**Load-Driven Motor Speed Control of a Conveyor Using PID Controller and Sorting the Items based on their attribute**

*A project report for the disciplinary project*



#### Pandit Dwarka Prasad Mishra, Indian Institute of Information Technology, Design and Manufacturing, Jabalpur

*AN INSTITUTE OF NATIONAL IMPORTANCE ESTABLISHED BY THE MINISTRY OF HOME AND RESOURCE MANAGEMENT*

Submitted by:

1. ***Aditya Naryan Jaiswal 22BME002***
2. ***Yash Pardhi 22BSM069***
3. ***Shubham Kumar 22BSM056***
4. ***Aryan 22BSM010***
5. ***Ankit Gopi Gahnoliya 22BSM019***

UNDER THE GUIDANCE OF

***PROF. SUNIL AGRAWAL***

***DEPARTMENT OF MECHANICAL ENGINEERING***

## Acknowledgment

We would like to express our deep gratitude towards **Prof. Sunil Agrawal,** Department of Mechanical Engineering , Pandit Dwarka Prasad Mishra Indian Institute of Information Technology, Design and Manufacturing, Jabalpur, for their guidance with unsurpassed knowledge and immense encouragement.

We would like to thank **Dr. Zahid Ansari, Head of Department, Department of Mechanical Engineering, PDPM IIITDMJ**, for giving us this opportunity to work on this project which helped us develop our skills in the field of Control system, industrial automation and fabrication skills.

We would like to thank all the teaching and non-teaching staff of the Department of Mechanical Engineering, PDPM IIITDMJ, for their immense support and guidance through reviews of the system.

Lastly, we would like to thank our parents, friends, and classmates for their immense support and encouragement during our project. We would like to thank everyone who supported us directly or indirectly during this time.

# TABLE OF CONTENTS

|  |  |
| --- | --- |
| 1) **Abstract** | 1 |
| 2)**Introduction** | 2 |
| 2.1) Closed Loop PID Control | 3 |
| 2.2) Sorting Based on Color Attribute and working of TCS3200 | 3,4 |
| 2.3) Motivation  2.4) Problem statement |  |
|  |  |
| 3) **Keywords and Definitions** |  |
| 3.1) PID controller | 5,6 |
| 3.2) Proportional, Integral and Derivative | 7 |
| 3.3) Setpoint  3.4) Pulse width Modulation | 7 |
| 4) **Methodology** | 8 |
| 4.1) System Design | 8 |
| 4.2) Component Integration | 8 |
| 4.3) Testing and calibration  4.4) Optimization  4.5) Deployment | 9 |
| 5) **Training and Testing** |  |
| 5.1) Training | 9 |
| 5.2) Testing | 10 |
| **Challenges** | 11 |
| **Key Leanings** | 11 |
| **Results** | 12 |
| **Future Scope** | 13 |

**ABSTRACT**

In modern manufacturing and logistics, optimizing conveyor systems for efficiency and precision is crucial. This study presents a load-driven motor speed control system for conveyor belts, **utilizing a Proportional-Integral-Derivative (PID) controller to dynamically adjust motor speed based on real-time load conditions**. **The PID controller ensures stable and responsive speed adjustments**, enhancing the conveyor's ability to handle varying load demands.

Additionally, **the system integrates an item sorting mechanism that categorizes items based on their attributes, such as size, weight, or type**. By combining advanced motor control with intelligent sorting, the proposed approach improves operational efficiency, reduces wear and tear on equipment, and enhances the overall throughput of conveyor systems in automated environments.

***1***

## Introduction

In the fast-paced world of modern manufacturing and logistics, conveyor systems are vital for efficiently transporting and processing items. These features are essential for enhancing the system's efficiency and accuracy.

To address the challenge of varying load conditions, the system employs a load-driven motor speed control mechanism. This is achieved through a **Proportional-Integral-Derivative (PID) controller**, which continuously adjusts the motor speed based on real-time feedback. The **PID controller, managed by an Arduino microcontroller, operates in a closed-loop configuration**. It uses three components to maintain optimal performance: **the Proportional (P) component adjusts speed in proportion to the difference between desired and actual speeds; the Integral (I) component corrects long-term errors by considering accumulated past discrepancies; and the Derivative (D) component predicts and mitigates future errors by analyzing the rate of change in load**. This dynamic adjustment ensures stable conveyor operation, minimizes wear and tear, and improves overall performance.

