Responses to Reviewers

Response to Reviewer Comments

1. Summary

We sincerely thank you for taking the time to review this manuscript. Please find the detailed responses below and the corresponding revisions in the re-submitted files.

2. Questions for General Evaluation	**Reviewer's Evaluation**
Figure 1 has error, please correct. Overlay words	Must be improved
Provide figures and photos of final prototype.	Must be improved

3. Point-by-point response to Comments and Suggestions for Authors

Comments : Figure 1 has error, please correct. Overlay words； Provide figures and photos of final prototype.

Response : We sincerely thank the reviewer for this detailed assessment. In response to this issue, the manuscript has been substantially revised. The following major changes have been implemented.

We have redrawn the previous figures 1 and 2: "System block diagram for the implemented dual-axis solar tracker" and "Main firmware flow implemented in the STM32 source code" (now it has become figure 5), to make them look clearer.

In Chapter 2: Design, we have included Figure 2: PCB Design, Figure 3: Circuit Schematic Design, Figure 4: Mechanical Structure of the Implemented Two-axis Solar Tracker, and Figure 6: Final Prototype of Our Project. In this way, teachers can more intuitively see our design prototypes and final products.

ECE 445

Senior Design Laboratory

Final Report

Intelligent Net-Energy Optimization System for Distributed Photovoltaic Nodes in Microgrids

---

Team #48

Ziru, Niu (ziruniu2@illinois.edu)
Yikai, Zhang (yikaiz2@illinois.edu)
Minghao, Fang (mf46@illinois.edu)
Yifei, Liu (yifei20@illinois.edu)

TA: Tiantong, Qiao

May 14, 2026

Abstract

This report presents a dual-axis solar tracking system with integrated lithium-battery power management. The design uses an STM32F103C8T6 microcontroller as the central controller, four light-dependent resistors (LDRs) as directional light sensors, two 28BYJ-48 stepper motors driven through ULN2003 arrays for pan and tilt motion, and a 0.96 OLED display for local status reporting. The power subsystem combines a 5 V, 120 mA solar panel, a TP4056 single-cell lithium-ion charger, an 18650 lithium-ion cell, and a 5 V, 1.2 A boost converter so the prototype can operate from stored solar energy. Firmware running on the STM32 samples the four light channels and two voltage channels through the internal analog-to-digital converter (ADC), averages measurements to reduce noise, estimates battery state of charge from voltage, detects charging state, and commands the motors until opposing light readings are balanced within a four-percentage-point deadband. The resulting system demonstrates a compact, low-cost architecture for small photovoltaic and outdoor sensing applications that need local energy autonomy, two-axis light seeking, manual override, and real-time display of energy status.

Contents

1 Introduction

1.1 Motivation

Small photovoltaic nodes, such as garden-scale solar chargers, greenhouse supplemental-light platforms, and outdoor sensing stations, often operate with limited local energy. In these applications, a fixed solar panel can lose available irradiance as the sun angle changes throughout the day. At the same time, a tracker consumes energy through its controller, sensors, display, and motors, so light seeking should be considered together with the node's local energy condition.

The goal of this project is to combine dual-axis light tracking and basic energy management in one low-cost embedded photovoltaic node. The completed prototype senses the relative light level above, below, left, and right of the panel, drives two axes of motion toward balanced illumination, charges a lithium-ion battery from a small solar panel, boosts the battery to a regulated 5 V system rail, and reports operating mode, light readings, battery voltage, battery capacity, charging voltage, and charging state on the OLED.

1.2 System Overview

Figure 1 shows the final project architecture. The system is built around an STM32F103C8T6 microcontroller, which provides the GPIO, timers, flash memory, UART support, and ADC channels needed to connect the light sensors, voltage and power measurements, stepper drivers, keys, and OLED display in a compact low-cost module stm32f103. The implemented prototype focuses on local autonomous operation: it senses directional light, controls the dual-axis motor drive, monitors the battery and charging inputs, and keeps a UART interface for optional future telemetry or remote-control extensions.

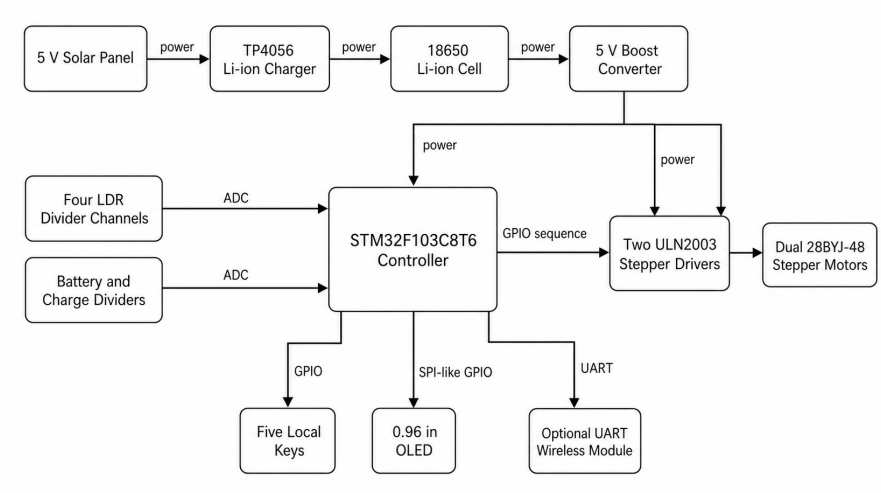

Figure 1: System block diagram for the implemented dual-axis solar tracker.

1.3 Performance Requirements

Table 1 summarizes the main requirements used to evaluate the implemented prototype. The limits reflect the current STM32F103C8 firmware and selected hardware modules.

Table 1: Top-level performance requirements

Function	Requirement
Automatic tracking	Compare the four LDR channels and drive each axis only when the opposing readings differ by more than four percentage points.
Manual control	Provide one mode key and four direction keys for local manual positioning.
Stepper drive	Control two 28BYJ-48 stepper motors through ULN2003 drivers with continuous step commands and per-axis acceleration.
Light sensing	Update sensor readings every 20 ms and average each LDR value over five samples.
Battery monitoring	Measure the 18650 battery voltage through ADC and estimate system power from the active motor load.
Charging detection	Detect charging from the solar/charger input using the measured charge voltage relative to the battery voltage.
Display	Show mode, four light readings, battery voltage, estimated power, charge voltage, and charge state on the OLED.
UART interface	Provide optional 9600 baud UART telemetry and remote commands.
Power system	Charge a single 18650 cell from a 5 V solar panel and boost the battery output to a regulated 5 V bus.

