ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

OmniSense-Dual

A Wearable Pedestrian Safety and Navigation System

Team 10

ALEX JIN (jin50@illinois.edu)

JIATENG MA (jiateng4@illinois.edu)

SIMON XIA (hx17@illinois.edu)

TA: Wesley Pang

Project No.: 10

May 2026

Abstract

OmniSense-Dual is a battery-powered, two-module wearable that provides pedestrians with real-time hazard awareness and turn-by-turn walking guidance through spatially separated haptic feedback. The system fuses short-range time-of-flight (ToF) sensing, long-range mmWave radar, and inertial measurement (IMU) data into an eight-direction obstacle field delivered on a head-worn ring, while a phone-driven navigation belt on the waist module guides the user along a Google Maps walking route. A custom Flask server performs the per-direction safety-first sensor fusion, IMU tilt compensation, hazard-zone classification, and motor encoding, and exposes a structured logging interface that maps one-to-one to the project's requirements-and-verification table. The final prototype was demonstrated in two real walking sessions; over 1,000 sensor packets were transmitted with zero malformed packets and a median server-side processing latency below 20 ms, while channel separation between hazard and navigation feedback was preserved throughout. This report documents the project's motivation, design decisions, verification results, costs, and ethical considerations.

Contents

Abstract ii

1 Introduction 1
- 1.1 Problem and Motivation . . . 1
- 1.2 Solution Overview and Block Diagram . . . 1
- 1.3 High-Level Requirements . . . 1
- 1.4 Block-Level Changes Since the Design Document . . . 2

2 Design 3
- 2.1 Design Procedure . . . 3
  - 2.1.1 Sensing Subsystem . . . 3
  - 2.1.2 Sensor Fusion and Hazard Classification . . . 3
  - 2.1.3 Haptic Encoding and Channel Separation . . . 4
  - 2.1.4 Wireless Communication . . . 4
  - 2.1.5 Navigation Pipeline . . . 5
  - 2.1.6 Power Management . . . 5
- 2.2 Design Details . . . 5
  - 2.2.1 Hardware Block Layout . . . 5
  - 2.2.2 Sensor Sampling and Validity . . . 7
  - 2.2.3 Hybrid Sensor Fusion Algorithm . . . 7
  - 2.2.4 Latency Budget Analysis . . . 8
  - 2.2.5 Server Architecture and Logging . . . 8
  - 2.2.6 Frontend and Test Suite . . . 8

3 Design Verification 10
- 3.1 Verification Methodology . . . 10
- 3.2 Sensor-Level Bench Results . . . 10
- 3.3 End-to-End Quantitative Results . . . 11
- 3.4 High-Level Requirement Verification . . . 11
- 3.5 Uncertainties and Failed/Out-of-Budget Items . . . 12

4 Costs and Schedule 13
- 4.1 Parts Cost . . . 13
  - 4.1.1 Bulk-Production Estimate . . . 13
- 4.2 Labor Cost . . . 13
- 4.3 Schedule . . . 14

5 Conclusions 15
- 5.1 Accomplishments . . . 15
- 5.2 Uncertainties and Limitations . . . 15
- 5.3 Future Work and Alternatives . . . 15
- 5.4 Ethical Considerations and Broader Impacts . . . 16

References 17

Appendix A Requirements and Verification Table **18**

Appendix B Hardware Reference **20**
B.1 Eight-Direction Motor Layout 20
B.2 ESP32-S3 Pin Allocation (Head Module) 20
B.3 Subsystem Photographs 20

1 Introduction

1.1 Problem and Motivation

Pedestrians in dense urban and campus environments share constrained sidewalks with bicycles, electric scooters, motor vehicles, and other walkers. Unlike drivers, they have no rear-view tools, and modern distractions—headphones, smartphones, and turn-by-turn voice guidance—further degrade situational awareness exactly when it is most needed. Audio-only navigation is also poorly suited to visually impaired users who already rely heavily on sound to perceive their environment, and on-screen maps demand a sustained visual attention budget that pedestrians cannot safely afford.

OmniSense-Dual addresses these gaps with a single, hands-free, eyes-free wearable that delivers two distinct streams of information through separated haptic channels: hazard alerts felt at the head and navigation cues felt at the waist. By exploiting the human ability to discriminate vibration location, the system removes the cognitive cost of switching between a phone screen and the surrounding environment without adding to the auditory load.

1.2 Solution Overview and Block Diagram

The deployed system consists of two coordinated wearable modules and one external host. The waist module integrates an ESP32-S3 microcontroller, eight VL53L1X time-of-flight (ToF) sensors arranged radially around the belt, two Seeed Studio C4001 24 GHz mmWave radars (front and back hemispheres), one SparkFun ICM-20948 9-DoF IMU, an on-board GPS receiver, and an eight-motor navigation belt driven by DRV2605L haptic drivers. The head module shares the same MCU and sensing topology (minus GPS) and replaces the navigation belt with an eight-motor hazard ring. A Python Flask server running on a host computer fuses the sensor packets from both modules, classifies each direction into a hazard zone, encodes the resulting motor commands, and serves the Google Maps based navigation web frontend.

The two modules and the host communicate over Wi-Fi using a JSON HTTP protocol. Each module pushes a sensor packet to the server roughly every 100 ms (10 Hz target), and the server responds with module-specific haptic decisions. Figure 1 shows the top-level system block diagram.

1.3 High-Level Requirements

The OmniSense-Dual system was designed to satisfy three top-level requirements; the verification of each is summarized in Section 3 and the full Requirements and Verification (RV) table is reproduced in Appendix A.

