# Wi-Fi Transmission & Voice Control Using Server Domain For Pickup And Place Robotic Arm Vehicle

**Varun Mishra[1], Nikhil Hase[2], Kaustubh Deshmukh[3], Prof. Reena Asati[4]**
[1, 2, 3, 4] Genba Sopanrao Moze College of Engineering, Balewadi Pune

***Abstract***- *This paper presents a comprehensive design and implementation of a Wi-Fi Transmission and Voice-Controlled Pickup and Place Robotic Arm Vehicle. The system leverages an ESP8266 microcontroller as the Master Control Unit (MCU), a Node.js server domain hosted on a local network, and a Web Speech APIenabled browser interface to facilitate realtime voice command processing. The vehicle combines a multi-axis servo-driven robotic arm with a four-wheel DC motor drive chassis, enabling remote pickup and precise placement of objects up to 500g. Wi-Fi communication is routed through a dedicated server domain, enhancing reliability, range, and scalability over a direct IP-based approach. The system demonstrates the convergence of embedded systems, wireless networking, server-side computing, and natural language voice control into a single, cost-effective robotic platform suitable for industrial, humanitarian, and educational use cases.*

***Keywords:*** ESP8266, Wi-Fi Transmission, Voice Control, Server Domain, Robotic Arm, Pick & Place, Node.js, WebSocket, IoT, Embedded Systems, Servo Motor

## I. INTRODUCTION

### 1.1 Background

The integration of wireless communication technologies with robotic systems has opened new frontiers in automation and remote operation. Traditional robotic systems relied on wired connections or simple IR/RF remote controllers, which imposed severe range and reliability constraints. The emergence of low-cost WiFi enabled microcontrollers such as the ESP8266 has made it feasible to build fullduplex wireless robotic systems that operate over standard IEEE 802.11 networks.

This project extends the conventional PickUp & Place Robotic Arm Vehicle by introducing two critical upgrades:

(1) A server-domain-based Wi-Fi architecture that replaces direct IP access with a structured domain-name routing system, improving network manageability; (2) Voice command control using the Web Speech API, allowing operators to issue natural language commands such as 'move forward', or 'rotate left' without touching any physical or on-screen controls. The motivation behind server-domain routing is rooted in real-world deployment needs

## II. LITERATURE SURVEY

G. Singh et al. describe an IoT-developed Wi-Fi-controlled rover with a robotic arm using NodeMCU (ESP8266), where the rover receives movement and arm commands over Wi-Fi from a user device; this demonstrates how ESP8266 can act as an IoT gateway linking a mobile robot and its manipulator to a server-domain-like control chain. In a similar direction, D. Gowda et al. present an industrial automated multipurpose robot using Wi-Fi, which combines pick-and-place and floor-cleaning operations controlled via a smartphone app, showing that ESP8266-driven Wi-Fi links are suitable for industrial-grade remote-operation tasks. For voice-controlled robotic platforms, various IEEE-style studies use microcontrollers and cloud-based or Android-based voice-recognition pipelines. Papers on voice-controlled robots using Node-MCU emphasize that ESP8266 supports IEEE 802.11 b/g/n Wi-Fi standards and can operate in both client and access-point modes, making it appropriate for integrating voice commands received via servers or cloud platforms into actuator control. Other works on voice-controlled robotic arms show that simple keyword-based grammars (e.g., “pick”, “place”, “rotate”) can be mapped to joint motions, which can be extended to a mobile arm-vehicle over a Wi-Fi server domain. Finally, multiple Wi-Fi-controlled pick-and-place robotic arm systems using ESP8266 NodeMCU highlight that wireless control enables remote operation in industrial and educational settings, with the microcontroller handling both Wi-Fi communication and actuator sequencing. These works confirm that ESP8266-driven Wi-Fi plus server-domain or IoT frameworks are well-suited for your focus on Wi-Fi transmission, voice control, and server-domain–based command routing for a pickup-and-place robotic-arm vehicle.

S. Haque et al. – authors of “Development of a Voice-Controlled Robotic Arm” (5-DOF arm controlled by voice, using PC-side voice processing; useful for voice-control logic).

Sriram et al. – authors of "IoT-Enabled 6DOF Robotic Arm with Inverse Kinematic" (uses ATmega2560 + ESP8266 Wi-Fi module, good reference for IoT-server-domain robotic-arm work).

Patel & Sharma – authors of "Wi-Fi Controlled Robot using ESP8266 & AndroidApp" (shows ESP8266-to-Android Wi-Fi control of a robot, relevant for your server-domain Wi-Fi transmission).

