Micro-Rover (Zade):

Build a WiFi-Controlled Robot Powered by a Pocket Power Bank

Introduction

What if you could drive a robot from your phone — without any dedicated controller, radio transmitter, or bulky battery pack? That's exactly what the Micro-Rover (nicknamed Zade) achieves. Powered by nothing more than a standard 5V USB power bank, this compact WiFi-controlled rover runs entirely on battery and can be steered from any web browser on your phone or laptop.

At its heart is an ESP8266 microcontroller — a tiny chip that hosts a full web server and creates its own WiFi hotspot. The rover features a beautiful web dashboard with a virtual joystick, real-time trajectory tracking, a "Draw Mode" where you sketch a path and the rover follows it, and even tilt-to-steer using your phone's accelerometer. All of this runs off a pocket-sized power bank you probably already own.

This project is perfect for the Battery-Powered Contest because it demonstrates how far you can push tiny, efficient hardware: no wall power, no heavy lead-acid cells, just a modern lithium USB power bank keeping an intelligent robot alive and fully operational.

Why Micro-Rover Stands Out

- ☑ Truly wireless — phone browser = remote control, no app installation needed
- ☑ Battery-only — runs off any standard 5V USB power bank
- ☑ Draw Mode — sketch a path and watch the rover trace it
- ☑ Tilt Control — steer by tilting your smartphone
- ☑ Real-time trajectory canvas — see where the rover has been
- ☑ Dual WiFi modes — standalone hotspot or home network integration
- ☑ Python API for programmers — send commands from a PC or Raspberry Pi

Supplies

Hardware

- ESP8266 microcontroller — NodeMCU v1.0 or Wemos D1 Mini (either works)
- Motor driver — L298N, L293D, or DRV8833
- 2WD differential drive robot chassis kit (includes motors, wheels, and frame)
- 5V USB power bank (any capacity — even a slim 5000 mAh bank works)
- USB-A to Micro-USB cable (to connect power bank to ESP8266 / motor driver)
- Jumper wires (male-to-male and male-to-female)
- Small breadboard or screwdriver for motor driver terminals

Software

- Arduino IDE (latest version) — free download from arduino.cc
- ESP8266 board package for Arduino IDE
- The Micro-Rover source code — available on GitHub at github.com/RemanDey/micro-rover
- (Optional) Python 3 with the requests library for advanced scripting

💡 Estimated Cost: The ESP8266 costs around ₹150–250 ($2–3 USD). A basic 2WD chassis kit runs ₹300–500 ($4–6 USD). An L298N motor driver is about ₹100–200 ($1–2 USD). Total project cost can be under ₹1000 (~$12 USD) if you already have a power bank.

Step 1: Assemble the Chassis

Begin by assembling your 2WD robot chassis according to its included instructions. Most kits follow this basic sequence:

1. Attach the two DC gear motors to the chassis frame using the provided screws.
2. Press the rubber wheels onto the motor shafts.
3. Mount the caster wheel (or ball caster) at the front or rear for the third point of support.
4. If using an L298N, mount it on the chassis platform. The L298N module has its own 5V regulator, so it can be powered from the power bank too.

⚠ Safety Tip: Test the rover elevated on a small box (wheels off the ground) before placing it on the floor. This lets you verify motor directions safely.

Step 2: Wire the Electronics

The ESP8266 communicates with the motor driver using four direction pins and two PWM (speed) pins. Connect them as shown in the table below:

ESP8266 Pin	Motor Driver Pin	Function
D1	IN1	Left Motor Forward
D2	IN2	Left Motor Backward
D6	IN3	Right Motor Forward
D7	IN4	Right Motor Backward
D0	ENA	Left Motor Speed (PWM)
D5	ENB	Right Motor Speed (PWM)
GND	GND	Common Ground

Power connections:

- Connect the power bank's USB output to the ESP8266 via the Micro-USB port.
- Power the motor driver separately — connect the power bank's second port (if available) or use the 5V output of the L298N's onboard regulator. Never power the motors from the ESP8266's 3.3V pin.
- Connect a common GND wire between the ESP8266 and the motor driver. This is critical — without it the motors will behave erratically.

Step 3: Set Up the Arduino IDE & Flash the Code

Install the ESP8266 Board Package

5. Open Arduino IDE and go to File > Preferences.
6. In the "Additional Boards Manager URLs" field, paste:

```
http://arduino.esp8266.com/stable/package_esp8266com_index.json
```

7. Go to Tools > Board > Boards Manager, search for esp8266, and click Install.

Download & Configure the Code

8. Download or clone the repository from github.com/RemanDey/micro-rover.
9. Open the file main/main.ino in Arduino IDE.
10. Find and update these two lines with your home WiFi credentials (optional — the rover also works as its own hotspot without this):

```
#define WIFI_STA_SSID  "YourHomeWiFiName"
#define WIFI_STA_PASS  "YourPassword"
```

11. Select your board: Tools > Board > NodeMCU 1.0 (ESP-12E Module) or Wemos D1 Mini.
12. Select the correct COM port under Tools > Port.
13. Click Upload. Wait for "Done uploading" to appear.

The Embedded Web Server

A key feature of this project is that the entire web dashboard (HTML, CSS, and JavaScript) is stored inside the ESP8266's flash memory as a compressed string. When you connect to the rover's IP address, the ESP8266 serves this page directly to your browser — no internet connection needed, no external server. This is what makes it truly battery-powered and self-contained.

