Electrical Engineering Department

Computers and Systems Program

Graduation Year

جامعة اسيوط
كلية الهندسة
Faculty of Engineering – Assiut University

Graduation Project

Design and Control of an Active Suspension System for Automobiles

Project Supervisors

Prof.Abdelfatah Mahmoud Mohamed

Dr.Abdelrahman Ahmed Ali Morsi

JUL 2024

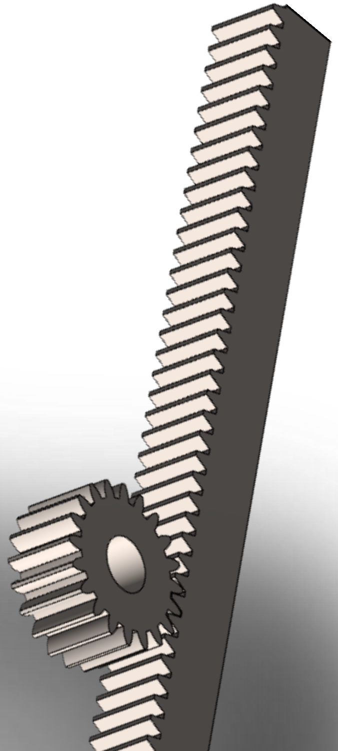
welcome

مرحبا

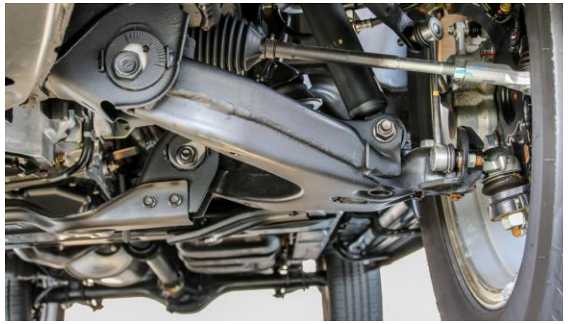
Dr. Ahmed Saad

Mechanical Engineering Department

Table of contents

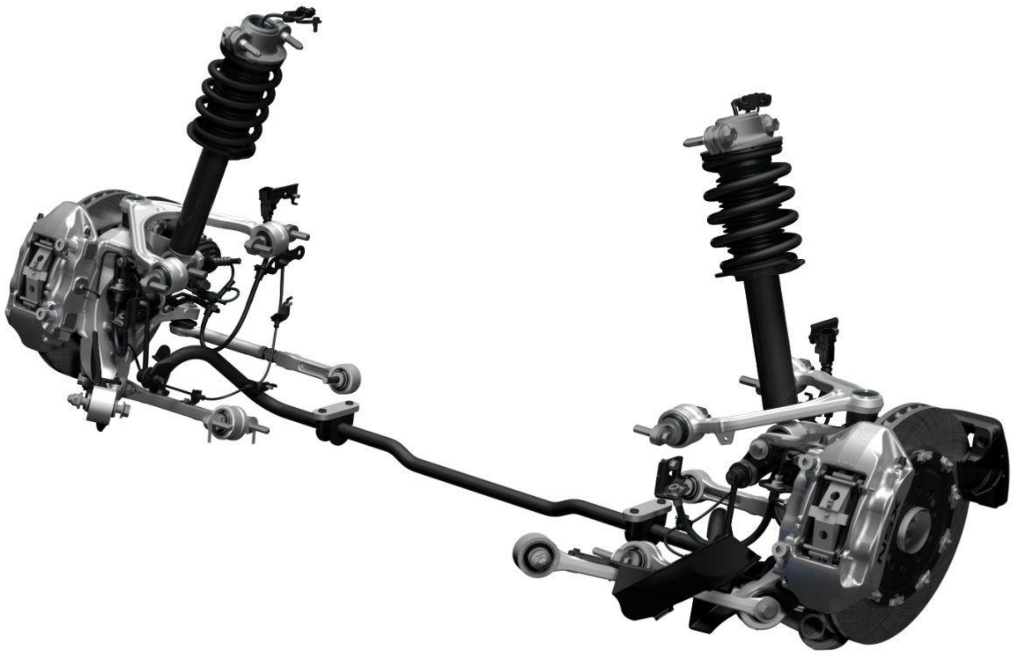


Suspension Types



MacPherson

- Space Efficiency
- Weight Reduction
- Manufacturing Cost
- Handling Performance



