PORTFOLIO CASE STUDY

Follow-Me Wagon

Autonomous Robot: From Idea to Working Prototype

May 2026 — June 2026

18 hrs	~$400	1 Month	0 → Live
Active Build Time	Total Parts Cost	Concept to Demo	Prior Experience

Section 1: What Was Built

The Follow-Me Wagon is an autonomous robotic vehicle built entirely from off-the-shelf consumer components. It tracks the operator's phone location using GPS, steers toward that position using a phase-based control algorithm, and operates through a four-mode state machine: Follow, Manual, Park, and Neutral.

The robot uses a LewanSoul/Hiwonder tank chassis as its drive platform, a Raspberry Pi 4 as its onboard computer, a NEO-6M GPS module for self-localization, and a phone-based browser interface that streams the operator's GPS coordinates to the Pi over WiFi. The operator's phone requires no app — a standard browser handles all communication.

The project was built as an explicit portfolio piece demonstrating one clear capability: taking a complex technical idea from zero to a working physical product in under twenty hours of active work, within a defined budget, with no prior experience in any relevant domain.

Core Capability

A proof-of-concept autonomous robot — designed, wired, programmed, tested, and demonstrated — built by someone with zero prior robotics, electronics, or programming experience, using AI as the primary technical guide throughout.

Section 2: Why It Was Built

The project was driven by a straightforward challenge: could a complete beginner take an idea for a physical autonomous robot and deliver a working prototype within a fixed time budget and a modest dollar budget — using AI as the expert guide?

The original concept was even more ambitious: an underwater sand-burrowing GPS-guided robot. AI analysis quickly revealed that concept was far beyond the scope of a first hardware project. The Follow-Me Wagon emerged as the right first project — technically meaningful, physically achievable, and demonstrable on camera.

The purpose was simultaneously personal, experimental, and portfolio-driven. Personally, it was a test of discipline and follow-through on an unfamiliar challenge. Experimentally, it was a structured test of whether AI-assisted learning could compress years of domain education into weeks of focused problem-solving. As a portfolio piece, it was designed to show a selection board not just a finished product, but the entire arc of how a non-expert builds something real.

Core Question

"Can someone with zero robotics, electronics, or programming experience deliver a working autonomous robot — on time, on budget, on purpose — through effective AI collaboration?"

Section 3: Starting Competence

Prior Robotics Experience	None — first hardware project of any kind
Prior Electronics/Wiring	None — first time crimping, wiring, or building circuits
Prior Programming Experience	None — first time writing or deploying code
Prior Multimeter Use	None — acquired and learned during this project
Hardware Assembly	Basic — personal computer assembly and general hand tools
AI Tool Experience	Growing — Claude used across multiple prior portfolio projects
Background	Primary job with demanding schedule; project built on weekends only
Skills Brought In	Structured planning, disciplined scoping, and the ability to ask precise questions

This project was executed at the outer edge of the builder's experience in every technical domain it touched. The only relevant prior skill was a disciplined approach to planning and problem decomposition — and the willingness to be uncomfortable.

Section 4: How AI Was Used

Ideation, Scoping & Design

The first eight hours of the project were not spent building anything. They were spent in structured conversation with Claude, working through the single hardest question in the project: how does a robot know where you are, where it is, and which direction to turn?

This phase evaluated four tracking approaches in depth — AirTag-based UWB ranging, multi-device Bluetooth RSSI, direct UWB ranging, and a custom iOS app — before ruling all of them out on technical grounds. AirTags are cryptographically locked to Apple's ecosystem. Bluetooth

RSSI is too noisy for moving outdoor targets. UWB moving-anchor drift is a fundamental physics problem, not a fixable implementation detail. A native iOS app would have added two to three weeks of Swift development.

The winning approach — streaming phone GPS coordinates via the browser's Geolocation API to a Flask web app on the Pi over WiFi — eliminated the iOS development tangent entirely and was the single highest-leverage decision in the project. That eight-hour investment in design review prevented weeks of wasted work and hundreds of dollars in parts that would not have delivered the intended result.