In addition to managing speed, the system incorporates a color-based sorting mechanism to enhance item processing. **The TCS 3200 color sensor is used to detect and differentiate items based on their color**. This intelligent sorting capability ensures that items are accurately categorized and routed, thereby **streamlining subsequent operations and reducing the need for manual intervention.**

Together, these advancements in load-driven speed control and color-based sorting optimize the c**onveyor system’s throughput, enhance operational efficiency, and reduce operational costs,** making it a more effective solution for managing diverse and high-volume material handling tasks.

***2***

#### Closed Loop PID Control

A PID controller is a control system that helps keep the conveyor running smoothly by adjusting the motor speed based on real-time feedback. The PID controller has three main parts:

* **Proportional (P)**: Adjusts motor speed according to the difference between the desired and actual load.
* **Integral (I)**: Corrects any long-term errors by considering past discrepancies.
* **Derivative (D)**: Predicts future errors based on how quickly the load is changing, helping to avoid sudden changes.

Using an Arduino, the PID controller ensures that the conveyor speed is adjusted precisely to handle varying loads, making the system more stable and efficient.

#### Sorting Based on Color Attribute and working of TCS3200

Sorting items by color is essential for organizing and processing them correctly. The TCS 3200 color sensor detects the color of each item as it moves along the conveyor. This sensor converts color information into data that Arduino can read.

The system then uses this data to sort items into different bins or paths based on their color. A servo motor, also controlled by the Arduino, helps direct the items to the right places. This makes sorting faster and more accurate.

**3**

**Working –**

The TCS3200 has an array of photodiodes with 4 different filters. A photodiode is simply a semiconductor device that converts light into current. The sensor has:

* 16 photodiodes with red filter – sensitive to red wavelength
* 16 photodiodes with green filter – sensitive to green wavelength
* 16 photodiodes with blue filter – sensitive to blue wavelength
* 16 photodiodes without filter



By selectively choosing the photodiode filter’s readings, you’re able to detect the intensity of the different colors. The sensor has a current-to-frequency converter that converts the photodiodes’ readings into a square wave with a frequency that is proportional to the light intensity of the chosen color. This frequency is then, read by the Arduino – this is shown in the figure below.

A diagram of a computer component

Description automatically generated

***4***

#### MOTIVATION

The logistics sector, particularly e-commerce giants like Amazon, faces significant challenges related to the efficient handling and processing of return items in their warehouses. The sheer volume of returns often leads to operational inefficiencies, increased costs, and delays in inventory management. Addressing these challenges requires innovative solutions to streamline material handling and reduce processing times. The motivation for developing a conveyor system with load-driven motor speed control using a PID controller, coupled with attribute-based sorting, stems from the need to mitigate these issues. By dynamically adjusting motor speeds to handle varying loads and accurately sorting items based on their attributes, this system aims to enhance throughput, minimize sorting errors, and reduce the time and costs associated with returns. Implementing such advanced technologies can significantly improve operational efficiency in warehouses, ultimately leading to better customer satisfaction and reduced financial losses for logistics companies.

Top of Form

Bottom of Form

#### PROBLEM STATEMENT

**Optimizing Material handling of conveyor belts to enhance the conveyor efficiency under load variation and reduce manual intervention.**

***5***

## Keywords and Definitions

* 1. **INTRODUCTION**

The domain analysis that we have done for the project mainly involved understanding the neural networks

1. **PID Controller**: A feedback control loop mechanism widely used in industrial control systems. PID stands for Proportional-Integral-Derivative, which are the three control actions used to maintain the desired output.

2.1 **Proportional Control (P)**: A component of the PID controller that produces an output that is directly proportional to the error (the difference between the desired setpoint and the actual process variable). The proportional gain determines how much correction is applied based on the current error.

2.2 **Integral Control (I)**: A component of the PID controller that addresses accumulated past errors by integrating the error over time. This helps eliminate steady-state error and ensures that the system reaches the setpoint.

2.3 **Derivative Control (D)**: A component of the PID controller that predicts future error based on its rate of change. It provides damping to the system by considering the rate at which the error is changing, which helps to reduce overshoot and oscillations.

3. **Setpoint**: The desired target value or position that the system is trying to achieve and maintain.

***6***

**4. Pulse Width Modulation (PWM)**: A technique used to encode the amplitude of a signal by varying the width of the pulses in a pulse train. It is commonly used for controlling the power delivered to electronic devices, such as motors and LEDs.