Adaptation Challenges

Mechanical Design

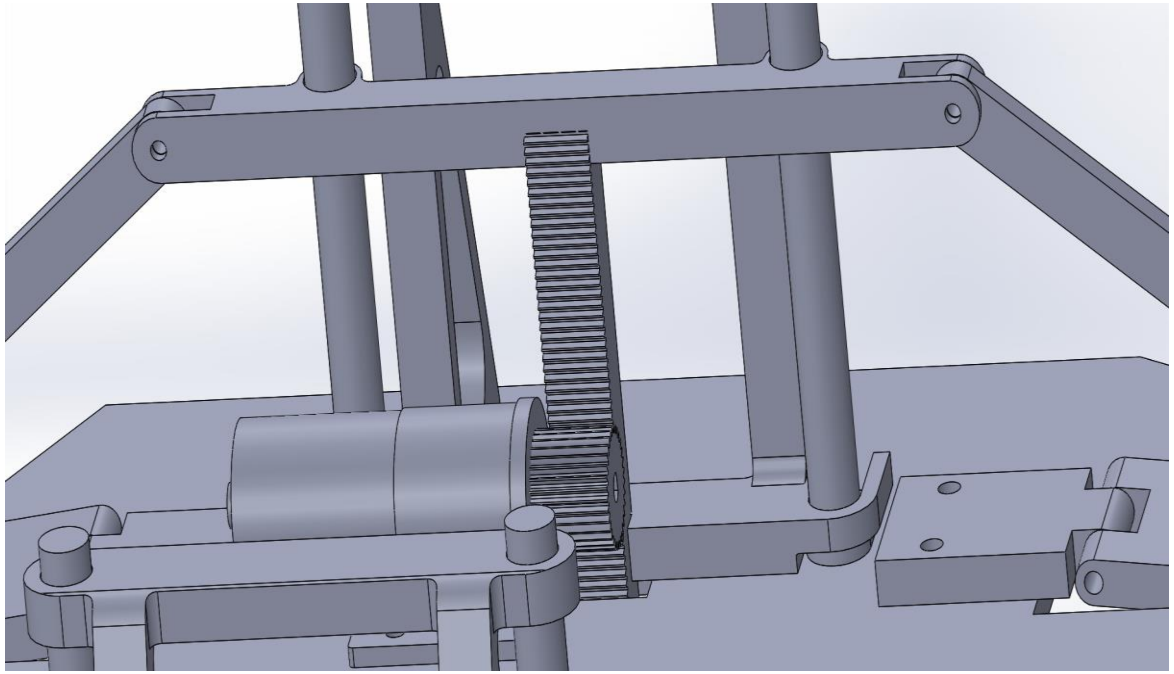
01

- Complexity
- Performance
- Energy Consumption
- Cost
- Flexibility

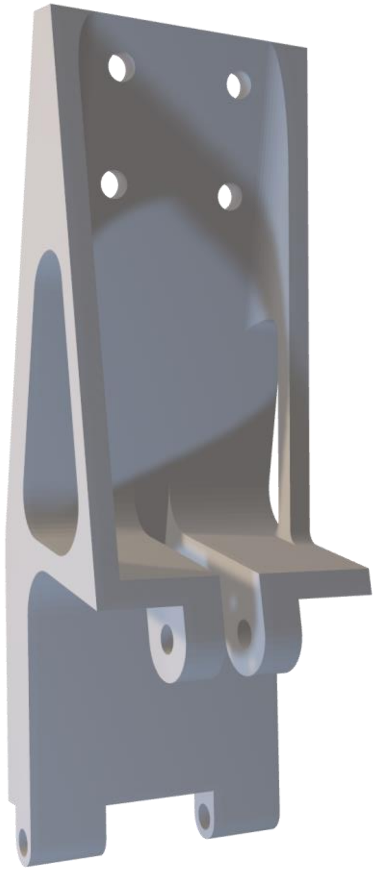
Mechanisms

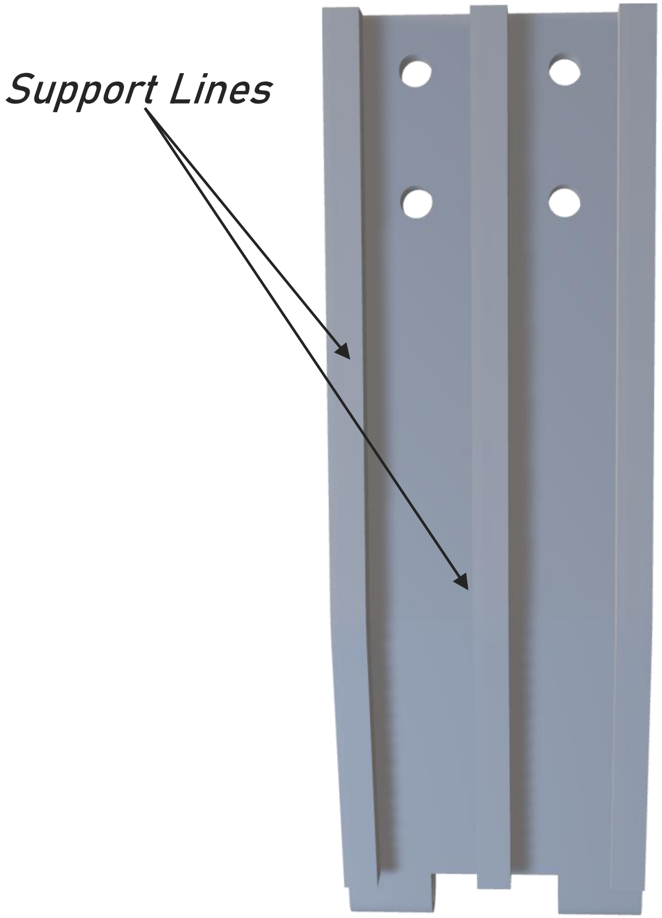
Power Screw

Rack And Pinion

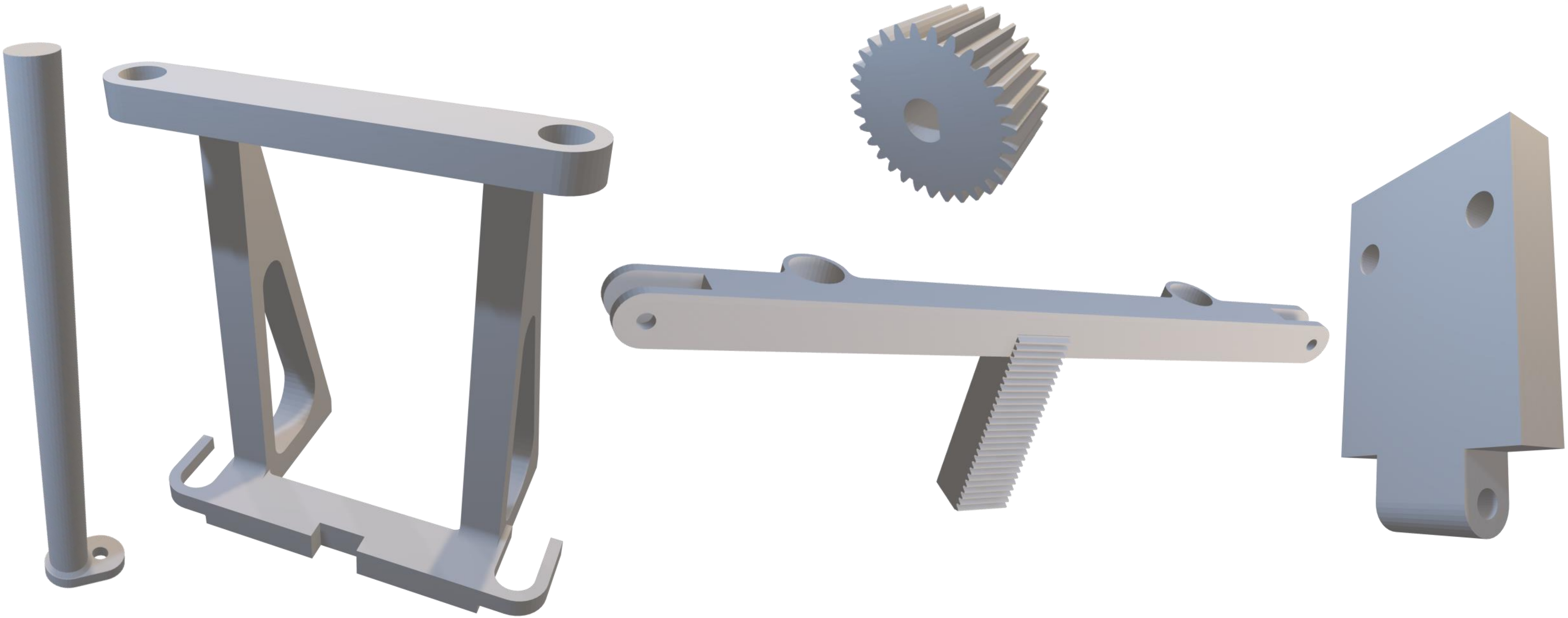


Design Parts





Design Parts



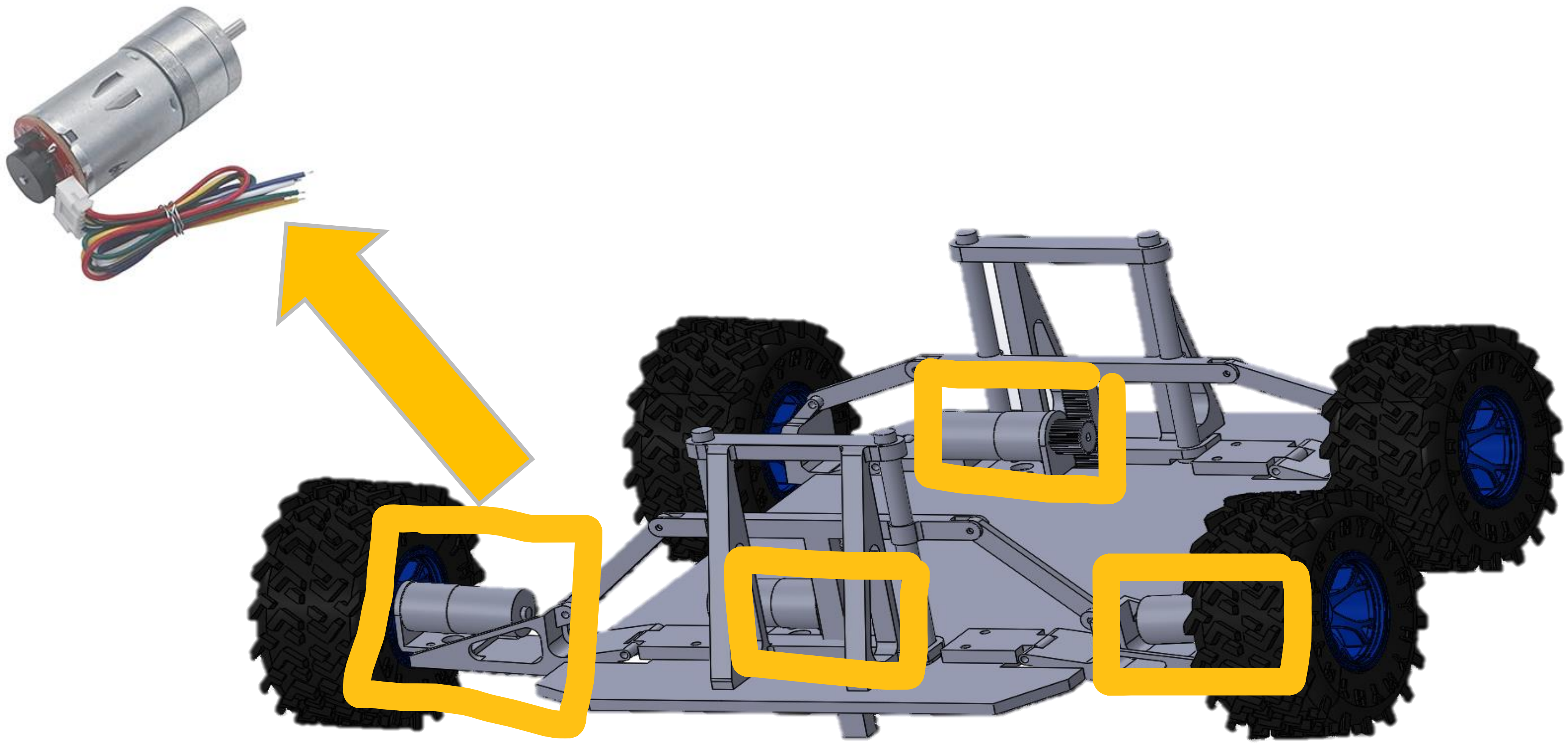
Electrical Components

- 12 Volt DC motors (4 motors)
- MPU6050 IMU Sensor
- L298N motor Driver
- Webcam
- PCB control circuit

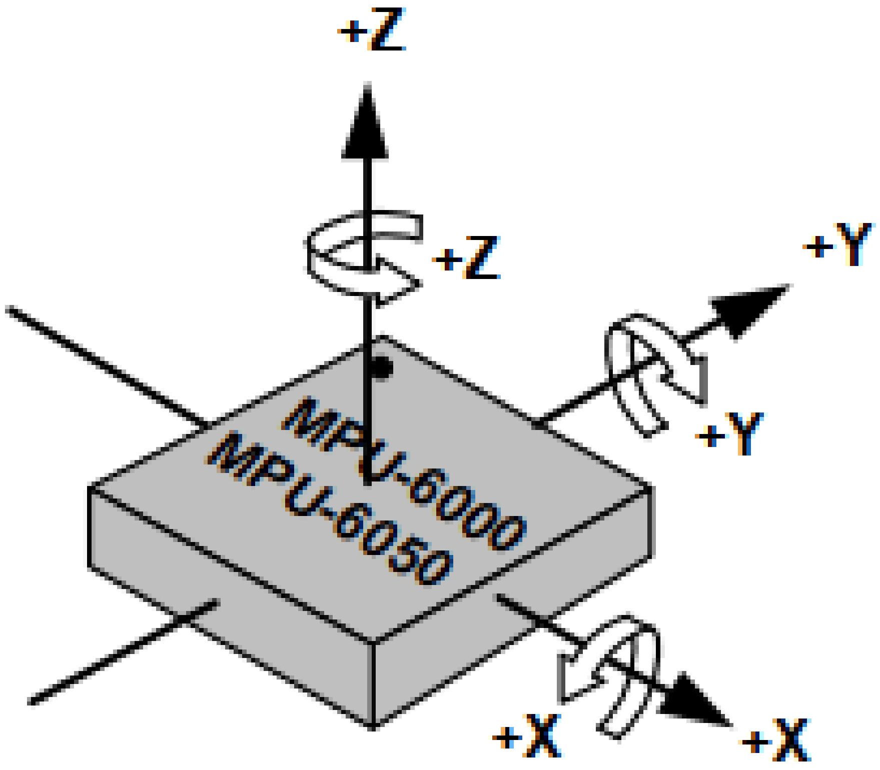
Controllers

- STM32F401RCT6 black pill board (main controller)
- Raspberry pi 4 model B (second main controller)
- Arduino Nano (secondary controller)

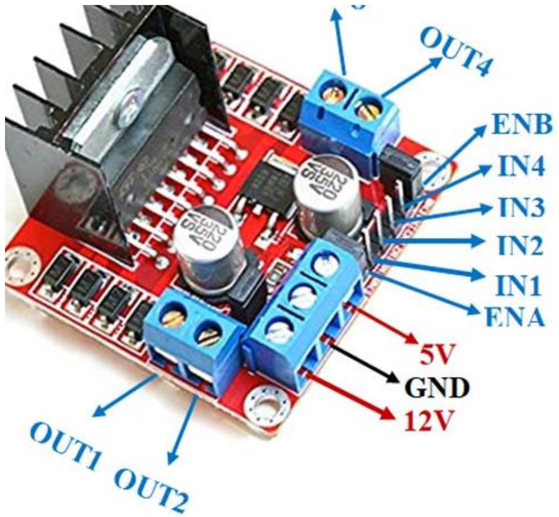
12 Volt DC motors

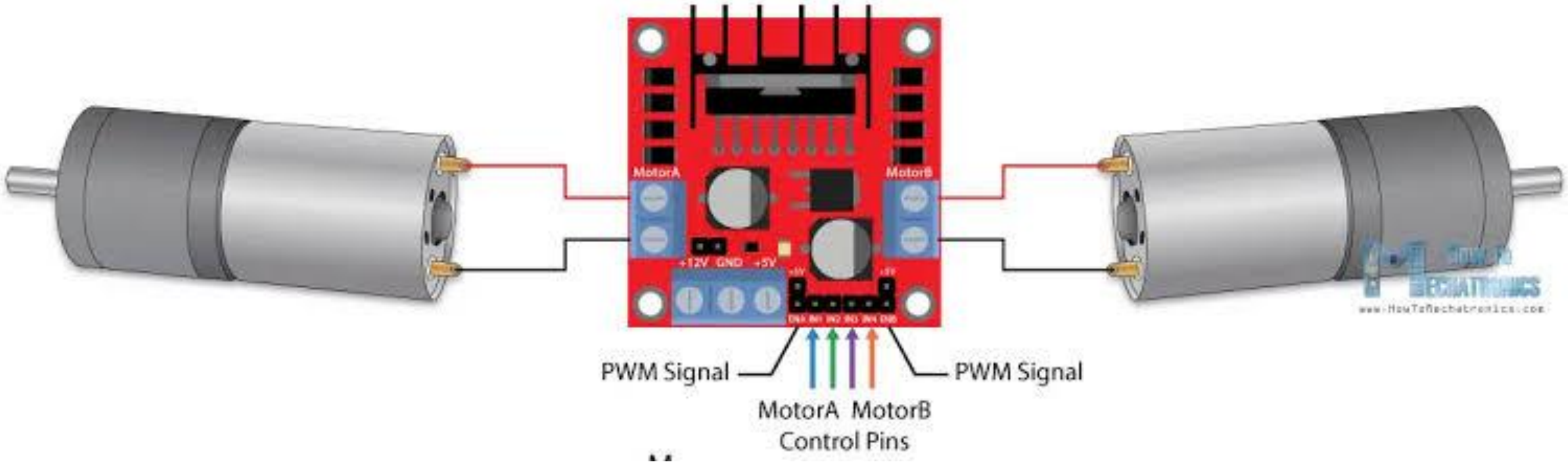


MPU6050 IMU Sensor

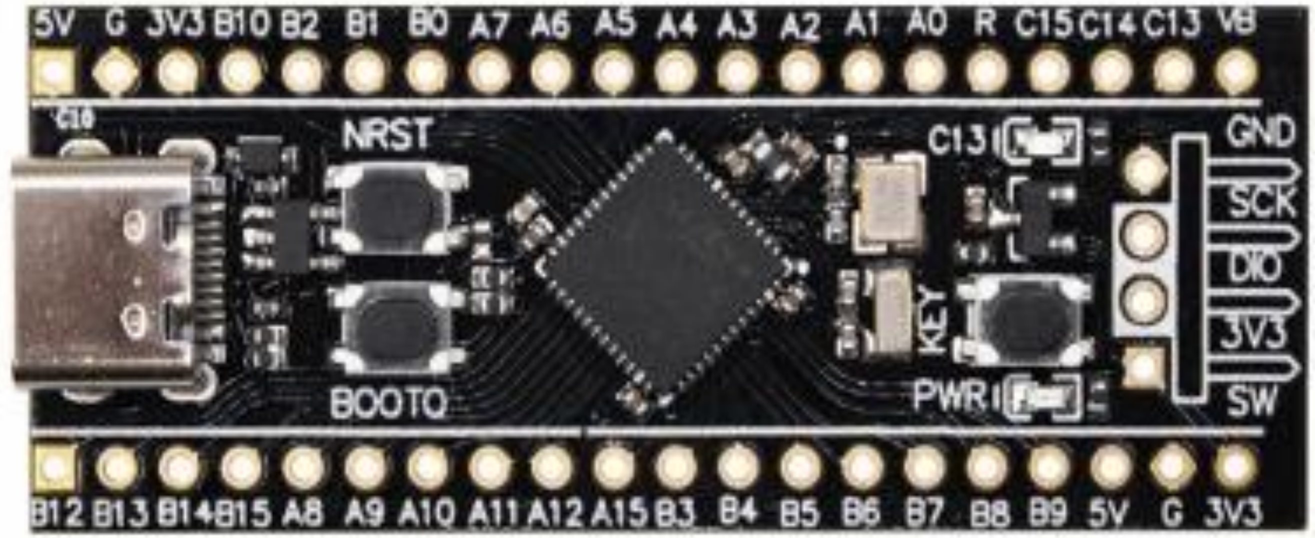


L298N motor Driver





STM32F401RCT6 black pill board (main controller)



