Educational Omnidirectional Mobile Manipulator with Mecanum Wheels

The Educational Omnidirectional Mobile Manipulator is a robotic mobile platform designed for STEM education, providing an interactive and practical approach to learning robotics, programming, and sensor technologies. The primary controller is Raspberry Pi Pico W, and the control is executed remotely via a Wi-Fi web interface, enabling management of the platform's movements and the robotic arm.

The software is developed in Python with extensibility in mind, and the hardware utilizes serial communication for interaction between modules. Experiments have been conducted to validate the algorithms, and the results obtained are presented.

1. Introduction

The Educational Omnidirectional Mobile Manipulator (Figure 1) is a mobile robot suitable for STEM (Science, Technology, Engineering, and Mathematics) educational purposes. It is capable of performing various tasks in the environment using a robotic arm. The robot is remotely operated via Wi-Fi access point and a web interface.

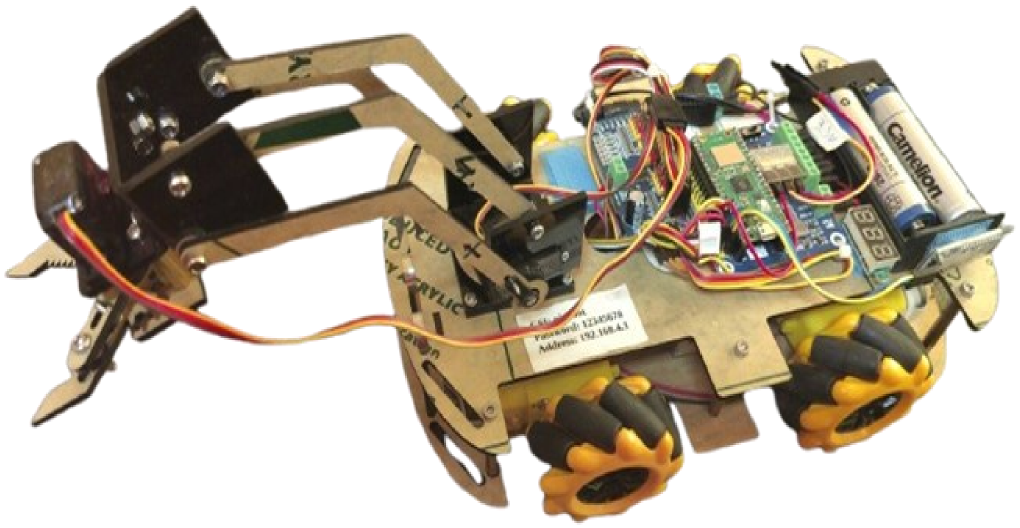

Fig. 1. Educational Omnidirectional Mobile Manipulator

Procedure for Connecting to the Educational Omnidirectional Mobile Manipulator as an Access Point:

1. Power Supply – The batteries must be fully charged prior to powering on the robot.
2. Connectivity – After powering on, the robot establishes a Wi-Fi network named "robot" (SSID). The device's Wi-Fi settings should be configured to select this network, enter the password, and establish a connection.

3. Access to the Interface – In a web browser, enter the IP address 192.168.4.1 to load the web interface.
4. Control – The interface provides control over the robot's movements, including its robotic arm.
5. Disconnection – The connection can be terminated by turning off Wi-Fi or selecting a different network. If necessary, the robot can be restarted.

2. Functionality and Control of the Robot

The Educational Omnidirectional Mobile Manipulator robot offers a range of capabilities for movement, object manipulation, and automated actions. All these functions are remotely controlled via a web interface (Figure 2), accessible through a Wi-Fi network.

