

# PID Control with DC Motor

By Team 20: Hardware Handlers:

Team Members:

Adyansh Kakran

Pranjali Bishnoi

Mohammad Zaid

Roja Sahoo

Guided By : Harikumar Kandath

# Motivation

1. To create a remote learning environment for learners which allows them to login remotely and perform experiments on the hardware.
2. Our project aims to create short user sessions of 15 minutes , in which a user can perform experiments remotely and learn about controlled system
3. To provide real time response for the experiment that the user wants to perform through graphs and live video.
4. PID algorithm has steady state error nearly zero to accomplish the designed criteria of the control system. Therefore, the algorithm works perfectly to get the desired angular position of DC motor.

# Our Implementation

Our Implementation involves the following components-

1. Hardware Setup (DC Motor+Encoder)
2. Arduino Code
3. Remote Labs Dashboard - PID Experiment Dashboard
4. User Credentials

Our user input on the dashboard includes ,  $K_p$ ,  $K_i$ ,  $K_d$  and required angle. (4 parameters)

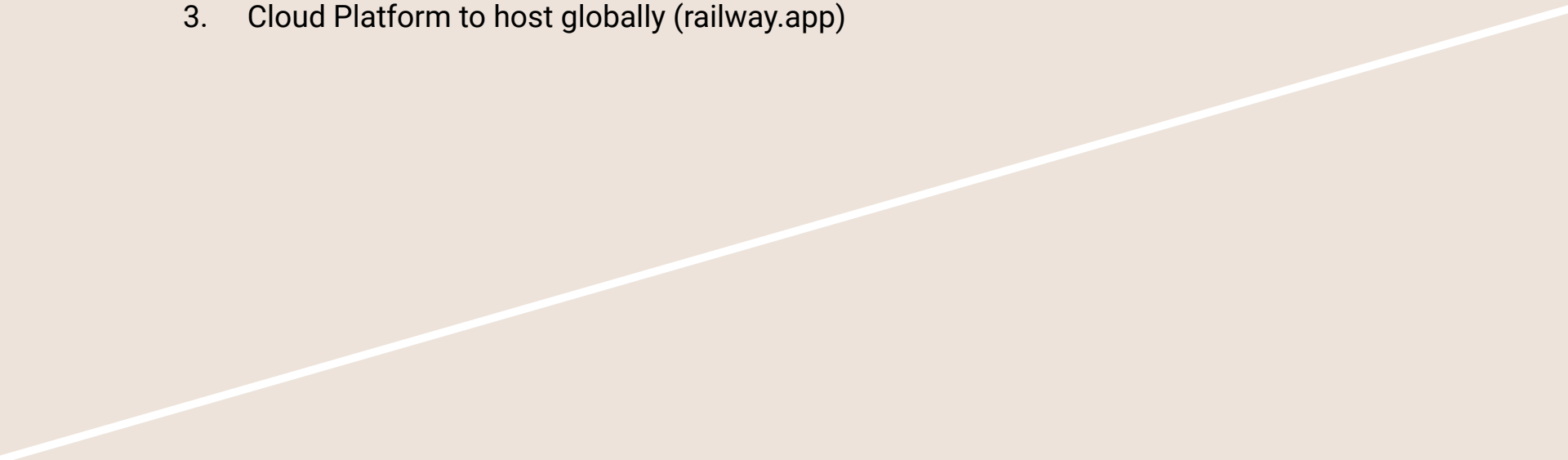
# Hardware Description

1. The motor encoder feeds the direction signal to the motor driver circuit.
2. The angle relevant data will be converted to a signal, then it will be fed to the PID controller.
3. We will adjust the system PID parameters until we get a stable response then the output of the PID should be a signal that maintains the motor angle.
4. Now we have the appropriate direction signal from the direction detection circuit, which should be fed to the motor driver to correct the motor position.