2 Design

2.1 Design Procedure

The reference document evaluates several implementation paths; Table 2 restates those trade-offs in the context of the final prototype rather than reproducing the original discussion project-reference. The selected solution favors easy prototyping, low cost, and direct compatibility with the required sensors and actuators.

Table 2: Major design decisions

Block	Selected approach	Reason for selection
Controller	STM32F103C8T6 Cortex-M3 board	Provides ADC channels, timers, UART, flash storage, and enough GPIO for two motors, OLED, keys, and sensors while remaining inexpensive and familiar to develop in Keil C.
Display	0.96 OLED, 128 by 64 pixels	Lower power and better contrast than character LCDs; enough area for mode, light, voltage, and charge fields; compatible with the SSD1306-style OLED controller documented in the project files oled-ssd1306.
Actuator	Two 28BYJ-48 geared stepper motors	Low-cost discrete positioning with simple four-phase control; sufficient for small panel/fixture motion.
Motor driver	ULN2003 Darlington arrays	Allows STM32 GPIO to switch the stepper motor coils safely without sourcing motor current directly uln2003.
Light sensing	Four LDR voltage dividers	Low-cost directional sensing; resistance decreases under stronger light, enabling relative upper/lower/left/right comparison through ADC channels.
Energy storage	One 18650 lithium-ion cell with TP4056 charger and 5 V boost module	The TP4056 provides a compact single-cell charge controller tp4056; the boost module supplies the 5 V bus required by the modules.

2.2 Hardware Architecture

The hardware is organized around the STM32 minimum system board. The four LDR dividers, the battery divider, and the charge-voltage divider feed ADC1. The OLED is driven through five GPIO pins using a software serial interface. Five key inputs provide mode selection and manual motor control. Two independent sets of four GPIO outputs

connect to ULN2003 motor drivers, which energize the phases of the two 28BYJ-48 stepper motors.

The firmware defines the system's electrical interface explicitly. ADC channel 7 measures charge voltage, channel 5 measures battery voltage, channel 4 measures the left LDR, channel 3 measures the upper LDR, channel 2 measures the right LDR, and channel 1 measures the lower LDR. The key inputs are PB12, PB13, PB14, PB15, and PA8. The tilt motor uses PA0, PC15, PC14, and PC13, and the left-right motor uses PA11, PA12, PA15, and PB3. The OLED uses PB4 through PB8 for clock, data, reset, data/command, and chip select.