Key Workflow Insight

Eight hours of upfront design discussion with AI prevented weeks of failed experiments. Every rejected tracking method — Bluetooth, WiFi triangulation, AirTag UWB, native iOS — was eliminated through reasoning before a single dollar was spent or a single wire was connected.

Pre-Procurement Review

Before the parts order was placed, Claude reviewed the full hardware design and flagged four critical compatibility issues that would have caused post-purchase failures:

- The HC-SR04 ultrasonic sensor's echo pin outputs 5V, which would have damaged the Pi's 3.3V GPIO without a voltage divider
- The BTS7960 motor driver's logic input required a TXS0108E level shifter for 3.3V compatibility
- The Pi has only two hardware PWM channels; the planned motor configuration required four, necessitating a PCA9685 PWM driver
- Compass placement near the motors would produce garbage readings due to magnetic interference

This review took approximately one day and prevented a week or more of post-purchase debugging. It also surfaced that the LewanSoul/Hiwonder tank chassis included a pre-installed integrated motor controller — a discovery that eliminated three planned components (BTS7960, TXS0108E, PCA9685) worth approximately $35, before the order was placed.

Build, Wiring & Hardware Diagnostics

Throughout the physical build, Claude functioned as an expert on call. The workflow was direct: encounter an unknown, take a screenshot, send it to Claude, receive a diagnosis and a specific corrective action. This covered wiring connections, Raspberry Pi GPIO pin assignments, I2C device detection, battery chain construction and multimeter verification, SFTP file transfers, and SSH terminal commands.

When the right motor failed mid-build, Claude guided a structured diagnostic sequence: port swap, cable swap, motor resistance measurement with a multimeter, and hand-rotation test. The diagnosis — partial motor failure based on asymmetric winding resistance — was captured with supporting data and submitted as a warranty claim. Hiwonder shipped a full chassis replacement.

When the replacement chassis arrived with a bent drive sprocket, the same diagnostic discipline was applied: photograph for warranty evidence, isolate the bend, assess repair viability, execute the fix by hand on a flat surface. The repair held.

Software Development & Real-Time Iteration

All Python code running on the Raspberry Pi was developed in real-time collaboration with Claude. The operator connected to the Pi via SSH using the Termius app on an iPhone, submitted problems and error outputs as screenshots, and received patched code in return — deployed via SFTP for multi-line files and sed commands for single-line tweaks.

The Follow mode control algorithm went through three full architectural rewrites and approximately six rounds of outdoor tuning. The core problem — deriving the wagon's heading from GPS position-delta rather than a compass — introduced significant noise at low speeds and produced over-rotation during turns. The solution was a phase-based controller: rather than correcting heading continuously at 10Hz, the wagon commits to a straight-driving phase for at least 2.2 seconds, then executes a time-locked arc, then returns to straight. This eliminated the twitchy overcorrection and produced stable following behavior.

Key Workflow Insight

The operator was outside with a phone, SSH'd into the Pi, describing in real time what the robot was doing wrong. Claude was writing code patches in response. The gap between 'the wagon is spinning in place' and 'here is the corrected controller logic' was minutes, not days.

Pivots & Scope Decisions

Several planned components were deferred to a future version after structured evaluation confirmed they were not required for the MVP:

- Compass (QMC5883L): All three boards from a purchased 3-pack reported the wrong I2C address — a clone product. Rather than reverse-engineer the protocol, heading was derived from GPS position-delta. Compass deferred to v2 with a specific replacement identified (Adafruit HMC5883L).
- Ultrasonic obstacle detection (HC-SR04): Dropped from v1 after weighing integration time against the low collision risk at operating speeds. Mitigated by speed caps and operator vigilance.
- LED status indicators, buzzer, and current sensing: Deferred — not blocking for MVP demonstration.
- Pi WiFi hotspot: Current build uses home WiFi. Field-portable independence is a v2 goal.

