BetaSpray: Climbing Route Visualization System

Design Document

Prakhar Gupta (`prakhar7`), Ingi Helgason (`ingih2`),
Maxwell Beach (`mlbeach2`)

ECE 445 — Spring 2026

Contents

1 Introduction

1.1 Problem and Solution

Spray walls are a popular fixture in modern climbing gyms, consisting of a dense grid of holds mounted on a single wall surface. Unlike pre-labeled routes, spray walls allow climbers to define their own routes by choosing any subset of holds. This flexibility makes spray walls an excellent training tool, but it introduces a significant usability problem: there is no practical way to record, recall, or share a custom route once a climber steps off the wall. Currently, climbers must memorize which holds belong to a given route or resort to taking photographs and manually annotating them. This process does not scale, significantly limiting the potential use of spray walls as community gathering points and as resources for training.

The International Federation of Sport Climbing (IFSC) has noted the growing importance of structured training tools for competitive climbing [1]: for this purpose, there are existing commercial solutions, such as the Kilter Board and MoonBoard [2], which use integrated LEDs to display routes. However, these solutions are proprietary, and can be cost-prohibitive to purchase and install: BetaSpray allows gyms to re-invigorate existing spray wall setups as community gathering points, and improves community access to shared information and training resources, while being an order of magnitude cheaper than COTS, pre-packaged solutions.

1.1.1 Our Solution

BetaSpray is a standalone device which scans a spray wall, identifies individual holds, and uses motor-aimed laser pointers to visually highlight the holds for that route in real time. It consists of: a **vision mapping subsystem** to capture images of the wall and detects hold positions using an onbard computer vision pipeline; a **projection subsystem** that directs the lasers towards target holds via 2-axis pan-tilt brackets; and a **user interface subsystem** to allow climbers to create, store, and replay routes through a web application hosted by the ESP32-S3 microcontroller. A **power subsystem** provides regulated 5 V and 3.3 V rails to all components.

After a wall layout is scanned and hold positions are extracted, users generate routes which BetaSpray can "play back", sequentially illuminating the corresponding holds. Routes are stored on the ESP32's integrated 16 MB QSPI flash, and are uploaded and managed via BetaSpray's lightweight HTTP server, using a phone or laptop. The PCB footprint is approximately 10 cm × 10 cm, to keep its footprint reasonably small and compliant with project requirements. The device is designed to sit on the floor in front of a spray wall, powered from a standard wall outlet. (*note: climbing gyms typically have large mats on the ground to break falls. BetaSpray would be placed on the far edge or just beyond these mats, for the safety of climbers*).

1.2 Visual Aid

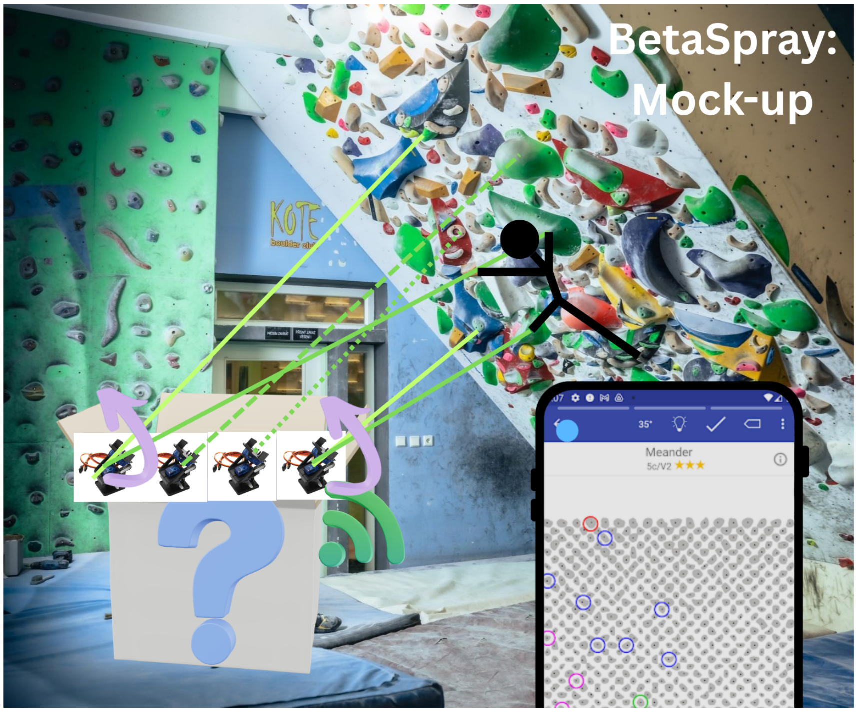

Figure 1: Mock up of BetaSpray in use: the device sits on the floor in front of a spray wall. The camera scans the wall and a phone or laptop connects over Wi-Fi to select a route, after which the servo-actuated laser gimbals illuminate the relevant holds in sequence.