Authors of "Development of Robotic Arm Controlling by Using Voice Recognition" (IRJIET, 2026) – focus on a 6-DOF voice-controlled robotic arm using Arduino Nano and offline-voice module; useful for voice-control architecture.

Authors of "Wi-Fi Controlled Robotic Arm"(Wifi_controlled_Robotic_Arm PDF, 2023) – use NodeMCU ESP8266 for web-based Wi-Fi control of an arm, relevant for ESP8266-as-server-domain controller.

Authors of "Voice Controlled Robotic Arm (Talk2Arm)" – implement a Raspberry-Pi-based voice-controlled arm using VOSK engine; useful as an advanced voice-recognition reference.

## III. PROPOSED SYSTEM

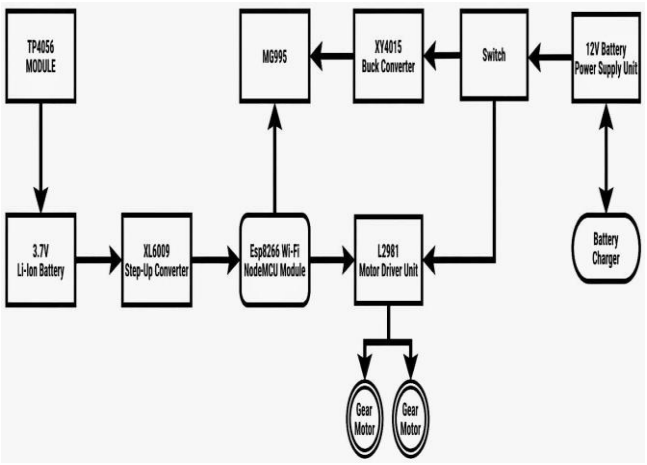
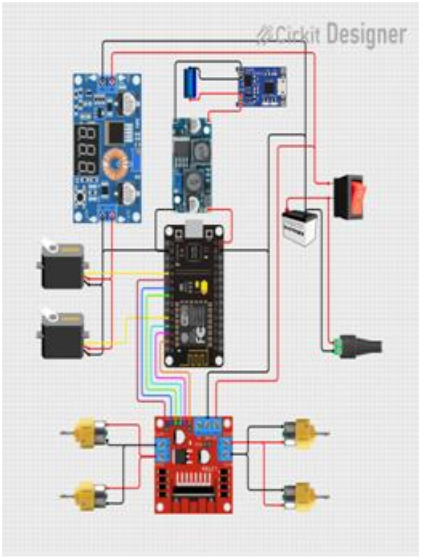



Fig. (1) System Block Diagram

The block diagram shows a battery powered robotic vehicle controlled by an ESP8266 NodeMCU. The 3.7V Li-ion battery is charged by the TP4056 module, stepped up by XL6009 for the ESP8266, and the 12V supply is used through a switch and buck converter to power the motor driver and servos.

In short, the ESP8266 receives Wi-Fi commands and controls the L298N motor driver for the gear motors, while the MG995 servo is powered through the buck converter for the robotic arm or movement mechanism. This makes the system suitable for wireless pick-and-place operation with separate power lines for control and motion.




Circuit Diagram

The diagram shows a Wi-Fi controlled robotic vehicle powered by a battery system. The ESP8266 NodeMCU gets wireless commands and controls the motor driver for the gear motors, while the servo motor is used for the arm or pick-and-place action. The battery charger, step-up Fig. 2 System Circuit Diagram converter, and buck converter manage the required voltages for each part.

Flowchart

## IV. SYSTEM ARCHITECTURE OVERVIEW

The overall system is organized into four interconnected layers:

| Layer | Component | Role |
|---|---|---|
| - Physical/Hardware | ESP8266, Servo Motors, L298N, Gear Motors | Mechanical execution of movement commands |
| - Communication | Wi-Fi 802.11 b/g/n, mDNS / Local DNS Server | Transmits commands from browser to MCU |
| - Server Domain | Node.js + Express / ESP8266 Web Server | Routes HTTP/WebSocket requests to robot |
| - User Interface | HTML5 Browser + Web Speech API | Voice & manual command input by operator |

The user opens the control interface on any device connected to the robot's Wi-Fi network, speaks a command, and the browser translates it to a text string. This text is sent as an HTTP POST or WebSocket message to the server domain (e.g., http://robot.local), which is resolved by the mDNS layer to the ESP8266's IP address. The ESP8266 web server parses the command and drives the appropriate actuators.