The form factor also pivoted mid-build. The original plan was to retrofit a Radio Flyer wagon by replacing its rear wheel assembly with the tank chassis. On inspection, fitting the chassis to the wagon body would have consumed significant time with zero demo value. The decision was made to use the tank chassis as the entire vehicle, with a basket bolted on top as the payload bay. This eliminated a multi-step mechanical subproject and was the right call.

Section 5: Timeline & Conditions

Project Start	May 2026 — first substantive design work
Demo Published	June 3, 2026
Total Duration	Approximately one month
Active Build Time	~18 hours
Work Rhythm	Weekends only — primary job and family obligations ran concurrently
External Deadline	None — self-imposed success criteria
Competing Demands	Primary job responsibilities, family projects, and ongoing personal education
Sessions Estimated	8–12 AI sessions across the project, with handoff documents at key milestones

The project was not continuous. Work occurred in weekend sessions with gaps between them. At each re-entry point, prior design decisions were documented in handoff summaries — a practice that made resumption fast and preserved the reasoning behind every pivot for the eventual portfolio narrative.

Section 6: Outcome & Evidence

The Follow-Me Wagon is a complete, working autonomous robot. It follows the operator's phone via GPS, executes a four-mode state machine with proper failsafes, is controlled entirely through a phone browser with no app required, and was delivered within the 20-hour active work target in approximately one month of weekend effort.

The success criteria set at the outset — it exists, it works, people see the value — were met in full. The demo video shows the robot operating in Follow mode, tracking the operator across open ground, executing turns, and stopping at the configured distance.

The project also generated a set of documented v2 upgrade paths: compass integration for closed-loop heading, ultrasonic obstacle detection, LED and buzzer status indicators, current sensing for battery monitoring, and Pi-hosted WiFi access point for field-portable operation. These are deferred features, not abandoned ones — each has a specific implementation path captured.

Demo Video: https://youtu.be/O5EXJdPDLwc

Section 7: Key Metrics

Chassis	LewanSoul/Hiwonder tank chassis with integrated Encoder Motor Module v1.4
Compute	Raspberry Pi 4 (4GB RAM)
Tracking Method	Phone GPS via browser Geolocation API → Flask app on Pi over WiFi
Heading Derivation	GPS position-delta (compass deferred to v2)
Operating Modes	Follow, Manual, Park, Neutral
Control Loop Rate	10 Hz
Follow Mode Speed (final)	55/100 — speed required for reliable GPS position-delta heading
Stop Distance	2.0 meters from operator
Resume Distance	3.0 meters (after stop, if operator walks away)
GPS Accuracy (outdoor)	±2–5 meters (phone and NEO-6M)
Motor Power	12V SLA battery (Mighty Max ML7-12, 7.2Ah) — isolated from Pi power
Pi Power	USB power bank — isolated from motor rail
Active Build Time	~18 hours
Total Duration	~1 month (May–June 2026), weekends only
Parts Cost (used)	~$372 pretax / ~$400 with tax
Original Budget Target	$180
Total Ordered	~$464 (includes unused/deferred components)
Control Algorithm	Phase-based: straight (≥2.2s) → time-locked arc → straight
Algorithm Versions	3 full rewrites (v3.0 → v3.1 → v3.2 → v3.3), ~6 outdoor tuning rounds
Tracking Methods Rejected	4 (AirTag UWB, direct UWB, BLE RSSI, native iOS app)
Hardware Failures	1 motor failure (warranty replaced), 1 bent sprocket (repaired)
Components Eliminated	BTS7960, TXS0108E, PCA9685 — replaced by chassis's integrated controller
v2 Upgrades Documented	Compass, ultrasonic, LED/buzzer, current sensing, Pi WiFi hotspot
Demo	https://youtu.be/O5EXJdPDLwc

Section 8: Lessons Learned

Lesson 1: The First Eight Hours Are the Most Valuable

The single most important realization from this project was how much leverage exists in the design phase before anything is built or bought. Eight hours of structured conversation —

working through how the robot would know its location, the operator's location, and which direction to turn — identified and eliminated four tracking approaches that would have failed. That investment prevented weeks of wasted build time and hundreds of dollars in parts that would not have achieved the goal. Front-loading the hard thinking is not a delay. It is the work.