Stm32f401rct6 Selected Pins

PH0 - OSC_IN	RCC_OSC_IN	
PH1 - OSC_OUT	RCC_OSC_OUT	
PA0-WKUP	TIM5_CH1	mf_pwm
PA1	GPIO_Output	mf_dir1
PA2	GPIO_Output	mf_dir2
PA3	GPIO_Output	mb_dir1
PA4	GPIO_Output	mb_dir2
PA5	TIM2_CH1	mb_pwm
PA6	TIM3_CH1	mr_pwm
PA7	GPIO_Output	mr_dir1
PB0	GPIO_Output	mr_dir2
PB1	GPIO_Output	ml_dir1
PB2	GPIO_Output	ml_dir2
PB10	TIM2_CH3	ml_pwm
PA9	USART1_TX	
PA10	USART1_RX	
PA11	GPIO_Input	ml_enc2
PA12	GPIO_EXTI12	ml_enc1
PA15	GPIO_Input	mr_enc2
PB3	GPIO_EXTI3	mr_enc1
PB4	GPIO_Input	mb_enc2
PB5	GPIO_EXTI5	mb_enc1
PB6	I2C1_SCL	
PB7	I2C1_SDA	
PB8	GPIO_Input	mf_enc2
PB9	GPIO_EXTI9	mf_enc1

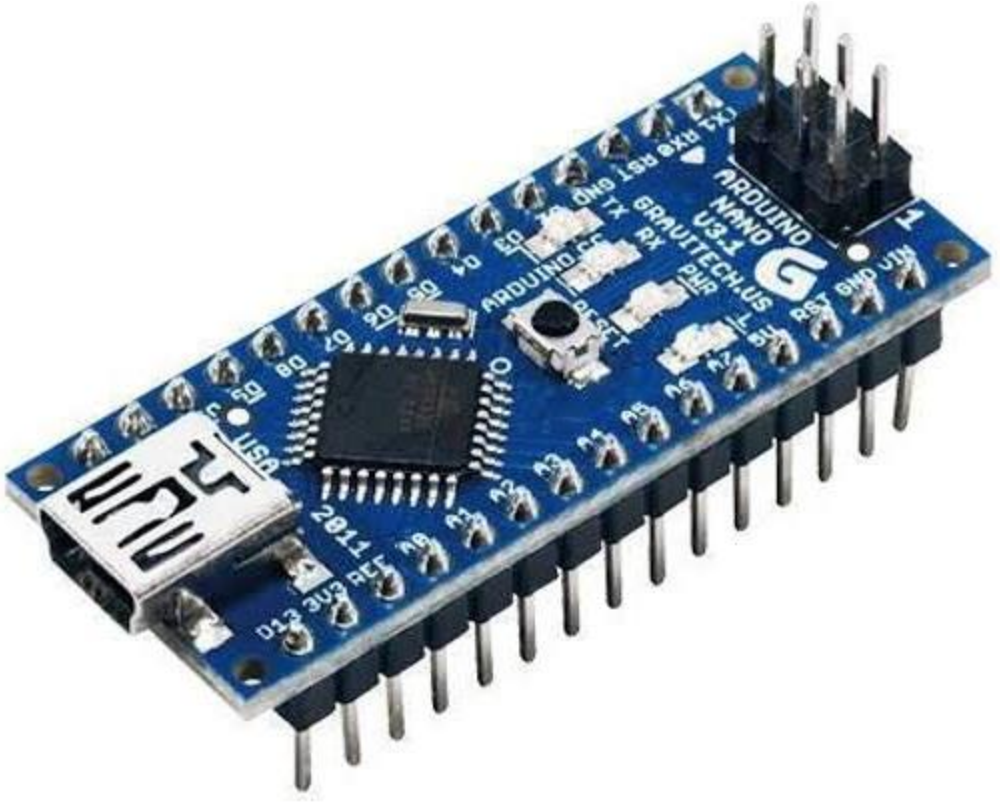
Raspberry pi 4 model B (second main controller)

Automatic Control

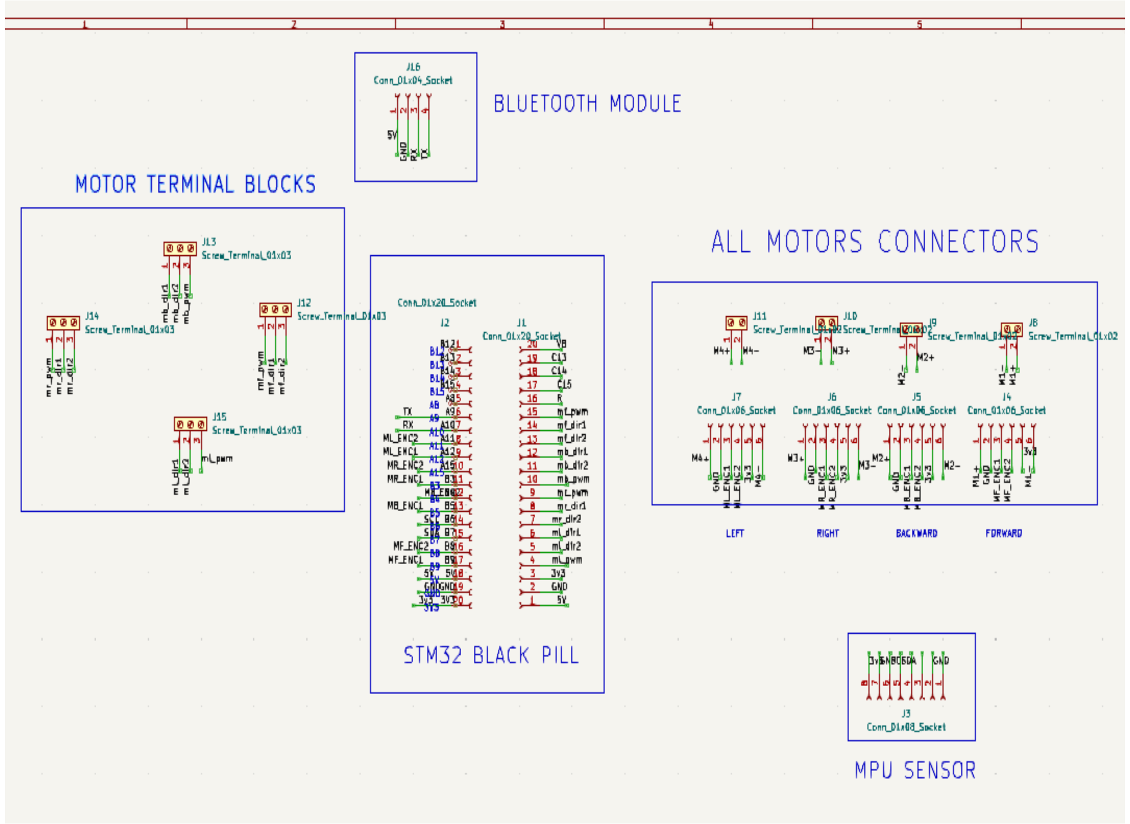
02

Used to be interfaced with webcam through a USB cable

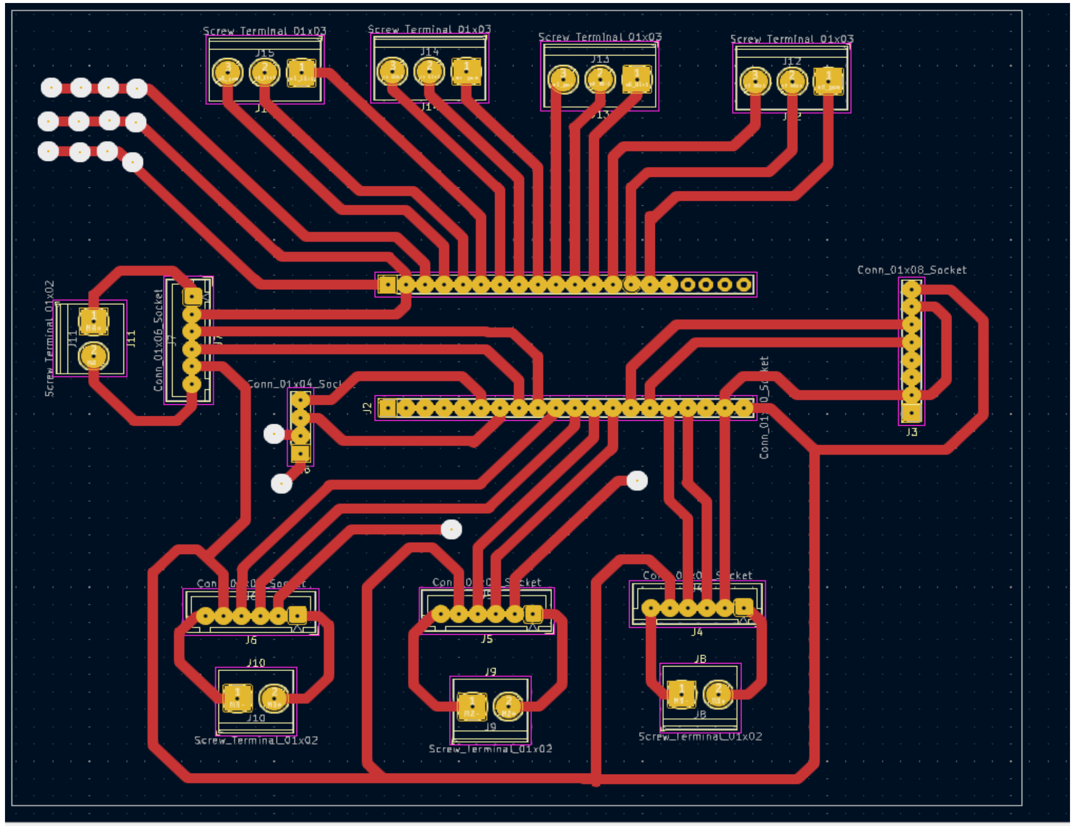
Why ?