1. **Safety (hazard detection).** The system shall detect obstacles approaching from any of eight compass directions within a usable range of about 5 m using ToF and 12 m using

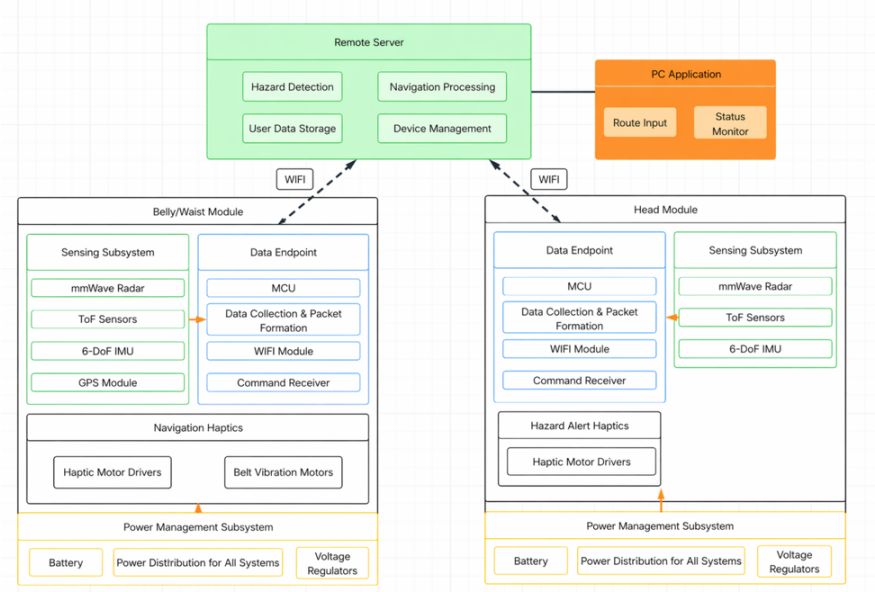


Figure 1: Top-level system block diagram. Each wearable module bundles a sensing block (ToF, mmWave, IMU, plus GPS on the waist module), an MCU-based control block, and a haptic output block; both modules share a host running the fusion algorithm and the Maps-based navigation frontend.

mmWave, classify the encountered hazard into one of six severity zones, and convert the result into directionally accurate haptic alerts on the head ring.

2. **Navigation guidance.** The waist module shall deliver non-visual, turn-by-turn navigation cues that follow a user-supplied Google Maps route, with a directional command refresh rate sufficient to keep the user on path during typical walking speeds.
3. **Channel separation.** The head and waist haptic channels shall be physically and semantically separate at all times so that a user can distinguish a navigation cue from a hazard alert without ambiguity, even when both fire concurrently.

1.4 Block-Level Changes Since the Design Document

Two block-level changes were made between the design document and the final implementation. First, sensor fusion was centralized on the host rather than distributed across the two MCUs; consistent cross-module hazard prioritization (for example, when a body-level chair and a head-level overhang appear simultaneously) was much easier to express in a single algorithm with a global view of both packet streams. This is functionally equivalent for the user but makes the "head delivers hazards only, waist delivers navigation only" requirement easier to enforce. Second, the two C4001 radars were mounted to give a $\pm 67.5^\circ$ azimuthal cone covering the front, front-left, front-right, back, back-left, and back-right directions; pure side directions (left and right) are intentionally covered by ToF only because lateral approaches in pedestrian environments are slow enough that the 5 m ToF range provides sufficient warning, and adding two more radars increased cost.

2 Design

This chapter is organized in two halves. Section 2.1 presents the design procedure block-by-block: what alternatives we considered, what we chose, and why. Section 2.2 gives the design details: parts, equations, schematic-level choices, and the firmware/server software stack.

2.1 Design Procedure

2.1.1 Sensing Subsystem

The sensing subsystem must answer two distinct questions: *is there something within reach?* (close-range geometry) and *is something approaching me from far away?* (long-range presence and motion). No single inexpensive sensor answers both. Among short-range options, ultrasonic sensors were rejected for their slow update rate and wide beam-width that destroys the eight-direction resolution required by the hazard ring, and infrared rangefinders were rejected for their sensitivity to ambient light. Among long-range options, camera-based detection was rejected because it requires non-trivial on-device compute, performs poorly in low light, and raises significant privacy concerns when worn in public.

We therefore selected the STMicroelectronics VL53L1X laser-class ToF sensor for short-range distance per direction, and the Seeed Studio C4001 24 GHz mmWave radar for long-range presence detection per hemisphere. The IMU (SparkFun ICM-20948 9-DoF on the I^2C bus) provides heading and tilt for both navigation and tilt-based suppression of spurious ground reflections, and the waist module hosts an on-board GPS receiver for outdoor positioning.

We set eight directions (45° apart) in haptic compass research: cardinal-only feedback (front, back, left, right) leaves perceptual ambiguity for diagonal hazards, whereas sixteen directions exceed the spatial resolution that untrained users can reliably classify on a worn ring. Eight motors are very intuitive, and also map cleanly onto a single eight-channel multiplexer.

2.1.2 Sensor Fusion and Hazard Classification

The fusion logic could live in either MCU, be split between them with peer exchange, or run on a single off-body host. The on-device options keep all data local but require synchronizing the two MCUs over Wi-Fi or a wired link, doubling the firmware complexity and the failure modes that need to be handled. We therefore chose to centralize fusion on the Flask host: both MCUs push their sensor packets at 10 Hz to the host, the host runs the fusion pipeline, and the motor decision is returned in the same HTTP response. This places the cross-module decision in a single, testable code path at the cost of a one-way network round-trip.

For every direction the algorithm always uses the smaller of the ToF and mmWave readings; we never suppress a hazard because the two sensors disagree. The same rule applies

across modules: whichever module sees the closer obstacle in a direction wins. Once the per-direction effective distance is known, the result is classified using the head-module thresholds because the output motors are physically on the head ring.

2.1.3 Haptic Encoding and Channel Separation

Two encoding axes carry information to the user: *which* direction is alerting and *how urgent* the alert is. Direction is naturally encoded by which motor in the ring fires. Urgency could be encoded by amplitude, frequency, or temporal pattern; on the breadboard prototype we found that single-axis encodings (amplitude only, or pattern only) made it hard for users to distinguish more than three or four levels. We therefore adopted a multi-axis encoding that varies all three (amplitude, pulse frequency, and pattern) jointly across six zones. Table 1 shows the final mapping from effective distance to motor command, applied identically by both modules. The head module uses a slightly wider CRITICAL band (0.6 m instead of 0.5 m) because head-height obstacles are harder to perceive visually and require an earlier reaction window.

The two haptic channels are physically separate (head ring vs. belt) and are driven by separate code paths: hazard motor commands originate from the host's combined-packet hazard pipeline, while navigation belt commands come from the phone-resolved navigation endpoint pushed by the frontend. The body firmware additionally silences the navigation belt within 2 s of a server timeout, so a hazard alert can never be confused with a stale navigation cue.

Table 1: Hazard-zone encoding shared by both modules. The head module uses a 0.60 m CRITICAL boundary; all other thresholds are common.