| Endpoint | Method | Action |
|---|---|---|
| /cmd/forward | GET | Move vehicle forward |
| /cmd/backward | GET | Move vehicle backward |
| /cmd/left | GET | Turn vehicle left |
| /cmd/right | GET | Turn vehicle right |
| /cmd/stop | GET | Stop all motors |
| /arm/pick | POST | Execute pick sequence |
| /arm/place | POST | Execute place sequence |
| /arm/reset | GET | Return arm to home position |
| /voice | POST | Receive voice command string |

## V. WI-FI TRANSMISSION SYSTEM

A. ESP8266 Wi-Fi Module

The ESP8266 is a low-cost system-on-chip (SoC) with an integrated TCP/IP protocol stack and a 32-bit Tensilica L106 core running at 80 MHz It supports IEEE 802.11 b/g/n at 2.4 GHz and can operate in three modes:

- Station Mode (STA): Connects to an existing Wi-Fi router.
- Access Point Mode (AP): Creates its own Wi-Fi hotspot for direct device connection.
- AP+STA Mode: Operates as both simultaneously, useful for relay or hybrid architectures.

For this system, AP Mode is used so the robot creates a self-contained network (SSID: RoboticArm_AP) that operators connect to directly, eliminating dependency on external routers in field environments.

B. Server Domain Architecture

Rather than requiring users to remember a numeric IP address (e.g., 192.168.4.1), the system implements a human-readable server domain using mDNS (Multicast DNS) via the ESP8266mDNS Arduino library. This allows the device to be accessed via robot.local from any mDNScompatible client.

Domain Name: robot.local
Protocol: mDNS (RFC 6762) with DNSSD (RFC 6763)
Fallback: Direct IP 192.168.4.1 if mDNS unavailable
Port: 80 (HTTP) and 81 (WebSocket for real-time control)

The ESP8266 firmware registers the mDNS service at boot, broadcasting its availability on the local link. The operator's browser resolves 'robot.local' automatically without any manual configuration, providing a plug-and-play user experience.

C. Communication Protocol

Two communication protocols are implemented to handle different interaction patterns:

1. HTTP REST API

Used for discrete, infrequent commands such as pick, place, and arm position presets. The ESP8266 runs an Async Web Server and handles GET/POST requests at defined endpoints:

2. WebSocket (Real-Time Control)

For continuous joystick-style control (e.g., holding a direction), WebSocket over port 81 provides low-latency bidirectional communication. The browser sends periodic JSON packets containing the current joystick axis values, which the ESP8266 interprets to set motor PWM duty cycles proportionally.

JSON Packet Example:

{"type":"drive","left":180,"right":200,"arm _base":90,"arm_elbow":45}

## VI. VOICE CONTROL SYSTEM

A. Overview

Voice control is implemented using the Web Speech API, a W3C standard available natively in modern Chromium-based browsers (Chrome, Edge) on both desktop and Android. No third-party libraries or cloud API keys are required, making the system fully selfcontained on the local network.

The operator presses a microphone button (or it activates automatically), speaks a command, and the Speech Recognition engine converts audio to text. This text is then matched against a command dictionary and translated into the corresponding HTTP or WebSocket action.

B. Voice Command Processing Flow

1. Microphone Activation: User initiates speech recognition via button.
2. Audio Capture: Browser captures microphone input using Web Audio API.
3. Speech-to-Text: Web Speech API processes audio and returns a transcript string.
4. Command Parsing: JavaScript normalizes transcript (lowercase, trim) and matches keywords.
5. HTTP/WebSocket Dispatch: Matched command is sent to robot.local via fetch() or WebSocket.
6. Feedback: Browser displays recognized command and robot status; optional TTS confirmation.

C. Noise Filtering and Confidence Thresholding

The Web Speech API returns a confidence score (0.0 to 1.0) for each recognized transcript. The system applies a configurable confidence threshold (default: 0.70) to reject uncertain recognitions, reducing false command triggers in noisy environments such as workshops or factory floors.