Printed Circuit Board (PCB)

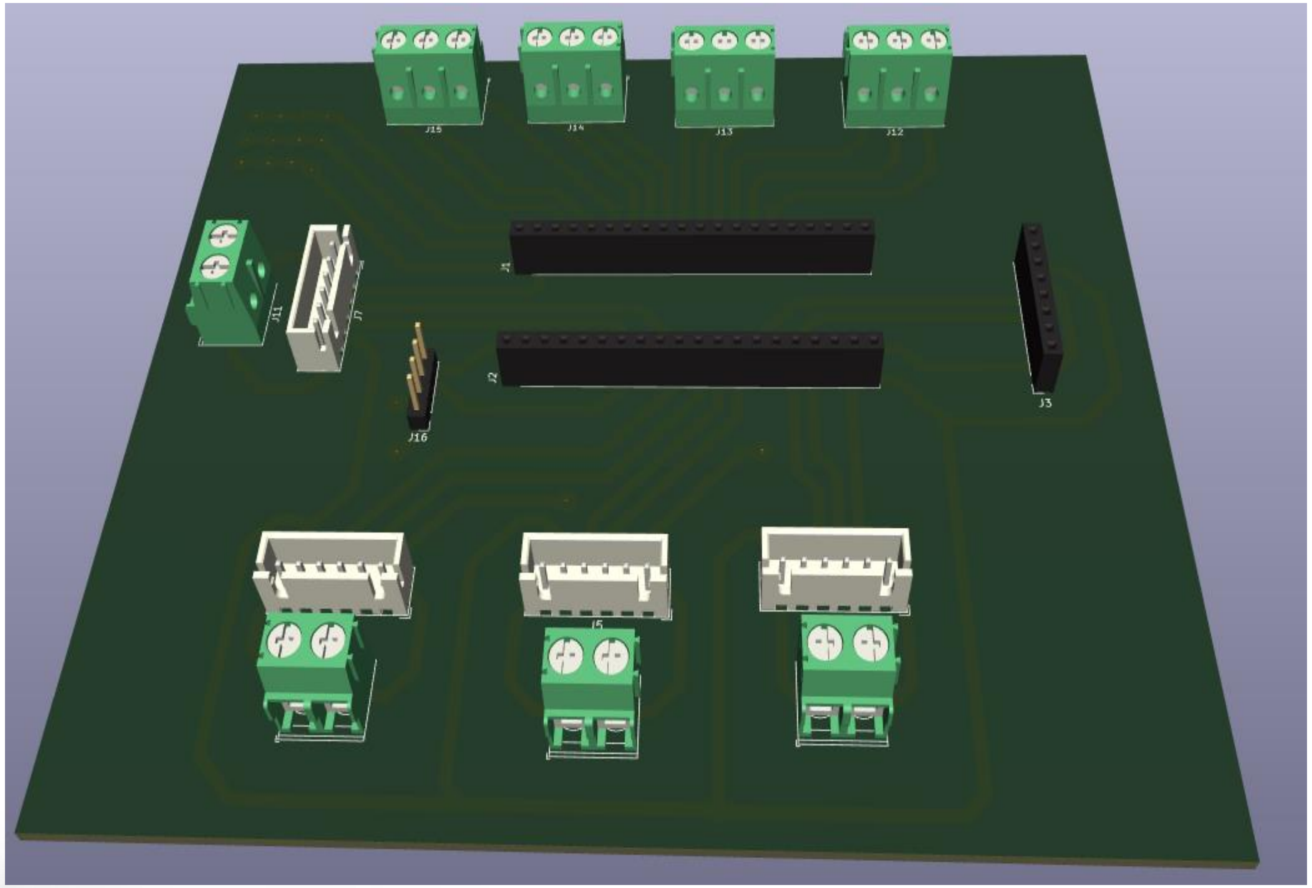


PCB schematic on KiCAD 7

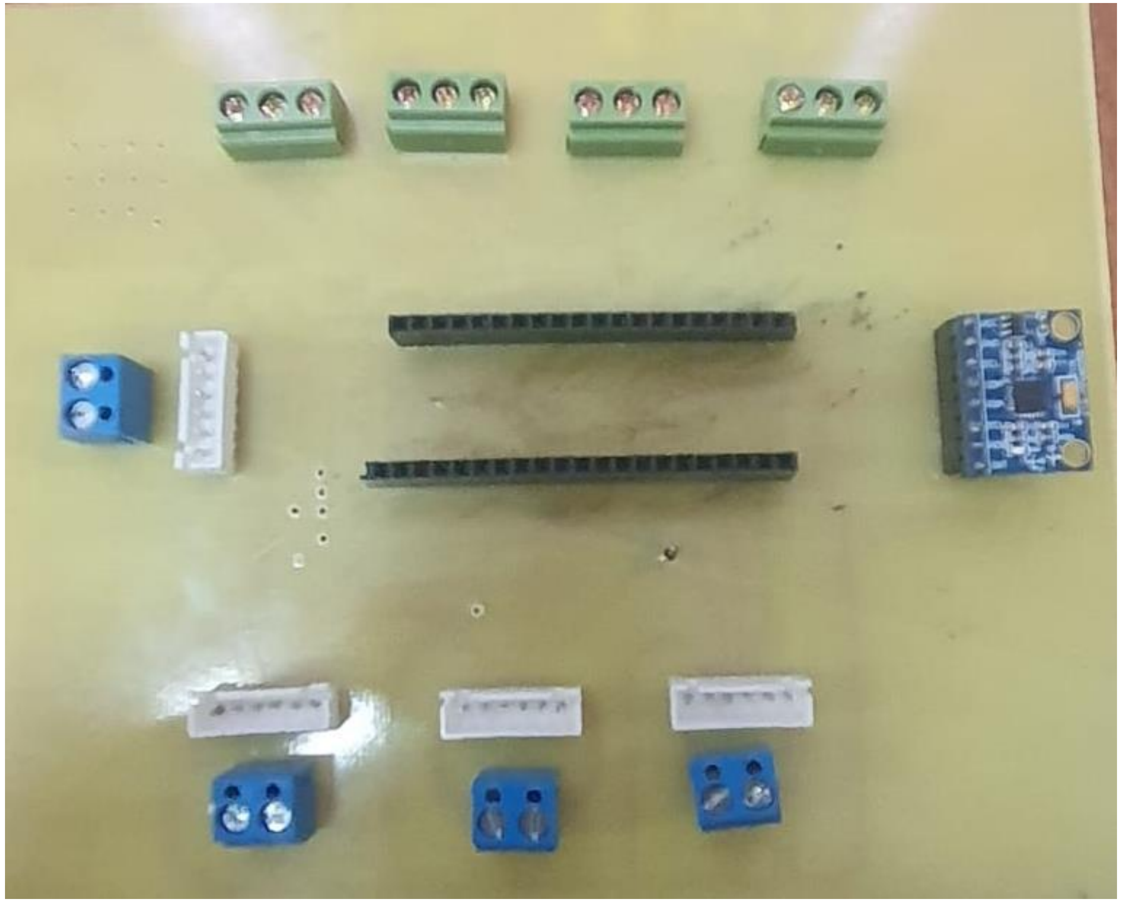


PCB Copper tracks

Printed Circuit Board (PCB)



3D Design

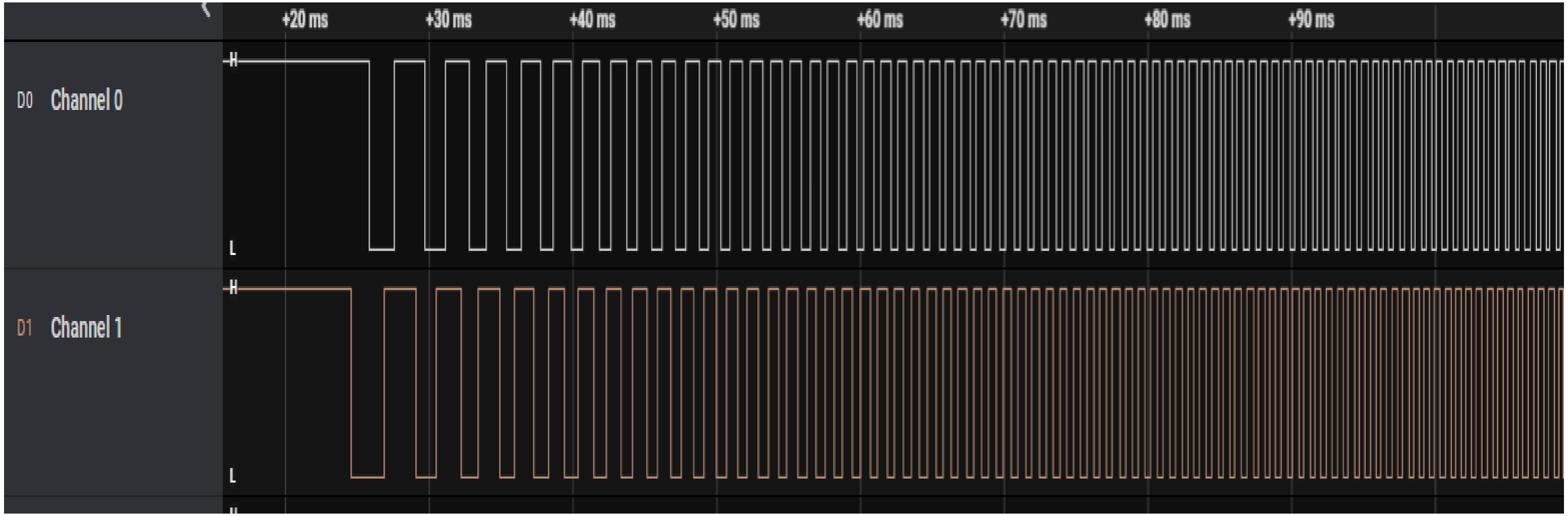


Real PCB

Position Control

Determining the accurate position of the motor we use sequence of steps:

First one: Analyzing the signals of the encoders.



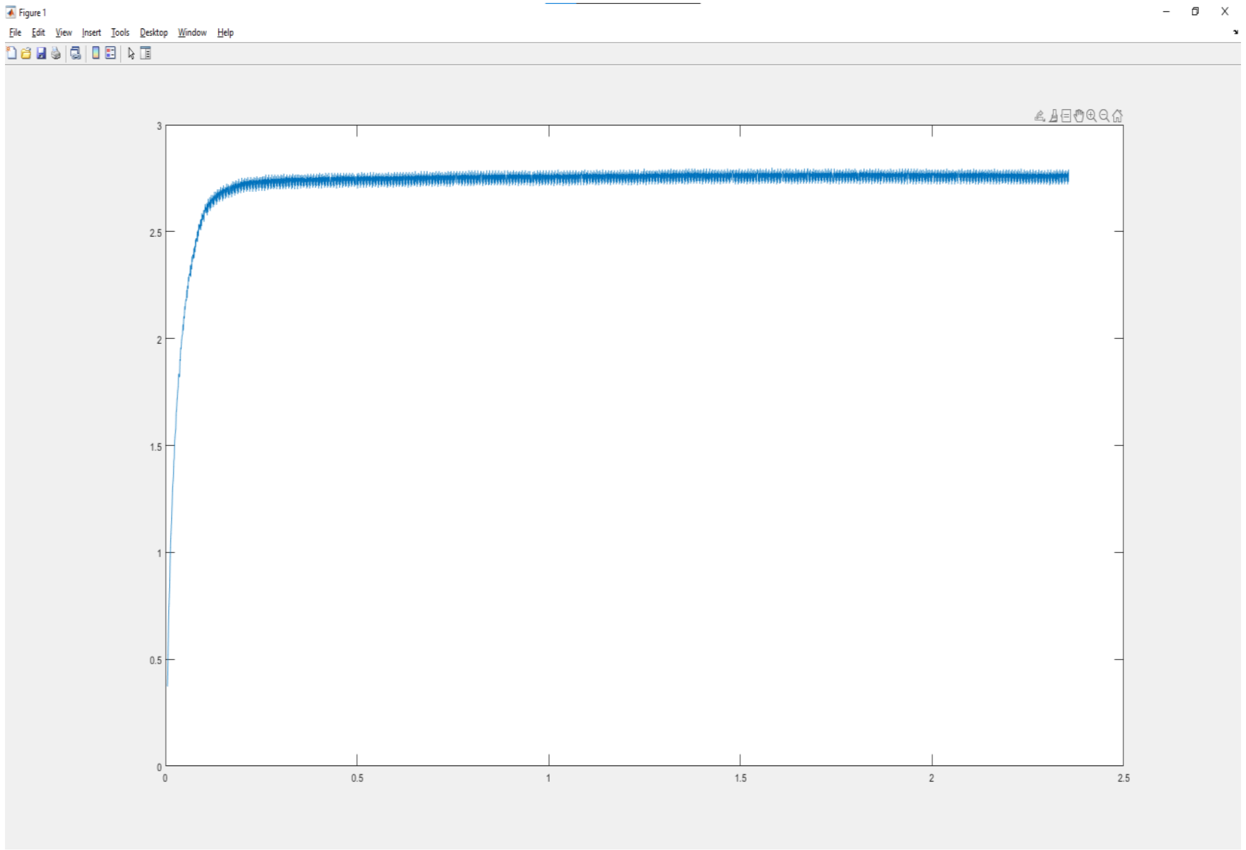
Encoder signals on 2 channels form a single motor