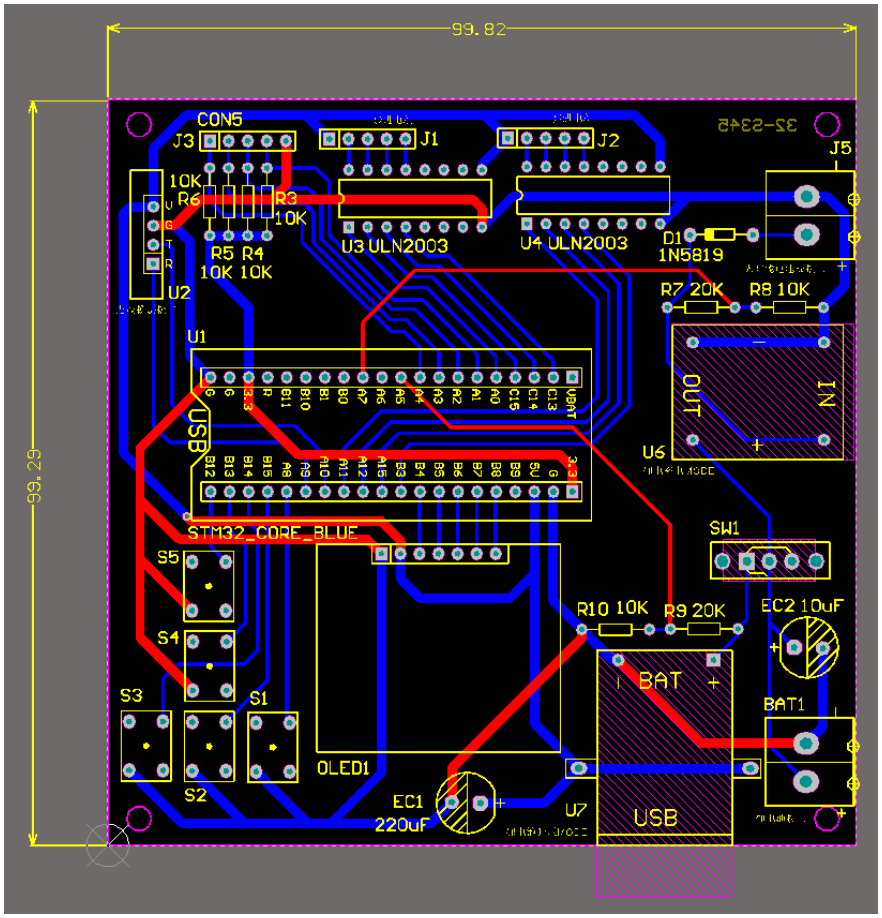

Figure 2: PCB design

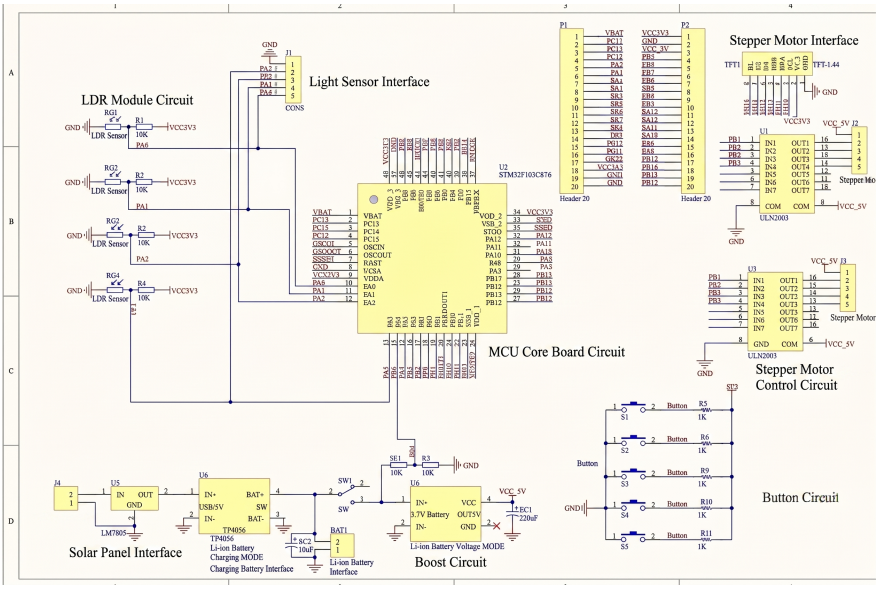

Figure 3: Circuit Schematic Design

2.3 Power Subsystem

The energy path is solar panel to TP4056 charger, charger to 18650 cell, and cell to boost converter. The project documentation specifies a 5 V, 120 mA polycrystalline solar panel and a 5 V, 1.2 A boost module. Because a single lithium-ion cell ranges from approximately 3.4 V to 4.2 V during normal use, the boost stage is required for the OLED, motor driver board, and other 5 V modules. The TP4056 charges one cell with a constant-current/constant-voltage profile and a fixed 4.2 V regulation point tp4056.

Voltage measurement uses a resistive divider before the ADC. The firmware multiplies the ADC input by three, so the divider ratio is treated as 3:1. The reconstructed voltage is

$$V_{\text{meas}} = \frac{N_{\text{ADC}}}{4096} \cdot 3.3\,\text{V} \cdot 3 \tag{1}$$

where N_{ADC} is the 12-bit ADC code. This keeps a 4.2 V cell at approximately 1.4 V on the ADC pin and a 5 V charging input at approximately 1.67 V, both below the STM32 ADC limit.

The firmware estimates battery capacity with a linear voltage model:

$$Q_{\text{bat}} = \begin{cases} 0, & V_{\text{bat}} < 3.40\,\text{V} \\ 100, & V_{\text{bat}} > 4.15\,\text{V} \\ 100 \cdot \frac{V_{\text{bat}} - 3.40\,\text{V}}{4.15\,\text{V} - 3.40\,\text{V}}, & \text{otherwise.} \end{cases} \tag{2}$$

This is not a precision coulomb counter, but it is appropriate for a compact user-facing state-of-charge indicator.

2.4 Mechanical Structure

Figure 4 shows the assembled mechanical layout of the tracker. The base plate carries the PCB and the battery holder, which keeps most of the electrical parts fixed during operation, This arrangement keeps the center of mass low and reduces cable motion when the upper stage rotates.

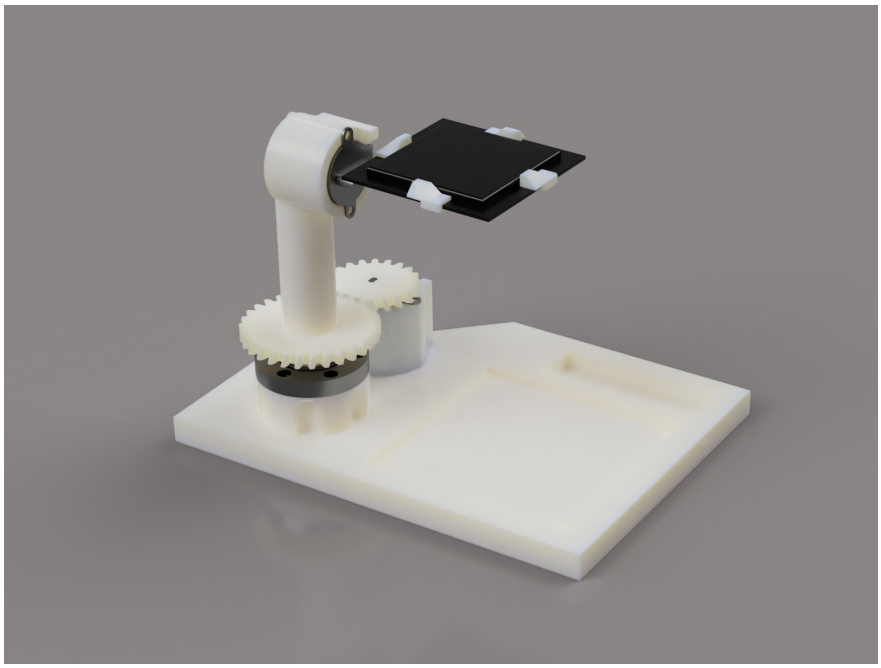

Figure 4: Mechanical structure of the implemented two-axis solar tracker.

The yaw motion is supported by a slewing-bearing stage mounted on the base. A stepper motor is installed on a vertical bracket beside the bearing, and its output is connected to the rotating stage through a reduction gear set. This gear reduction is important in our design because the yaw motor does not only move the panel itself, it also needs to carry the upper support, the pitch motor, the panel frame, and part of the sensor wiring. Without reduction, the available motor torque would be close to the practical limit, especially when the structure starts from rest. The reduction gear set increases the output torque at the yaw stage and slows down the motion, which makes small tracking corrections easier to control.

The yaw transmission uses helical gears instead of a simple straight spur gear pair. Since the teeth of a helical gear come into contact gradually, the torque transfer is smoother during rotation. This helps reduce vibration and small impacts when the stepper motor moves in short intermittent steps. In this project, the tracker often needs to start, stop, and make small corrections around the balanced-light position, so smoother gear engagement is useful for keeping the upper stage stable.

The pitch axis is placed on the rotating upper stage. A second stepper motor is mounted on the upper bracket and connected to the solar-panel frame through a horizontal shaft. This motor directly adjusts the panel tilt angle, while the lower motor controls the azimuth angle

through the slewing-bearing stage. In this way, the two axes are mechanically separated, and the firmware can command the yaw and pitch motors independently.

2.5 Light Sensing and Tracking Control

The light-sensing part is built around four photoresistors mounted along the four edges of the solar panel. They correspond to the upper, lower, left, and right directions of the panel. Instead of placing the sensing surfaces parallel to the main panel, each photoresistor is tilted outward by 45°. This small mechanical detail makes the sensing result more useful. When the incoming light is not perpendicular to the panel, the incident angle on the four photoresistors becomes noticeably different, so the measured light difference between opposite sides becomes larger. In other words, the 45° outward placement makes it easier for the controller to tell which side the light source is coming from.