The functionalities include:

- Maneuverability – Thanks to the omnidirectional wheels, the robot can move in all directions. Movement is controlled via a set of buttons in the web interface, allowing navigation in the following directions:
    - Left Forward – movement diagonally forward-left.
    - Forward – straight ahead movement.
    - Right Forward – movement diagonally forward-right.
    - Left – movement to the left.
    - STOP – halt the robot.
    - Right – movement to the right.
    - Left Back – movement diagonally backward-left.
    - Back – backward movement.
    - Right Back – movement diagonally backward-right.
    - Rotate Left – rotation in place to the left.
    - STOP – halt rotation.
    - Rotate Right – rotation in place to the right.
- Remote Control – Operation is carried out via a web browser, with the user connecting to the robot’s Wi-Fi network and utilizing an intuitive control interface.
- Object Manipulation – The robotic arm is equipped with three degrees of freedom, enabling grasping, lifting, and repositioning objects. In the web interface, this is achieved through sliders and text fields that regulate the angles of the servomotors. The arm is controlled by sliders and numeric input fields that specify the rotation angles:
    - Base – rotation of the arm’s base.

- Arm – adjustment of the shoulder's tilt.
- Claw – control of the gripper's open/close action.

Changing the value in the text fields automatically updates the corresponding slider position, and adjusting the slider updates the numeric value. An additional Reset button restores the sliders and text fields to their initial values.

- Automated Tasks – Besides manual control, the robot supports setting up movements and manipulations that can be programmed for automation.
- Environmental Object Detection – Ultrasonic sensors provide obstacle information, enabling the robot to avoid collisions.

