Prototype III and Customer Feedback

Vidvath Tanjore

Andrew Walisser

Connor Rennie

Wesley Savage

Andrew Bettin

University of Ottawa

GNG 1103 - Introduction to Engineering Design

Mr. David A. Knox

Sunday, March 23rd, 2025

Table of Contents

Introduction......1

Feedback......2

Table 1: Feedback Provided by the TA and PMs...... 2.1

Updated Design of the Collection and Storage mechanism...... 3

Objectives......4

Analysis of Critical Components and Systems......5

Prototyping Test Plan...... 6

Results of the Test on the Code......7

Test of the circuit......8

Test of the plastic parts......9

Justification for choosing this prototype as the final design......10

Conclusion......11

Introduction

The purpose of this report is to present the final prototype, which will be tested and demonstrated to the client on Design Day. As outlined in previous reports, the primary objective of this project is to design a device that enables the EMED team at Canadian Nuclear Laboratories to safely collect a 30-80 milligram sample of metal from inside a fuel channel supplying uranium fuel to a CANDU reactor.

Up to the date of this report, the project needs were identified and categorized based on their importance, and design criteria were established to address each of these needs, leading to the creation of two initial prototypes. A comprehensive list of materials required for the product's construction was developed, including detailed cost estimates. Additionally, a series of tests were designed and conducted to evaluate the performance of each prototype.

The purpose of this report is to incorporate the feedback provided by the client, teaching assistant (TA), and project managers (PMs), as well as insights gained from the previous prototype tests.

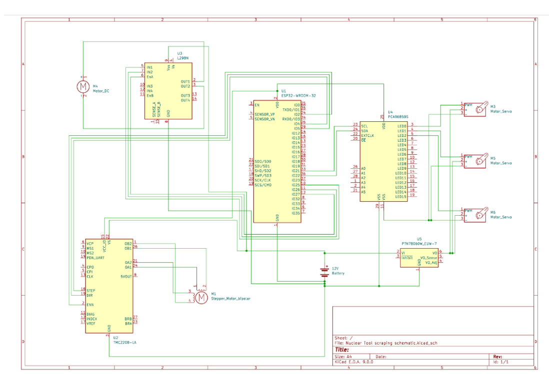
Based on this feedback, necessary design modifications have been implemented to produce the final prototype.

Furthermore, this report outlines the test plan for Prototype III, describing the specific tests to be conducted and presenting the results. The goal is to ensure that the final prototype meets all established design criteria and project objectives, leading to a successful demonstration on Design Day.

Feedback

The table below outlines the feedback provided by the TA and PMs regarding Prototype II:

Table 1: Feedback Provided by the TA and PMs.

Part of the Design.	**Description of the Part.**	**Feedback provided regarding the part.**
Power supply.	The electricity required for the functioning of the various components in the device is provided by a 11 V LiPo battery, through the circuit given below 	The teaching assistant (TA) and project managers (PMs) approved the circuit and the use of the listed components without raising any concerns.

<table>
<tr>
<td></td>
<td>

Figure 1: Diagram representing the circuit connecting all the electrical components of the device.

In addition to the various motors and microprocessors used in the device, the circuit shown in Figure 6 consists of the following components:

1. 11 V LiPo battery as the power source.
2. 6V buck convertor to act as a step down module for the servo motors.
3. 1 uF and 100 uF capacitors to store energy and prevent voltage spikes.
4. L298N driver for the DC motor.
5. PCA9865 driver for the servo motor.

All of these electrical components will be connected together using wires with connectors at the ends.

