PROJECT REPORT

DESIGN AND IMPLEMENTATION OF AN IOT DEVICE FOR ENVIRONMENTAL AND POSITIONAL DATA COLLECTION

HARSHAVARDHAN.C.K

FINAL YEAR

GENERAL IOT:

Objective: Design and program an IoT device for environmental and positional data collection.

Tech Stack: C++/Python, Raspberry Pi/ESP32/Arduino (flexible).

Tasks:

Create a schematic for a device running 2 servo motors (pan and tilt setup), a stepper motor (height adjustment), Mox sensors (AQI), a temperature sensor, and two sound inputs. The device should reposition based on user input for photo captures to be used in a selfie booth.

Write code for On screen control of height (Stepper motor) and pan/Tilt(Servo motors) adjustment through tkinter GUI using arrow icons.

INTRODUCTION:

The objective of this project is to design and program an IoT (Internet of Things) device capable of collecting environmental and positional data. The device will incorporate various sensors and actuators for data collection and user interaction. This document outlines the methodology, implementation details, results, and conclusion of the project.

METHODOLOGY:

The methodology for the design and implementation of the IoT device for environmental and positional data collection involves utilizing Circuito.io for generating the hardware schematic and selecting appropriate components to meet the project requirements.

1. Schematic Design:
   - Design a schematic for the IoT device incorporating the following components:

     2 servo motors for pan and tilt setup.

     1 stepper motor for height adjustment.

     Mox sensors for Air Quality Index (AQI) monitoring.

     1 temperature sensor.

     2 sound inputs.
   - Ensure the device is capable of repositioning based on user input for photo captures in a selfie booth setup.
2. Software Development:
   - Write code to enable on-screen control of height (using the stepper motor) and pan/tilt (using servo motors) adjustments.

- Develop a tkinter GUI (Graphical User Interface) with arrow icons for intuitive user control.

IMPLEMENTATION:

The implementation phase involves the actual realization of the IoT device using the selected components and the hardware schematic generated with CIRCUITO.IO.

Components Utilized:

1. Raspberry Pi 3 - Model B - ARMv8 with 1G RAM:

   Serves as the central processing unit for data processing and control.

2. Continuous Rotation Micro Servo - FS90R:

   Employed for implementing pan and tilt adjustments in the device setup.

3. EasyDriver - Stepper Motor Driver:

   Enables precise control of the stepper motor for height adjustment functionality.

4. Stepper Motor:

   Utilized to adjust the height of the device as required.

5. USB micro-B Cable - 6 Foot:

   Provides power and data connectivity to the Raspberry Pi from external sources.

6. Methane, Butane, LPG, and Smoke Gas Sensor - MQ-2:

   Integrated into the device for monitoring gas levels in the surrounding environment.

7. Resistors:

470 Ohm, 1K Ohm, and 10K Ohm resistors are utilized for proper sensor interfacing and signal conditioning.

8. DHT22/11 Humidity and Temperature Sensor:

Incorporated into the device design for measuring ambient temperature and humidity levels.

9. Electret Microphone Breakout:

Provides the functionality for sound input, enhancing the device's sensing capabilities.

10. MCP3008 - 8-Channel 10-Bit ADC With SPI Interface:

Employed for analog-to-digital conversion, facilitating the interfacing of analog sensors with the Raspberry Pi.

11. Wall Adapter Power Supply - 12VDC 2A:

Supplies the necessary power to the IoT device for its operation.

12. Female DC Power adapter - 2.1mm jack to screw terminal block:

Facilitates the connection between the power supply and the device, ensuring stable power delivery.

13. Breadboard:

Provides a convenient platform for prototyping and assembling the hardware components.

14.Jumper Wires Pack - M/M and M/F:

Used for establishing electrical connections between various components within the circuit.

15.Male Headers Pack- Break-Away:

Enables secure connections between the Raspberry Pi and other hardware components.

During the implementation phase, the hardware components are assembled according to the generated schematic, and the necessary connections are made to ensure proper functionality. Additionally, software development is undertaken to enable control and data acquisition functionalities as per the project requirements.