Each photoresistor is connected as part of a voltage divider and is read by the STM32 internal ADC. The firmware then converts the ADC result into an inverted relative light percentage:

$$L = 100 - 100 \cdot \frac{N_{\mathrm{ADC}}}{4096} \tag{3}$$

where N_{ADC} is the 12-bit ADC output. With this definition, a larger L means stronger light on that sensor. The four processed readings are stored as L_{up}, L_{down}, L_{left}, and L_{right}.

The firmware samples the four light channels periodically instead of reacting to a single ADC reading. A raw sampling event is triggered every 20 ms, and each displayed or control-used light value is obtained by averaging five samples. This gives an effective update interval of about 100 ms for the filtered light values. The averaging step is simple, but it is useful because the LDR readings can change slightly due to ADC noise, hand motion around the light source, or small shadows on the sensor board.

The tracking controller compares the two opposite sensor pairs. For the vertical and horizontal axis, the firmware computes

$$e_{\mathrm{UD}} = L_{\mathrm{up}} - L_{\mathrm{down}}, \tag{4}$$

$$e_{\mathrm{LR}} = L_{\mathrm{left}} - L_{\mathrm{right}}. \tag{5}$$

These two errors give the direction of the light imbalance. If the upper sensor reads much stronger than the lower sensor, the panel should be adjusted in the corresponding pitch direction. If the left sensor reads much stronger than the right sensor, the yaw axis should move toward the left side, and vice versa.

A deadband of four percentage points is used for both axes:

$$|e_{\mathrm{UD}}| \le 4, \qquad |e_{\mathrm{LR}}| \le 4. \tag{6}$$

When the error of one axis is inside this range, the firmware treats that axis as already balanced and stops the corresponding motor. When the error is larger than the deadband, the firmware keeps commanding that axis to move toward the brighter side. This avoids unnecessary back-and-forth motion when the panel is already close to the best position. It

also prevents small ADC fluctuations from making the motors chatter around the balance point.

In auto mode, the two axes are controlled independently. The up-down comparison only affects the pitch motor, while the left-right comparison only affects the yaw motor. This keeps the control rule easy to debug: each axis simply tries to reduce the light difference between its own opposite sensor pair. Once both differences fall within the four-point deadband, the panel is considered aligned with the dominant light direction, and the drive layer stops the motors.

2.6 Stepper Motor Drive

The two motion axes are driven by 28BYJ-48 geared stepper motors through ULN2003 driver boards. Each motor has four control phases, and the firmware drives them with an eight-state half-step sequence. Compared with a simple full-step sequence, half-step driving gives smoother low-speed motion and finer position resolution, which is helpful for small tracking corrections near the balanced-light position.

The motor control is not handled by blocking delay loops in the main program. Instead, the firmware uses the SysTick interrupt to update the stepper-motor service every 1 ms. The main loop is still responsible for reading sensors, refreshing the OLED, scanning keys, and deciding whether each axis should move, but the actual stepping is handled in the periodic interrupt routine. This separation is important because the OLED update and other tasks can take a noticeable amount of time. If motor stepping depended directly on the main loop, the motor motion would become uneven whenever the main loop was busy.

Each motor is controlled by a current position and a target position. In the code, these are represented by the motor's current step count and target step count. During each motor-service update, the firmware checks whether the current position has reached the target. If not, it advances the motor by one half-step in the required direction. In this way, the high-level controller only needs to update the target position, while the lower-level drive routine handles the actual step-by-step motion.

A lookahead target is used in auto-tracking mode. Instead of commanding only one step ahead each time, the firmware sets the target position about 20 half-steps ahead of the current position:

$$N_{\text{target}} = N_{\text{current}} \pm 20. \tag{7}$$

This value corresponds to the firmware constant MOTOR_LOOKAHEAD. The reason for doing this is practical. If the target were only one step ahead, the interrupt routine could finish that step in a few milliseconds and then wait for the main loop to issue another command. Since the main loop may be delayed by OLED communication or ADC processing, the motor would move in a start-stop pattern. With the lookahead target, the interrupt routine always has several steps available to execute, so the motor keeps moving smoothly while the light imbalance remains outside the deadband.

The lookahead mechanism is also easy to stop. When the light readings return to the balanced range, or when a manual key is released, the stop function resets the target position

to the current position and de-energizes the motor coils. This means the motor does not continue through the remaining lookahead steps after the controller decides that the axis should stop. In normal operation, the lookahead only keeps motion continuous; it does not remove the controller's ability to stop the axis quickly.

A linear acceleration profile is added to reduce missed steps at startup. The 28BYJ-48 motor cannot always jump directly to the desired cruise stepping frequency under load, especially when the yaw axis is driving the gear set and the rotating upper stage. If the stepping interval is too short at the beginning, the rotor may fail to follow the changing magnetic field and the motor can vibrate or lose steps. To avoid this, the firmware starts each movement with a longer step interval and then gradually shortens the interval until the cruise speed is reached.

The two axes use different speed settings because their mechanical loads are not the same. The pitch motor starts with a step interval of about 4 ms and accelerates toward a 2 ms interval, while the yaw motor starts more slowly, from about 5 ms toward 3 ms. The yaw axis is given the slower profile because it drives the rotating stage through the gear set and has higher effective inertia. After the motor stops, the acceleration counter is reset, so the next movement again starts from the safer low-speed region. This makes the motion more reliable during repeated tracking corrections.

2.7 User Interface and Firmware

Figure 5 summarizes the main firmware flow on the STM32. After power-on, the controller initializes the UART interface, key inputs, LEDs, ADC1, flash-backed settings, stepper-motor driver variables, and the OLED. The two axes are then moved to predefined initial targets, and the OLED shows the startup screen before the program enters the main loop.

The main loop is organized around several software flags rather than long blocking delays. Key scanning runs continuously, while ADC processing is performed only when the read flag is set. OLED refresh and flash saving are also handled in the loop, so mode changes can be stored without interrupting the motor-service routine described in Section ??. Serial telemetry is sent every 800 ms, which is slow enough not to overload the main loop but still fast enough to observe the system state during testing.