Position Control

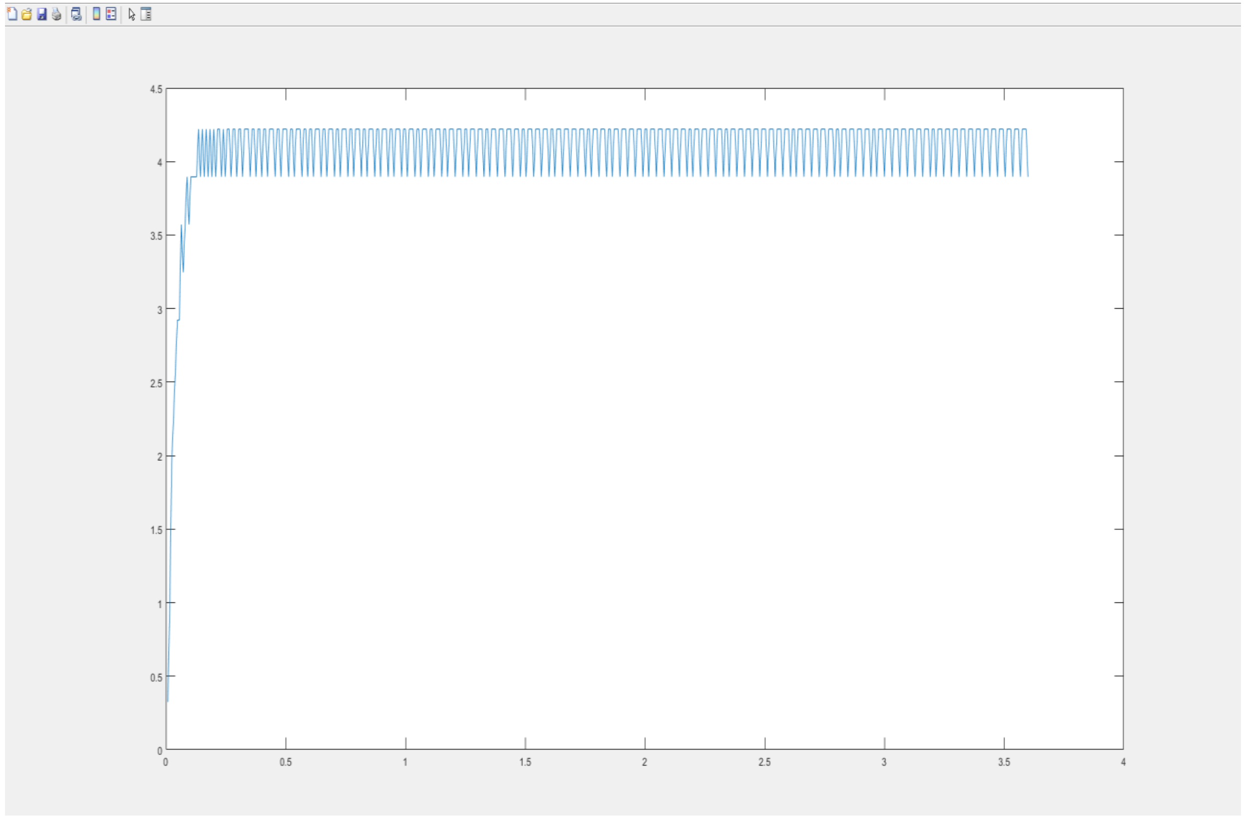
Automatic Control

02

Second one: Finding the Response of dc motor using interval time of pulses



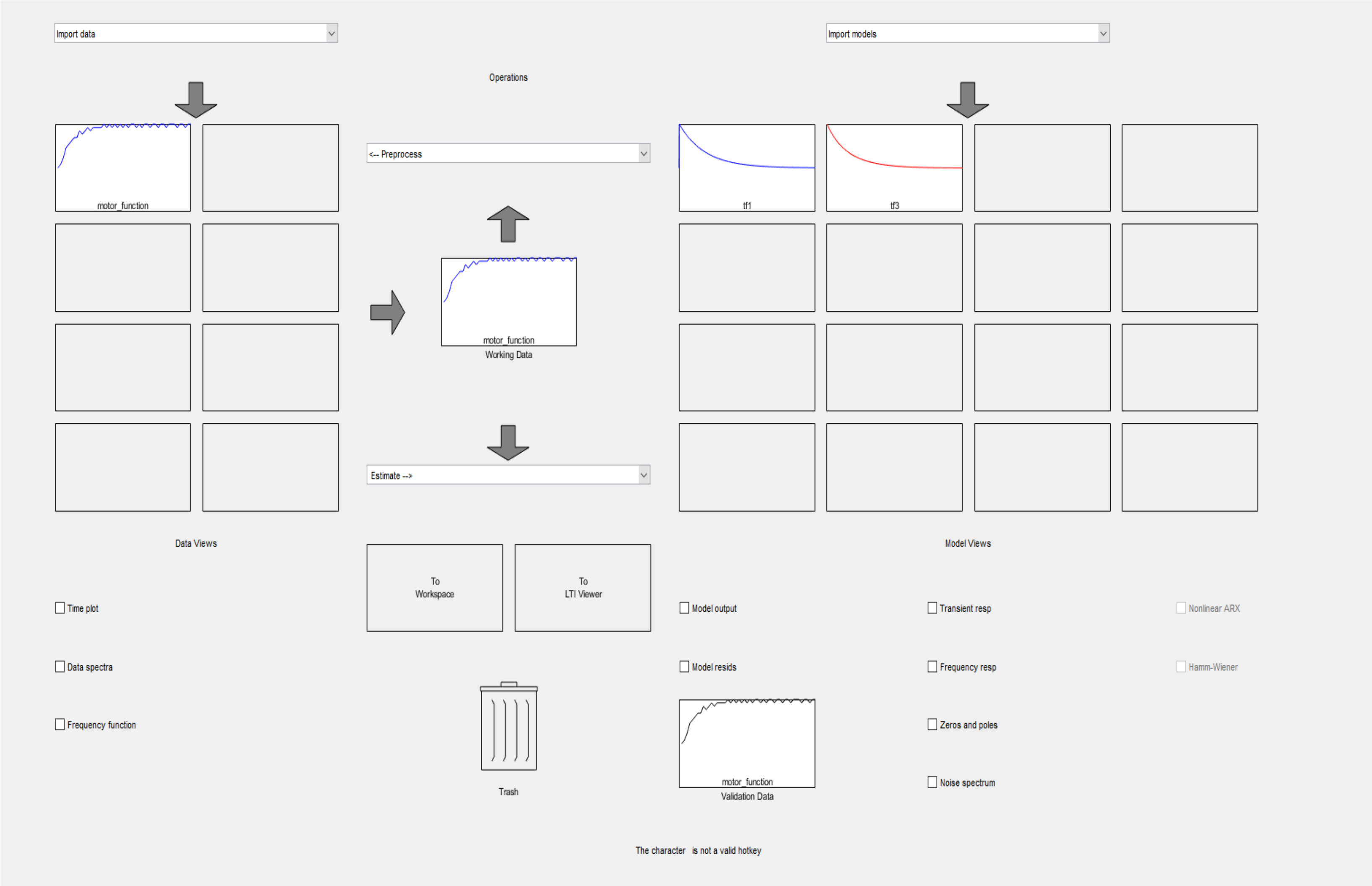
Third one: response of dc motor using in terms of pulses per second



