CSE461

Introduction to Robotics Lab

Lab No. : 03
Group : 06
Section : 03
Semester : Fall_2025

Group members :

Muaz Mohammad Zimam	23101529
Abdullah Al Mahmud	22201502
Safin Ahamed Ifty	22299283
Razia Marzan Mou	22299123
Quazi Sumaya Sultana Prionti	22101454

Submitted Date: 08-11-2025
Submitted to - Mahadi Ibne Bakar & Aabrar Islam

1. Objectives:

The purpose of this lab is to learn how to interface an ultrasonic distance sensor (HC-SR04) and an LED with a Raspberry Pi microcontroller.
Through the experiment, we aim to:

- Measure distance using an ultrasonic sensor.
- Use that measured distance to automatically turn an LED on or off.

By completing this task, we will understand how the Raspberry Pi's GPIO (General Purpose Input/Output) pins can be programmed for both input (sensor) and output (LED) control using Python.

2. Equipments:

Raspberry Pi 4B board

Ultrasonic Sensor (HC-SR04)

LED (Light Emitting Diode)

Resistors (220Ω × 3)

Jumper wires

Breadboard

3. Experimental Setup:

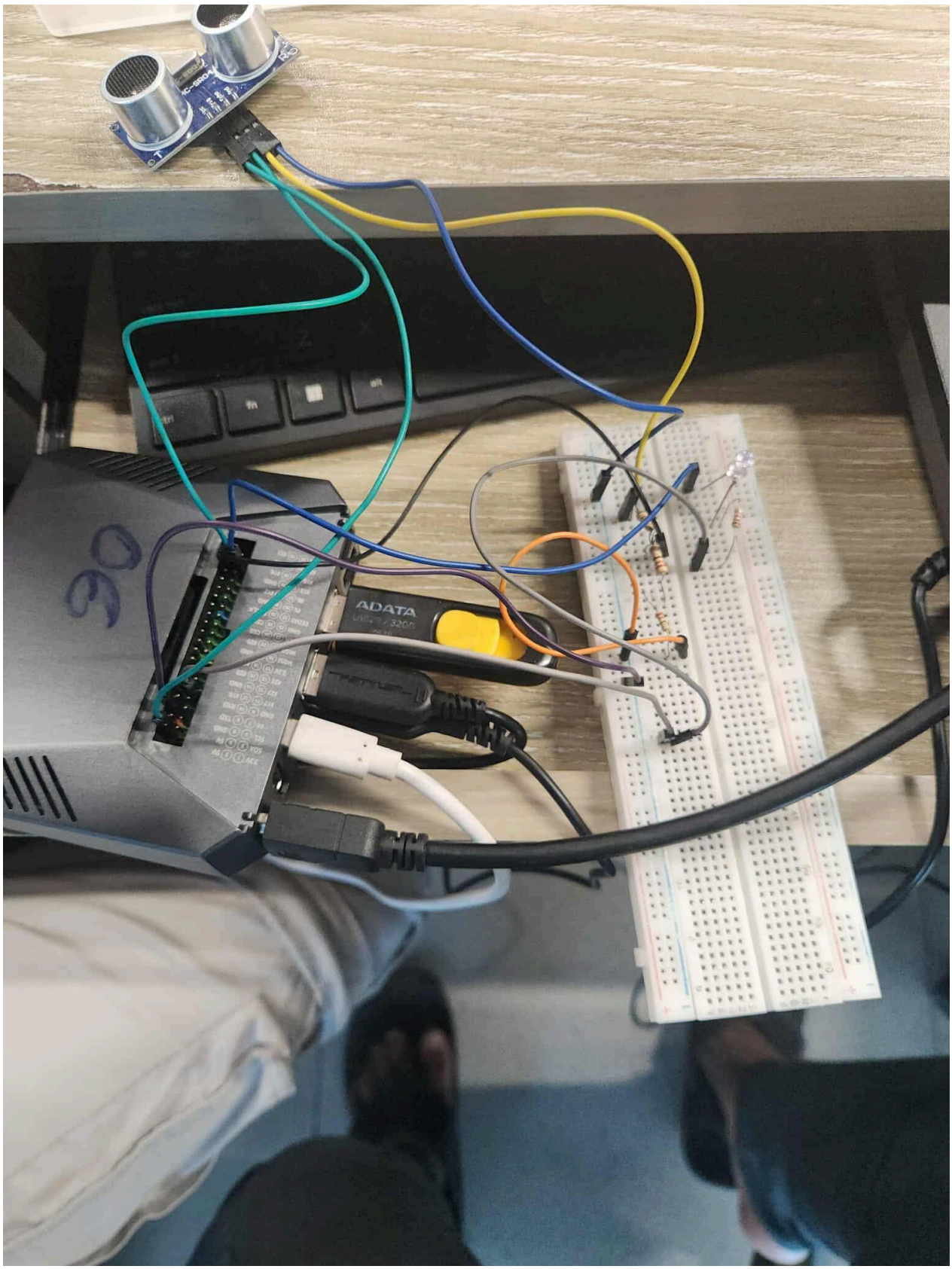
(Picture + Explanation)

