CSE461 LAB
Introduction to Robotics
LAB 2

Topic: Measuring distance using ultrasonic sensor.

Submitted By:
Group 8
Tahsin Ashrafee Susmit (20301088)
Isratul Hasan (20301072)
Suraiya Binte Akbar (20301007)
Lamia Khan Shoily (20301085)
Samia Abdullah (20101595)

Description:

The ultrasonic distance sensor, with a remarkable precision of up to 3mm, is a flexible and affordable option for non-contact distance measurements between 2 and 400 cm. This sensor, which consists of an ultrasonic transmitter, receiver, and control circuit, is made to be straightforward and simple to use. It's easy setup and four pins (VCC (Power), Trig (Trigger), Echo (Receive), and GND (Ground)) make it a great option for a wide range of range-finding applications. In order to reduce irregular or "bouncy" data, the sensor also includes extra control circuitry, guaranteeing dependable performance in a variety of applications. To effectively utilize the ultrasonic distance sensor with a Raspberry Pi, we grasp its fundamental principles of emitting and receiving ultrasonic pulses for distance measurement. In order to precisely measure distance using the sensor, we attempted to compose Python code hands-on. We also test the sensor in a variety of situations to evaluate its performance in order to comprehend the significance of precision in distance measuring.

Components:

1. Raspberry Pi
2. Ultrasonic Sensor (HC-SR04)
3. Breadboard
4. Jumper Wires
5. Five resistors with a 220 ohm resistance
5. MicroSD Card
6. USB Cable
7. Monitor
8. Keyboard
9. Mouse

Circuit diagram:

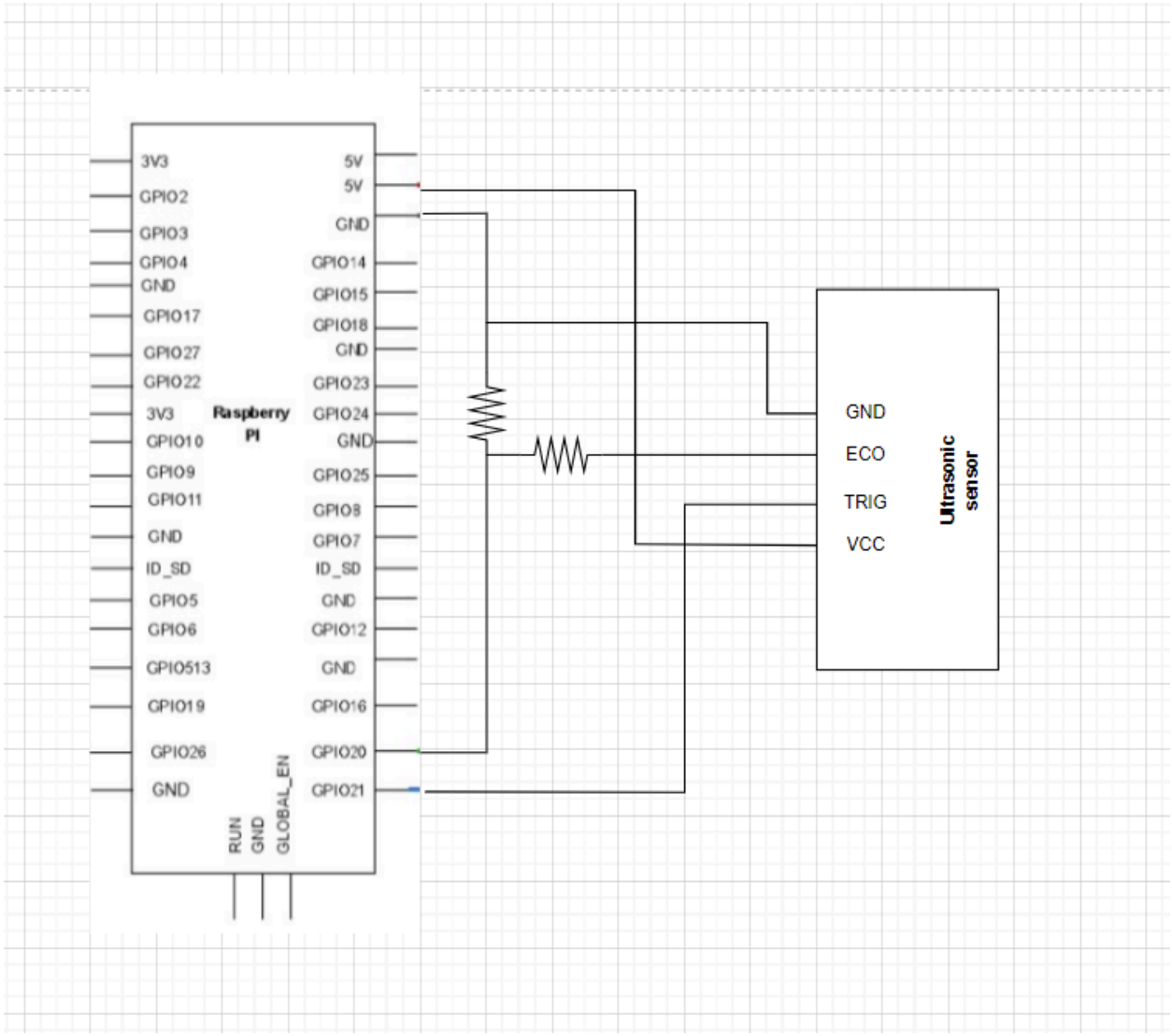
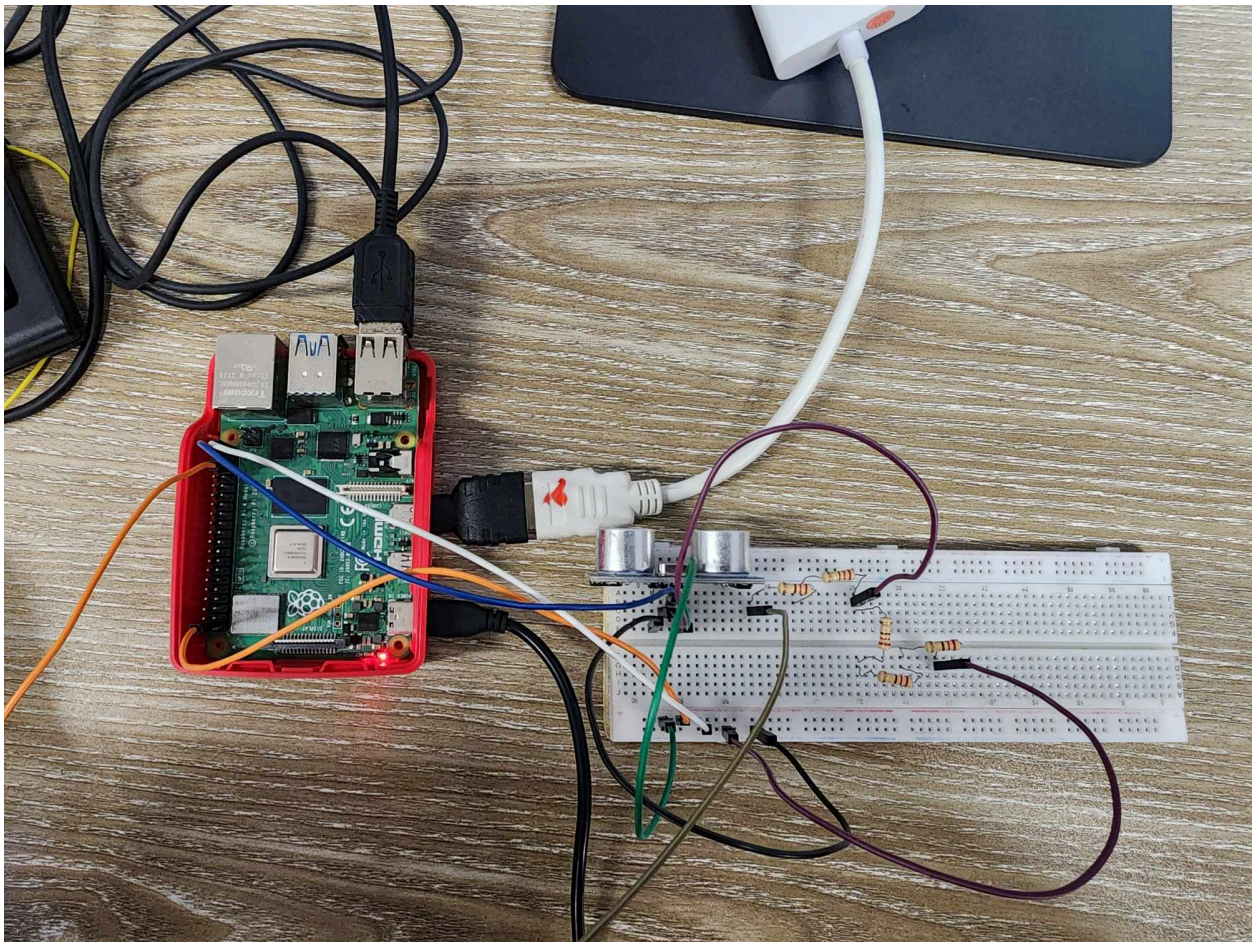
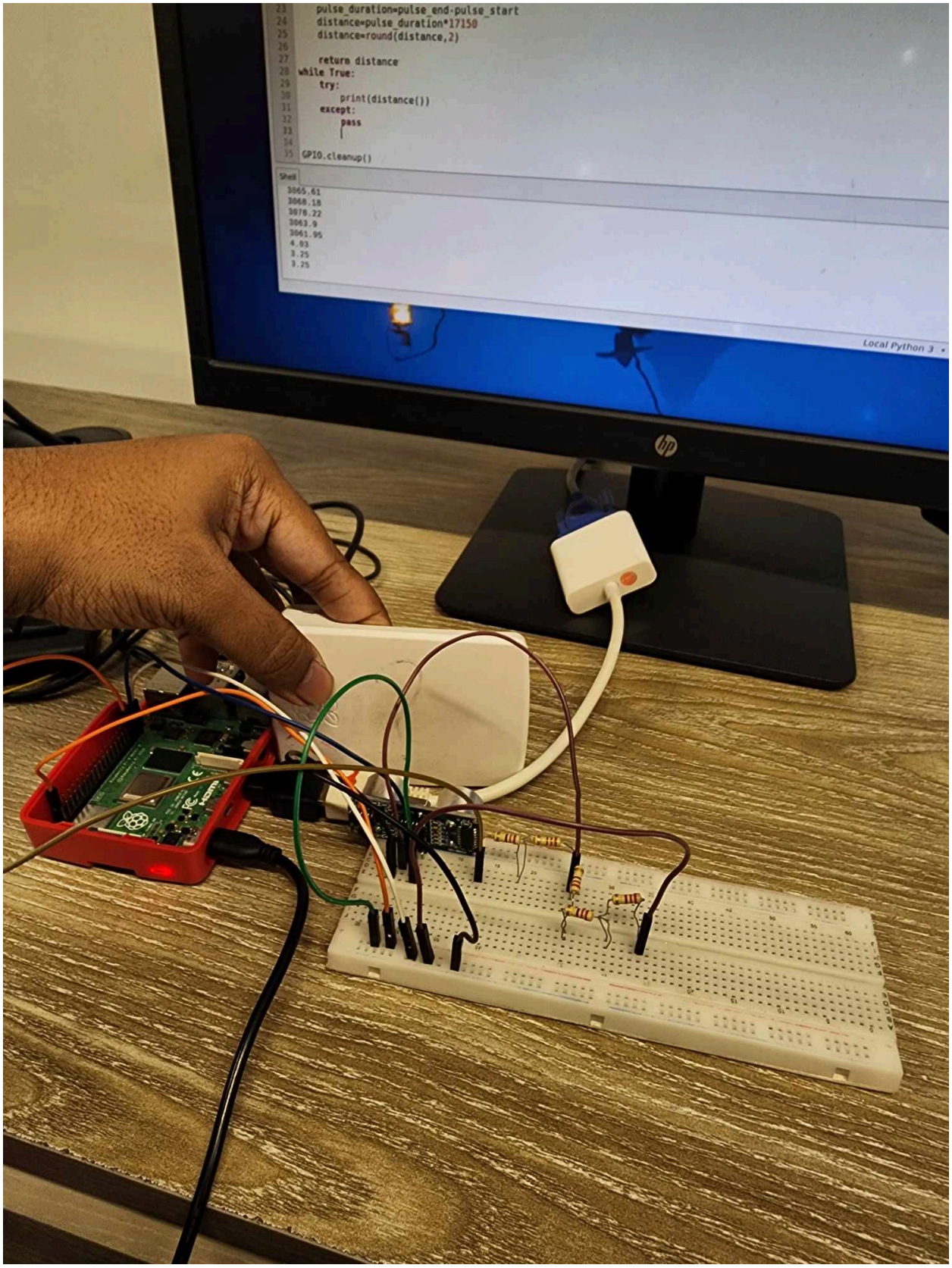