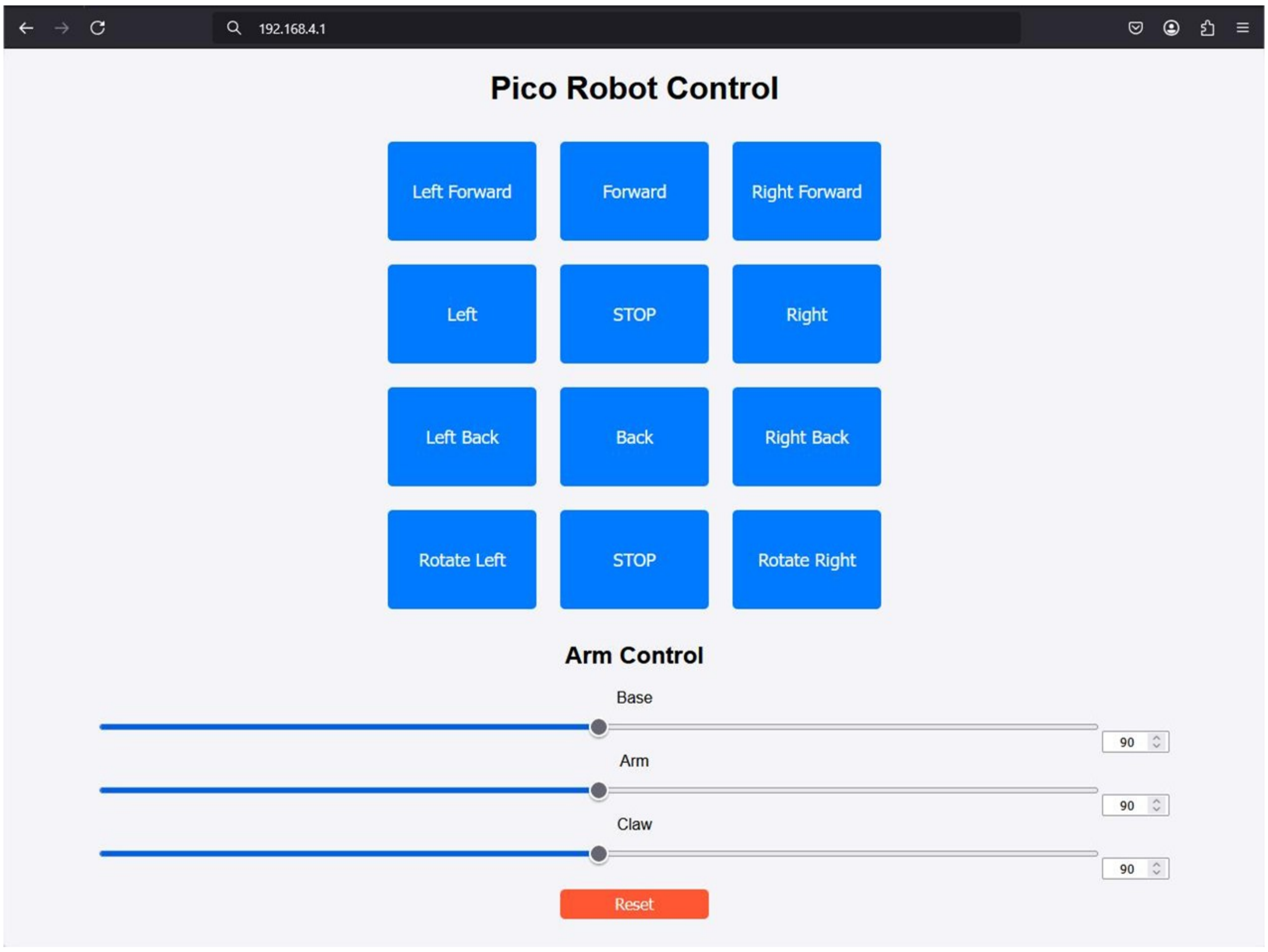

Fig. 2. Web control interface for Educational Omnidirectional Mobile Manipulator .

3. Hardware Architecture of the Educational Omnidirectional Mobile Manipulator

Figure 3 illustrates the architecture of the robotic system based on the Raspberry Pi Pico W, which manages motors and sensors through communication with peripheral components. The Educational Omnidirectional Mobile Manipulator comprises the following modules:

1. Power Supply System
   - Lithium-Ion battery providing primary power (8V).
   - Buck/Boost converter regulating voltage to 5V for sensitive electronic components.
2. Motor Control Module
   - DC Motor Driver receives 8V from the battery and a 5V logic control signal. The driver controls four motors connected to omnidirectional wheels, enabling the robot's maneuverability.
   - PCA9685 PWM controller extends the number of PWM outputs for controlling additional devices. Three channels (Ch0, Ch1, Ch2) of the PCA9685 are used to control the servomotors of the robotic arm.
3. Control Module
   - Raspberry Pi Pico W communicates with other components via the I2C bus.
4. Sensor
   - Ultrasonic sensor connected to the Pico W is used for obstacle detection.

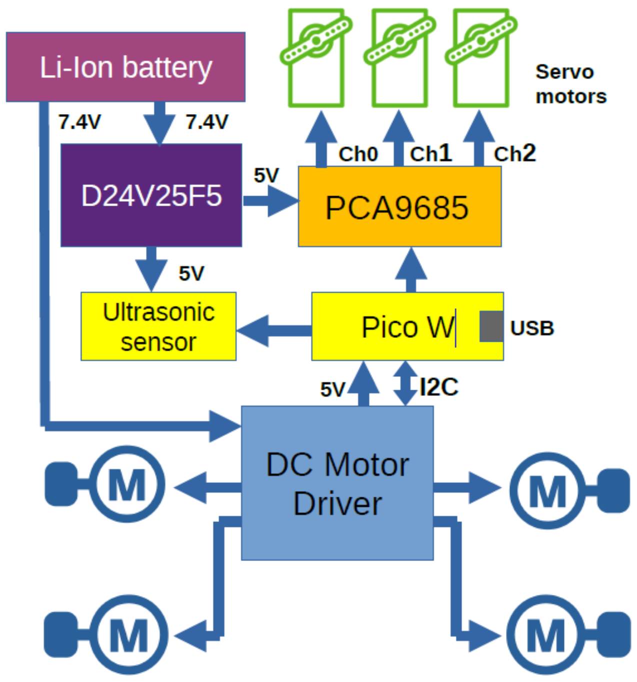

Figure 3. Block architecture of the Educational Omnidirectional Mobile Manipulator.