1.3 High-Level Requirements

1. The laser projection subsystem shall direct each laser pointer to its target hold with a positional accuracy of ± 5 cm at a projection distance of 3 m, sufficient to unambiguously identify individual holds on a standard spray wall.

2. The vision mapping subsystem shall detect and localize at least 90% of holds on a spray wall under standard gym lighting conditions ($\geq$300 lux) within 30 seconds of initiating a scan.

3. The user interface subsystem shall allow a user to create, save, and replay a route with an end-to-end response latency of less than 200 ms from command input to first laser actuation.

4. All laser modules shall be Class 2 ($<$1 mW continuous visible output) to comply with IEC 60825-1 eye safety requirements [3].

2 Design

2.1 Block Diagram

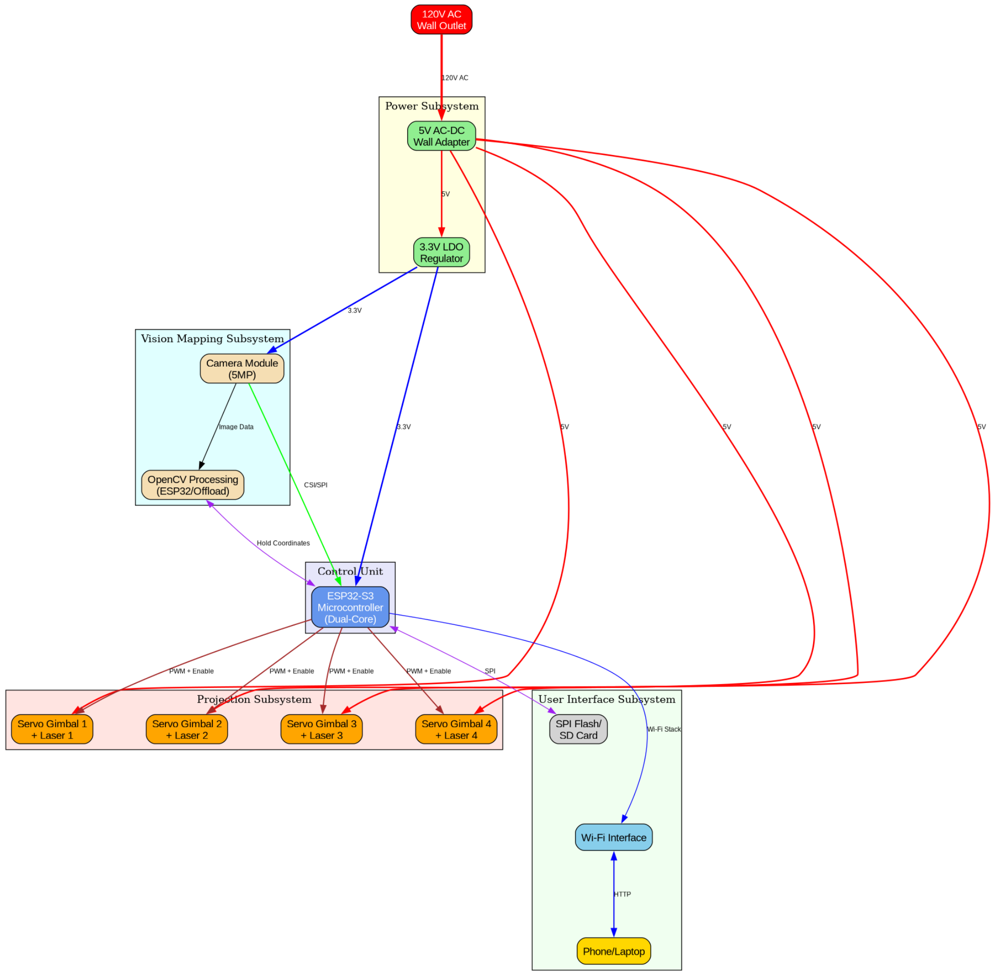

Figure 2: Top-level block diagram of the BetaSpray system. Voltage rails (5 V, 3.3 V) and data buses (PWM, SPI, Wi-Fi) are labeled on each inter-system connection.