Lesson 2: AI as Expert on Call Changes the Build Experience

Having an expert available in real time — one who could receive a screenshot of an unfamiliar error, diagnose the root cause, and return a specific corrective action in minutes — compressed what would otherwise be days of forum research into hours of focused execution. The experience was not "AI did the work." It was "AI made it possible to do work that would otherwise have been inaccessible." The operator still had to crimp every wire, read every multimeter result, carry the chassis outside, and observe what the robot actually did. AI cannot see your hardware. You are the only one who can.

Lesson 3: Hardware Fails — That Is Normal, Not a Problem

A motor failed mid-build. A replacement chassis arrived with a bent sprocket. Compass boards turned out to be non-standard clones. Each of these was frustrating in the moment and entirely normal in any hardware project. The response to each was the same: isolate the problem methodically, document the findings, make the best available decision, and continue. Becoming comfortable with the uncomfortable is not a soft skill in hardware development. It is the core skill.

Lesson 4: Scope Discipline Is an Engineering Decision

Every feature that was deferred — the compass, the ultrasonic, the LED indicators, the Pi hotspot — was a deliberate decision, not a failure. The question for each was: can the MVP succeed without this? When the answer was yes, the feature moved to v2 with a documented implementation path. Holding the line on "it exists, it works, people see the value" allowed the project to ship. A robot that demonstrates autonomous following on camera is more valuable than a more capable robot that is still being built.

Lesson 5: You Don't Have to Know What You're Doing to Figure Out How to Do It

The most honest summary of this project is also its most transferable lesson. Every domain in this build — robotics, electronics, wiring, GPS, Python, control theory — was new. None of it was waited on. The approach was to ask precise questions, understand the answers well enough to make decisions, and execute. That cycle, repeated across eighteen hours of active work, produced a working autonomous robot from a standing start. The barrier to entry on hard technical problems is not knowledge. It is the willingness to start without it.

Future Upgrades (v2 — Documented, Not Executed)

The following improvements are planned for a future version. Each was deferred from v1 by deliberate scope decision, not abandoned:

Compass Integration	Replace GPS position-delta heading with a genuine Adafruit HMC5883L compass (~$15). Enables accurate closed-loop heading at any speed, eliminates the current speed-floor constraint for reliable following.
Ultrasonic Obstacle Stop	HC-SR04 with voltage divider already in the parts inventory. Adds automatic stop when a physical obstacle is detected — removes the current reliance on operator vigilance at close range.
LED Status Ring	Visual indicator for current operating mode (Follow vs. Park vs. Manual). Deferred as non-blocking for the MVP demo.
Buzzer Alert	Audio alert on connection loss or GPS signal dropout. Simple addition post-MVP.
Current Sensing	Battery voltage monitoring via ADC. Enables low-voltage cutoff and removes the current manual monitoring requirement.
Pi WiFi Hotspot	Pi creates its own access point so the demo runs anywhere without home WiFi. Currently deferred — build assumes home WiFi reaches the test area.

What This Project Proves

- Zero-to-prototype execution: A working autonomous robot was designed, wired, programmed, and demonstrated in 18 hours of active work across one month — with no prior experience in any relevant domain.
- Front-loaded design discipline: Eight hours of pre-build AI-assisted design review prevented weeks of failed experiments and hundreds of dollars in unusable parts.
- Real hardware problem-solving: Motor failure diagnosis, warranty claims, bent sprocket repair, and clone component identification — all executed by a first-time hardware builder through structured troubleshooting.
- Adaptive algorithm development: Three architectural rewrites of the Follow mode controller, six outdoor tuning rounds, and a phase-based control breakthrough — all developed through real-time AI collaboration in the field.
- Honest capability documentation: Every limitation is named, every deferred feature has a path forward, and the demo shows exactly what works — a credibility signal as important as the technical achievement itself.

--- END OF CASE STUDY ---

Demo: https://youtu.be/O5EXJdPDLwc