The hardware architecture of the Educational Omnidirectional Mobile Manipulator combines energy-efficient and functional modules, enabling high maneuverability, autonomy, and interactivity. The integration of Raspberry Pi Pico W, PCA9685, and Pico-Motor-Driver provides flexible control, while navigation sensors make the robot more intelligent and adaptive in various environments. To ensure stable and continuous power supply, the Educational Omnidirectional Mobile Manipulator uses two Li-Ion batteries (2600 mAh), which provide operational autonomy. A DC converter supplies stable power to some of the modules.

Hardware modules used in the Educational Omnidirectional Mobile Manipulator:

- BliliDIY 0.5V-33V Boost/Buck Converter: a voltage converter (Figure 4) that provides a stable 5V output voltage regardless of the battery charge level.

TUSOTEK
AUTOMATIC BOOST/BUCK VOLTAGE REGULATOR
INPUT VOLTAGE:DC3.3-30V
OUTPUT VOLTAGE:DC0.5-33V/5A(MAX
Key Function:
IN
OUT
REG
DISPLAY
ADJ

Fig. 4. BliliDIY 0.5V-33V Boost/Buck Converter

Main Features:

- Input Voltage: 3.3V - 30V DC
- Output Voltage: 0.5V - 33V DC (adjustable)
- Maximum Current: 5A
- Automatic Regulation: Functions as both a Step-Up (Boost) and Step-Down (Buck) converter, allowing adaptation to various power supply requirements.
- Built-in Display: Enables real-time monitoring of input and output voltages.
- Control Functions: Voltage adjustment is performed via short or long button presses, providing flexibility in power supply configuration.

Microcontroller: Raspberry Pi Pico W

The Raspberry Pi Pico W (Figure 5) is the primary computing module of the robot, responsible for controlling all peripheral devices and providing communication with the user.

Fig. 5. Raspberry Pi Pico W

Functions and Features:

- Dual-core ARM Cortex-M0+ processor with a clock frequency of 133 MHz
- Built-in Wi-Fi module enabling wireless control via a web interface
- I2C, SPI, and PWM interfaces used for communication with peripheral modules
- Compact size and low power consumption, making it suitable for autonomous mobile robots

Pico-Motor-Driver (Waveshare)

The Pico-Motor-Driver (Figure 6) is a module for controlling direct current (DC) motors, specifically designed for use with the Raspberry Pi Pico. It allows for simultaneous control of multiple DC motors, providing the ability to adjust speed and rotation direction.

Fig. 6. Pico-Motor-Driver

Main Features:

• Motor Control: Supports up to 4 DC motors with independent speed and direction regulation.
• Power Supply Voltage: Operates within a range of 6V to 12V DC, providing flexibility

in choosing the power source.
• PWM Control: The integrated PCA9685 chip offers 12-bit hardware PWM control, enabling precise speed regulation of the motors.
• H-Bridge Drivers: Utilizes a dual TB6612FNG H-bridge driver, which ensures high efficiency and low heat generation during operation.
• I²C Communication: The Pico-Motor-Driver module is controlled via the I²C interface and supports up to 32 different I²C addresses, configurable through five address jumpers. These jumpers determine the module's I²C address, with each of the five positions being either open (disconnected) or closed (connected), creating various address combinations. This allows for configuring up to 32 unique addresses ($2^5 = 32$).
If all jumpers are open, the driver address defaults to 0.
• Built-in 5V Regulator: The integrated 5V regulator can supply up to 3A of output current, enabling direct powering of the module from a battery through the VIN terminal.
• Compatibility: The standard Raspberry Pi Pico header allows for easy connection and support for all models in the Raspberry Pi Pico series.

These features make the Pico-Motor-Driver an ideal solution for projects requiring control of multiple DC motors with high precision and efficiency.