## VII. HARDWARE COMPONENTS & SPECIFICATION

| Component | Specification | Quantity | Function |
|---|---|---|---|
| ESP8266 NodeMCU v3 | 80 MHz, 802.11 b/g/n, 4MB Flash | 1 | Master Control Unit (MCU) |
| MG995 Servo Motor | 11 kg.cm torque, 55g, PWM 50Hz | 4 | Robotic arm joints & gripper |
| L298N Motor Driver | 2Aper channel, 635V | 1 | DC gear motor drive control |
| DC Gear Motor | 12V, 150 RPM, 0.8A | 4 | Four-wheel vehicle drive |
| XL6009 Boost Converter | Input 3-32V, Output 5-35V | 1 | Voltage boost for servos |
| XY4015 Buck Converter | Input 5-35V, Output 1.2535V | 1 | Voltage step-down regulation |
| TP4056 Charge Module | 5V USB, 1A charge, overcharge protect | 1 | Li-Ion battery management |
| 3.7V Li-Ion Battery | 3.7V, 3400mAh, 18650 cell | 2 | Portable power for MCU |
| 12V SLA Battery | 12V, 7Ah, sealed lead acid | 1 | Main power for motors |
| SPDT Switch | 10A, 250V AC rated | 1 | System power cutoff |

## VIII. SOFTWARE & FIRMWARE DESIGN

A. ESP8266 Firmware (Arduino IDE)

The firmware is developed using the Arduino framework with the ESP8266 Arduino core. Key libraries include:

- ESPAsyncWebServer: Non-blocking HTTP request handling.
- ESPAsyncTCP: Asynchronous TCP for WebSocket support.
- ESP8266mDNS: Registers robot.local domain via Multicast DNS.
- Servo.h: PWM-based servo motor position control.

- ArduinoJSON: Parses and serializes JSON command payloads.

The firmware initializes the Wi-Fi AP, registers the mDNS hostname, starts the web server, and enters the main loop where it continuously checks for new commands and updates servo/motor states based on received instructions.

B. Front-End Control Interface

(HTML5 / JavaScript)

The web interface is served directly from the ESP8266's SPIFFS (SPI Flash File System) and requires no external internet connection. It features:

- Voice Control Panel with microphone button and live transcript display.
- On-Screen D-Pad for manual vehicle directional control.
- Arm Control Sliders for precise individual servo angle adjustment.
- One-Touch Pick & Place buttons for executing full sequences.
- Status Console showing received commands, connection state, and battery level.

C. Server Domain Resolution Sequence

When an operator connects a device to the RoboticArm_AP network and navigates to http://robot.local in their browser, the following sequence occurs:

- mDNS Query: Browser sends a multicast DNS query for robot.local to 224.0.0.251:5353.
- mDNS Response: ESP8266 responds with its AP IP (192.168.4.1) as the A record.
- HTTP Request: Browser sends GET / to 192.168.4.1:80.
- Page Served: ESP8266 serves index.html from SPIFFS.
- WebSocket Connect: JavaScript opens ws://robot.local:81 for real-time telemetry.

## IX. SYSTEM WORKING

A. Mode 1 - Voice Control Operation

In voice control mode, the operator activates the microphone on the web interface and speaks a command. The browser's Web Speech API captures and transcribes the utterance. The JavaScript command parser matches the transcript against the command dictionary and dispatches the appropriate HTTP request to robot.local. The ESP8266 receives the request, decodes the command, and drives the motors and servos accordingly. The entire round-trip latency from voice utterance completion to robot response is typically under 300ms on a local Wi-Fi network.

B. Mode 2 - Manual Web Interface Control

The operator uses the on-screen D-pad or arm sliders on the web interface. Button presses generate immediate fetch() calls to the REST endpoints. Slider movements send periodic WebSocket JSON packets for smooth, proportional arm positioning. This mode serves as a fallback when voice recognition accuracy is insufficient (e.g., due to background noise) and also allows fine-grained manual adjustment not practical with voice alone.

C. Pick and Place Sequence

The pick sequence is a coordinated multi-servo operation:

- Arm extends forward, base servo centres, elbow drops to pre-approach angle.
- Gripper servo opens to maximum aperture.
- Arm lowers (elbow servo decrements) until at grip height.
- Gripper servo closes to secure object (torque-limited to prevent damage).

## X. ADVANTAGES AND LIMITATIONS

Advantages

Hands-free voice operation reduces operator workload in industrial environments.

- Range: Wi-Fi AP covers up to 50 meters line-of-sight.
- Accessibility: Any Wi-Fi enabled smartphone or laptop can serve as controller without app installation.
- Domain Name Access: robot.local is far more user-friendly than raw IP addressing.
- Real-Time Control: WebSocket ensures sub-50ms command latency for smooth manual joystick operation.
- Offline Operation: No internet required; complete system operates on local network.