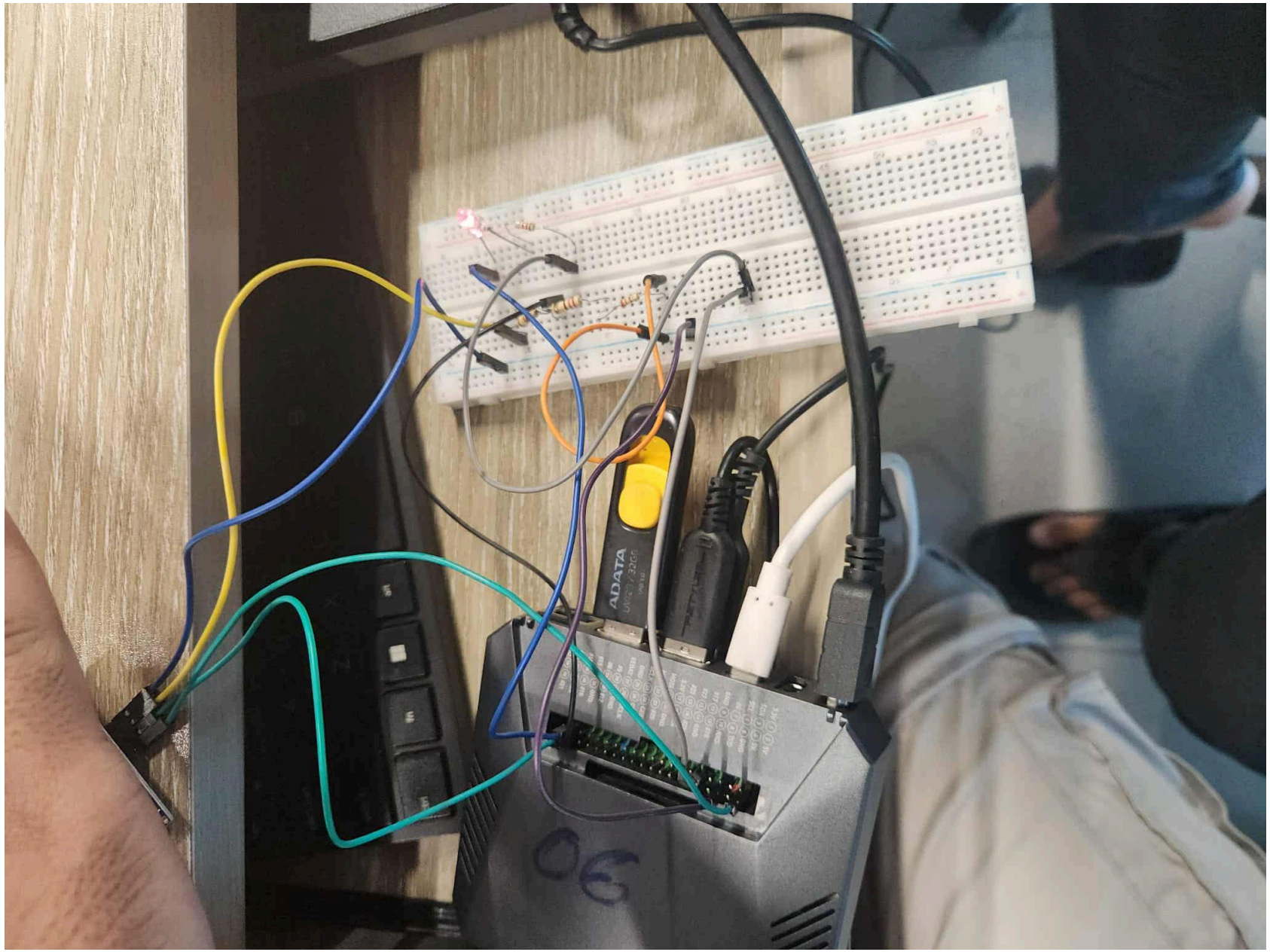
The circuit consists of connecting both the HC-SR04 ultrasonic sensor and an LED to the Raspberry Pi's GPIO pins on a breadboard. Connections