**5. RGB values** - RGB means Red Green Blue, i.e. the primary colors in additive color synthesis. A RGB file consists in composite layers of Red, Green and Blue, each being coded on 256 levels from 0 to 255. For example, black corresponds to the levels R=0, G=0, B=0, and white corresponds to the levels R=255, G=255, B=255.

***7***

#### Methodology

**FOR PID CONTROLLER -**

**1. Hardware Setup**

* **Components:**
  + **DC Motor**: The motor whose speed will be controlled.
  + **Encoder**: A rotary encoder attached to the motor to provide feedback on position and speed.
  + **Motor Driver**: Interface between the DC motor and the microcontroller, enabling control over the motor's direction and speed.
  + **Arduino Microcontroller**: Used to implement the control algorithm.
* **Pin Configuration:**
  + Encoder signals are connected to the microcontroller’s digital pins: ENCA (pin 2) and ENCB (pin 3).
  + Motor control signals are connected via PWM (pin 5), IN1 (pin 6), and IN2 (pin 7) pins to control motor speed and direction.

**2. Software Implementation**

* **Encoder Feedback:**
  + An interrupt service routine (ISR) is triggered by the rising edge of the encoder’s ENCA channel. This ISR reads the value of ENCB to determine the rotation direction and updates the motor's position and velocity
  + The use of no Interrupts () and interrupts() functions ensures safe access to shared variables by preventing interruptions during critical sections of the code.

8

* **Velocity Calculation:**
  + Motor velocity is computed using two methods:
    1. **Method 1:** Velocity is calculated by measuring the time difference between readings and the change in position, then converting the result from counts per second to RPM.
    2. **Method 2:** Velocity is calculated directly within the ISR based on the elapsed time between encoder pulses.
* **Filtering:**
  + Velocity readings are processed through a low-pass filter to reduce noise and provide a more stable input for the PID controller.
* **PID Control:**
  + A PID controller regulates the motor’s speed to a set target value. The controller calculates the difference between the target speed and the filtered speed (error) and adjusts the motor control signal (u) accordingly.
  + The controller parameters include:
    1. **Proportional Gain (kp)**: Determines the controller's response to the current error.
    2. **Integral Gain (ki)**: Accumulates past errors to address steady-state error.
    3. **Derivative Gain (kd)**: Reacts to the rate of change of the error, though it is set to zero in this implementation.
* **Motor Control:**
  + The direction and speed of the motor are controlled based on the PID output. The direction is determined by the sign of the control signal, while the speed is controlled by adjusting the PWM duty cycle.

9

* + Motor control signals are sent to the motor driver, which adjusts the motor according to the calculated speed and direction.

**3. System Integration and Testing**

* The system is tested by setting various target speeds and observing the motor’s response to these settings. The performance of the PID controller is analysed, and adjustments to the controller gains are made to optimize the system's behaviour.
* **Data Logging:**
  + The system outputs data such as target speed, actual speed, and control signal values via the serial port for further analysis.

**4. Conclusion**

This methodology describes the step-by-step approach used to design and implement a motor speed control system using a PID controller. Key components, software logic, and testing procedures are detailed to provide a comprehensive understanding of the system's operation and performance.

1. **Optimization**:
   * **Performance Evaluation**: Assess the system’s performance in terms of speed control, sorting accuracy, and throughput. Make necessary adjustments to improve efficiency and reliability.
2. **Deployment**:
   * **Implementation**: Deploy the system in a controlled environment or pilot warehouse setting. Monitor its performance and make final adjustments to ensure it meets operational requirements.

This methodology ensures a systematic approach to developing and implementing an efficient conveyor system with advanced control and sorting features.

10

**FOR COLOR SENSOR SYSTEM**

**1. Hardware Setup:**

* **Components Used:**
  + GY-31 TCS3200 TCS230 colour sensor module
  + Servos (3 units) for red, green, and blue colour indications
  + Arduino microcontroller
  + Connecting wires
  + Breadboard
* **Pin Configuration:**
  + The colour sensor module is connected to the Arduino with specific pins:
    - S0 to pin 8
    - S1 to pin 7
    - S2 to pin 6
    - S3 to pin 5
    - Sensor output to pin 4
  + The servos are connected to pins 10, 11, and 12 of the Arduino, corresponding to red, green, and blue colour detection respectively.