</td>
<td></td>
</tr>
<tr>
<td>Sample collection and storage.</td>
<td>

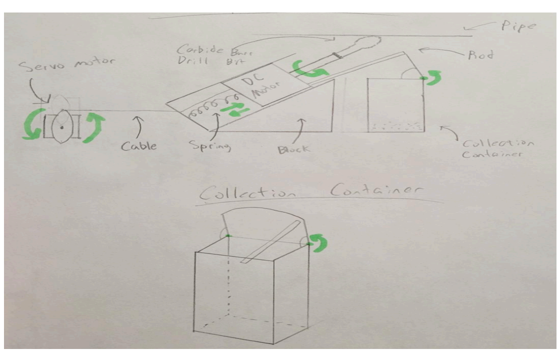


Figure 2: Collection and scraping mechanism.

</td>
<td>A concern was raised about the spring's ability to support the DC motor and keep it in a fixed position during the scraping process. If the body of the DC motor moves due to vibrations, it would prevent consistent contact between the drill bit and the surface of the tube, reducing the</td>
</tr>
</table>

	The concept represented in Figure 2 works in the following manner: The servo motor actuates, releasing a spring that allows the drill bit to come into contact with the pipe while the DC motor begins spinning allowing the drill bit to scrape off a sample. The rod between the DC motor and the collection system allows it to be opened as the drill bit extends towards the tube and closes when the drill bit is retracted.	efficiency of the scraping process. Additionally, another concern was raised regarding the use of the cable to adjust the spring's position. The issue is that the spring has a high coefficient, making it difficult to compress or expand. The need for an excessive amount of force can cause the cable to snap, rendering the mechanism useless.

Note: The design parts mentioned in Table 1 are those that received feedback. It is important to note that the designs for the communication, drivetrain, and fail-safe components were approved in previous deliverables. As a result, no changes will be made to these components in Prototype III.

Updated Design of the Collection and Storage mechanism

Based on the feedback provided, the following changes were made to the design of the collection and scraping mechanism:

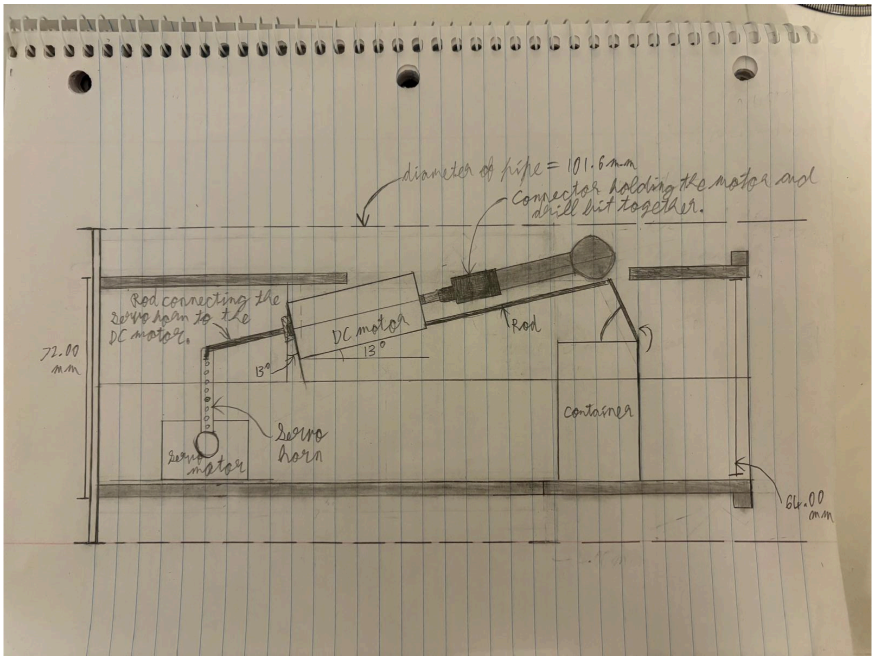


Figure 3: Updated design of the scraping and collection system.

As shown in Figure 3, the spring has been replaced with a more stable rod mechanism. The front end of the rod is connected to the protruding portion of the DC motor, while the back end is attached to one of the holes in the servo horn. As the servo rotates clockwise, the rod pushes the DC motor and drill bit against the surface of the tube, maintaining constant contact without any back-and-forth movement, which would have occurred with the spring.