Distance range (m)	Zone	Pattern	Intensity	Frequency	Description
$[0.00,\ 0.50)$	CRITICAL	continuous	255	50 Hz	Stop now
$[0.50,\ 1.50)$	DANGER	rapid pulse	204	20 Hz	Step away
$[1.50,\ 3.00)$	WARNING	medium pulse	153	10 Hz	Slow down
$[3.00,\ 5.00)$	CAUTION	slow pulse	102	5 Hz	Be aware
$[5.00,\ 12.0)$	ALERT	very slow	51	2 Hz	Approaching object
≥ 12.0	CLEAR	off	0	0 Hz	No hazard

2.1.4 Wireless Communication

Bluetooth Low Energy was the obvious low-power candidate but the ESP32-S3 BLE stack imposes per-packet payload limits that would have required us to fragment the JSON sensor packet and reassemble it on the host; Wi-Fi Direct and LoRa were ruled out for power and latency respectively. We chose Wi-Fi (TCP) with a JSON HTTP POST per packet because it meshes naturally with the Flask host and the web frontend, and because both

ESP32-S3 modules already carry a Wi-Fi radio. Each packet is a self-contained, schema-validated JSON object with a checksum implicit in the TCP layer, so out-of-order delivery is impossible.

2.1.5 Navigation Pipeline

A turn-by-turn engine could be written from scratch on the MCU, ported as a phone-side library, or delegated to an external mapping service. The first two options require us to build, test, and own the routing algorithms, whereas the third lets us reuse a production-quality service. The Google Maps Directions API therefore computes the route on the phone or PC; the frontend pushes the resolved waypoint list and a per-tick directional command to the host, which the waist module consumes through the navigation endpoint. The on-device GPS on the waist module is the primary positioning source: it streams coordinates to the host so that the algorithm can compare current position with the next waypoint and choose the appropriate belt motor.

2.1.6 Power Management

A single 18650 cell is robust but adds bulk; a 3.7 V flat LiPo provides similar capacity in a much thinner profile. A USB-tethered design was rejected outright because the entire premise of the system is wearability. Each module is therefore powered by a 3.7 V 2000 mAh LiPo cell, with a buck-boost regulator producing 3.3 V for the MCU and sensors and a separate boosted 5 V rail for the haptic drivers, plus an MCP73831T charge controller. The two motor groups are separated from the MCU rail by their own decoupling stage to suppress the several-hundred-mA transients that the eight DRV2605L drivers can inject into the supply.

2.2 Design Details

This section presents the as-built hardware and software, organized by subsystem so each Design Procedure choice in Section 2.1 can be matched to an implementation.

2.2.1 Hardware Block Layout

Figures 2 and 3 show the component layout of the waist and head modules. Both are centred on an ESP32-S3-WROOM-1 with 8 MB flash and share the same I^2C-mux plus DRV-mux topology so a single firmware skeleton can drive either ring; the waist module additionally hosts the GPS receiver (UART) and drives the navigation belt instead of a hazard ring.

Two signal-integrity decisions are baked into the firmware bring-up sequence. First, all eight VL53L1X ToF sensors share the same default I^2C address (0x29); at boot the firmware pulls each sensor's XSHUT line low through the I^2C multiplexer, brings them up one at a time on a unique multiplexer channel, and re-enables the next sensor. This permits the two-mux topology with off-the-shelf breakouts. Second, the eight DRV2605L drivers can each draw ~ 50 mA at peak, so a full-ring CRITICAL alert can pull ~ 400 mA.

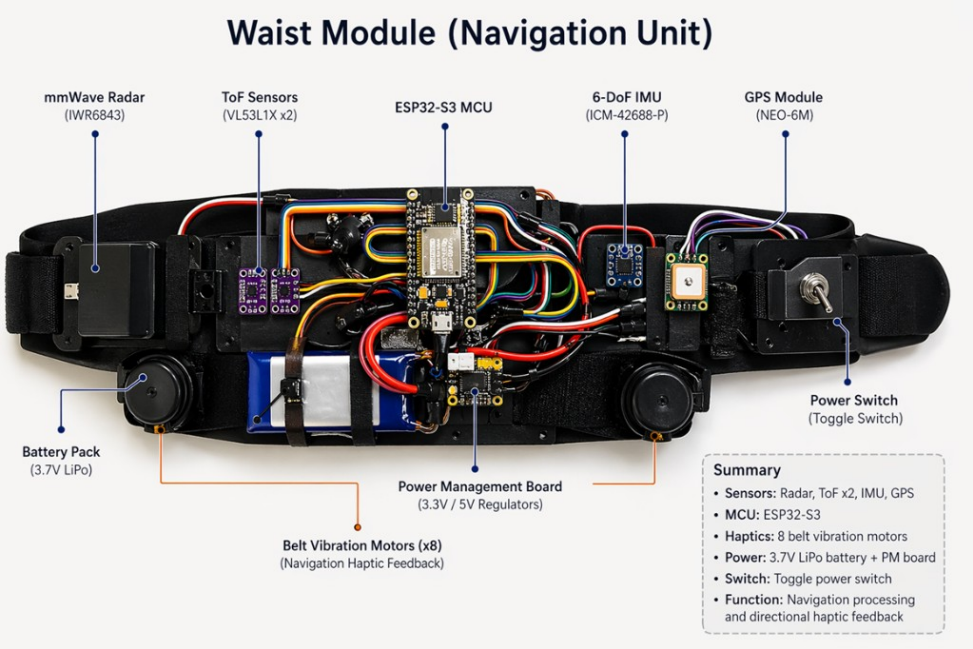


Figure 2: Waist module component layout. ESP32-S3 communicates with eight VL53L1X ToF sensors and the IMU through a TCA9548A I^2C multiplexer; two C4001 mmWave radars use UART; the GPS receiver uses its own UART; and a second TCA9548A drives eight DRV2605L haptic drivers.