The system is built around an ESP32-S3 microcontroller, which orchestrates all four subsystems. The power subsystem takes in power delivered over USB-C and provides 5 V and 3.3 V rails. The vision mapping subsystem feeds raw camera frames into the ESP32-S3 over an 8-bit DVP parallel bus. The projection subsystem receives PWM commands from the ESP32-S3 to drive up to four 2-axis servo gimbals (note: this value is derived from the system's goal of pointing at the next four holds). The user interface subsystem communicates with client devices over 802.11 Wi-Fi using a lightweight HTTP server. Route data is persisted via the 16 MB QSPI flash integrated in the module, which is of type ESP32-S3-WROOM-1-N16R8 [4].

2.2 Physical Design

The BetaSpray device is designed to rest on the floor in front of the spray wall. The PCB measures approximately 10 cm × 10 cm and mounts to the enclosure base via four 1/8 in mounting holes located 1/4 in from each corner. The OV5640 camera module is elevated and angled to cover the full wall extent from the floor-level vantage point. The four laser gimbals are arranged in a horizontal row with sufficient spacing to avoid mechanical interference at the extremes of their angular travel ($\pm 90°$ per axis).

2.3 Power Subsystem

2.3.1 Design

The power subsystem converts a standard 5 V USB-C wall adapter power input into the two rails required by the system. A 3.3 V low-dropout (LDO) linear regulator (the current plan is to use the LM3940IT-3.3) steps the 5 V rail down for the ESP32-S3, OV5640 camera, and logic-level peripherals. Currently, we are assuming that the laser modules and servo motors shall operate directly from the 5 V rail (our reasoning currently being to keep sensitive signals for control components separate from noisy power-like signals from the motors).

Decoupling capacitors are placed near major power consumers, in a manner we believe to be appropriate: 100 μF electrolytic bulk capacitors are placed near the 5 V and 3.3 V rails to absorb low-frequency transients from servo motor switching. Per Espressif's Hardware Design Guidelines [5], a collection of decoupling capacitors (100 nF, 1 μ, and 10 μF) are placed near the ESP32-S3's power input, to address a mixture of low-frequency (Wi-Fi current draw), high-frequency, and mid-band frequencies to prevent power rail droop.

A USB-C receptacle is used to provide power. USB-Serial support is built in to the S3 series microcontroller we order, such that the USB-C connector also provides the facility to program BetaSpray.

2.3.2 Sufficiency of LDO: Analysis

The LM3940IT-3.3 [6] seems well-suited for our application. It is a true low-dropout regularor rated to support a 1 A output current. Per the Espressif's posted guidelines (Section 1.3.2 [5]), this is sufficient to power the microcontroller, which should exhibit a peak current draw of approximately 500 mA. Per a datasheet describing Arducam OV5640 camera modules [7], we expect the camera on its most agressive settings to have a peak current draw of approximately 300 mA.

We ensure that the LDO has sufficient input and output capacitances, per the typical application values provided by Texas Instruments, such that this rated current can be maintained.

From our initial research we anticipate minimal risk of power rail droop under typical current draw patterns (with our pessimistic momentary peak current draw at the LDO output assumed to be $\sim$800 mA). The ESP32 high current events should be transient, and the OV5640 camera parameters can be adjusted to improve power/draw characteristics, it seems. Any additional components on the 3.3V bus should have low, predictable current draw, keeping us well within the operating regime of the LDO.

2.3.3 Draft Schematic

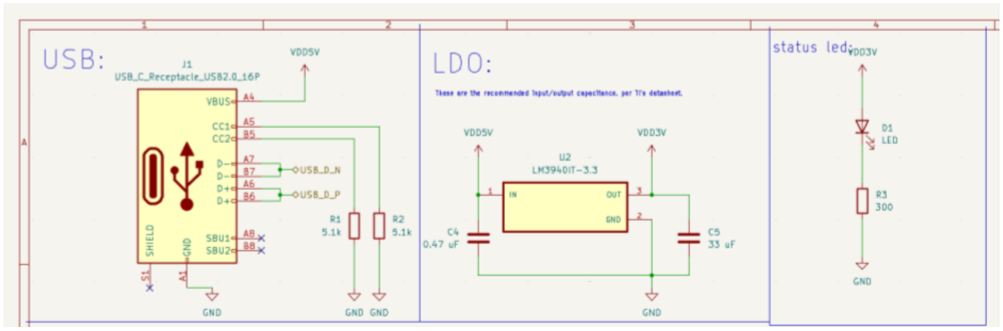

Figure 3: Working draft of power subsystem schematic. The LDO is paired with its minimum input capacitances.
Note that the decoupling capacitances not immediately related to the LDO are isolated to those subsystem/components' sheets.