It is also important to note that the rod components, particularly the points of contact with the DC motor and the servo horn, have been carefully dimensioned. This ensures sufficient friction at these connections, preventing the components from falling apart during operation.

Additionally, no changes have been made to the collection system, as no negative feedback was provided regarding its performance.

A new addition made to this mechanism is the part used to connect the shaft of the DC motor and the shank of the drill bit.

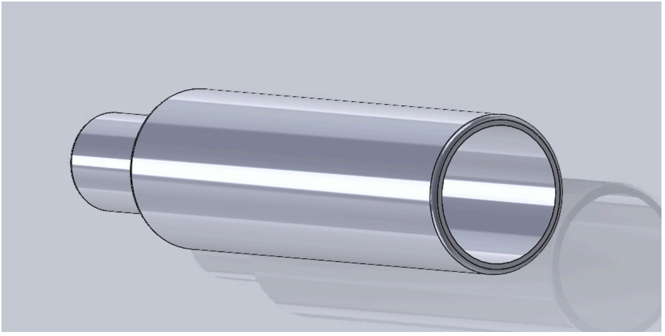


Figure 4: Front end of the DC motor - shank mount.

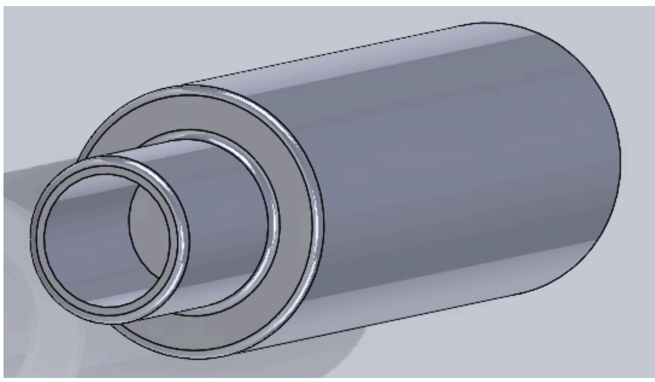


Figure 5: Rear end of the DC motor - shank mount.

As shown in Figures 4 and 5, the part features holes on either side that are large enough to accommodate the diameters of both the shank and the shaft. Additionally, the hole designed for the shank is long enough to provide support and prevent it from bending downwards.

Updated Bill of Materials

Please click on the PDF file attached to this submission to access the updated bill of materials. It is important to note that in this update, no change has been made to the list of parts. However, the total cost has been reduced by purchasing the parts from websites that were offering the products at lower prices. These websites were not chosen initially due to their poor reputation and due to concerns regarding the timely arrival of the parts.

Objectives

The objective of this prototype is to conduct physical tests on various components, including the movement, collection and scraping mechanisms, code, electronics, and clamping system. Up to this point, most of the tests were performed using CAD simulations for the movement and clamping mechanisms, as well as tools like TinkerCAD to analyze the functionality of the code and other electronic components.

These virtual tests were necessary because the bill of materials had not yet been approved, preventing the purchase of the required parts. Now that the parts are available, physical testing can be conducted to evaluate the prototype's performance in real-world conditions.

It is expected that the results of the physical tests will closely align with those obtained from the virtual simulations. Any significant discrepancies between the two sets of results will be identified, and corrected to ensure that the device works as it should.

Analysis of Critical Components and Systems

The critical components and systems of this prototype are similar to those in Prototype II. However, certain new components have been added as critical. Below is an updated list of the critical components identified in this project:

1. **Clamping Mechanism** – The clamping mechanism is essential because, without a secure grip on the tube, the rest of the device will not be able to function properly.

2. **Rack and Pinion System** – This system must operate smoothly to enable forward and backward movement. In addition to ensuring smooth operation, the speed of the system is also critical, as it directly affects the efficiency of the device.

3. **Scraping System** – The scraping mechanism must efficiently collect between 30 and 80 grams of the sample. To achieve this, the scraping tool and motor need to make contact with the tube's surface during sample collection and retract when not in use.