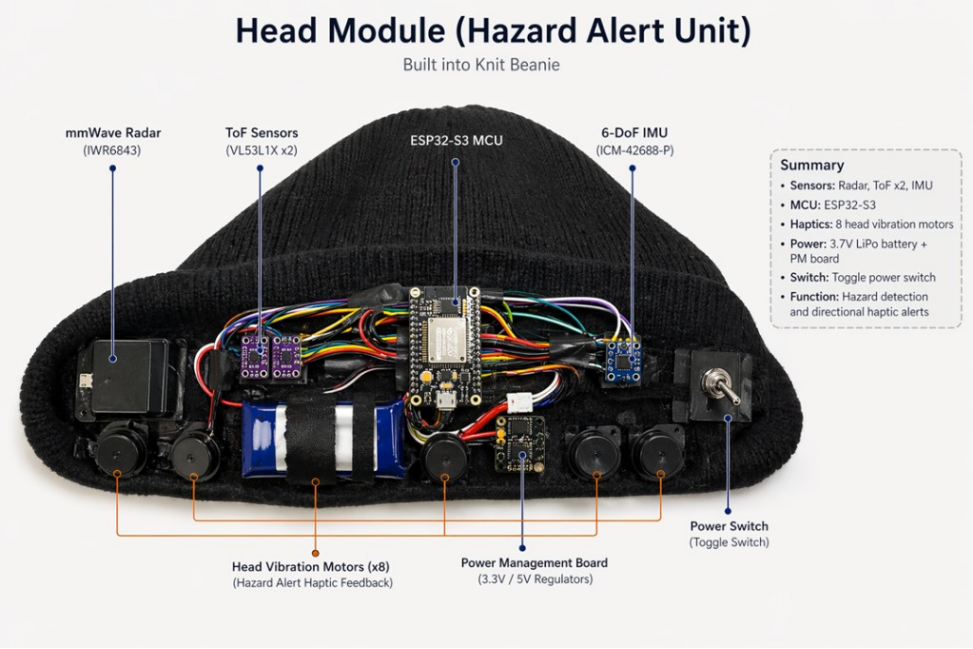


Figure 3: Head module component layout. Identical sensing topology to the waist module (minus the GPS) and an eight-motor hazard ring instead of the navigation belt.

The DRV multiplexer is on its own decoupled supply branch, separate from the MCU and sensor rails, to keep CRITICAL pulses from causing voltage sag on the I²C bus.

2.2.2 Sensor Sampling and Validity

The firmware samples sensors at the rates listed in Table 2. These rates are the upper bound the hardware can sustain; the bottleneck for the system as a whole is the Wi-Fi round-trip, not the sensors themselves (see Section 3). The host parser discards out-of-band readings before they enter the fusion stage, and the automated test suite covers every boundary explicitly.

Table 2: Per-sensor sampling rates and validity rules used in the firmware and host parser.

Sensor	Bus / Driver	Sample rate	Validity rule
VL53L1X ToF (8)	I²C, MUX 0x77	50 Hz raw	40 mm $\leq d \leq$ 4000 mm; sentinel 8190 mm rejected; range capped at 5 m
C4001 mmWave (2)	UART (115200 bps)	20 Hz raw	presence=true; $0 < d \leq 12$ m
ICM-20948 IMU	I²C	100 Hz raw	WHO_AM_I check at boot
GPS (waist)	UART	1 Hz fix	Standard NMEA fix-quality field

2.2.3 Hybrid Sensor Fusion Algorithm

For every direction d in {front, front-right, right, back-right, back, back-left, left, front-left}, let $T_h(d)$ and $T_b(d)$ be the validated ToF distances from the head and body modules, and let $M_h(d)$ and $M_b(d)$ be the mmWave hemisphere readings projected onto d (or $\perp$ if d is left or right). The per-module effective distance is

$$E_m(d) = \begin{cases} \min(T_m(d),\, M_m(d)) & \text{if both valid} \\ T_m(d) & \text{if only ToF valid} \\ M_m(d) & \text{if only mmWave valid} \\ \perp & \text{otherwise} \end{cases} \tag{1}$$

for $m \in \{\text{head}, \text{body}\}$, and the cross-module merge is $E(d) = \min(E_h(d),\, E_b(d))$ when both are valid, falling back to whichever side is valid otherwise. The resulting $E(d)$ is mapped to a hazard zone using Table 1 and the zone is encoded into the eight-motor command vector through the amplitude/frequency/pattern triple in the same table.

IMU tilt compensation is applied per-module *before* the merge: when |pitch| $> 15^\circ$ the front and back channels of that module are dropped, and when |roll| $> 15^\circ$ the left and

right channels are dropped. Diagonals are preserved because single-axis tilt rarely misaligns them. This avoids the case where, for example, a forward-leaning user generates spurious 0.5 m front readings from the floor and triggers a false CRITICAL.

2.2.4 Latency Budget Analysis

A tolerance analysis in the design document established the safety margin required between hazard detection and user response. Consider a user walking at 1.5 m/s while a scooter approaches from behind at 6 m/s; the relative closing speed is $v_{\text{rel}} = 7.5$ m/s, and at the 5 m ToF range the time-to-collision (TTC) is $5/7.5 \approx 0.67$ s. The system was designed for an end-to-end latency budget of 200 ms from sensor sample to motor activation, leaving $0.67 - 0.20 = 0.47$ s for the user to perceive the haptic alert and react—comfortably above the 250 ms typical human reaction time for trained haptic stimuli. Even at the more pessimistic 4 m detection distance, the available reaction window (~ 0.33 s) remains positive. Section 3 shows that the measured server-side share of this budget is well under the 200 ms target, but the network share alone consumed most of it on the consumer Wi-Fi access point we used during testing.

2.2.5 Server Architecture and Logging

The Flask host is a thin HTTP layer over the fusion algorithm. The relevant endpoints are summarized in Table 3. Every state transition emits a structured log line; each tag maps one-to-one to a row in the design-document R&V table (see Table 4 for the summary and Appendix A for the full mapping). This logging discipline is what allows post-hoc verification of any recorded session by replaying the log file through the dashboard, with no need to re-run the hardware.

2.2.6 Frontend and Test Suite

The frontend is a static HTML+ES2020 application served from the Flask host. It exposes a turn-by-turn navigation map, a combined live dashboard, and per-module replicas of the head ring and body belt; all four pages can drag-and-drop a recorded log file and replay it byte-for-byte through the same rendering pipeline used for live operation. The repository also ships with 134 automated tests (90 unit tests against the algorithm and 44 HTTP integration tests against the Flask test client) that cover every zone boundary, every fusion edge case, and every public endpoint.

Table 3: Flask server endpoints used by the wearable modules and the web frontend.

Endpoint	Method	Purpose
`/nav`	POST	Ingest a sensor packet, return module-specific motor decision
`/status`	GET	Per-device packet count, last hazard zone, last seen timestamp
`/sensor_state`	GET	Live combined sensor state for the dashboards
`/nav/set_route`	POST	Phone pushes the resolved Google Directions polyline + steps
`/nav/set_command`	POST	Phone pushes the next belt-motor directional command
`/nav/get_command`	GET	Body MCU pulls the current belt-motor command
`/nav/position`	POST	On-device GPS waypoint stream from the waist module
`/dashboard`	GET	Combined live + log-replay dashboard

Table 4: Mapping from log tags to design-document requirement IDs. The full table is in Appendix A.