The OLED is used as the local feedback interface for the prototype. It reports the current mode, the four directional light readings, battery voltage, estimated battery percentage, charging voltage, and charge state. This is especially useful during demo operation because the system can show whether it is in automatic tracking mode, whether the LDR readings are balanced, and whether the power subsystem is detecting an input source.

The local key interface provides two basic operating modes. In automatic mode, the firmware uses the light-sensing logic from Section ?? and lets the motors track the dominant light direction. In manual mode, the direction keys directly command up, down, left, and right motion, which is useful for testing the two motor axes without relying on the light sensors. The source code also keeps a simple UART command path. A host can switch modes with

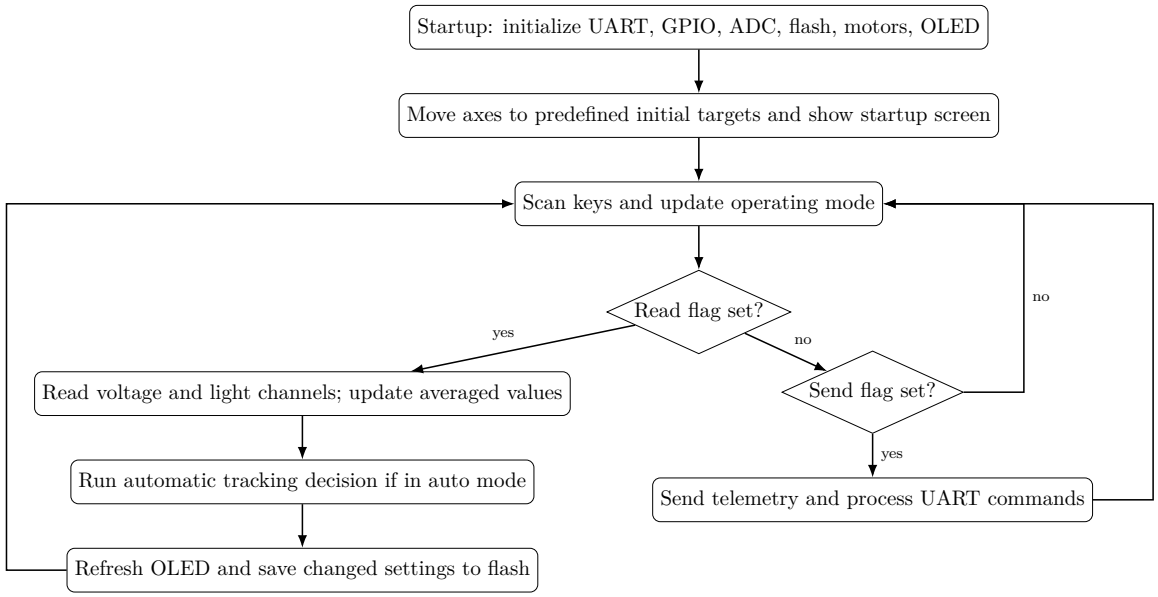

Figure 5: Main firmware flow implemented in the STM32 source code.

the sM command, or send individual direction commands using sU, sD, sL, and sR. This serial path was mainly kept for debugging and telemetry, while normal demonstration can still be done from the local keys and OLED.

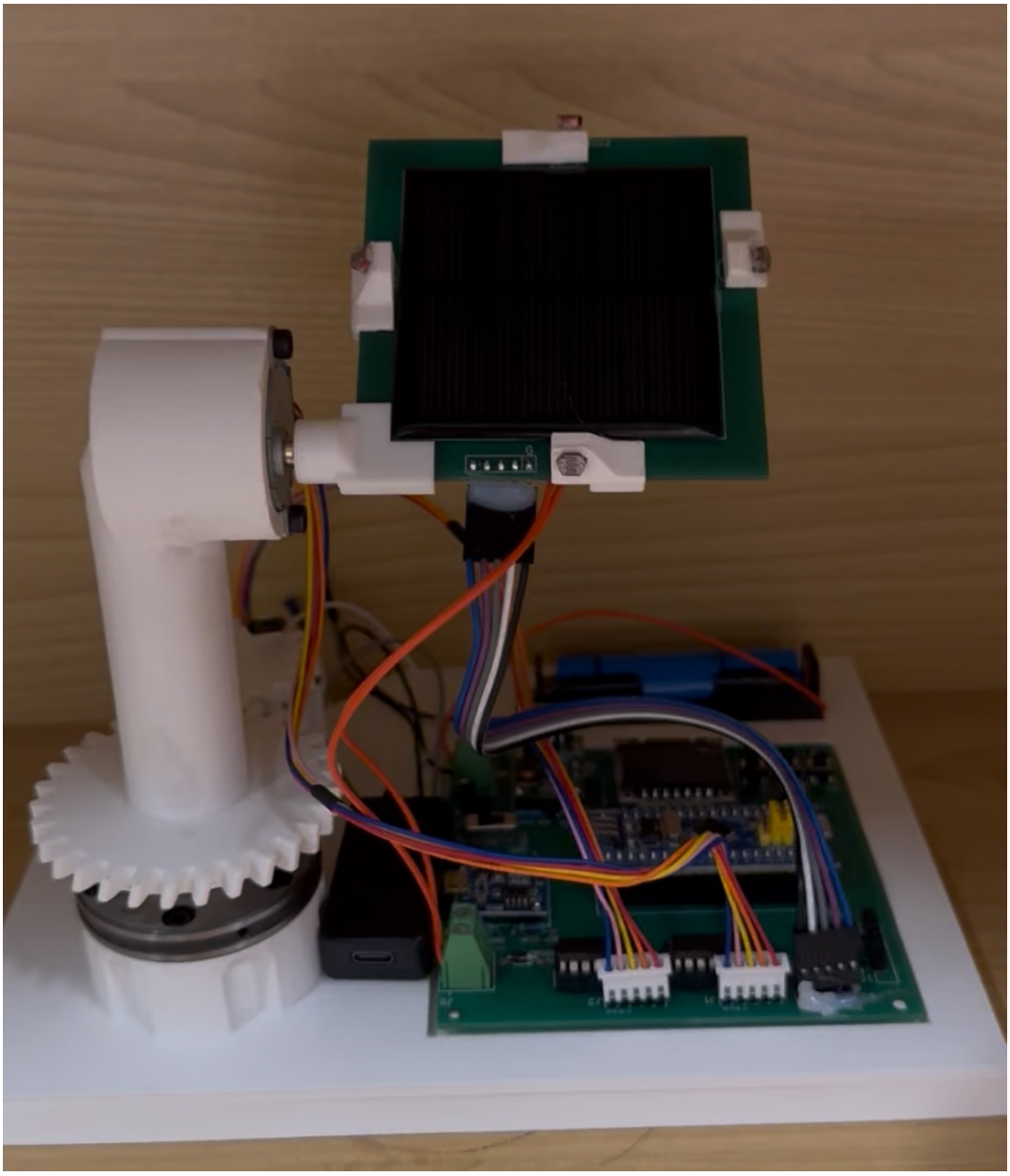

Figure 6: Final prototype of our project.