**2. Software Implementation:**

* **Arduino IDE:**
  + The code is written and uploaded using the Arduino IDE.
  + The code initializes the required pins and configures the colour sensor module to measure the pulse width of reflected light corresponding to red, green, and blue colours.
* **Calibration:**
  + The colour sensor is first calibrated to obtain the minimum and maximum pulse width values for red, green, and blue colours.
  + These values are then mapped to a range of 0-255, which represents the intensity of each colour.
* **Colour Detection:**
  + The sensor measures the pulse width for each colour by selecting the appropriate photodiode set using the S2 and S3 pins.
  + The pulse width is then converted to a colour value using the map() function.

11

* **Servo Control:**
  + Based on the detected colour values, the corresponding servo is activated.
  + For instance, if Red is the dominant colour, the Red servo moves to a specific position and then returns to its initial position after a delay.
  + Similar logic is applied for blue and green colours.

**3. Data Processing:**

* The code continuously reads the colour values from the sensor and processes them to identify the colour.
* If the detected colour meets the predefined criteria, the corresponding servo is triggered to perform a predefined action.

**4. Serial Communication:**

* The detected colour values and the identified colour are printed to the Serial Monitor for debugging and verification purposes.

**5. Testing and Validation:**

* The system is tested with various colour samples to ensure accurate detection and servo response.
* Adjustments to the calibration values and servo movements are made based on the results.

**6. Final Integration:**

* The system is integrated into a larger setup where the colour detection triggers other actions or processes.
* The entire process from colour detection to servo activation is optimized for real-time operation.

## Training and Testing

**1. Introduction**

In this project, we focused on two key components: a color sensor and a PID (Proportional-Integral-Derivative) controller for DC motor speed control. The goal was to calibrate the color sensor for accurate color detection and to tune the PID controller to achieve optimal speed control of a DC motor.

12

1. **Color Sensor Calibration and Testing**

**2.1 Calibration Process**

* **Initial Setup**: The color sensor was connected to the microcontroller, and the system was set up to read raw data from the sensor.
* **White and Black Calibration**: Calibration was performed using known reference colors:
  + **White Calibration**: The sensor was pointed at a white surface to establish the baseline for detecting the maximum reflectance and to set the reference values for the RGB channels.
  + **Black Calibration**: The sensor was then calibrated with a black surface to determine the minimum reflectance values and establish the baseline for zero reflectance.

**2.2 Color Detection**

* **Testing Procedure**: After calibration, the color sensor was tested with various color samples (e.g., red, green, blue, yellow, etc.).
* **Data Collection**: The RGB values corresponding to each color were recorded and compared to the expected values.

**Accuracy Assessment**: The detected colors were assessed for accuracy by comparing the sensor's output with known color values. Deviations were noted and adjustments were made to improve accuracy.

**2.3 Results**

* **Performance Analysis**: The sensor demonstrated accurate color detection for the calibrated colors (white and black) and for a range of other colors. Minor adjustments were made to enhance detection accuracy based on the results.

**3. PID DC Motor Speed Control**

**3.1 PID Controller Tuning**

13

* **Initial Setup**: The PID controller was implemented to regulate the speed of a DC motor. Initial values for Kp (Proportional gain), Ki (Integral gain), and Kd (Derivative gain) were set based on standard recommendations.
* **Tuning Process**:
  + **Proportional Gain (Kp)**: Adjusted to determine how much correction is applied based on the current error. Increasing Kp generally improves response time but can lead to overshooting.
  + **Integral Gain (Ki)**: Fine-tuned to eliminate steady-state errors. Adjustments were made to balance the integral response without causing excessive overshoot or oscillations.
  + **Derivative Gain (Kd)**: Adjusted to predict future error and provide damping. The right balance of Kd helped in reducing oscillations and improving system stability.

**3.2 Testing Procedure**

* **Speed Setpoint Testing**: The PID controller was tested with various speed setpoints to observe how well it maintained the desired speed.
* **Response Analysis**: The system's response to changes in setpoint, disturbances, and noise was analyzed. The controller's ability to maintain stable speed with minimal overshoot and settling time was evaluated.
* **Optimization**: Based on the test results, further adjustments were made to Kp, Ki, and Kd to achieve the best performance. This included reducing overshoot, improving settling time, and ensuring system stability.