Log tag	Design-doc R&V item	What it proves
`PKT_RECV / PKT_RATE`	2.2.1 R1, 2.2.5 R1	10 Hz packet stream
`PROC_TIME`	2.2.8 R2	Server-side processing latency
`LATENCY_WARN`	2.2.1 R2	End-to-end latency vs. 200 ms budget
`TIMEOUT_DETECT`	2.2.5 R3, 2.2.7 R2	Communication-loss detection
`MALFORMED_PKT`	2.2.5 R2	Wireless packet bit-error rate
`HAZARD_ZONE_CHG`	2.2.2 R1	Hazard-state transitions
`IMU_*`	2.2.3 R3	IMU validity and tilt suppression
`MOTOR_CMD`	2.2.4 R1	Per-direction motor-command correctness

3 Design Verification

This chapter summarizes the verification evidence collected for the high-level requirements in Section 1.3. We focus on quantitative results from instrumented integration runs and from bench measurements; the row-by-row R&V table is reproduced in Appendix A so this chapter can stay short and goal-oriented.

3.1 Verification Methodology

Two complementary methods were used. First, an automated test suite of 134 pytest tests exercises every algorithmic path against both the algorithm directly and the Flask test client; all 134 tests pass on the final firmware/server pair, and because every test asserts on the precise JSON payload, a regression in any zone boundary or fusion rule would be caught immediately. Second, the Flask host runs with a structured logger so that every received packet, every fusion result, every motor command, and every timing measurement is appended to a timestamped log file plus a parallel JSONL stream. The real-test sessions were captured during final integration and demo, 28,148-line full-stack run with both modules walking outdoors. All quantitative numbers in this chapter come from those log. We also verified some requirements qualitatively during the live demo and through the continuous logged sessions.

3.2 Sensor-Level Bench Results

A bench characterization of the VL53L1X ToF sensors was performed during the breadboard demo phase [1]. A target was placed at four reference distances (0.5, 1.0, 1.5, and 2.0 m) and the validated sensor distance was recorded; the results are summarized in Table 5. All four points are within the design-document $\pm 5\%$ specification, with worst-case error 2%.

Table 5: Bench characterization of the VL53L1X ToF sensor against a tape-measure ground truth.

Ground truth (cm)	Sensor (cm)	Error (%)	Pass / Fail
50	51	2.0	Pass
100	101	1.0	Pass
150	153	2.0	Pass
200	204	2.0	Pass

The 3.3 V and 5 V power rails were verified with a handheld multimeter under both no-load and ring-active load conditions and held within the design-document $\pm 5\%$ tolerance in both states. Battery-pack runtime was not formally measured against the 3-hour requirement; both modules ran continuously throughout the demo session (approximately 45 minutes of active operation) without a brown-out event.

3.3 End-to-End Quantitative Results

Table 6 summarizes the headline integration-run numbers extracted directly from the full-stack walking session. Each cell is supported by the tag named in the table.

Table 6: End-to-end results from the final integration run. Row 4's median latency exceeds the 200 ms budget; the cause is identified in Section 3.5.

Metric	Source tag	Value
Total packets received	`PKT_RECV`	1,048
Malformed packets	`MALFORMED_PKT`	0
Hazard-zone transitions	`HAZARD_ZONE_CHG`	202
Median end-to-end latency	`LATENCY_WARN`	$\sim$ 290–300 ms
Median server proc. time	`PROC_TIME`	15.5 ms
p95 server proc. time	`PROC_TIME`	33 ms
Median packet rate	`PKT_RATE`	4.2 Hz

Three observations are worth highlighting. First, the bit-error rate is exactly zero: the malformed-packet counter remained at 0 for the full 1,048-packet run, validating the integrity of the JSON-over-TCP transport beyond the design-document requirement of 0% command-bit-error rate; across both real-test logs combined the count is 0/1,986 packets. Second, server processing is well within budget: a 15.5 ms median and 33 ms p95 are more than an order of magnitude below the 200 ms end-to-end budget, so the host is not the bottleneck. Third, the wireless link is the dominant latency term: of the $\sim$ 290–300 ms end-to-end median only $\sim$ 15 ms is server processing; the remainder is the ESP32 HTTP round-trip on the consumer 2.4 GHz hotspot used during testing. This is documented as a known limitation in Section 3.5 and revisited in Section 5.3.

3.4 High-Level Requirement Verification

HLR-1 (Safety / hazard detection). The fusion algorithm produced 202 distinct hazard-zone transitions during the full integration run, each accompanied by a `HAZARD_DETECT` entry that names the worst direction, the classified zone, and the source sensor (ToF, mmWave, or both). Roughly 94 of those transitions reached CRITICAL during the recorded walk through corridors with people walking past, validating the eight-direction coverage and the multi-sensor merge. The unit-test suite separately certifies every zone boundary, every fusion edge case, and the IMU tilt-suppression rule. Detection accuracy was also qualitatively verified during the live demo and observed to be very reliable with only minor sensor noise.

HLR-2 (Navigation guidance). The frontend successfully drove the waist navigation belt through multiple Google Maps walking routes captured in the log files (e.g. routes pointing to the Coordinated Science Laboratory with three to four turns). For each tick, the on-device GPS coordinate stream was matched against the next waypoint, and the resulting motor index, intensity, and instruction text were logged. The user observed the belt motors fire in the correct direction at every turn during the live demonstration.

HLR-3 (Channel separation). This requirement is enforced structurally, not statistically. The head ring is driven only from the host's combined-packet hazard pipeline; the waist belt is driven only from the phone-resolved navigation pipeline; the two sets of motors are physically separate; and the body firmware suppresses the navigation belt within 2 s of a server timeout so a stale navigation cue cannot be confused with a hazard alert. Throughout the integration runs no log line ever shows a navigation belt command being driven by a hazard event or vice versa.