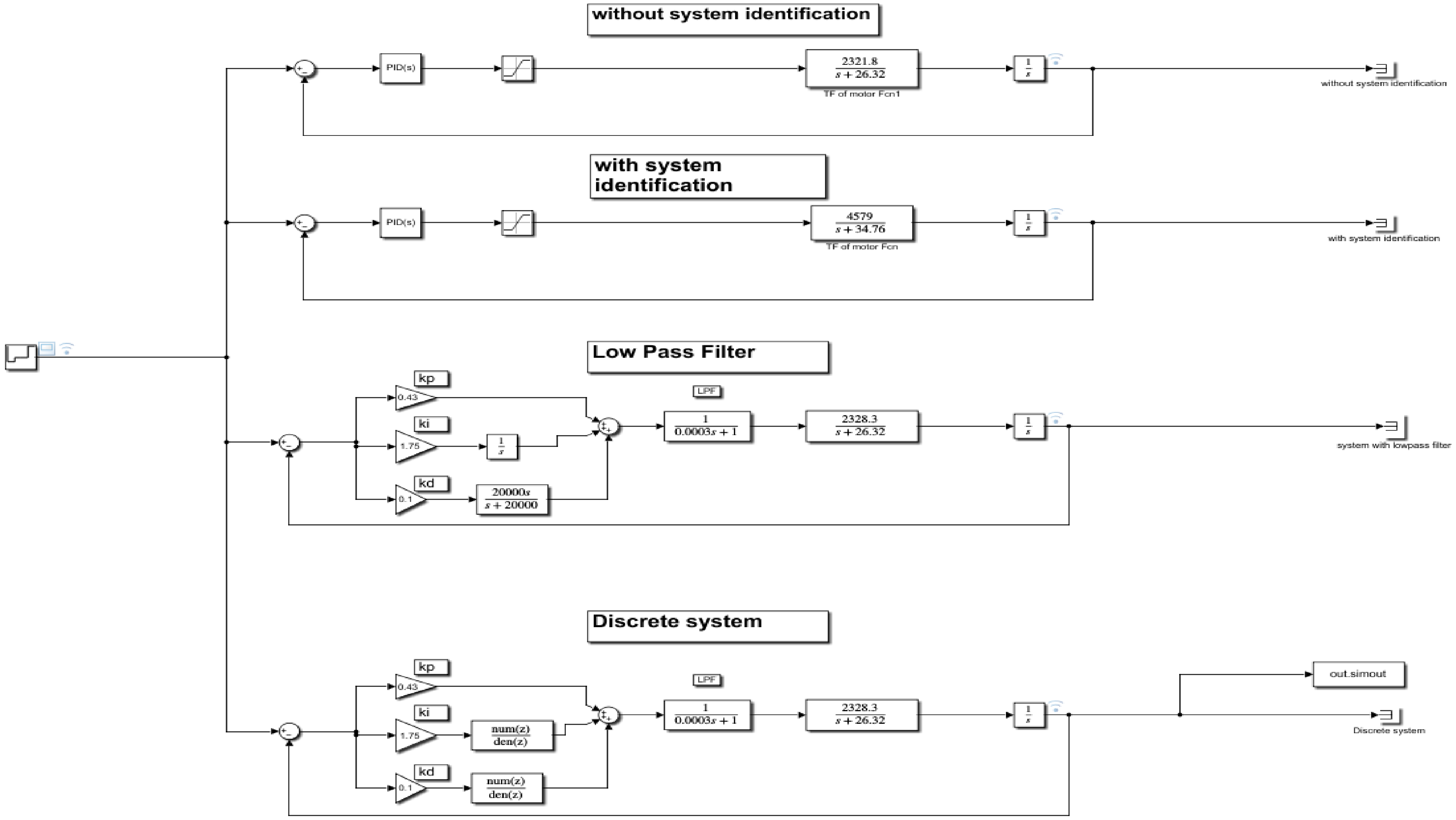
System Identification method in MATLAB

Automatic Control

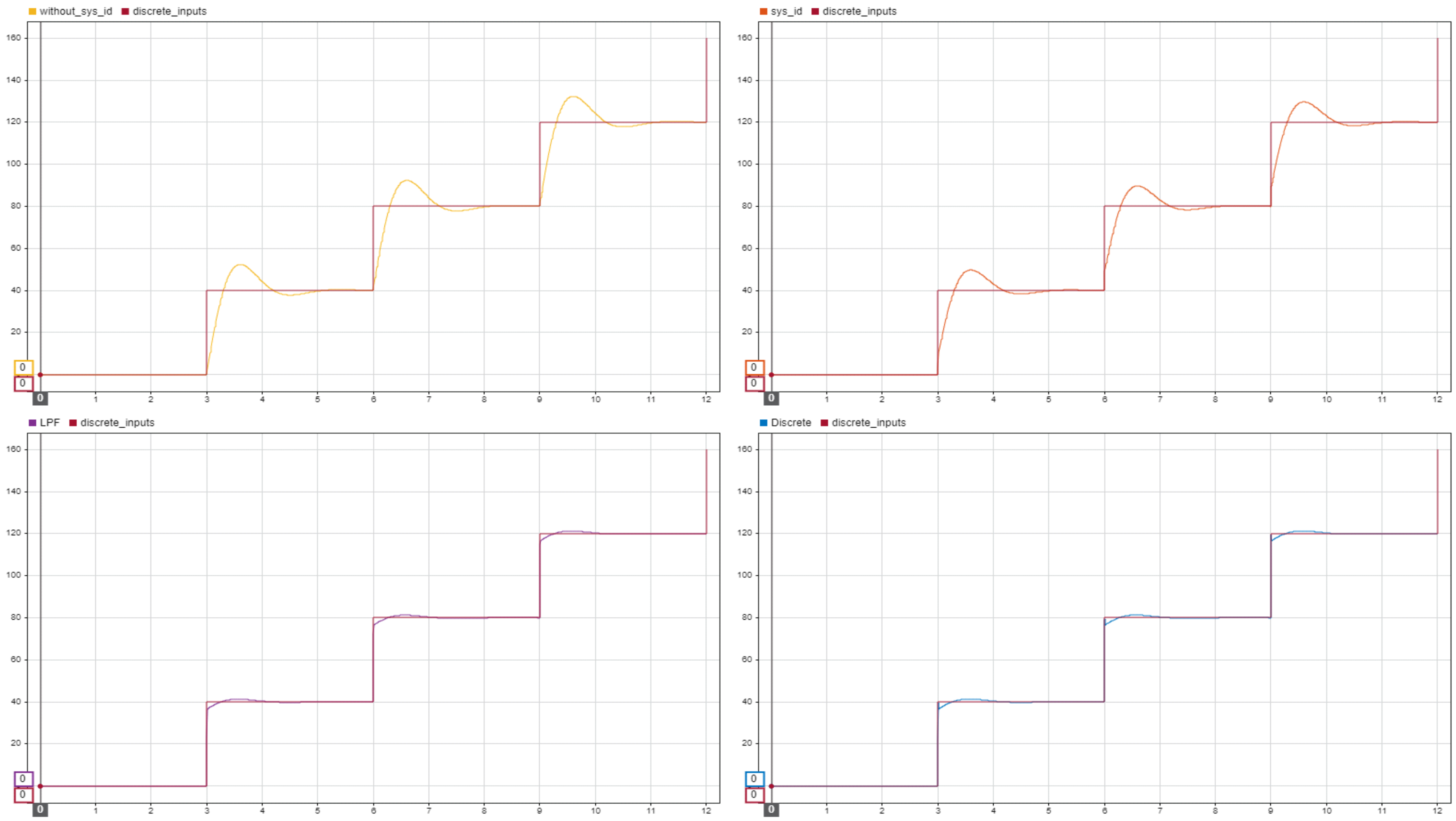
02



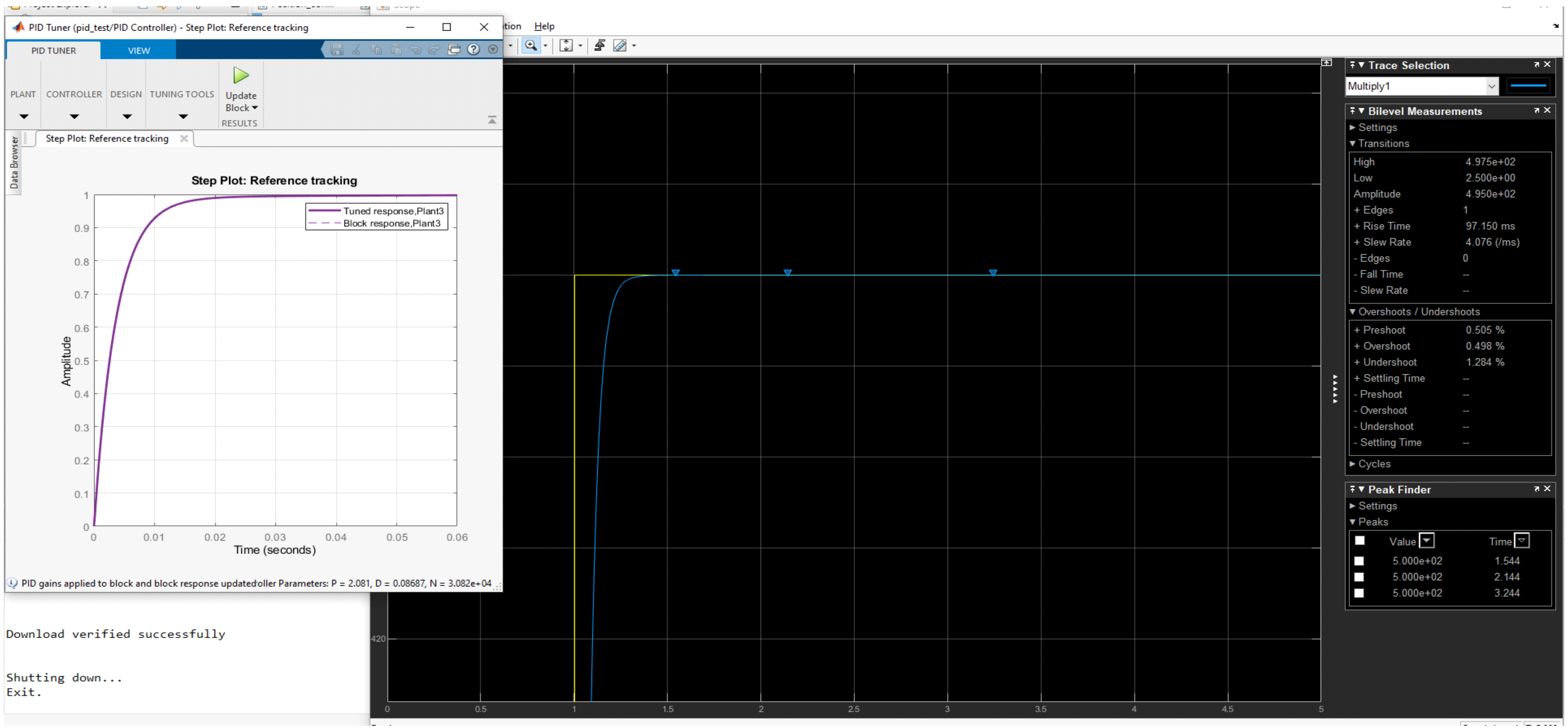
Building Complete control systems with stairs input



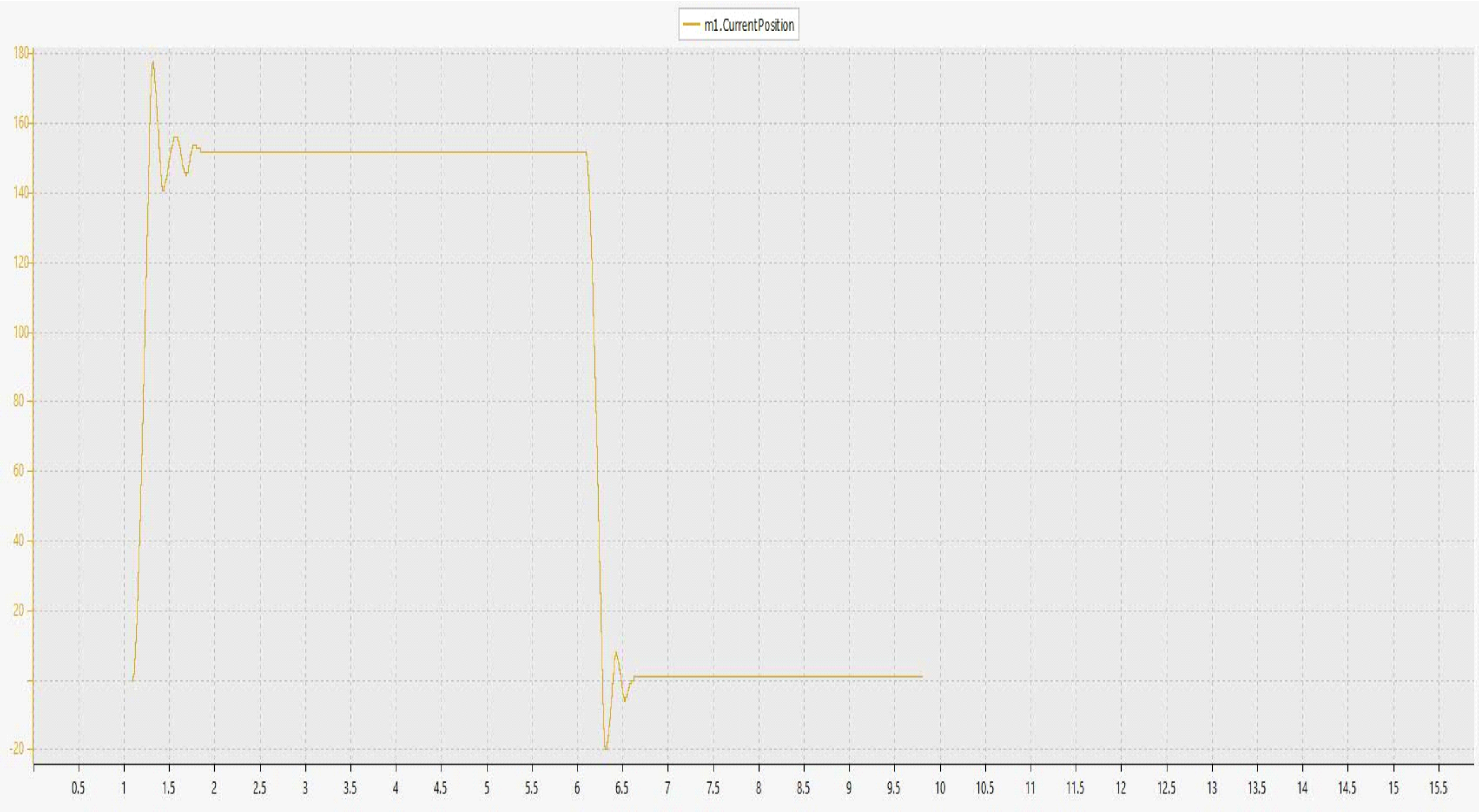
The output of each block diagram



PID Tuning Parameters in Simulink



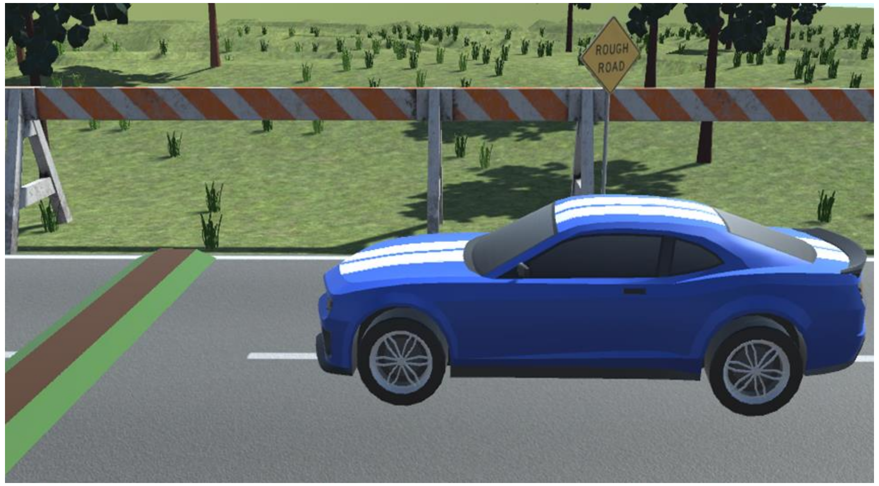
Position Control with Real dc motor, with the same Parameters of PID tuner in Simulink with some tuning.