3 Verification

3.1 Verification Strategy

This section presents the testing performed on the completed solar-tracking system and its major functional blocks to demonstrate that the design goals established during the design review were met. Quantitative results are drawn from two bench experiments, in which live telemetry was logged over USART1 at 1.25 Hz to produce time-stamped CSV traces of operating mode, LDR readings, battery voltage, estimated system power, and panel charging-input voltage. The consolidated requirements and verification table is provided in Appendix A; the discussion below highlights the key high-level requirements and notes any that were not fully verified.

3.2 Battery Voltage Across Operating States

Figure 7 shows the captured battery terminal voltage V_{BAT} and the firmware's estimated instantaneous system power P across four back-to-back operating phases: idle (both motors off), left–right motor running, up–down motor running, and active solar charging. The voltage dividers in Equation 1 protect the ADC while preserving useful resolution, and the firmware sampling pipeline averages ten consecutive ADC reads per channel before each 200 ms display refresh.

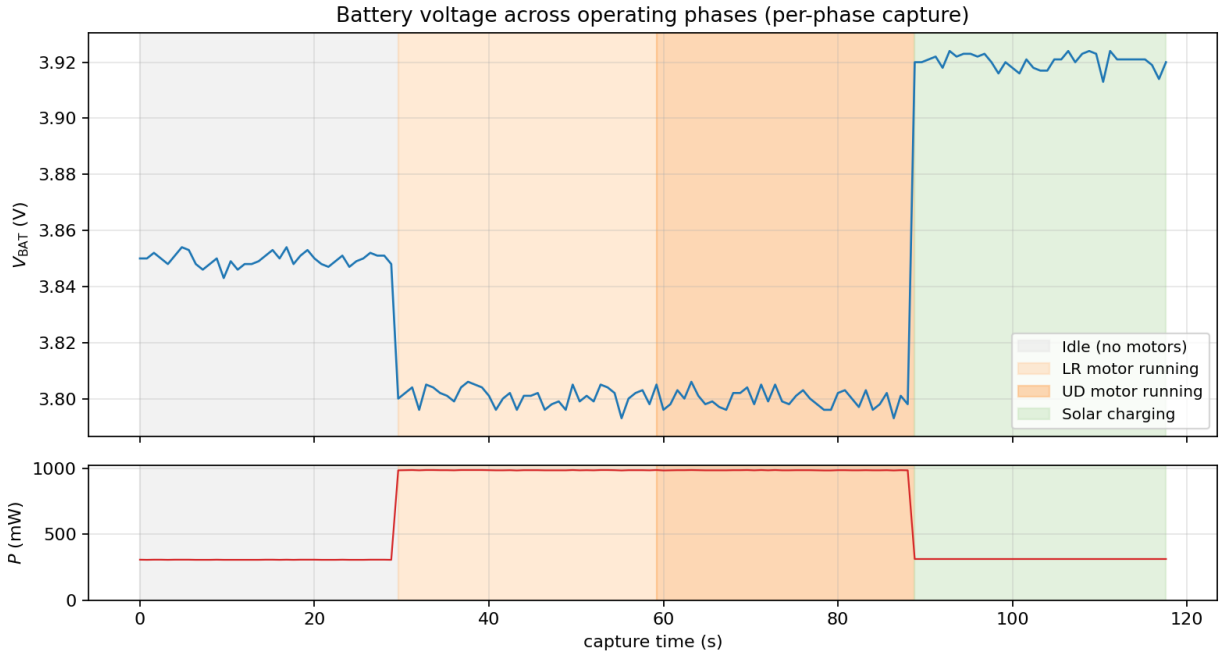

Figure 7: Battery terminal voltage V_{BAT} and estimated system power P recorded across four back-to-back operating phases. Each phase was captured for approximately 30 s; the segments are concatenated on a single capture-time axis with each phase shaded.

Three quantitative observations follow from the trace.

(1) In idle, the firmware reports approximately 300 mW, consistent with the system-power model $P = V_{\mathrm{BAT}} \cdot I_{\mathrm{baseline}}$ at $I_{\mathrm{baseline}} = 80\,\mathrm{mA}$. The terminal voltage drift over thirty seconds is below the firmware's 10 mV display resolution, confirming that the resting cell is the appropriate reference for the state-of-charge mapping in Equation 2.

(2) During motor activity, V_{BAT} sags by approximately 40 mV relative to the idle baseline. The magnitude is consistent with a cell internal resistance near 100 mΩ and a step in load current from 80 mA to roughly 440 mA when both stepper drivers are energized. The voltage returns to within a few millivolts of the prior idle level within two seconds of motor stop, demonstrating that the cell is not significantly stressed by the duty cycle expected of the tracker.

(3) Under active charging, V_{BAT} rises monotonically and the firmware's charge-detection flag asserts. The flag uses the relative criterion $V_{\mathrm{CHG}} > V_{\mathrm{BAT}} + 0.01\,\mathrm{V}$, after the phantom-suppression model clamps the panel reading to zero when no solar input is present.

These three behaviors satisfy the power-subsystem requirements: the ADC remains within its operating range, motor bursts do not pull the cell below the firmware's low-voltage threshold during ordinary operation, and the charging path delivers measurable input that is correctly classified by the firmware.

3.3 Auto-Mode Tracking Response

Figure 8 verifies the closed-loop tracking law. The system was placed in automatic mode with a static lamp aimed at the panel; once the stage had settled, the lamp was displaced laterally and the disturbance time was stamped into the log. The panel charging-input voltage V_{CHG} is plotted directly because it is the physical quantity the tracker is implicitly maximising: a higher V_{CHG} indicates better alignment between the panel surface normal and the dominant light source.