In this prototype, the rod responsible for moving the DC motor and drill bit must support the weight of the motor-drill bit mechanism. Additionally, it must keep the mechanism in a fixed position to ensure an efficient scraping process. For the system to function properly, the ends of the rod must remain securely attached to both the servo horn and the DC motor.

4. **Collection System** – The collection system must gather all scraped samples while preventing contamination by ensuring that no material falls into other parts of the device. Additionally, the system must be effective in both horizontal and vertical orientations.

5. **Motors and Sensors** – These components must function efficiently to perform their respective roles, ensuring the overall operation of the device. Additionally, in this prototype, the motors must produce enough power to complete their respective tasks.

6. **Plastic Parts:** Most of the parts in this prototype are made out of PLA plastic. These parts must be strong enough to withstand any forces applied on them.

Prototyping Test Plan

Table 2: Table representing the test plan for prototype II.

Part being tested	How will it be tested ?	Type of test	What is being tested ?
Movement System (Rack and Pinion)	In this prototype, the test will be a physical one in which the device will be operated within a 15 ft long pipe, and the distance traveled per inchworm	The physical test will be a focused, high-fidelity test. The main body of the device will be tested along with all the other components such as the motor drivers, ESP-32	The physical test will demonstrate the rack and pinion system's ability to move the device in and out of the tube without

	movement will be measured. Through multiple trials, the range of movement per cycle will be determined, allowing for the calculation of the number of movements required to travel a distance of 15 feet.	microprocessors, etc.	any difficulties. Additionally, it will also serve as a means of testing the ESP-32 microprocessors' ability to communicate with each other to measure the total distance traveled by the device. It is essential to ensure that the device moves a predetermined length consistently and reliably across multiple executions. The distance traveled will be recorded in meters.
Clamping system.	The system will be put through a physical test in which the device will be placed inside the tube to allow the clamps to extend up to its diameter (101.6 mm) and grip its surface.	The physical test will be a high-fidelity test which will involve a test on all the components of the ends of the device. This includes the body part, the four clamping legs, and the servo motor that operates the system.	The goal of the physical test is to test the extension and the gripping of the clamps, which will then allow the rest of the device to move.

Scraping System	To evaluate the scraping system, the device will be placed inside the tube and the scraping tool will be actuated by allowing the servo motor to rotate clockwise, thus resulting in the movement of the drill bit to the surface of the tube. The test will involve timing the duration required to remove 30-80 mg of material. This trial will be repeated multiple times, and the results will be averaged to determine the precise operating duration for the scraper. This test will be conducted in horizontal and vertical orientations of the tube.	This test will be a focused, high-fidelity evaluation. The scraping arm will be fully constructed. Additionally, the tube in which this device will be tested will be made out of metal.	This evaluation will measure the amount of material removed from the test surface. Additionally, the rate of material removal will be recorded to determine an average. Data will be collected in milligrams of material scraped, total scraping time in seconds, and material removal rate in milligrams per second.
Container	The primary requirement for testing	This test will be a focused, low-fidelity evaluation. It is expected	This test will measure the amount of material that falls into the collection

	the container is to ensure it captures the material and successfully closes its lid. To conduct this test, the container will be positioned beneath the scraper during operation to verify that the majority of the sample is collected. Following this, the functionality of the lid will be confirmed to ensure proper closure. This test will be conducted in horizontal and vertical orientations of the tube.	to coincide with the testing of the scraper tool, as both components are functionally connected. The objective is to achieve a high collection rate of the material removed during the scraping process.	container. The objective is to ensure that the majority, if not all, of the scraped material is captured within the container rather than dispersing into the surrounding tube. The evaluation will involve quantifying the material collected in the container versus the amount that falls into the tube and calculating the ratio between these two values in milligrams. The idea of conducting the test in horizontal and vertical orientations of the device is to ensure complete collection of the sample and to prevent it from being deposited to other parts of the device. Special emphasis will be given to this part of the test as the client has highlighted a discrepancy in this area, and it is