- VCC → Raspberry Pi 5 V pin
- GND → Raspberry Pi Ground
- TRIG → GPIO 21 (Output)
- ECHO → GPIO 20 (Input, through voltage divider)
- LED → GPIO 16 through a 220 Ω resistor

Because the ECHO pin of the sensor outputs 5 V and Raspberry Pi GPIO pins tolerate only 3.3 V, a voltage divider circuit is used. One 220 Ω resistor acts as R1R_1R1, and two 220 Ω resistors in series (total 440 Ω) act as R2R_2R2. Applying the voltage-divider formula gives:

which is safe for the GPIO input. The ultrasonic sensor sends out high-frequency sound waves; when they bounce back from an obstacle, the sensor calculates the time delay and converts it into distance. The LED will light up whenever this measured distance is less than 5 cm.


ADATA


ADATA

4. Code: (If Applicable)

```
#TASK-02
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
TRIG = 21 #GPIO21
ECHO = 20 #GPIO20

GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

def distance():
      GPIO.output(TRIG, False)
      time.sleep(0.5)
      GPIO.output(TRIG, True)
      time.sleep(0.00001)
      GPIO.output(TRIG, False)
      pulse_start = time.time()
      while GPIO.input(ECHO)==0:
            pulse_start = time.time()
      while GPIO.input(ECHO)==1:
            pulse_end = time.time()
      pulse_duration = pulse_end - pulse_start
      distance = pulse_duration * ((343*100)/2)
      distance = round(distance, 2)

      return distance

while True:
    print(distance())

GPIO.cleanup()


#LAB TASK

import RPi.GPIO as GPIO
import time
From gpiozero import LED
GPIO.setmode(GPIO.BCM)
TRIG = 21 #GPIO21
ECHO = 20 #GPIO20
led=LED(26)
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

def distance():
      GPIO.output(TRIG, False)
      time.sleep(0.5)
```

```
      GPIO.output(TRIG, True)
      time.sleep(0.00001)
      GPIO.output(TRIG, False)
      pulse_start = time.time()
      while GPIO.input(ECHO)==0:
            pulse_start = time.time()
      while GPIO.input(ECHO)==1:
            pulse_end = time.time()
      pulse_duration = pulse_end - pulse_start
      distance = pulse_duration * ((343*100)/2)
      distance = round(distance, 2)

      return distance

while True:
    print(distance())
    d=distance()
    If d<5:
      led.on()
    Else:
      led.off()
GPIO.cleanup()
```

5. Results (Output of the experiment):

When the program runs, the measured distance is continuously displayed in the terminal.
Examples:

Distance: 13.42 cm

Distance: 8.56 cm

Distance: 4.98 cm

When the object is farther than 5 cm, the LED remains off. When an object comes closer than 5 cm, the LED turns on automatically. As the object moves away, the LED turns off again. This confirms the Raspberry Pi successfully reads the sensor data and controls the LED in real time.

6. Discussions/Answers:

(May contain conclusions or learnings from the result along with problems faced and solving methodology.)

This combined experiment demonstrates how sensors and actuators can interact through the Raspberry Pi. The ultrasonic sensor measures distance using the **time-of-flight** of sound waves, while the LED provides a visible output based on that reading.

Concepts Learned

- How to configure Raspberry Pi GPIO pins as inputs and outputs.
- The working principle of the HC-SR04 sensor and how to compute distance from pulse duration.
- How to protect GPIO pins using a voltage-divider circuit.
- Basic automation logic using conditional statements in Python.

Challenges Faced

- At first, the distance readings fluctuated because of loose jumper connections.
- The LED did not respond until the correct GPIO pin was defined in code.
- The 5 V Echo signal required proper voltage reduction; after adding the resistor divider, the readings stabilized.

Learning Outcomes

- Understanding how microcontrollers interact with the physical world through sensors.
- Implementing decision-making logic to trigger actions automatically.
- Recognizing the importance of precise timing and correct wiring in embedded systems.