**3.3 Results**

* **Performance Evaluation**: The PID controller effectively maintained the DC motor’s speed with optimized settings. The system showed improved stability, reduced overshoot, and minimized steady-state error after tuning.

**4. Conclusion**

* **Color Sensor**: The calibration using white and black reference points enabled accurate color detection for a variety of colors. The sensor’s performance met the expected accuracy requirements after fine-tuning.
* **PID Controller**: The PID tuning process successfully optimized the DC motor speed control. Adjustments to Kp, Ki, and Kd resulted in improved system performance, characterized by stable operation and minimal error.

14

**5. Future Work**

* **Color Sensor**: Further calibration with a broader range of colors and under different lighting conditions could enhance accuracy.
* **PID Controller**: Exploring advanced tuning methods or adaptive control strategies might provide even better performance in dynamic environments

## Challenges

**PID Tuning**: Achieving optimal PID parameters can be complex, requiring precise tuning to balance speed stability, responsiveness, and minimal oscillations.

**Sensor Accuracy**: The TCS 3200 color sensor may face challenges in accurately detecting colors under varying lighting conditions and speeds, affecting sorting precision.

**Load Variability**: Handling varying load weights smoothly without causing delays or jams demands robust motor control and consistent speed adjustments.

**Integration Issues**: Ensuring seamless communication and compatibility between the Arduino, PID controller, color sensor, and servo motor can be technically challenging.

**Calibration**: Regular calibration of sensors and control algorithms is necessary to maintain system accuracy and efficiency over time.

15

## Key Learnings

1. **PID Tuning Importance**: Proper tuning of PID parameters is crucial for achieving stable and responsive motor control in varying load conditions.
2. **Sensor Calibration**: Accurate color detection relies on effective calibration of the TCS 3200 sensor, especially under different lighting and operational conditions.
3. **Load Management**: Managing varying loads efficiently requires precise speed adjustments and robust control mechanisms to prevent system disruptions.
4. **Component Integration**: Successful integration of hardware components and software requires careful planning and testing to ensure compatibility and reliable performance.
5. **System Optimization**: Continuous testing and calibration are essential for maintaining system accuracy and optimizing overall efficiency in real-world applications.

These insights are fundamental for developing and maintaining an effective conveyor system with dynamic control and sorting capabilities.

Top of Form

16

Bottom of Form

## Results

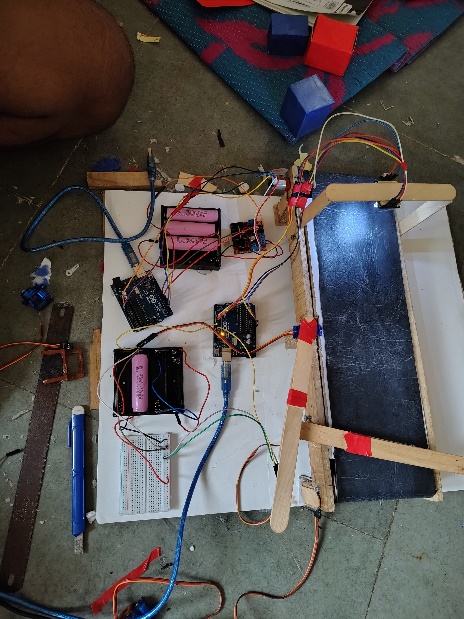
The results after all our training and testing are shown below.

After multiple testing of different Kp Ki and Kd values we found the best resilt as :

* 1. Kp – 1.600
  2. Ki – 0.500
  3. Kd – 0.0



Also, the color sorting mechanism is working well in sorting the red, blue and green color boxes.



17

## Future Scope

In the future, the conveyor system could be improved by using machine learning to better handle different loads and predict maintenance needs. Upgrading to advanced cameras could allow the system to sort items based on more than just color, such as size or shape. Adding IoT technology could enable real-time monitoring and automatic adjustments, making the system even more efficient and adaptable.

# SOURCES USED

For our project, we used the following sources:

 Tutorials **Point:**

* **PID Controller Tutorial**: Basic introduction and examples of PID control.
  + Tutorials Point PID Controller
* **Sensors Tutorial**: Covers various types of sensors, including color sensors.
  + Tutorials Point Sensors

 Adafruit **Learning System:**

* **Color Sensors**: Guides and tutorials for using color sensors with Arduino and other platforms.
  + Adafruit Learning System

18

***THANK YOU***