			expected that the team does a good job of correcting it.
Code and electronic components.	The code used to operate the microprocessor, motors, and sensors will be physically tested alongside these components to ensure proper functionality.	The physical test will be a high-fidelity one involving the code and other electronic components such as the microprocessor, motors, and sensors.	The goal is to ensure that the DC motor, stepper motor, and servo motors operate when they should. In order for this to happen, the written code must be correctly interpreted by the microprocessor and motor drivers, which work together to supply electricity from the battery to the motor. All of these components will be tested together. Additionally, the ESP-32 inside the device must send signals to the ESP-32 placed at the entrance of the tube. These signals will then be interpreted as the distance travelled by the device inside the tube. This system will also be tested.

Circuit	The circuit connecting all the electronic components such as the motors, drivers, microprocessors, capacitors, etc, must be tested to ensure a stable flow of current to all the components, with no loose connections.	This will be a high - fidelity test involving all the electronic components in this device.	The goal of this test is to ensure that all the electronic components in the device receive the right amount of electricity to function properly. Additionally, this test will ensure that there are no loose connections among components, and that there are no voltage drops (which lead to a reduction in the component's ability to function as it should).
Plastic Components	The plastic components in this device will be tested by applying tension and compression forces on them.	This test will be a high-fidelity test as all the plastic parts will be subjected to tests with the same amount of force that they will experience in real-life conditions.	The goal of this test is to verify that the plastic parts can withstand the forces applied to them without breaking, as any breakage would render the device unusable.

Results of the Test on the Code

Part A: Test of the code's ability to control the motors.