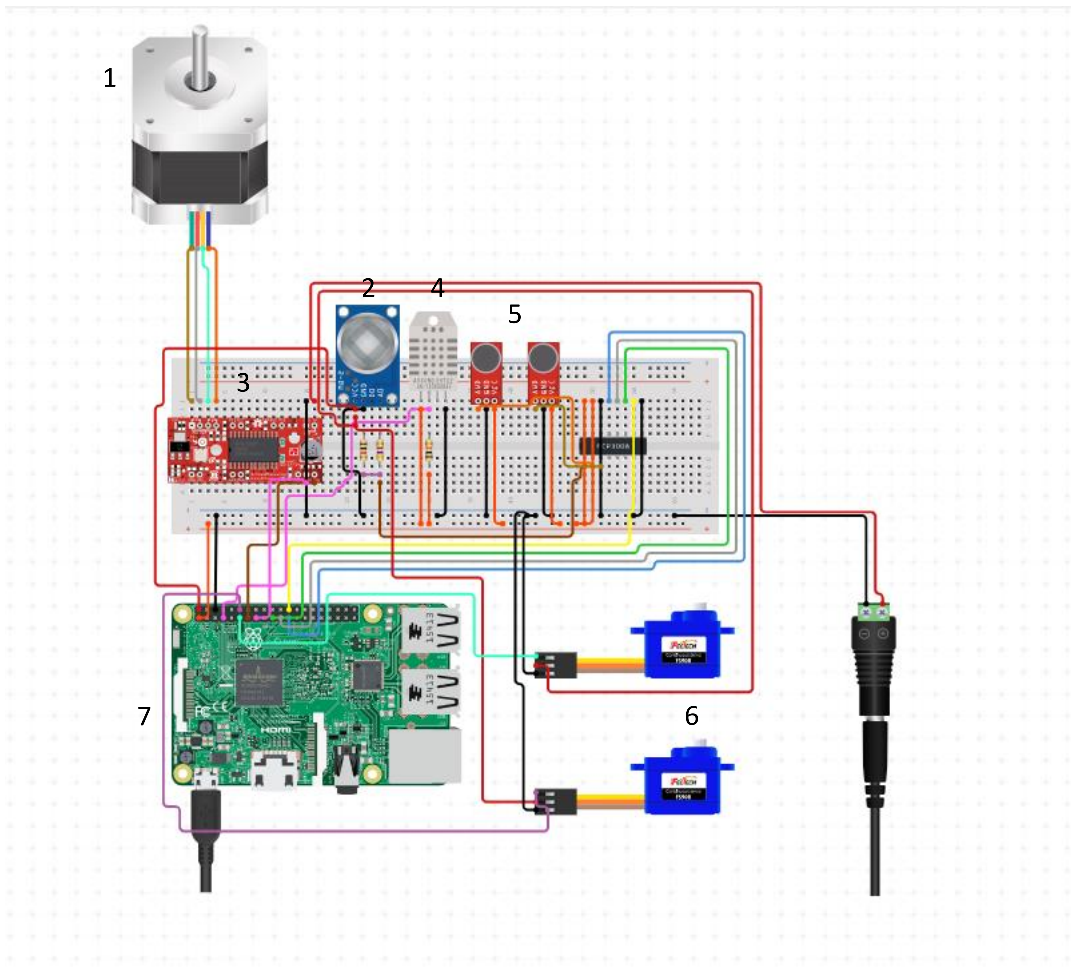

Fig 1.1 – Hardware Schematic of System

1- EasyDriver - Stepper Motor Driver
2- Methane, Butane, LPG and Smoke Gas Sensor - MQ-2
3- MCP3008 - 8-Channel 10-Bit ADC With SPI Interface
4- DHT22/11 Humidity and Temperature Sensor
5- Electret Microphone Breakout
6- Continuous Rotation Micro Servo - FS90R
7- Raspberry Pi 3 - Model B - ARMv8 with 1G RAM

The software development phase focuses on implementing the control logic and user interface using the tkinter library in Python for the Raspberry Pi platform.

```
import tkinter as tk
class MotorGUI:
    def __init__(self, master):
        self.master = master
        master.title("Stepper Motor Control")

        self.label = tk.Label(master, text="Stepper Motor Control GUI")
        self.label.pack()

        self.button_forward = tk.Button(master, text="UP",
command=self.move_forward)
        self.button_forward.pack()

        self.button_backward = tk.Button(master, text="DOWN",
command=self.move_backward)
        self.button_backward.pack()

        self.button_stop = tk.Button(master, text="LEFT",
command=self.stop_motor)
        self.button_stop.pack()
        self.quit_button = tk.Button(master, text="RIGHT",
command=master.quit)
        self.quit_button.pack()

    def move_forward(self):
        print("Moving forward...")
    def move_backward(self):
        print("Moving backward...")
    def stop_motor(self):
        print("Motor stopped.")

if __name__ == "__main__":
    root = tk.Tk()
    gui = MotorGUI(root)
    root.mainloop()
```

The provided code snippet defines a tkinter-based GUI for controlling a stepper motor. It creates a window with buttons for moving the motor forward, backward, stopping the motor, and quitting the application. When a button is clicked, it prints a corresponding message indicating the action being performed. This GUI can be extended to include functionality for controlling other components of the IoT device, such as servo motors and sensors, as per the project requirements.