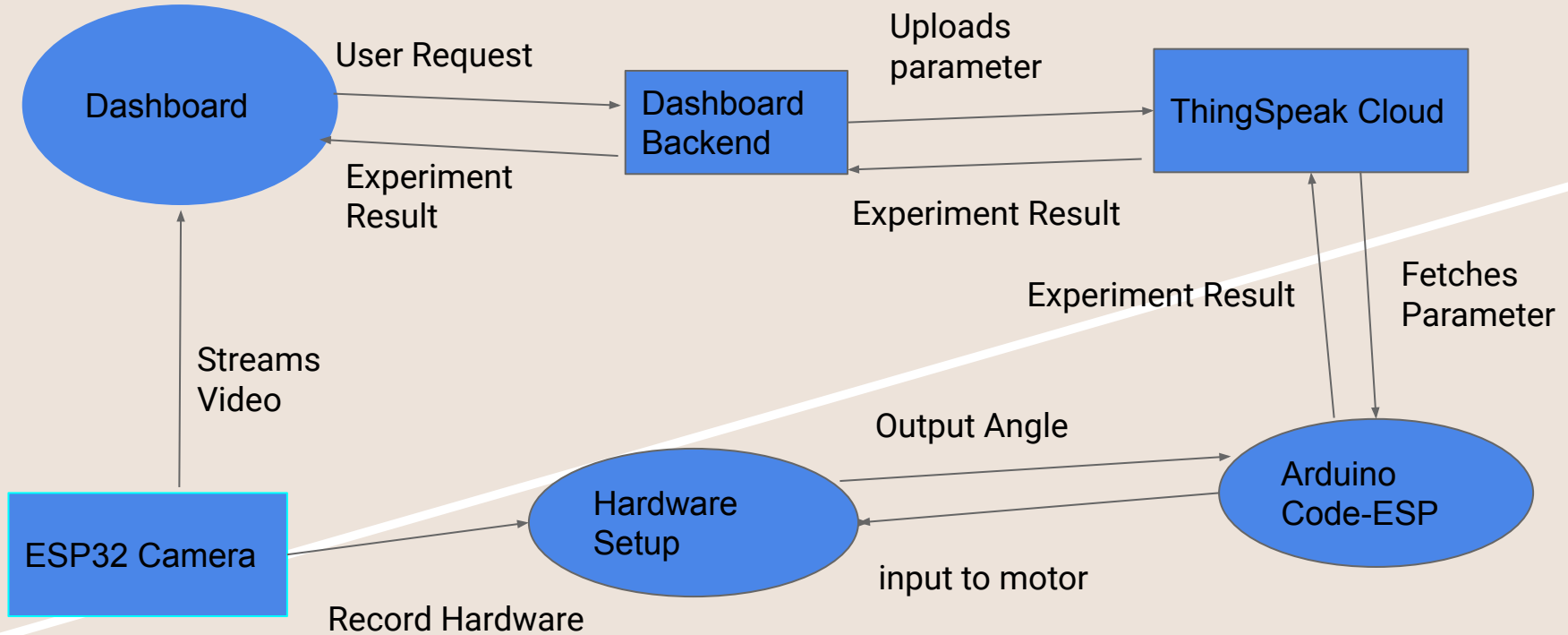
# Software Description

1. We made our Dashboard using NodeJs for backend and HTML,CSS and Javascript for frontend.
2. New User first need to register on our website first. We are using Redis as our User Database
3. Once the user is logged in, the experiment page is available, user can feed  $k_p$ ,  $k_i$ ,  $k_d$  and required angle to interact with the hardware.
4. The parameters ( $k_p$ ,  $k_i$ ,  $k_d$ ) are uploaded to thingspeak from our backend.
5. After the parameters are uploaded to thingspeak, ESP32 fetches the parameter, and uploads the resultant angle vs time data on thingspeak.
6. The results of experiment, which are already uploaded on thingspeak now are shown in the form of graph using charts.js library by our Dashboard.

# Tech Stack

1. Node JS,Express JS and Redis for Backend of Dashboard
  2. HTML,CSS and Javascript for Frontend of Dashboard
  3. Cloud Platform to host globally (railway.app)
- 

## Data Flow in the System



# Data Analysis

1. The aim is to optimize the time taken by the DC motor to set the pointer to the angle given as input i.e minimize the deviation from the target angle as much as possible. We are analysing the error and the variation of angle with respect to time
2. We are making graphs of change of angle with respect to time,fetching data from ThingSpeak and making graphs using javascript and showing it to our dashboard
3. We are also streaming the hardware response using esp32 webcam



# Challenges Faced -

## Hardware-

1. We came across an error in connecting the circuit to common ground.
2. The error calculation i.e measuring the deviation of the shaft of the DC motor from the target angle.

## Software

1. We faced issue in writing different APIs to take care of all the routes for our Dashboard and to make the login flow as smooth as possible.  
Once the user is logged in ,no need to login again for another request.  
We used browser cookies for ensuring logged in user is saved.
2. Since ThingSpeak has an upload limit of 1 second,so we had to store all the updates in a buffer and update all at once in an interval.We are using bulk update api provided by thingspeak to take care of this.

# Summary

1. Our project after completion will help students all over the globe to study the advantages of using the PID controller to control any device using feedback control laws.
2. It can make an impact in remote communities which are resource deprived and do not have access to such complicated hardware/software.