. The torque required is u given by u = - kx.

Here, x denotes the state variables of the system.

To calculate the frequency change, we are using PID control by considering the input u required to be the error. The values of Kp, Ki, Kd are calculated by trial and error.

The LEDC peripheral of the ESP32 can be used to set the required PWM. It has to be set by using channel configuration, timer configuration and changing the PWM accordingly.

The duty cycle is kept constant but frequency is varied based on the calculations.

We also set the direction of movement by setting the GPIO pin HIGH and LOW.

Hence, the stepper motor moves clockwise or anticlockwise based on the direction required to be moved with a frequency stabilizing the pendulum.

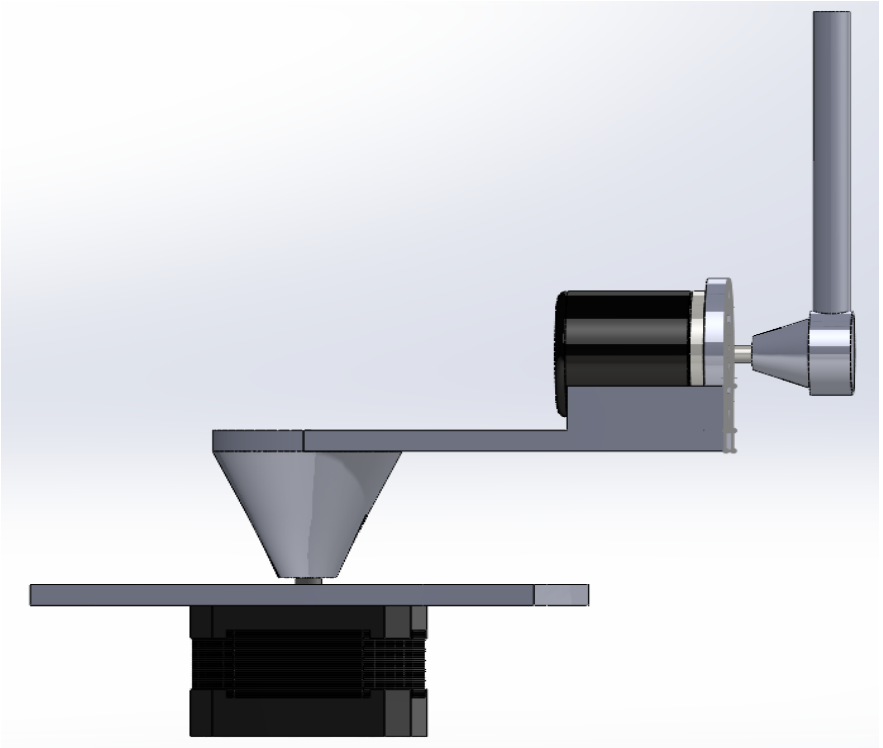
ASSEMBLY

Our assembly process began with a laser-cut base plate, featuring slots for securing screws with spacers in between both the plates. This base plate was a crucial component, serving as the foundation for our system. The top plate is secured in place using four screws.

Next, our attention turned to the NEMA-17 motor. With careful precision, we affixed this motor to the top plate, utilizing screws to ensure a secure connection. The motor's shaft protruded through the top plate.

The horizontal bar housing the rotary encoder was the next component to be integrated into our design. We firmly attached the rotary encoder to its dedicated case, using screws for a stable connection. The encoder's shaft was then inserted into the specially designed slot within the pendulum,

With the key components in place, our attention shifted to the assembly of the horizontal plate. This plate was lowered onto the upright shaft. The slot on the horizontal plate aligned perfectly with the shaft, guaranteeing a snug fit and ensuring that our system was ready for successful operation.



CHALLENGES ENCOUNTERED:

ISSUES IN SOFTWARE SIMULATION:

We commenced with our inverted pendulum project with a MATLAB simulation, specifically, the software animation of a 3D inverted pendulum model. A crucial aspect of this simulation was the necessity to establish the system's state equations. These equations were derived using the Lagrangian function, a process that proved to be quite challenging due to the complexity involving numerous terms and factors.

ISSUES IN RUNNING THE MOTOR:

When running the code to control the motor speed, we encountered difficulties marked by significant frequency variations. These fluctuations posed a challenge in effectively managing the motor's speed and the overall system stability.

To address this issue, we added extra weight to the 3D-printed pendulum rod, which initially was quite light. The additional weight aided in achieving a more balanced and stable configuration, helping to minimize the challenges associated with motor control and speed variations.

PROBLEMS IN TUNING:

In our implementation of PID control for frequency adjustment, tuning the PID parameters is essential for system stabilization. To begin this process, we initially ran the motor to determine its direction of rotation, as indicated by the encoder readings.

Our next step involved achieving the desired frequency by adjusting the motor's speed intensity, based on the calculated torque. The values of the PID parameters, Kp, Ki, and Kd, are determined through iterative trial and error in subsequent runs, ensuring the system behaves as intended.

FINAL ASSEMBLY :

CONCLUSION:

We achieved several milestones in our project:

1. Understanding LQR: We gained a deep understanding of how Linear Quadratic Regulator (LQR) works and its applications in control systems.

2. SolidWorks Experience: We honed our 3D modeling and design skills through hands-on experience with SolidWorks.

3. 3D Printing: Our project allowed us to explore 3D printing technology, which helped us in providing practical understanding of how it can be used to create complex objects with ease.

4. Embedded C Proficiency:We gained valuable experience in embedded C programming and explored the usage of LEDC code for precise control.

RESULT:

Despite our accomplishments, the complex nature of the variables presented challenges, and regrettably, we did not achieve our primary objective of stabilizing the inverted pendulum.

Nonetheless, this project has enriched us with valuable knowledge and skills that will undoubtedly prove beneficial in our future endeavors. Each hurdle we encountered has

contributed to our growth and understanding, enhancing our capacity to tackle complex projects more effectively.

It's worth emphasizing the significance of our system, which has broad applications in the vertical stabilization of dynamic systems. These systems are integral to the control and stability of various applications, including Segways, rocket launches, and even human posture.

BIBLIOGRAPHY:

- Linear Algebra
- Control Bootcamp
- State Equations Reference
- ESP-IDF Framework
- LED-C PIN Documentation