The following code is used to control the DC motor that operates the drill bit and the servo motor that operates the retraction and extension of the drill bit. This code was used on the ESP-32 microprocessor and the drivers of the motors.

```
from machine import Pin, PWM
 import network
 import socket
 import time

# Configuration
 SSID = 'Wified'
 PASSWORD = 'justin69'

# Define GPIO pins for motor control
 IN1 = Pin(12, Pin.OUT)  # IN1 on L298N
 IN2 = Pin(14, Pin.OUT)  # IN2 on L298N
 ENA = PWM(Pin(13), freq=1000)  # ENA on L298N for speed control (PWM)

# Motor states
 MOTOR_STATES = {
        "forward": False,
        "backward": False,
        "speed": 512,  # Default speed (50%)
        "start_time": None  # Track when the motor started
 }

# Function to rotate the motor forward
 def motor_forward(speed):
        IN1.value(1)
        IN2.value(0)
        ENA.duty(speed)
        MOTOR_STATES["forward"] = True
```

```
        MOTOR_STATES["backward"] = False
        MOTOR_STATES["start_time"] = time.time()  # Start the timer

# Function to rotate the motor backward
def motor_backward(speed):
        IN1.value(0)
        IN2.value(1)
        ENA.duty(speed)
        MOTOR_STATES["forward"] = False
        MOTOR_STATES["backward"] = True
        MOTOR_STATES["start_time"] = time.time()  # Start the timer

# Function to stop the motor
def motor_stop():
        IN1.value(0)
        IN2.value(0)
        ENA.duty(0)
        MOTOR_STATES["forward"] = False
        MOTOR_STATES["backward"] = False
        MOTOR_STATES["start_time"] = None  # Stop the timer

# Function to calculate elapsed time
def get_elapsed_time():
        if MOTOR_STATES["start_time"] is not None:
        return int(time.time() - MOTOR_STATES["start_time"])
        return 0

# Connect to WiFi
def connect_wifi():
        wlan = network.WLAN(network.STA_IF)
        wlan.active(True)
        wlan.connect(SSID, PASSWORD)
```

```
        for _ in range(20):
        if wlan.isconnected():
        return wlan.ifconfig()[0]
        time.sleep(1)
        raise RuntimeError("WiFi connection failed")

# HTML for the web interface
 HTML = """HTTP/1.0 200 OK
 Content-Type: text/html

<html>
<head>
<style>
        body {
        font-family: Arial, sans-serif;
        background-color: black;
        color: white;
        text-align: center;
        margin: 0;
        overflow: hidden;
        }
        h1 {
        font-size: 3em;
        animation: colorChange 1s infinite;
        }
        .light-grid {
        display: grid;
        grid-template-columns: repeat(10, 1fr);
        grid-template-rows: repeat(10, 1fr);
        width: 100vw;
        height: 100vh;
        position: absolute;
```

```
        top: 0;
        left: 0;
        z-index: -1;
        }
        .light {
        width: 100%;
        height: 100%;
        animation: flash 0.5s infinite;
        }
        @keyframes flash {
        0%, 100% { opacity: 1; }
        50% { opacity: 0; }
        }
        @keyframes colorChange {
        0% { color: red; }
        25% { color: green; }
        50% { color: blue; }
        75% { color: yellow; }
        100% { color: purple; }
        }
    </style>
    <script>
        function updateTimer() {
        fetch('/elapsed_time')
            .then(response => response.text())
            .then(data => {
            document.getElementById('timer').innerText = data;
            });
        }
        setInterval(updateTimer, 1000); // Update timer every second
```

```
    // Randomize light colors
    const lights = document.querySelectorAll('.light');
    setInterval(() => {
    lights.forEach(light => {
        const randomColor = `rgb(${Math.floor(Math.random() * 256)},
${Math.floor(Math.random() * 256)}, ${Math.floor(Math.random() * 256)})`;
        light.style.backgroundColor = randomColor;
    });
    }, 500); // Change colors every 500ms
</script>
</head>
<body>
<h1>DC Motor Control</h1>
<div class="light-grid">
<!-- Generate 100 lights -->
    ${Array(100).fill('<div class="light"></div>').join('')}
</div>
<p>Motor State: %MOTOR_STATE%</p>
<p>Speed: %SPEED%</p>
<p>Drilling Time: <span id="timer">%ELAPSED_TIME%</span> seconds</p>
<form action="/forward">
<button>Forward</button>
</form>
<form action="/backward">
<button>Backward</button>
</form>
<form action="/stop">
<button>Stop</button>
</form>
<form action="/speed">
<input type="range" name="speed" min="0" max="1023" value="%SPEED%">
```

```
<button>Set Speed</button>
</form>
</body>
</html>
 """

# Handle HTTP requests
 def handle_request(client):
        request = client.recv(1024).decode()

        try:
        if request.startswith('GET /forward'):
        motor_forward(MOTOR_STATES["speed"])
        elif request.startswith('GET /backward'):
        motor_backward(MOTOR_STATES["speed"])
        elif request.startswith('GET /stop'):
        motor_stop()
        elif request.startswith('GET /speed'):
        speed = int(request.split('speed=')[1].split(' ')[0])
        MOTOR_STATES["speed"] = speed
        if MOTOR_STATES["forward"]:
                motor_forward(speed)
        elif MOTOR_STATES["backward"]:
                motor_backward(speed)
        elif request.startswith('GET /elapsed_time'):
        elapsed_time = get_elapsed_time()
        client.send("HTTP/1.0 200 OK\r\nContent-Type: text/plain\r\n\r\n" + str(elapsed_time))
        client.close()
        return
        except Exception as e:
        print("Request error:", e)
```

```
        # Generate response
        motor_state = "Forward" if MOTOR_STATES["forward"] else "Backward" if
MOTOR_STATES["backward"] else "Stopped"
        elapsed_time = get_elapsed_time()
        response = HTML.replace("%MOTOR_STATE%", motor_state) \
                .replace("%SPEED%", str(MOTOR_STATES["speed"])) \
                .replace("%ELAPSED_TIME%", str(elapsed_time))
        client.send("HTTP/1.0 200 OK\r\nContent-Type: text/html\r\n\r\n" + response)
        client.close()

# Main function
 def main():
        try:
        # Initialize motor to stop
        motor_stop()

        # Connect to WiFi
        ip = connect_wifi()
        print("IP:", ip)

        # Start web server
        sock = socket.socket()
        sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        sock.bind(('0.0.0.0', 80))
        sock.listen(1)
        print("Web server started")

       while True:
        client, addr = sock.accept()
        print("Client connected:", addr)
        try:
                handle_request(client)
        except Exception as e:
```

```
            print("Error:", e)
        finally:
            client.close()
            print("Client disconnected")

    except Exception as e:
    print("Fatal error:", e)
    time.sleep(5)
    reset()

if __name__ == "__main__":
    main()
```