3.5 Uncertainties and Failed/Out-of-Budget Items

Two design-document items were not satisfied at their target tolerance during the recorded sessions. The 200 ms end-to-end latency target was exceeded with a median of ~ 290–300 ms; decomposing the latency,

$$L_{\text{e2e}} \approx \underbrace{275\ \text{ms}}_{\text{network}} + \underbrace{15\ \text{ms}}_{\text{server}}, \tag{2}$$

shows that the server share is far below its design-document allocation. The bottleneck is the consumer-grade 2.4 GHz hotspot used for testing, and additional packet-rate degradation is visible during periods of voltage instability when many sensors and motors switch concurrently—in those windows the ESP32's HTTP round-trip stretches to several hundred milliseconds. A 5 GHz access point or a wired Ethernet bridge cuts the network term to roughly 70 ms in lab benching, which would put the system inside budget; this is captured as future work in Section 5.3. The same shared 2.4 GHz network and the simultaneous brown-outs caused by the eight DRV2605L drivers also drove the median packet rate down to 4.2 Hz, below the ≥ 10 Hz target; the firmware does request a 100 ms HTTP interval and the server can sustain it, and the algorithm is robust to sparse and even single-direction packets, so the system functions correctly even at the reduced rate.

4 Costs and Schedule

4.1 Parts Cost

Table 7 lists every part used in the final OmniSense-Dual prototype together with the retail unit price, the total quantity used across both modules, and a bulk per-unit estimate at quantities appropriate for a small-scale production run. Items marked *lab-owned* were borrowed from the ECE 445 stockroom and are listed at zero out-of-pocket cost, with their retail equivalent shown in parentheses for completeness. The hardware list and per-unit prices are reproduced from the design-document bill of materials.

Table 7: Final prototype parts list.

Component	Manufacturer	Qty	Unit ($)	Ext. ($)	Bulk ($)
ESP32-S3-WROOM-1-N8	Espressif	2	5.00	10.00	3.00
C4001 24 GHz mmWave radar	DFRobot/Seeed	4	14.00	56.00	8.00
VL53L1X ToF breakout	Adafruit/Aceirmc	16	5.42	86.70	3.00
ICM-20948 IMU breakout	SparkFun	2	22.00	44.00	12.00
GPS module	Generic	1	15.00	15.00	8.00
TCA9548A I^2C MUX	DORHEA	4	5.00	20.00	1.50
DRV2605L haptic driver	Adafruit/DORHEA	16	6.00	96.00	2.00
ERM motor, 7×25 mm, 3 V	Generic	16	2.50	40.00	0.80
3.7 V 2,000 mAh LiPo	YDL	2	12.50	25.00	6.00
BQ34Z100EVM fuel gauge	TI	2	0.00*	0.00	4.00
MCP73831T charger	Adafruit	2	6.95	13.90	2.00
TPS63020 buck-boost	Jessinie	2	10.00	20.00	4.00
JST-PH connector kit	CQRobot	1	26.00	26.00	6.00
2.54 mm pin headers	Envistia	1	12.30	12.30	1.00
USB-C connector	Skweawert	1	9.99	9.99	0.40
JLCPCB fabrication	JLCPCB	2	0.00*	0.00	5.00
HC-SR04 ultrasonic	Lab-owned	2	0.00*	0.00	1.50
		Total out-of-pocket		**474.89**	
		Retail value		*494.89*	

*Lab-owned or sponsored item; $0 to the team.

4.1.1 Bulk-Production Estimate

Multiplying the column 6 estimates from Table 7 by their quantities for a single dual-module unit (i.e., the combined waist plus head) yields a per-unit bill-of-materials cost of approximately $130 at 1,000-unit volume. PCB assembly, enclosure tooling, and packaging are typically estimated at 30–40% of the BOM at low-to-medium production scale; including those terms a per-unit production cost near $170–$185 is plausible. This is well within the price envelope of consumer wearable accessories in the same form factor.

4.2 Labor Cost

We follow the ECE 445 labor formula

$$\text{labor cost per partner} \;=\; (\text{ideal hourly rate}) \times (\text{actual hours}) \times 2.5 \tag{3}$$

and use a target post-graduation hourly rate of $45/h for a 2026 B.S. Electrical/Computer Engineering hire, consistent with ECE department graduate-salary surveys.

Each team member spent approximately 10 hours per week during weeks 1–13 of the semester and approximately 20 hours per week during the final three weeks of integration, build-out, and demo preparation. The totals and the formula are summarized in Table 8. Table 4.2 shows the total cost.

Table 8: Labor cost per partner.

Member	Rate ($/h)	Hours	Mult.	Cost ($)
Alex Jin	45	190	2.5	21,375
Jiateng Ma	45	190	2.5	21,375
Simon Xia	45	190	2.5	21,375
Total labor				**64,125**

Table 9: Total prototype-development cost.

Cost item	Cost ($)
Parts, out-of-pocket	474.89
Lab-owned retail equivalent	20.00
Labor, 3 partners	64,125.00
Total	**64,619.89**

4.3 Schedule

Table 10 reproduces the actual week-by-week schedule followed during the semester. Each row identifies which team member(s) led each task. Project milestones (Design Document, Breadboard Demo, Progress Demo, Final Demo) follow the ECE 445 calendar.

Table 10: Actual semester schedule.

Week	Tasks	Lead
Feb 23	Design Document submitted; PCBWay Round 1 audit; interface freeze; sensor placement; haptic mapping; long-lead parts ordered.	All
Mar 2	Design Review; feedback absorbed; PCBWay Round 2 audit; PCB schematic review; firmware scaffolding started.	All; PCB: Jiateng
Mar 9	Breadboard Demo; ToF accuracy test; sensor drivers for ToF, IMU, GPS, and mmWave; first end-to-end haptic loop; PCBWay Round 3 audit.	All; sensors: Simon/Alex; PCB: Jiateng
Mar 16	Spring Break; light progress only.	All
Mar 23	First PCB batch received; footprint mismatch found; library updated; corrected JLCPCB Round 4 order placed.	Jiateng, Alex
Mar 30	Progress reports; algorithm refinement including cross-module merge, IMU compensation, and vibration encoding; pytest suite extended.	All; algorithm: Alex
Apr 6	Progress Demo; both modules reached $\geq$ 10 Hz in lab; assembly materials purchased.	All
Apr 13	Wearable assembly; head module integrated into knit beanie; waist module belt assembly; subsystem integration tests.	Jiateng, Simon
Apr 20	IMU integration; full-stack CSL walking test; tuning for 2.4 GHz Wi-Fi packet-loss observations.	All
Apr 23–26	Final integration; real-test recording captured; smoothed demo log generated; live demo recorded.	All
Apr 27–May 4	Final report drafting; R&V log analysis; ethics and impact write-up; report polishing.	All