The Educational Omnidirectional Mobile Manipulator utilizes a connection between the Raspberry Pi Pico W and the Pico-Motor-Driver with address 0 in the following master-slave configuration:

- The Raspberry Pi Pico W functions as the I²C master (primary device).
- The Pico-Motor-Driver operates as an I²C slave (secondary device), receiving commands to control the motors.
  By default, the I²C bus of the Pico W is configured as follows:

Signal	Pico W (GPIO пин)	Pico-Motor-Driver (пин)
SDA (Serial Data)	GP20	SDA
SCL (Serial Clock)	GP21	SCL
VCC (Захранване)	5V	VCC (pin 39)
GND (Маса)	GND	GND (pin 38)

Role of the pins:

- SDA (GP20) and SCL (GP21) – these two pins are used for communication between the Raspberry Pi Pico W and the motor driver. The Pico W sends commands via SDA, while SCL synchronizes the communication. If SDA and SCL are not connected properly, the Pico W will not be able to communicate with the driver, and the motors will not respond to commands.
- GND and VCC – provide electrical connection between the two devices.

Control PWM board: PCA9685
A 16-channel PWM controller

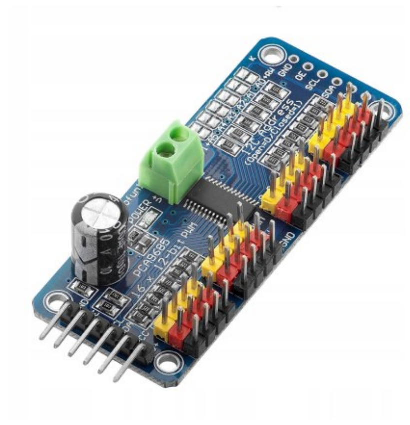

Fig. 7. PCA9685

Main Features:

- Allows control of up to 16 PWM devices simultaneously
- I2C interface providing efficient communication with the Raspberry Pi Pico W
- Precise 12-bit pulse-width modulation (PWM) control, ensuring smooth movements of the robotic arm
- Compact design and low power consumption

The Educational Omnidirectional Mobile Manipulator is equipped with a robotic arm that has three degrees of freedom (3 DOF). The arm is a laser-cut structure that guarantees stability and lightness. It is controlled by three MG90S servomotors, which facilitate the following movements:

1. Rotation of the base – sets the orientation of the arm
2. Raising and lowering the shoulder – allows reaching different heights
3. Gripping mechanism (clamp) – opens and closes to grasp objects

Ultrasonic Sensor CS100A

The CS100A is an ultrasonic sensor (see Figure 8) used for distance measurement, obstacle detection, and navigation. It operates by sending and receiving ultrasonic waves to calculate the distance to objects in the environment.

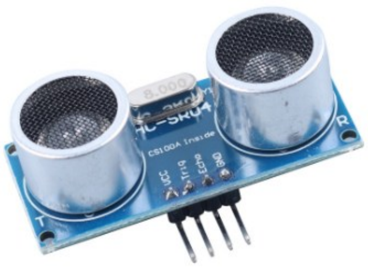

Fig. 8. CS100A

Sensor Characteristics:

• Power Supply: 3.3V - 5V DC

• Measurement Range: 2 cm - 500 cm

• Signal Type: Digital (not analog)

• Communication: Operates via Trigger (TRIG) and Echo (ECHO) pins

The Educational Omnidirectional Mobile Manipulator uses the CS100A sensor for obstacle detection and autonomous navigation. The sensor measures the distance to objects in front of the robot, enabling the system to make decisions for collision avoidance and movement planning. This functionality enhances the robot's adaptability in various environments and improves its navigation precision.

4. Software Architecture of the Educational Omnidirectional Mobile Manipulator

The software of the Educational Omnidirectional Mobile Manipulator is organized into several main modules developed in the Python programming language. These modules facilitate communication, motor control, management of the robotic arm, and provide a web interface for the user.

The primary software modules developed for the Educational Omnidirectional Mobile Manipulator are presented in Table 1:

Table 1. Main Python Programming Modules

File	Function
main.py	The main startup script that initializes the robot and manages communication over Wi-Fi.
Educational Omnidirectional Mobile Manipulator _main.py	Responsible for creating the Wi-Fi access point and processing HTTP requests.
index.html	Web interface for controlling movement and a robotic arm via a browser.
Educational Omnidirectional Mobile Manipulator .py	Main control class for the robot, including movement methods.
Educational Omnidirectional	Motor control class using PCA9685 and H-bridge driver.