The test was successful as the code reader did not raise any errors and the motors worked as they should.

Part B: Test of the code used by the microprocessors to measure distance.

The following code was plugged into the two ESP-32 microprocessors. The distance test was conducted by placing the two microprocessors at a certain distance apart from each other. The test proved to be successful as the microprocessors were able to send signals to each other and measure the distance. Additionally, the calculated distance was similar to the measurement of the distance using a measuring tape.

Station code:

```
import network
 import socket
 import time
 import math

# Wi-Fi credentials
 SSID = "ESP32_OpenAP"
```

```
# Function to estimate distance from RSSI with better calibration
def calculate_distance(rssi):
    # Calibrated RSSI at 1 meter distance (adjusted for your environment)
    tx_power = -66  # RSSI value at 1 meter distance (calibrated)

    # Path loss exponent (adjust based on your environment)
    n = 2.5  # A typical value for indoor environments, can range from 2.0 to 4.0
    if rssi == 0:
    return -1.0  # No signal

    # Distance estimation based on RSSI value in meters
    distance_in_meters = 10 ** ((tx_power - rssi) / (10 * n))

    # Convert distance to feet (1 meter = 3.28084 feet)
    distance_in_feet = distance_in_meters * 3.28084

    return distance_in_feet

# Close any existing socket if open
try:
    s.close()
except:
    pass

# Create a socket and start the web server
try:
    addr = socket.getaddrinfo('0.0.0.0', 80)[0][-1]  # Use port 80 (HTTP)
    s = socket.socket()
    s.bind(addr)
    s.listen(1)
    print("Web server started on", addr)
except Exception as e:
```

```
        print("Failed to start the server:", e)
        exit()

# Initialize the station (STA) mode and connect to the AP
 sta = network.WLAN(network.STA_IF)
 sta.active(True)

print("Connecting to AP...")

# Attempt to connect to the Wi-Fi AP
 sta.connect(SSID)

# Wait for connection and check status
 timeout = 10  # 10-second timeout
 start_time = time.time()
 while not sta.isconnected():
        if time.time() - start_time > timeout:
        print("Connection failed: Timeout")
        break
        time.sleep(0.5)
        print("Connecting to AP...")

if sta.isconnected():
        print("Connected to AP")
        print("Station IP Address:", sta.ifconfig()[0])
 else:
        print("Failed to connect to AP")
        print("Status:", sta.status())
        exit()

while True:
        try:
        # Wait for a client connection
```

```
cl, addr = s.accept()
print("Client connected from", addr)

# Measure RSSI
rssi = sta.status('rssi')

# Calculate the estimated distance from RSSI (in feet)
distance = calculate_distance(rssi)
if distance < 0:
print("No valid RSSI signal received.")
distance = "N/A"  # Handle invalid RSSI

print(f"RSSI: {rssi} dBm, Estimated Distance: {distance:.2f} feet")

# HTTP response with dynamic content (Distance in feet)
response = """HTTP/1.1 200 OK
 Content-Type: text/html
<meta http-equiv="refresh" content="5">

<!DOCTYPE html>
<html>
<head>
<title>ESP32 Distance Measurement</title>
</head>
<body>
<h1>Welcome to ESP32 Distance Measurement!</h1>
<p>Station IP: {}</p>
<p>Current RSSI: {} dBm</p>
<p>Estimated Distance: {:.2f} feet</p>
</body>
</html>
 """.format(sta.ifconfig()[0], rssi, distance)
```

```
        # Send the HTTP response to the client
        cl.send(response)
        cl.close()
        print("Response sent and connection closed")

        except Exception as e:
        print("Error handling request:", e)

        time.sleep(1)  # Wait before accepting the next connection
```