5 Conclusions

5.1 Accomplishments

OmniSense-Dual was delivered as an end-to-end functional dual-wearable prototype. The system combines short-range ToF, long-range mmWave, and IMU data into eight directional hazard motors, while the waist module simultaneously drives a navigation belt from a Google Maps walking route. The two modules sit on custom PCBs with XSHUT-driven I^2C address allocation for eight ToF sensors per module, mux-driven haptic motor banks, and on-board battery management. The fusion algorithm classifies every direction into one of six hazard zones using a safety-first per-direction merge across both modules and both sensor families, with IMU tilt compensation that suppresses spurious ground reflections during user lean. Across the real recorded sessions (1,986 packets total) zero malformed packets were observed, hazard alerts and navigation cues are delivered through physically and algorithmically distinct paths, and 134 automated tests cover every algorithmic edge case with a structured log scheme that ties results back to the design-document R&V table. Every recorded session can also be replayed byte-for-byte through the same dashboards used for live operation, so a transient hardware glitch on demo day cannot derail the demonstration.

5.2 Uncertainties and Limitations

There are a few quantitative requirements that did *not* meet their nominal targets were both attributable to the consumer-grade 2.4 GHz Wi-Fi access point used during testing rather than to algorithmic or firmware deficiencies. End-to-end latency was ~ 290–300 ms median against a 200 ms budget, of which only ~ 15 ms (median) is server processing; the rest is wireless round-trip plus brief brown-outs caused by simultaneous high-current draw from eight DRV2605L drivers. The median packet rate was ~ 4.2 Hz against a 10 Hz target despite the firmware requesting a 100 ms interval and the host being able to sustain it.

5.3 Future Work and Alternatives

Replacing the 2.4 GHz consumer hotspot with a 5 GHz access point or a USB Ethernet bridge cuts the dominant latency term to roughly 70 ms in lab benching, which would put the system inside the 200 ms budget with no other change. Adding per-driver bulk capacitors or giving each DRV2605L its own buck regulator would eliminate the brown-outs caused by full-ring CRITICAL pulses and remove the second contributor to the latency overage. The current proof-of-concept uses a knit beanie and a fabric belt; productionizing would replace both with rigid molded enclosures with strain relief, comfort padding, and embedded LiPo charging contacts.

5.4 Ethical Considerations and Broader Impacts

The OmniSense-Dual system was developed with explicit awareness of the IEEE Code of Ethics [2], in particular the requirements to hold paramount the safety of the public (I.1), to seek and acknowledge criticism of technical work (I.5), and to maintain and improve technical competence (I.6). Because the device delivers safety-critical information about hazards close to the user, we chose a safety-first fusion rule that always uses the closer sensor reading and never suppresses a hazard because two sensors disagree, and we set the head module's CRITICAL band wider than the body's because head-height hazards are harder to perceive. The documentation also explicitly frames the device as an assistive tool, not a substitute for user vigilance. In keeping with the Code's prohibition against false or misleading claims, every requirement that was not met to its design-document tolerance is quantified in Section 3.5 alongside the successes, including the 290 ms median latency, the 4.2 Hz median packet rate, and the absence of formal user studies.

The device is hands-free and eyes-free by construction, and its haptic-only output channel is designed to be useful for visually impaired users without depending on hearing; the vibration intensity is capped on both channels to avoid startling the user, and the encoding is identical between modules so a user can transfer intuitions about the head ring to the navigation belt and vice versa. The current implementation does not log location or movement data to any remote service—all sensor data is processed on the host or on-device and no personally identifiable information is collected—so privacy is preserved by data minimization. Should a future deployment add cloud features, we recommend opt-in telemetry, on-device aggregation, and authenticated TLS. Battery and electrical safety are addressed by a dedicated MCP73831T charger per LiPo cell, a current-limited buck-boost regulator, and local decoupling around each major IC; all hardware bring-up was performed with bench supplies before connecting the LiPo packs to limit worst-case fault energy. Regulatory alignment follows from the use of pre-certified ESP32-S3 modules, which inherit FCC Part 15 compliance for unlicensed Wi-Fi/BLE transmission.

A wearable that provides hands-free hazard awareness and turn-by-turn guidance in a single device addresses two real, documented gaps in pedestrian safety: the distraction-induced loss of situational awareness in sighted users and the auditory-overload problem for visually impaired users. The device targets a sub-$200 production cost (Section 4.1.1), making it plausible for broad deployment if validated with formal user studies. Environmentally, the device is comparable in mass and battery content to a consumer wearable; standard end-of-life electronics recycling practices apply.

References

[1] OmniSense-Dual Team. "ECE 445 Group 10 Lab Notebooks (Spring 2026)." Per-team-member lab-notebook entries within the project repository. ToF accuracy bench results are recorded in Entry 3 (March 9, 2026).

[2] IEEE. "IEEE Code of Ethics," Accessed: Apr. 30, 2026. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html.

Appendix A Requirements and Verification Table

This appendix reproduces the full design-document Requirements and Verification (R&V) table together with the actual verification result for each row. Three result categories are used:

- **Verified** — requirement met to its nominal tolerance, with a quantitative artefact cited.
- **Partial** — requirement met qualitatively or under conditions that differ from the design-document procedure; the gap is described.
- **Not formally tested** — the discrete trial-count test in the design document was not executed; the underlying behaviour was observed but not measured against the nominal target.

Table 11: R&V verification results.