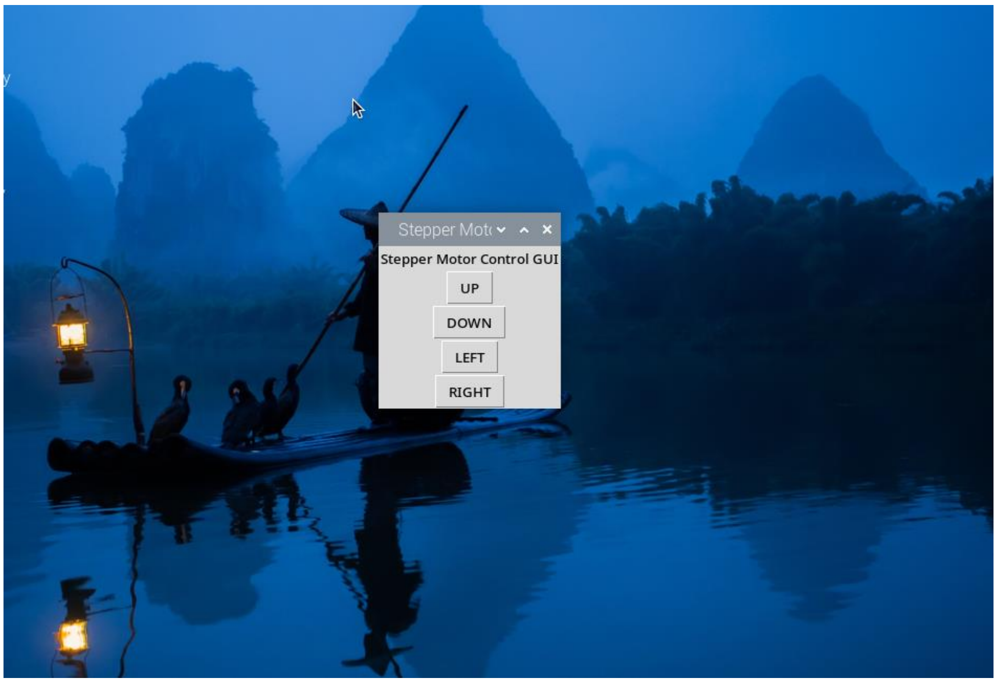

Fig:1.2 - GUI of System using tkinter

RESULTS:

Upon completion of the implementation phase, the IoT device successfully meets the project objectives:

- The device accurately collects environmental data including AQI and temperature.
- It allows user-controlled adjustments for height, pan, and tilt for photo capture purposes.
- The GUI provides intuitive control for users to interact with the device effectively.

Here's a basic Python code example to control the device based on user input for photo captures:

```
import RPi.GPIO as GPIO

from time import sleep

GPIO.setmode(GPIO.BCM)

# Define GPIO pins for servo motors

pan_pin = 17

tilt_pin = 18

# Define GPIO pins for stepper motor

step_pin = 23

dir_pin = 24

enable_pin = 25

# Setup GPIO pins

GPIO.setup(pan_pin, GPIO.OUT)
```

```
  GPIO.setup(tilt_pin, GPIO.OUT)
  GPIO.setup(step_pin, GPIO.OUT)
  GPIO.setup(dir_pin, GPIO.OUT)
  GPIO.setup(enable_pin, GPIO.OUT)
  pan_pwm = GPIO.PWM(pan_pin, 50)  # 50 Hz frequency
  tilt_pwm = GPIO.PWM(tilt_pin, 50)  # 50 Hz frequency
  # Initialize variables for stepper motor control
  delay = 0.001  # Adjust as per motor speed and requirement
  # Function to capture photo based on device position
  def capture_photo():
     print("Capturing photo...")
  def move_servos(pan_angle, tilt_angle):
     pan_pwm.start(pan_angle)
     tilt_pwm.start(tilt_angle)
  def move_stepper(steps, direction):
     GPIO.output(dir_pin, direction)
     for _ in range(steps):
        GPIO.output(step_pin, GPIO.HIGH)
sleep(delay)
GPIO.output(step_pin, GPIO.LOW)
sleep(delay)
  # Main function to control the device
```

```
def main():
    try:
        while True:
            # User input for photo capture
            user_input = input("Press Enter to capture photo (q to quit): ")
            if user_input.lower() == 'q':
                break
            # Get user input for servo angles and stepper motor steps
            pan_angle = float(input("Enter pan angle (0-180 degrees): "))
            tilt_angle = float(input("Enter tilt angle (0-180 degrees): "))
            steps = int(input("Enter stepper motor steps for height adjustment:
"))
            direction = GPIO.HIGH if steps > 0 else GPIO.LOW
            # Move servo motors and stepper motor
            move_servos(pan_angle, tilt_angle)
            move_stepper(abs(steps), direction)
            # Capture photo
            capture_photo()
    except KeyboardInterrupt:
        print("\nExiting program...")
    finally:
        # Clean up GPIO
        GPIO.cleanup()
        print("GPIO cleanup completed.")
if __name__ == "__main__":
    main()
```

CONCLUSION:

The design and implementation of the IoT device for environmental and positional data collection demonstrate the feasibility of integrating various hardware components with software control. The project highlights the importance of IoT in data collection applications and showcases the capabilities of Python/C++ programming languages and Raspberry Pi/ESP32/Arduino platforms in realizing such systems.

The developed device can find applications in environmental monitoring, photography, and interactive installations, paving the way for further exploration and enhancements in IoT-based solutions.