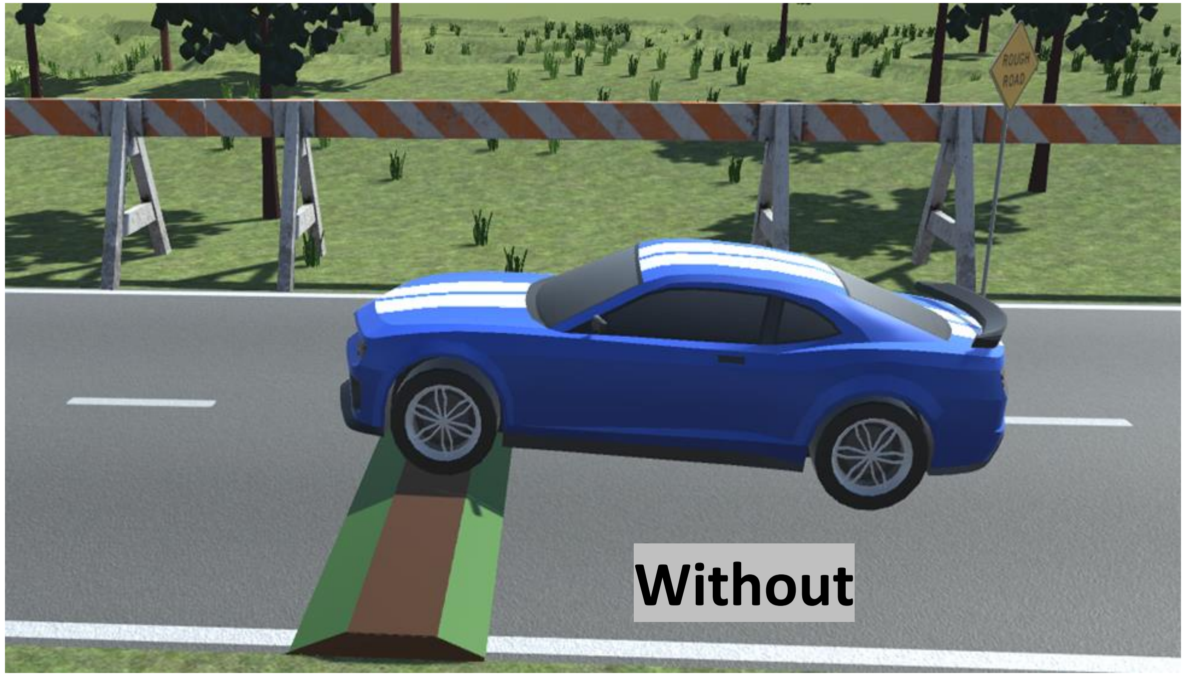


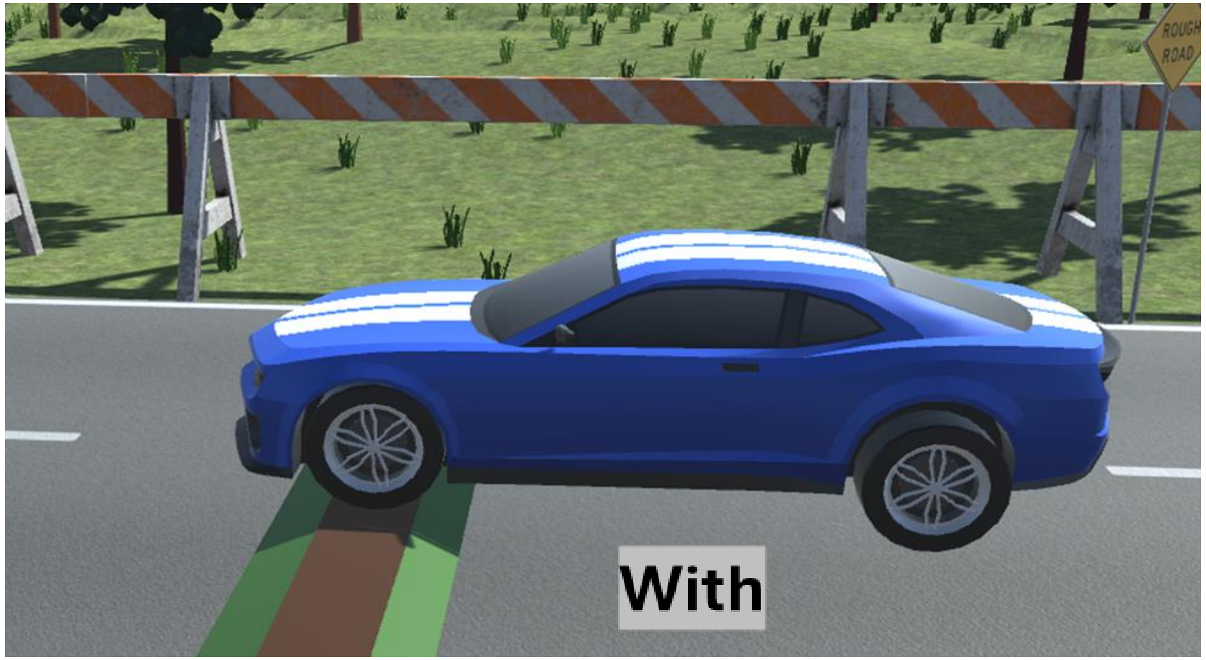
Development Tools

- How It is programmed
- 3D engine (Unity)
- IDE (Visual Studio)
- Programming Language (C#)
- Asset Store: Package in Unity

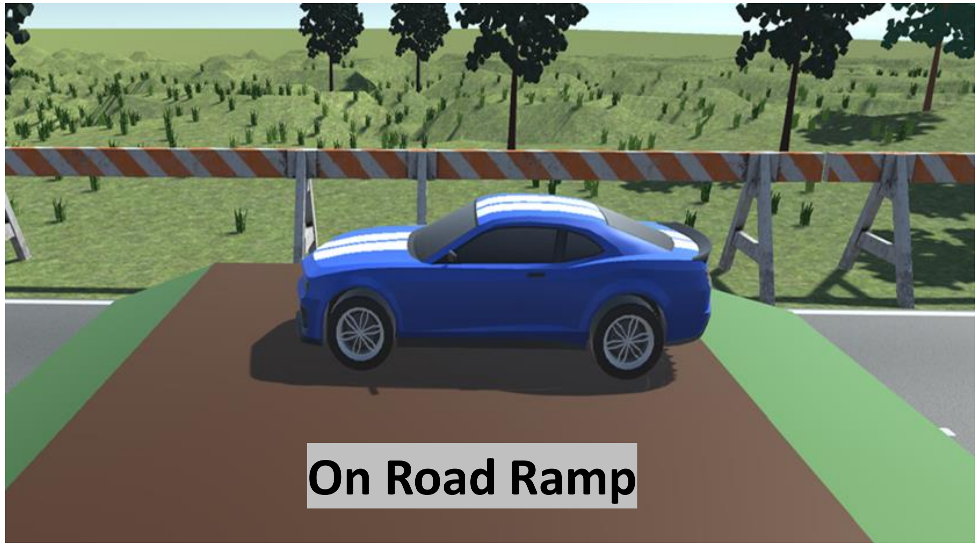
Effect on Road Hump

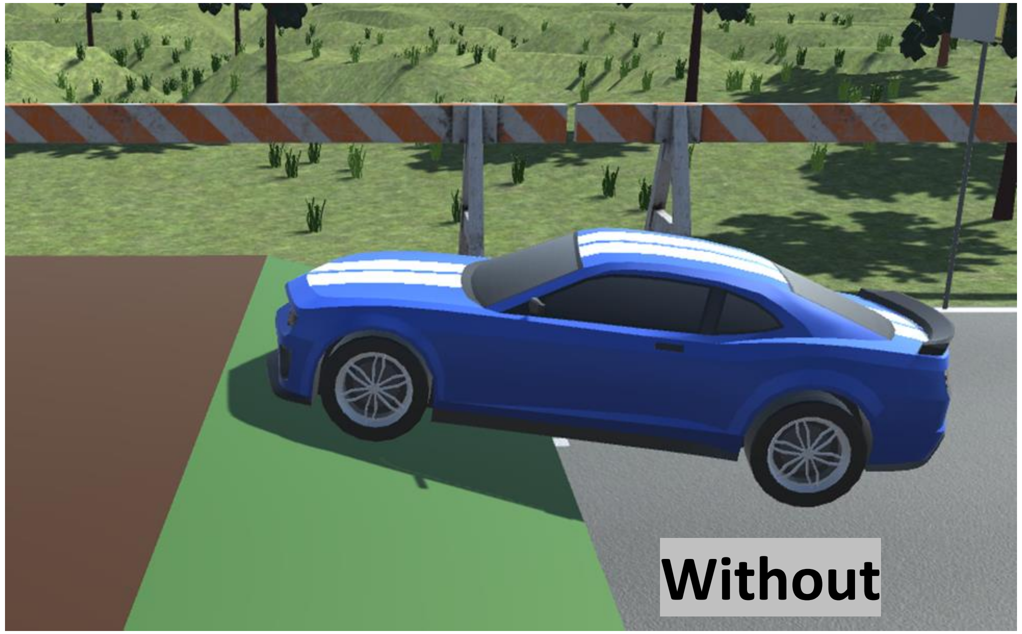


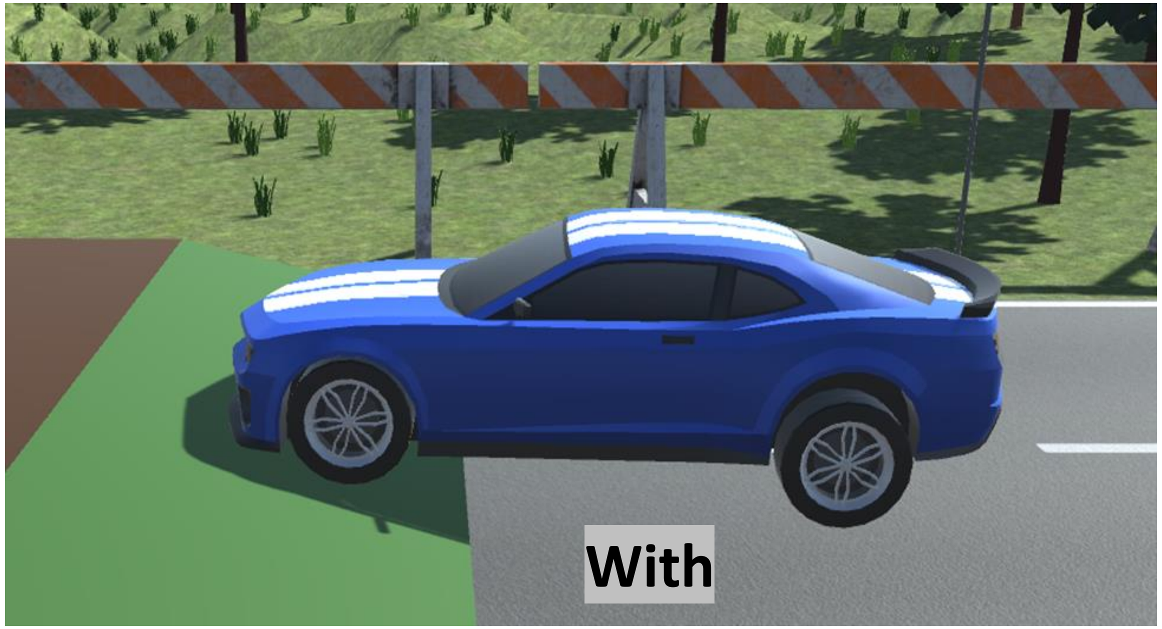




Effect on Road Ramp







Components

Web Camera

Raspberry Pi Model 4 B

Controller Used: Raspberry Pi Model 4 B

A small single-board computer consisting of

USB Ports: For connecting peripherals like keyboards, mice, and storage devices.

HDMI Port: For connecting to a monitor or TV.