Step 4: Connect & Control

Option A — Direct Connection (No Home WiFi Needed)

14. Plug in your power bank to the rover.
15. On your phone or laptop, go to WiFi settings and connect to the network named micro_rover.
16. Enter password: 12345678
17. Open your browser and navigate to: http://micro-rover.local (or check the Serial Monitor at 115200 baud for the rover's IP address if mDNS doesn't resolve).

Option B — Home Network Connection

18. Make sure you entered your WiFi credentials in Step 3.
19. After upload, the rover joins your home WiFi. Check Serial Monitor for its IP.
20. On any device on the same network, browse to http://micro-rover.local or the IP shown.

Step 5: Exploring the Control Modes

Virtual Joystick

Drag the blue circle on the dashboard to steer. The further you drag from center, the faster the rover moves. Release to stop. Works on both touchscreens and mouse.

Keyboard Control

Use W / A / S / D keys or the Arrow keys on a keyboard. Tap a key to move, release to stop. Great for precise manoeuvering.

Draw Mode — The Standout Feature

This is where the Micro-Rover truly shines:

- Click the "Draw Mode: ON" button on the dashboard.
- Use the joystick area or keyboard to draw any path you want — the drawn trajectory appears in yellow on the canvas.
- Click "Follow Path" — the rover will replay your drawn path in the real world.

This feature works by recording the sequence of movement commands you input while in Draw Mode, then replaying them with the same timing. It's simple, elegant, and impressive to demonstrate.

Tilt Control

On a smartphone, tap the Tilt Control toggle. The browser reads your phone's accelerometer data and translates tilting forward/backward/left/right into motor commands. No extra hardware required — just your phone's built-in sensors.

Real-Time Trajectory Canvas

The dashboard continuously draws the rover's estimated path on a canvas, giving you a bird's-eye view of where it has been. The simulation uses the motor command history to calculate approximate position and heading.

Step 6: Python API for Advanced Users

For those who want to automate the rover or integrate it into larger projects, the repository includes Python scripts in the scripts/ folder. These send HTTP requests directly to the rover's REST API.

Install the dependency:

```
pip install requests
```

Run the keyboard-control script from your PC:

```
python scripts/keyboard_control.py
```

Or send raw commands using the API endpoint directly:

```
GET http://<ROVER_IP>/move?left=800&right=800   # drive forward at ~78% speed
```

Values range from -1023 (full reverse) to +1023 (full forward) for each motor independently. This makes it straightforward to write autonomous routines, obstacle-avoidance logic, or control the rover from a Raspberry Pi.

Battery Performance & Power Tips

Power Bank Tips for Maximum Runtime:

- A 5000 mAh power bank will run the rover for approximately 3–5 hours of normal use.
- Ensure the power bank supports continuous low-current output — some shut off automatically when they detect low draw (the ESP8266 is very efficient). Wiggling the motors briefly on startup keeps the power bank awake.
- For extended sessions, an 10,000 mAh bank gives all-day operation.
- The L298N motor driver wastes some energy as heat; the DRV8833 is more efficient and recommended if you want maximum battery life.
- Keep the ESP8266 in Station mode (home WiFi) rather than AP mode to reduce radio power consumption slightly.

How It Works — Under the Hood

The ESP8266 as a Web Server

The ESP8266 runs the ESPAsyncWebServer library to handle HTTP requests asynchronously. This means it can serve the web dashboard to multiple clients and respond to motor commands simultaneously without blocking. The entire HTML/CSS/JavaScript frontend is gzip-compressed and stored in program flash memory (PROGMEM), which is why no SD card or external storage is needed.

Dual WiFi Architecture

The firmware starts both an Access Point (AP) and a Station (STA) simultaneously. In AP mode, the rover broadcasts its own hotspot (micro_rover) so you can always connect directly. In STA mode, it joins your home network, making it accessible from any device on that network. mDNS broadcasting (via the ESP8266mDNS library) lets you use the friendly address http://micro-rover.local instead of memorizing an IP.

Motor Control & PWM

Motor speed is controlled via analogWrite() on the ENA and ENB pins, generating PWM signals that the motor driver converts to variable voltage. Direction is set by the IN1–IN4 pins. Differential steering (like a tank) is achieved by varying the speed of the left and right motors independently — turn left by slowing the left motor, turn right by slowing the right.

Troubleshooting & Tips

- Motors spin in the wrong direction? Swap the wire pairs on that motor at the terminal block (or swap IN1 and IN2 in the code).
- Can't reach http://micro-rover.local? Use the IP address shown in the Serial Monitor at 115200 baud instead.
- Power bank shuts off automatically? Add a small load (e.g., an LED) or use a power bank known to support "always-on" mode.
- WiFi connection unstable? Move the rover closer to the router, or use AP mode (direct connect) for the most reliable connection.
- Draw Mode path is inaccurate? The trajectory simulation is open-loop (no encoders). Accuracy improves on smooth, flat surfaces.

Conclusion

The Micro-Rover (Zade) proves that a remarkable amount of capability can run on a pocket-sized power bank. With a web dashboard that works on any browser, multiple control modes including the clever Draw Mode, and a clean REST API for programmers, this is a project that grows with your interests — from a weekend build to a platform for learning robotics, computer networking, and embedded systems.

All source code is open-source under the MIT License. Fork it, modify it, extend it — and most importantly, have fun driving it.

🔗 **GitHub: github.com/RemanDey/micro-rover**