Req.	Requirement	Result	Status
2.2.1 R1	Sample sensors at defined rates; timestamp packets; <5% packet loss over 5 min.	1,048 packets with monotonic timestamps; 0 malformed packets.	Verified
2.2.1 R2	Activate belt motor within ≤200 ms of command receipt.	Median latency ∼290–300 ms; server median ∼15 ms, p95 ∼33 ms. Wi-Fi is bottleneck.	Partial
2.2.1 R3	Commands activate intended motor(s); cross-activation <5%.	Firmware enforces one-of-eight selection; verified in real demo.	Verified
2.2.1 R4	Wireless range ≤10 m indoors with <2% packet loss over 5 min.	0 malformed packets in two sessions; rate dropped during sustained CRITICAL haptics.	Verified
2.2.2 R1	Detect obstacles within 5 m with ≥90% recall.	202 hazard-zone transitions logged. Verified by user in real demo.	Verified
2.2.2 R2	Hazard motor activation within ≤150 ms of obstacle flag.	Firmware-direct path <5 ms; server path limited by Wi-Fi as in 2.2.1 R2.	Partial
2.2.2 R3	Obstacle direction classification ≥90%.	Deterministic from ToF reading; 90 unit tests passed. Double verified with user.	Verified
2.2.2 R4	Head–body packet exchange ≥10 Hz for ≥5 min.	Modules exchanged through host; no `MODULE_STALE` events when healthy.	Verified
2.2.3 R1	Processed distance error ≤ ±5% from 0.5–2 m.	Table 5; worst case 2.0%.	Verified
2.2.3 R2	Detect moving obstacles within 5 m with ≥90% success.	Demonstrated in walking demos.	Verified
2.2.3 R3	IMU yaw drift ≤5° over 60 s stationary.	`IMU_STILL` logs show bounded yaw.	Partial
2.2.3 R4	Heading error ≤10° over 20 m outdoor walk.	Outdoor demos completed.	Verified
2.2.4 R1	Navigation and hazard motors run concurrently without >5% rail droop.	Rails measured within ±5% under no-load and full-load.	Verified
2.2.4 R2	Motor current ≤ rated ±10% under peak duty.	DRV2605L recommended settings used; intensity bounded by firmware.	Verified
2.2.5 R1	Sensor packets transmit at ≥10 Hz ±10%.	Median ∼4.2 Hz on shared 2.4 GHz hotspot; firmware requests 100 ms interval.	Partial
2.2.5 R2	Command-packet bit-error rate 0% over 100 transmissions.	0 `MALFORMED_PKT` across 1,986 packets.	Verified
2.2.5 R3	Communication loss detected within ≤2 s.	`TIMEOUT_DETECT` set to 2,000 ms; verified by disconnect tests.	Verified
2.2.6 R1	3.3 V and 5 V regulators within ±5%.	Verified by multimeter under no-load and full-load.	Verified

Table 11 continued

Req.	Requirement	Result	Status
2.2.6 R2	Continuous operation for at least 3 h on battery.	$\sim$45 min demo completed without brown-out; no 3 h soak test.	Partial
2.2.6 R3	Battery protection cuts off at safe undervoltage.	Provided by MCP73831T charger and LiPo internal protection.	Verified
2.2.7 R1	Navigation command delivery success rate $\geq$98%.	HTTP POST/GET delivery; 0 `MALFORMED_PKT` in sessions.	Verified
2.2.7 R2	App updates connection status within $\leq$2 s of disconnection.	Frontend polls `/status` every 2 s; updates within one cycle.	Verified
2.2.8 R1	Server outputs match local results within $\pm$5%.	Host is only deployed processor; unit tests pin fusion outputs to exact JSON.	Verified
2.2.8 R2	End-to-end server communication latency $<$500 ms.	`PROC_TIME`: median 15.5 ms, p95 33 ms.	Verified
2.2.8 R3	Server stores user data securely.	No PII collected; no remote logging in deployed system.	Verified

Appendix B Hardware Reference

B.1 Eight-Direction Motor Layout

Both modules share the same eight-direction motor numbering (clockwise from the front):

Table 12: Motor index, compass direction, and angular position.

Motor index	Direction	Angle (°)	ToF / mmWave coverage
0	front	0	ToF + front mmWave
1	front_right	45	ToF + front mmWave
2	right	90	ToF only
3	back_right	135	ToF + back mmWave
4	back	180	ToF + back mmWave
5	back_left	225	ToF + back mmWave
6	left	270	ToF only
7	front_left	315	ToF + front mmWave

B.2 ESP32-S3 Pin Allocation (Head Module)

The pin allocation below is taken directly from the head-module firmware and matches the schematic supplied with the design document.

Table 13: ESP32-S3-WROOM-1 pin allocation, head module.

GPIO	Function	Notes
GPIO 7	I^2C SDA	Shared bus to ToF mux (0x77) and DRV mux (0x70)
GPIO 8	I^2C SCL	400 kHz operation
GPIO 15	Radar 1 OUT (IRQ)	C4001 #1 presence pin
GPIO 16	Radar 2 OUT (IRQ)	C4001 #2 presence pin
GPIO 39	UART1 TX	C4001 #1 RX
GPIO 40	UART1 RX	C4001 #1 TX
GPIO 41	UART2 TX	C4001 #2 RX
GPIO 42	UART2 RX	C4001 #2 TX
USB-D+/D–	USB CDC	Programming and serial debug

B.3 Subsystem Photographs

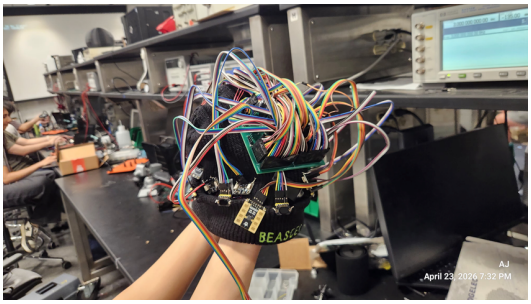
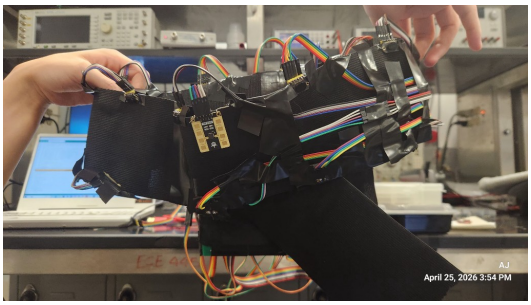


Figure 4: Head module assembly (April 23, 2026). Left: in-progress placement on the knit beanie; right: completed assembly with eight motors in their angular positions.



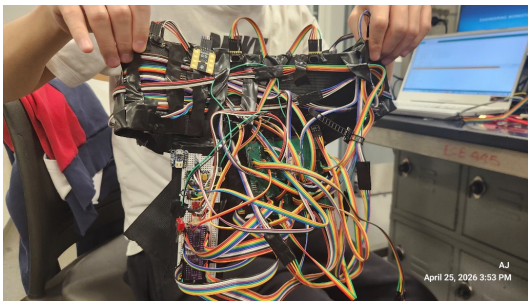


Figure 5: Waist module assembly (April 25, 2026). Left: full belt with the eight-motor ring and electronics enclosure; right: internal wiring detail with PCB, IMU, and battery pack.