Ethernet Port: For wired internet connection

Wi-Fi and Bluetooth: For wireless connectivity

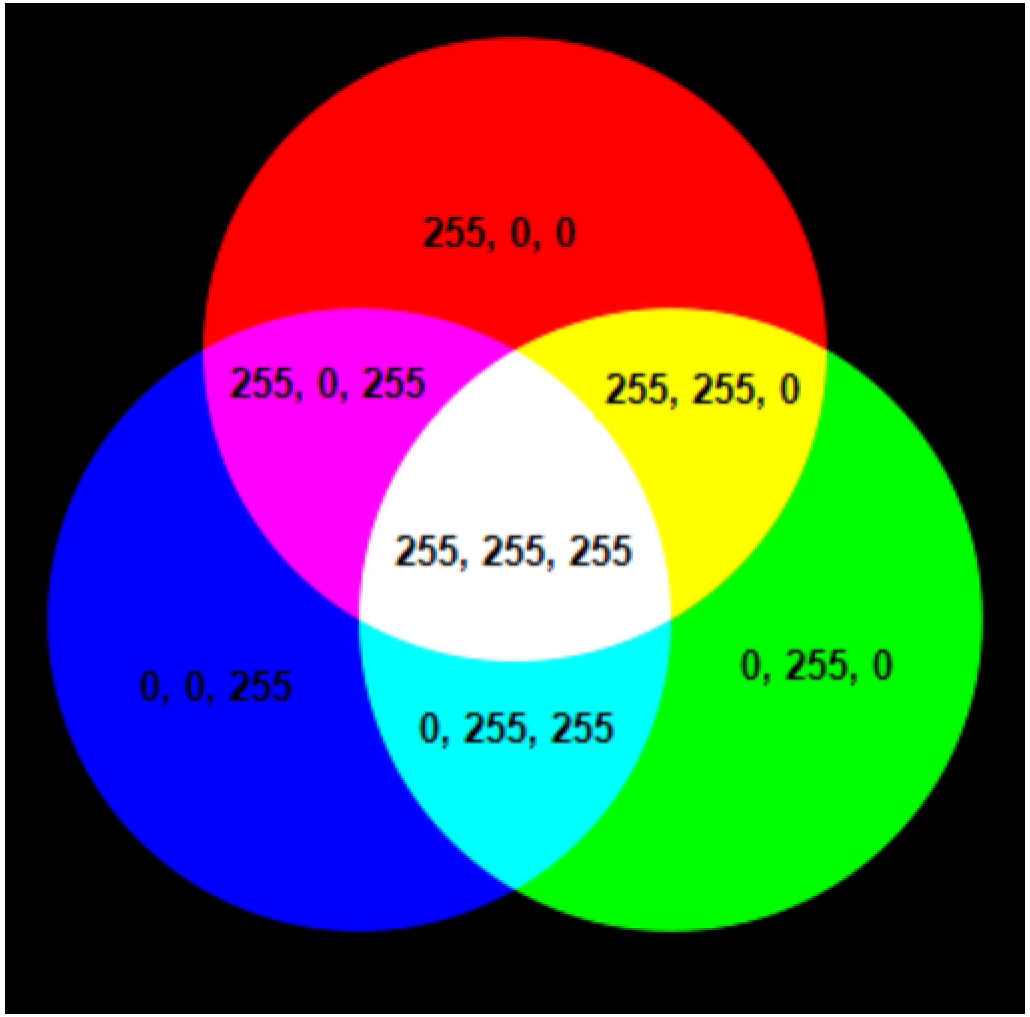
GPIO Pins: General Purpose I/O pins for connecting sensors, LEDs, motors, and other electronic components.

Tools Used

1- **Jupyter Notebook**: a project to develop open-source software, open standards, and services for interactive computing across multiple programming language, which supports python cell by cell running for better debugging

2-**OpenCV**: a popular computer vision library written in C/C++ with bindings for Python, OpenCV provides easy ways of manipulating color spaces which is useful in image processing.

In digital images, colors are typically represented using the RGB (Red, Green, Blue). All colors are represented based on these three colors only by mixing values from ranging from 0 ➔ 255



RGB, HSL and HSV

How Color Identification Using OpenCV?

Convert from RGB to HSV. Why?

1- Robustness to Lighting Variations:

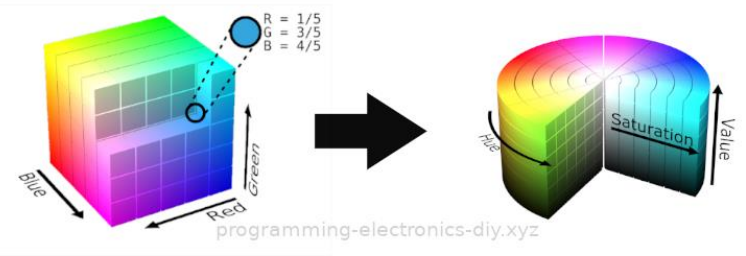
In the RGB color model, changes in lighting can significantly alter the RGB values of a color, making it difficult to detect colors accurately. In contrast, the hue component in HSV is less sensitive to changes in lighting and shadows, making color detection more robust.

4. Simplified Thresholding:

When performing color-based segmentation or detection, it's easier to define thresholds in the HSV space:

- For instance, to detect a specific color, you can set a range for the hue value while allowing a wider range for saturation and value. This simplifies the process of isolating a particular color from an image.

RGB vs HSL



<u>Hue</u>: Represents the type of color (e.g., red, green, blue).

<u>Saturation</u>: the purity of the color. Higher saturation means more vivid colors.

<u>Value</u>: brightness of the color. Higher value means a brighter color.

Color Detection

- Using Python and OpenCV to detect a monochromatic colour object.
- We will use a webcam to detect objects of a single color .
- convert the colors from RGB to HSV format.
- HSV stands for hue(color), saturation(intensity), and value(Brightness).
- Color Range (0-255)

Color Detection

- Import **necessary** libraries

```python
import numpy as np
import cv2
import matplotlib.pyplot as plt
```

- Red color boundaries (HSV Value)

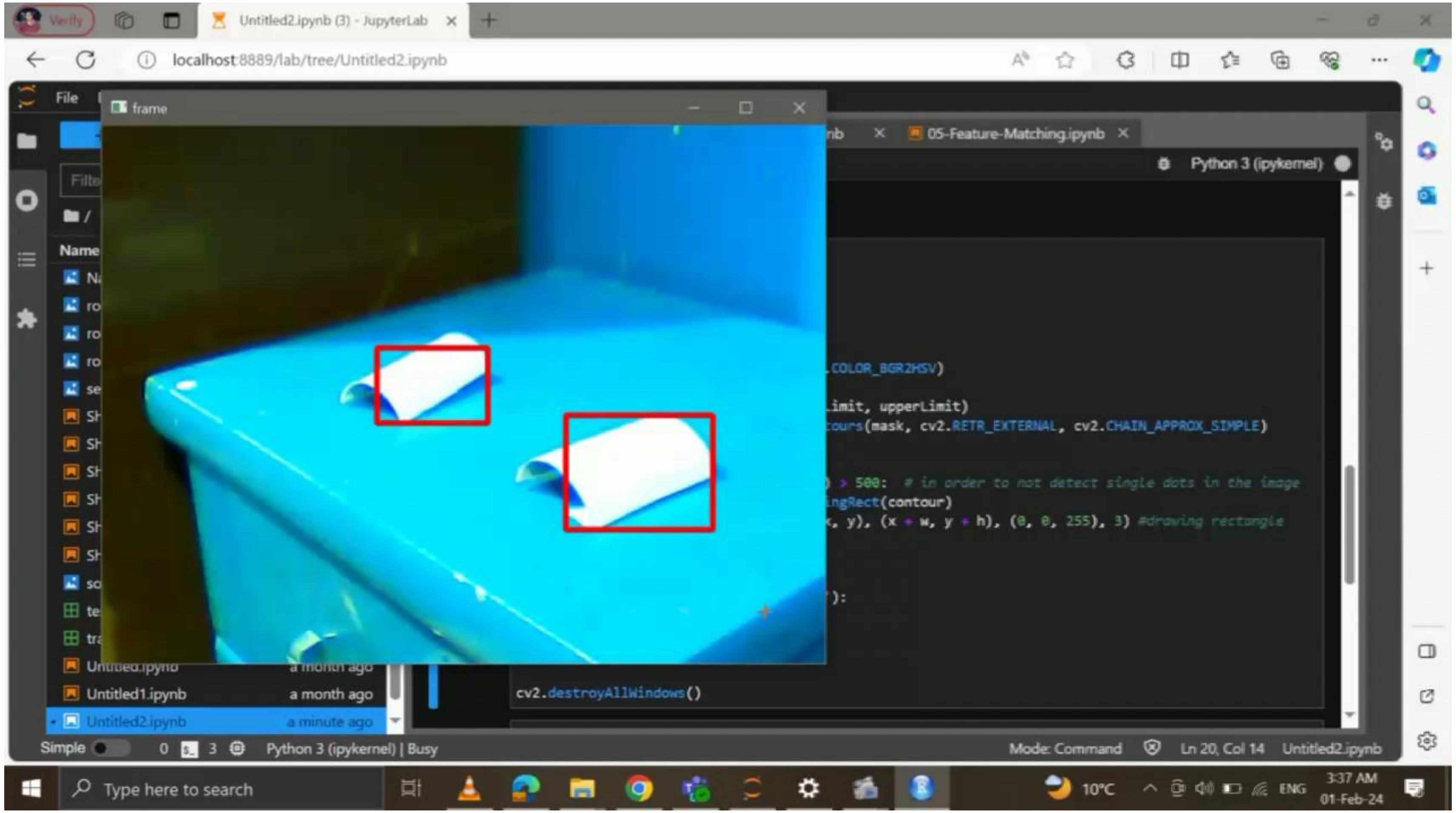
```python
lower_range = np.array([0, 120 ,70 ])#
upper_range = np.array([10, 255, 255])
```

- Open Webcam

```python
cap = cv2.VideoCapture(0)
```

```python
while True:

    ret, frame = cap.read()
    hsv = cv2.cvtColor(frame,cv2.COLOR_BGR2HSV) # convert from BGR to HSV
    mask = cv2.inRange(hsv,lower_range,upper_range) # focus on regions of interest
    _, mask = cv2.threshold(mask,254,255,cv2.THRESH_BINARY) #filter noise
    contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE) # boundaries
    if len(contours) != 0:
        for contour in contours:
            if cv2.contourArea(contour) > 1000:  # to filter out noise. ex: small dots in the image
                x, y, w, h = cv2.boundingRect(contour) # Computes the bounding rectangle
                cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 3) # draw the rectangle
                Detected = 1

    cv2.imshow('frame', frame)

    if cv2.waitKey(1)&0xFF==27:
        break

cap.release() # releasing the webcam.

cv2.destroyAllWindows() # Closes all OpenCV windows.
```

Color Detection Test



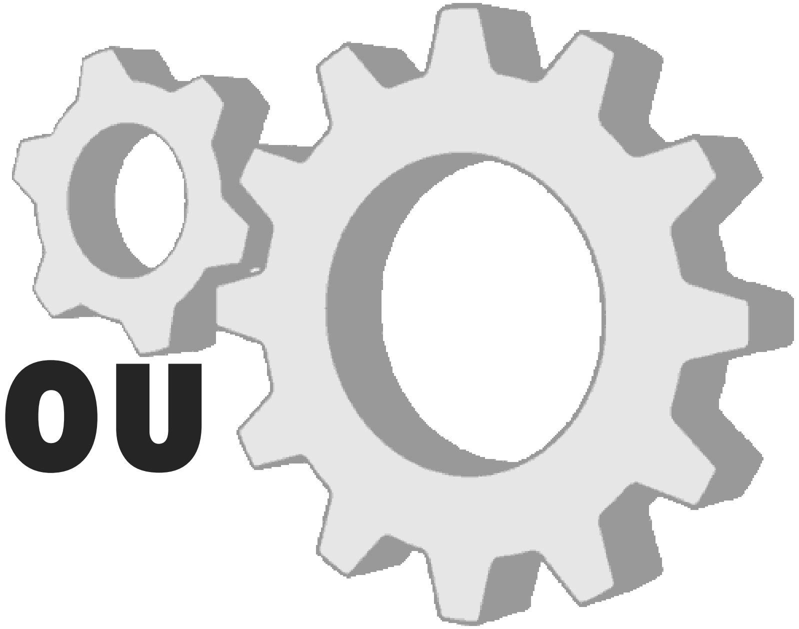
Suggestions and Improvements

- **Full Car Suspension instead of Half Car Suspension**:
  - More reliable under various driving conditions.
  - Helps in maintaining vehicle balance and stability.
- **Metal parts instead of the 3D printed parts**:
  - Strength (bear significant loads) and lives longer.
  - Precision ensuring better fit especially between the rack and pinion gear.

Suggestions and Improvements

- **Full Scale PCB rather than etched one**:
  - Less errors, easier to debug and higher quality.
  - Outsourcing PCB manufacturing saves time, allowing to focus on design and development rather than the time-consuming process of manual etching.
- **Autonomous**:
  - Monitor and adjust the suspension in real-time based on road conditions.
  - Machine learning algorithms that can learn from driving data and continuously improve the system’s ability to predict and react to road conditions.

Thank You



Active Suspension