Figure: Raspberry Pi connection with an ultrasonic sensor and resistors.

Circuit setup:




pulse_duration=pulse_end-pulse_start
distance=pulse_duration*17150
distance=round(distance,2)
return distance
while True:
try:
print(distance())
except:
pass
GPIO.cleanup()
Shell
3065.61
3068.18
3078.22
3063.9
3061.95
4.03
3.25
3.25
Local Python 3

Code:

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

TRIG = 21
ECHO = 20

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 * 17150
    distance = round(distance, 2)

    return distance

print(distance())
GPIO.cleanup()
```

Discussion:

We used an HC-SR04 sensor connected to the Raspberry Pi via a breadboard in the experiment centered on measuring distance using an ultrasonic sensor and Raspberry Pi. To ensure secure and reliable connections, we also used resistors and jumper wires. By utilising the Raspberry Pi 4's processing capacity and networking capabilities, the configuration offered an adaptable

framework for combining and evaluating sensor data. The project demonstrated how easy and efficient it is to build a useful distance measurement system using common parts like resistors, jumper wires and breadboards. The addition of resistors improved the dependability and precision of distance measurements by shielding sensitive parts from harm and reducing unwanted noise.

Question/Answer:

1. **Why are the resistors used?**

Ans: In distance measurement circuits employing ultrasonic sensors, resistors are utilised primarily for voltage scaling and signal conditioning purposes. The use of two resistors in a voltage divider configuration allows for scaling down the output voltage of the ultrasonic sensor to a level suitable for interfacing with microcontrollers or other processing units. Additionally, employing two resistors enables fine-tuning of the sensor's sensitivity and range, facilitating calibration for specific application requirements. Overall, the inclusion of resistors in the circuit serves to optimize the sensor's performance and ensure accurate distance measurements.