Soft Access Point:

```
import network

# Configure the ESP32 as an open access point
 ap = network.WLAN(network.AP_IF)

# Activate the AP interface
 ap.active(True)

# Set the ESSID (Wi-Fi name)
 ap.config(essid="ESP32_OpenAP", authmode=network.AUTH_OPEN)

# Print the status and IP address of the AP
 print("AP Active:", ap.active())
 print("AP IP:", ap.ifconfig()[0])
```

Test of the circuit

The circuit shown in Figure 1 was constructed and tested by supplying electricity to all the components. This test was successful as there were no drops in voltage or loose connections in the circuit. Additionally, all the components received an adequate amount of electricity to function properly. Below are some images of the constructed circuit.

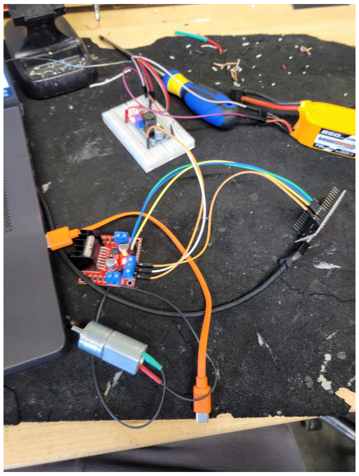


Figure 6: Fully constructed circuit.

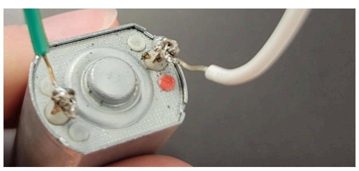


Figure 7: Soldered connection between the jumper cables and the prongs of the DC motor.

Test of the plastic parts

The following plastic parts were subjected to tests in which a member of the team used their hands to apply a large compression or tension force on the surface. This test proved to be successful as none of the parts broke.

However, it must be noted that the quantity of force applied on each part was not measured, but it is safe to assume that the amount of force applied during the tests is the same as in real-life conditions.

Figure 8: 3D printed plastic part.

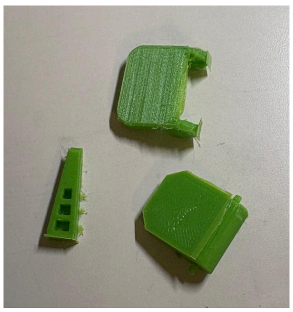


Figure 9: 3D printed plastic part.

Justification for choosing this prototype as the final design

This is the third prototype constructed by the team, incorporating all feedback provided by the TA, PMs, and client regarding previous versions. The design has been refined based on this input, ensuring significant improvements.

Additionally, this prototype underwent high-fidelity, physical tests that closely simulated real-life conditions. The successful results of these tests demonstrate that the design performs effectively and reliably. As a result, this prototype is considered the most accurate and reliable version, making it the best choice for the final design.

Conclusion

After analysing the feedback provided regarding the first and second prototypes, and finalising the design of the third prototype, a comprehensive round of physical tests were planned and conducted on each component of the device. These tests proved to be successful, thereby making this prototype the most perfect one for design day.

After carefully analyzing the feedback from the first and second prototypes and finalizing the design of the third prototype, a comprehensive series of physical tests were planned and conducted on each component of the device. These tests were successful, confirming the effectiveness of the design. In the future deliverables, this prototype will be presented to the client for evaluation, and a user manual will be created to ensure that the users of the device have all the information required to operate it.