File	Function
Mobile Manipulator _motors.py	
Educational Omnidirectional Mobile Manipulator _arm.py	Class for controlling a robotic arm, using PCA9685 for controlling the servomotors.
pca9685.py	Communication driver with PCA9685 that provides PWM signals for motors and servomotors.

The structure and communication between modules are as follows:

• Motor Control
The file robot_motors.py utilizes PCA9685 to control four DC motors.

- The motors are managed via an I²C interface (SDA: GPIO20, SCL: GPIO21).
- The module contains a class MotorDriver, which provides methods for starting, stopping, and controlling the speed of the motors.

• Robotic Arm Control
The file robot_arm.py manages three MG90S servomotors using PCA9685.

- Control is executed through I²C communication.
- Smooth movement of the servomotors is facilitated via the method smooth_move_servo().
- Includes a function for resetting the arm (reset_servos()).

• Wi-Fi Communication and Web Interface
The file robot_main.py creates a Wi-Fi access point and handles requests from index.html.

- Upon startup, the Educational Omnidirectional Mobile Manipulator creates an SSID: "robot", which users can connect to.
- When a user presses a button in the web interface, a request is sent to the robot as an HTTP request.
- Movement and manipulation of the robotic arm can be controlled via sliders and buttons.

The software workflow encompasses the following stages:

1. Power-up and initialization – main.py initializes the Educational Omnidirectional Mobile Manipulator.
2. Creating the Wi-Fi network – robot_main.py launches an Access Point (SSID: "robot").

3. Web control interface – index.html provides a graphical control panel for operation.
4. Command reception – The Educational Omnidirectional Mobile Manipulator handles requests from the web interface via robot_main.py.
5. Movement and manipulation – robot.py initiates movement and arm control methods.
6. Connection termination – When the user disconnects, the Wi-Fi connection can be terminated.

The Python modules (files) related to the workflow of the Educational Omnidirectional Mobile Manipulator are shown in Table 2:

Table 2. Python modules related to the workflow

№	Action	Fail
1.	Action	main.py
2.	System initialization	robot_main.py
3.	Creating a Wi-Fi access point	index.html
4.	Interactive web interface	robot_main.py
5.	Receiving and processing commands	robot.py
6.	Motor control	robot_arm.py
7.	Arm control	robot_main.py
8.	Executing commands from the web interface	robot.py

This architecture provides flexible and efficient control of the Educational Omnidirectional Mobile Manipulator, allowing for the expansion of functionality with new modules and improvements.

The main software components that illustrate the workflow of the Educational Omnidirectional Mobile Manipulator are as follows:

1. *Power-up and Initialization*
   File: main.py
   Upon startup, the Educational Omnidirectional Mobile Manipulator loads the core modules and initializes the control system.

```
import sys
sys.path.append('/Educational Omnidirectional Mobile Manipulator ')
from robot_main import *  # The main control module
```

2. *Creating a Wi-Fi Network*

File: robot_main.py
After startup, the Educational Omnidirectional Mobile Manipulator creates a Wi-Fi access point (SSID: "robot") that the user can connect to.

```
import network
import rp2

ssid = 'robot'
password = '12345678'

def create_WiFi_AP():
   ap = network.WLAN(network.AP_IF)
   ap.config(essid=ssid, password=password)
   ap.active(True)
   while ap.active == False:
      pass
   print("Access point active", ap.ifconfig())

# Стартиране на Access Point
create_WiFi_AP()
```

3. *Web interface – robot control*

A user interface designed for controlling movement or navigation, such as in robotics, simulations, or interactive applications

```
<button onclick="submitAction('./forward?')">Forward</button>
<button onclick="submitAction('./left?')">Left</button>
<button onclick="submitAction('./right?')">Right</button>

<script>
function submitAction(url) {
   fetch(url);
}
</script>
```