Two behaviors are explicitly verified. First, before the disturbance the trace holds at a stable peak, confirming that the firmware deadband $|L_{\mathrm{up}} - L_{\mathrm{down}}| \leq 4$ (and the matching condition on the left–right pair) terminates motion once the four LDR sensors are balanced and that the resulting orientation maximises V_{CHG}. Second, after the disturbance V_{CHG} drops sharply and then returns toward its pre-disturbance value within several seconds, demonstrating that the closed loop actively re-acquires the source rather than coasting open-loop. The recovery time observed here is an upper bound rather than a tight estimate, because the UART telemetry frame rate (1.25 Hz, set by the firmware send period of 800 ms) aliases transients faster than approximately 1.6 s; the qualitative dip-and-recover signature, however, is unambiguous.

Manual control was verified jointly with the closed loop. Because the key handler invokes the same stepper service used by the automatic controller, observing correct motion under manual key input also validates the GPIO–ULN2003–coil signal path that the auto mode depends on.

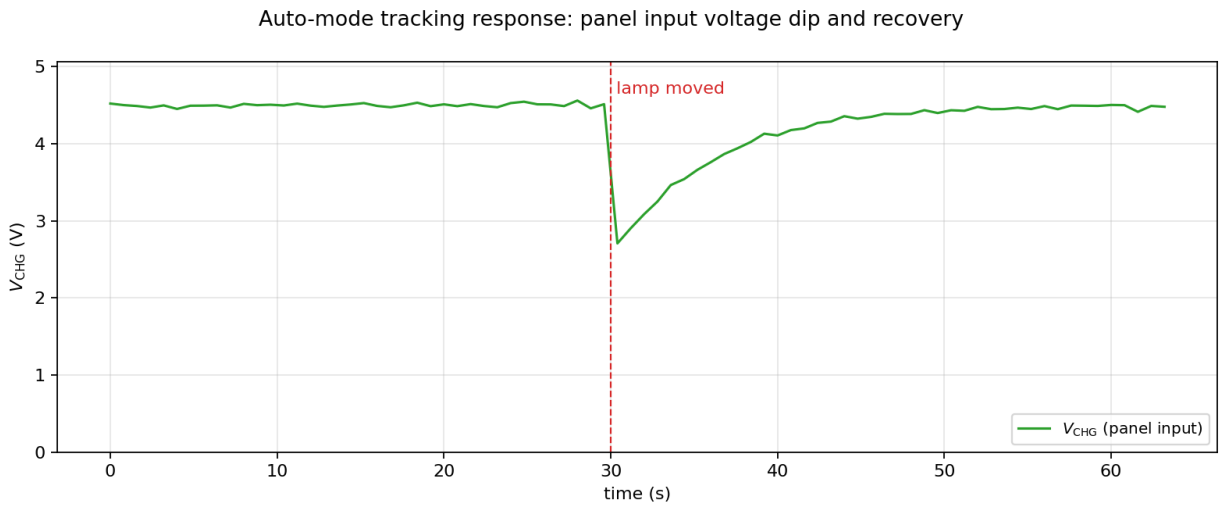

Figure 8: Auto-mode tracking response. The vertical dashed line marks the moment the lamp was displaced; the panel input voltage V_{CHG} dips and then recovers as the firmware drives the appropriate stepper axes (Equations ?? and ??) to re-balance the four LDR sensors.

4 Costs

4.1 Parts Cost

The project bill of materials lists the controller board, display, motor drive components, solar charging circuit, battery hardware, light-sensing components, connectors, wiring, and assembly materials. Since the supplied BOM gives total prices rather than unit prices, Table 3 reports the listed prototype cost directly in RMB. The total cost is ¥97, which is well below the 600 RMB prototype target in the reference document project-reference.

Table 3: Prototype parts cost from the supplied BOM

Part or module	Model / note	Quantity	Total cost (¥)
5.08 mm two-pin terminal block	2-pin terminal	2	2
18650 lithium-ion cell	18650 cell	1	7
18650 battery holder	18650 holder	1	3
Polarized capacitor	220 µF	1	1
Polarized capacitor	10 µF	1	1
5 V four-phase stepper motor	Stepper motor	2	8
5 V solar panel	Solar panel	1	14
Long solder wire, red	Red wire	1	2
Long solder wire, black	Black wire	1	2
Through-hole resistor	10 kΩ	6	1
High-precision photoresistor	LDR sensor	4	1
6 mm × 6 mm push button	Button switch	5	1
Two-position slide switch	Power switch	1	1
0.96 OLED	0.96 OLED	1	14
STM32 core board	STM32 core board	1	8
ULN2003 IC or stepper driver module	ULN2003	2	6
TP4056 lithium-ion charging module	Li-ion charger module	1	2
L7805CV voltage regulator	LM7805	1	1
USB 5 V boost module	Li-ion boost module	1	2
Photoresistor copper-clad sensor board	Blank LDR board	1	4
5-pin DuPont cable	5-pin cable	1	1
IC socket	16-pin socket	2	1
2.54 mm single-row right-angle header	5-pin header	1	1
2.54 mm single-row pin header	5-pin header	2	1
2.54 mm single-row pin header	8-pin header	1	1
2.54 mm single-row socket header	8-pin socket	1	1
2.54 mm single-row socket header	3-pin socket	2	1
2.54 mm single-row socket header	20-pin socket	2	1
Universal prototyping board	Perfboard	1	3
Solder wire	Assembly material	1	5
	Total prototype parts cost		97

4.2 Labor Cost

Following the ECE 445 cost-estimation guideline, labor is estimated as the product of hours worked, hourly rate, and a 2.5 overhead multiplier ece445-guide. Table 4 uses a conservative estimate of four team members, each contributing 90 h, with a nominal labor rate of \$40/h.

Table 4: Estimated labor cost

Contributor	Hours	Rate (\$/h)	Cost with 2.5 factor (\$)
Student 1	90	40	9000
Student 2	90	40	9000
Student 3	90	40	9000
Student 4	90	40	9000
	Estimated labor total		36 000

5 Conclusions

The final design demonstrates a compact STM32-based photovoltaic node that integrates dual-axis light tracking, lithium-battery charging, boosted system power, local display, manual control, and optional serial telemetry. This addresses the main motivation of the project: a small solar-powered node should not only orient its panel toward stronger illumination, but also provide local information about its energy condition. The completed prototype senses directional light, drives the panel toward balanced illumination, monitors the battery and charging inputs, and reports operating mode, light readings, battery voltage, battery output power, charging voltage, and charging state through the OLED display.