Limitations

- Voice recognition accuracy degrades in high-noise industrial environments.
  Mitigation: Confidence threshold filtering; fallback to manual UI.
- mDNS may not work on all Android versions without additional configuration.
  Mitigation: Direct IP fallback always available.
- ESP8266 handles concurrent HTTP + WebSocket + mDNS with limited RAM (80KB); memory optimization required

## XI. APPLICATIONS

| Component | Specification | Quantity | Function |
|---|---|---|---|
| ESP8266 NodeMCU v3 | 80 MHz, 802.11 b/g/n, 4MB Flash | 1 | Master Control Unit (MCU) |
| MG995 Servo Motor | 11 kg.cm torque, 55g, PWM 50Hz | 4 | Robotic arm joints & gripper |
| L298N Motor Driver | 2A per channel, 635V | 1 | DC gear motor drive control |
| DC Gear Motor | 12V, 150 RPM, 0.8A | 4 | Four-wheel vehicle drive |
| XL6009 Boost Converter | Input 3-32V, Output 5-35V | 1 | Voltage boost for servos |
| XY4015 Buck Converter | Input 5-35V, Output 1.2535V | 1 | Voltage step-down regulation |
| TP4056 Charge Module | 5V USB, 1A charge, overcharge protect | 1 | Li-Ion battery management |
| 3.7V Li-Ion Battery | 3.7V, 3400mAh, 18650 cell | 2 | Portable power for MCU |
| 12V SLA Battery | 12V, 7Ah, sealed leadacid | 1 | Main power for motors |
| SPDT Switch | 10A, 250V AC rated | 1 | System power cutoff |

## XII. FUTURE ENHANCEMENTS

- Internet-Accessible Domain: Replace local mDNS with a globally accessible domain via a VPN tunnel or cloud relay, enabling remote operation from anywhere.
- Battery Management UI: Add real-time State-ofCharge display on the web interface by reading LiIon voltage via ADC.
- Multi-Robot Coordination: Extend the server domain architecture to manage a fleet of robots from a single dashboard using unique subdomains (robot1.local, robot2.local).
- Increased Payload: Upgrade to RS-485 serial servo buses with higher-torque motors (25 kg.cm) to handle payloads up to 2 kg.

## XIII. CONCLUSION

This paper demonstrates a practical and costeffective integration of Wi-Fi transmission via a server domain architecture and voice control for a Pickup and Place Robotic Arm Vehicle. By replacing direct IP access with an mDNS-based domain (robot.local) and augmenting the operator interface with Web Speech API-powered voice commands, the system significantly improves usability, accessibility, and operational flexibility over conventional Wi-Fi robotic designs.

The ESP8266 MCU proves capable of simultaneously managing AP networking, mDNS broadcasting, HTTP REST serving, WebSocket communication, and real-time servo and motor actuation within its constrained resources when firmware is carefully optimized. The layered architecture -- from physical hardware through wireless communication, server domain resolution, and voice user interface -- provides a clear template for further research and commercial development of low-cost intelligent robotic systems.

Future work will focus on extending the system with computer vision, increasing payload capacity, and enabling secure internet-based remote control, progressing the platform from a local lab prototype toward a deployable industrial robotic assistant.

Acknowledgement

The authors would like to express their sincere gratitude to Prof. Sujata Girawale (Project Guide) and Prof. Sushma Patwardhan (HOD, E&TC Department) at Genba Sopanrao Moze College of Engineering, Balewadi, Pune, for their invaluable guidance, continuous support, and

encouragement throughout the academic year 2025-2026. The authors also thank the Electronics and Telecommunication Engineering Department for providing laboratory facilities and hardware resources essential for the development and testing of this system.

## REFERENCES

[1] Ryan Fernandes, Paurash Deboo, Sahil Bhuta, and Giridhar Chavan, "Automated Guided Robotic Armed Vehicle," discussing a prototype of an automated guided vehicle with a mounted robotic arm for improved material handling in industrial settings.
[2] Ivan Grigorov and Georgi Georgiev, "WiFi Control of Robotic Systems via ESP8266," International Journal of Engineering Research, 2022.
[3] W3C Web Speech API Specification, https://wicg.github.io/speech-api/, accessed 2025.
[4] S . Haque et al. – authors of “Development of a Voice-Controlled Robotic Arm” (5-DOF arm controlled by voice, using PC-side voice processing; useful for voice-control logic).
[5] Patel & Sharma – authors of “Wi-Fi Controlled Robot using ESP8266 & Android App”
[6] (showsESP8266-to-Android Wi-Fi control of a robot, relevant for your server-domain Wi-Fi transmission).
[7] Authors of “Development of Robotic Arm Controlling by Using Voice Recognition” (IRJIET, 2026) – focus on a 6-DOF voice-controlled robotic arm using Arduino Nano and offline-voice module; useful for voice-control architecture.