4. *Processing HTTP requests and executing commands*

File: robot_main.py
When the button labeled "Educational Omnidirectional Mobile Manipulator" is pressed, the system receives an HTTP request and executes the corresponding movement.

```
import socket
from robot import rcobot

robot = robot()  # Създаване на обект за управление на робота

def serve(connection):
   while True:
      client = connection.accept()[0]
      request = client.recv(1024)
      request = str(request).split()[1]

      if request == '/forward?':
          robot.goForward()
      elif request == '/left?':
          robot.moveLeft()
      elif request == '/right?':
          robot.moveRight()

      client.send('HTTP/1.0 200 OK\r\nContent-type: text/html\r\n\r\n')
      client.close()
```

5. *Motion Control*
   File: robot.py
   Functions for movement through control of four DC motors.

```
from robot_motors import MotorDriver

class robot:
    def __init__(self):
        self.m = MotorDriver()  # Инициализация на моторния драйвер

    def goForward(self, speed=50):
        self.m.TurnMotor('LeftFront', 'forward', speed)
        self.m.TurnMotor('RightFront', 'forward', speed)
        self.m.TurnMotor('LeftBack', 'forward', speed)
        self.m.TurnMotor('RightBack', 'forward', speed)
```

6. *Control of the Robotic Arm*
   File: robot_arm.py
   Control of the three servomotors via PCA9685.

```
from pca9685 import PCA9685
from machine import I2C, Pin

class RobotArm:
    def __init__(self):
        self.i2c = I2C(1, sda=Pin(2), scl=Pin(3))
        self.pca = PCA9685(i2c=self.i2c)
        self.pca.freq(50)

    def control_servo(self, channel, angle):
        pulse = int(102 + (angle / 180.0) * (512 - 102))
        self.pca.pwm(channel, 0, pulse)  # Изпращане на PWM signal
```

7. *Processing of hand control requests from the web interface*

File: robot_main.py
When the slider values on the web interface are changed, a request is sent to the robot.

```
if 'servo_base_slider' in request:
    slider_angle = int(request.split('servo_base_slider=')[1].split('&')[0])
    robot.arm.control_servo(0, slider_angle)  # Завъртане на основата

if 'servo_arm_slider' in request:
    slider_angle = int(request.split('servo_arm_slider=')[1].split('&')[0])
    robot.arm.control_servo(1, slider_angle)  # Контрол на рамото

if 'servo_claw_slider' in request:
    slider_angle = int(request.split('servo_claw_slider=')[1].split('&')[0])
    robot.arm.control_servo(2, slider_angle)  # Grip control
```

8. *Session Termination and Shutdown*

File: robot.py
Method to stop all motors when finishing operation.

```
def stop_all_motors(self):
    self.m.StopAllMotors()
```

5. Conclusion and Future Improvements

The developed Educational Omnidirectional Mobile Manipulator is a mobile robotic platform designed for STEM education, offering control capabilities via a web interface and Wi-Fi communication. Thanks to its omnidirectional wheels, robotic arm, and navigation sensors, it can perform various tasks related to movement, object manipulation, and obstacle avoidance.

The software architecture of the Educational Omnidirectional Mobile Manipulator is built from modular components, ensuring easy adaptation and extension of functionality. The interactive web interface allows for straightforward control, while the use of I²C communication between components guarantees stability and efficiency during operation.

Future enhancements for the Educational Omnidirectional Mobile Manipulator may include:

- Addition of computer vision
  - Integration of a camera for object and color recognition.
  - Implementation of algorithms for color and shape recognition of objects.
- Autonomous navigation
  - Deployment of lidar or additional sensors to improve navigation capabilities.
  - Development of algorithms for autonomous decision-making based on sensor data.
- Voice control
  - Integration of a voice assistant for command input through speech recognition.
- Bluetooth and IoT connectivity

  - Support for Bluetooth communication to connect with mobile devices.
  - Incorporation of a cloud-based IoT platform for real-time management and monitoring.
- Enhanced energy efficiency
  - Optimization of power consumption through sleep modes and dynamic power management.
  - Potential use of solar panels as an auxiliary energy source.