The firmware implements the core engineering behavior with explicit ADC scaling, sample averaging, a four-percentage-point tracking deadband, flash-backed settings, periodic display updates, and a periodic motor stepping service. These features allow the system to function as a self-contained low-cost demonstrator for small distributed photovoltaic applications.

Several limitations remain. The 5 V, 120 mA panel is suitable for demonstration and light-duty charging, but it cannot support a continuous high motor duty cycle in weak sunlight. The LDR readings are relative rather than calibrated in lux, so the system is best interpreted as a directional tracker rather than a precision irradiance instrument. A deployable outdoor version would also require weatherproofing, mechanical travel limits, fuse or resettable protection on the battery path, and more complete lithium-ion protection against over-discharge.

5.1 Ethical and Broader Impacts

The IEEE Code of Ethics emphasizes public safety, honest representation of limitations, and responsible handling of technology risks [1]. The most important safety issue in this project is the lithium-ion battery. A final product should use a protected cell or a protection board, prevent reverse polarity, avoid exposed conductive terminals, and clearly state charging limits. The optional camera and wireless versions also raise privacy and cybersecurity concerns; those versions should avoid unencrypted remote access and should make recording status visible to users.

The broader impact of the project is positive when deployed responsibly. A low-cost self-powered tracker can improve solar utilization for small educational, agricultural, and sensing applications, reducing dependence on disposable batteries or grid wiring in remote locations. At the same time, the environmental benefit depends on durable construction and proper recycling of the lithium cell and electronics at end of life.

References

[1] IEEE. “”IEEE Code of Ethics”,” Accessed: Feb. 8, 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html.

Appendix A Requirement and Verification Table

Table 5: Detailed requirement and verification table

Requirement	Verification method	Acceptance criterion	Result
The controller shall read four directional light sensors.	Inspect ADC channel mapping and display output for upper, down, left, and right light fields.	Four independent values are sampled and displayed on a 0–100 relative scale.	Implemented
The automatic mode shall track toward balanced illumination.	Apply a directional light difference and observe the firmware command the relevant motor axis.	Axis moves when difference exceeds 4 and stops inside the deadband.	Implemented
The manual mode shall allow local motor control.	Press the four direction keys while in manual mode.	Vertical and horizontal axes respond to the corresponding key pair.	Implemented
The mode key shall switch between automatic and manual operation.	Press KEY1 and observe the mode field and motor behavior.	Mode toggles and is reflected on the OLED.	Implemented
The battery voltage shall be measured safely by the STM32 ADC.	Calculate divider output for 4.2 V battery and 5 V charge input.	ADC input remains below 3.3 V.	Pass by design
The charging state shall be displayed.	Apply strong and weak external light.	OLED reports charging above the threshold and not charging below it.	Implemented
The OLED shall display all top-level user data.	Inspect the display strings generated by the firmware.	Mode, four light values, battery voltage, output power, charge voltage, and charge state are present.	Implemented
The settings shall survive power cycling.	Change mode and inspect flash-save call path.	Updated setting is written to the reserved flash sector after a mode change.	Implemented
The optional serial interface shall report system data.	Inspect the periodic UART send path.	The same display strings are transmitted every 800 ms.	Implemented in firmware

Appendix B Implementation Details

B.1 Pin Map

Table 6: STM32 pin usage from the supplied firmware

Function	STM32 pins or channels	Notes
Charge voltage ADC	ADC1 channel 7, PA7	Scaled by divider and reconstructed with factor 3.
Battery voltage ADC	ADC1 channel 5, PA5	Averaged over 30 samples for display.
Light ADC inputs	ADC1 channels 4, 3, 2, 1	Left, up, right, and down LDRs, respectively.
Keys	PB12, PB13, PB14, PB15, PA8	Mode, vertical pair, and horizontal pair.
Tilt stepper motor	PA0, PC15, PC14, PC13	ULN2003-driven four-phase sequence.
Left-right stepper motor	PA11, PA12, PA15, PB3	ULN2003-driven four-phase sequence.
OLED	PB4, PB5, PB6, PB7, PB8	Clock, data, reset, data/command, chip select.
UART	USART1 at 9600 baud	Optional wireless or serial telemetry path.

B.2 Firmware Constants

Table 7: Key firmware constants

Constant	Value	Purpose
CHANGE_VOLT	3.8 V	Charge-detection threshold.
RONGCHAZHI_UD	4	Upper/lower light deadband in percentage points.
RONGCHAZHI_LR	4	Left/right light deadband in percentage points.
zzAngle, fzAngle	$-10°$, $10°$	Incremental motor commands.
Battery averaging count	30 samples	Reduces displayed voltage jitter.
Light averaging count	5 samples	Reduces LDR noise before control decisions.
Telemetry period	800 ms	Periodic UART send interval.

Appendix C Bill of Materials

Table 8: Condensed bill of materials from the project package

Item	Quantity	Purpose
STM32F103C8T6 core board	1	Main controller, ADC, GPIO, UART, flash settings.
OLED-0.96-7-pin display	1	Local status display.
28BYJ-48 stepper motor	2	Dual-axis mechanical actuation.
ULN2003 driver	1 or 2 boards	High-current stepper coil switching.
TP4056 lithium battery charger module	1	Single-cell lithium-ion charging from solar input.
18650 lithium-ion cell and holder	1 each	Energy storage and removable battery mounting.
5 V, 1.2 A boost module	1	Boosts lithium-cell voltage to system 5 V.
5 V solar panel	1	Photovoltaic charging source.
5506 LDRs and divider resistors	4 LDRs plus passives	Directional light sensing.
Buttons and two-position switch	5 buttons plus power switch	User input and power control.
1N5819 Schottky diode	1	Reverse-current or polarity protection in the power path.
Capacitors and resistors	As required	Filtering, reset, divider, and support circuitry.
Headers, sockets, terminals, Dupont wires, solder, PCB	As required	Interconnect and assembly.
Optional Bluetooth/Wi-Fi/camera/cloud module	0 or 1	Wireless monitoring and remote control, depending on build option.