2.3.4 Flowchart

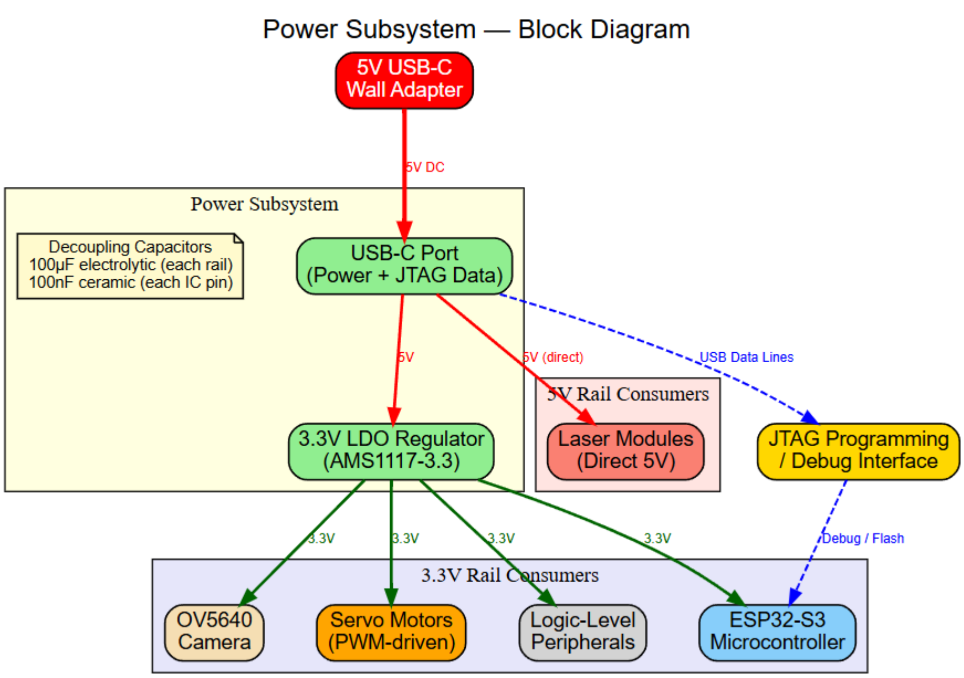

Figure 4: Power subsystem flowchart.

2.3.5 Requirements and Verification

Requirement	Verification
The 5 V rail must supply sufficient current to power the LDO input and laser modules continuously.	Measure rail voltage with a digital multimeter with all lasers enabled and the LDO is under full load. Verify rail stays within $5\,\text{V} \pm 0.25\,\text{V}$.
The $100\,\mu$F bulk capacitors on both rails must suppress PWM switching ripple to $<$250 mV pk-pk under normal load (though preferably much lower). *This is required such that the LDO provides a sufficiently accurate 3.3 V output.*[1]	Measure rail ripple on an oscilloscope during PWM operation. Verify ripple $< 250\,\text{mV}$ pk-pk.

[1] We find it difficult to estimate how much ripple we should expect, or how much is acceptable on our 5 V rail.

2.4 Vision Mapping Subsystem

2.4.1 Design

The vision mapping subsystem is responsible for scanning the spray wall and detecting the positions of climbing holds. An OV5640 camera module (5 MP, up to 2592×1944) with autofocus communicates with the ESP32-S3 via an 8-bit DVP (Digital Video Port) parallel interface. The DVP signals—8 pixel-data lines (D2–D9), HREF, VSYNC, PCLK—are routed to dedicated ESP32-S3 camera-interface GPIO pins avoiding strapping pins (GPIO0, 3, 45, 46), PSRAM SPI pins (GPIO35–37), and USB pins (GPIO19–20). Camera configuration is handled over I^2C (SCCB protocol) on a bus. While I^2C supports sharing a bus between different devices, we only use the OV5640 and the ESP in this case.

The external XCLK line is left unconnected; the OV5640 uses its internal 24 MHz oscillator.

The wall-scanning pipeline proceeds in four stages:

1. **Capture:** Trigger the OV5640 and read one full-resolution frame over DVP.
2. **Flash save:** Write the raw frame as a file on the FatFS volume (e.g. write as `/img/wall.raw`). This decouples capture and processing iterations, and allows CV to be re-run without recapturing.
3. **Hold detection:** Load the saved frame; de-distort using a stored lens calibration matrix; convert to HSV; apply color thresholding and Canny edge detection to locate hold contours. Discard contours below a minimum area threshold.
4. **Angular coordinate extraction:** Convert each detected hold centroid from pixel coordinates to angular coordinates (ϕ_i, θ_i) using the geometric model in Section 2.5. Write the complete angular coordinate list to a FatFS file (e.g. `/holds.json`).

Post-MVP, WATCH shall use the same camera to detect whether the climber is contacting the currently highlighted hold; see the state machine below.

2.4.2 Flowchart

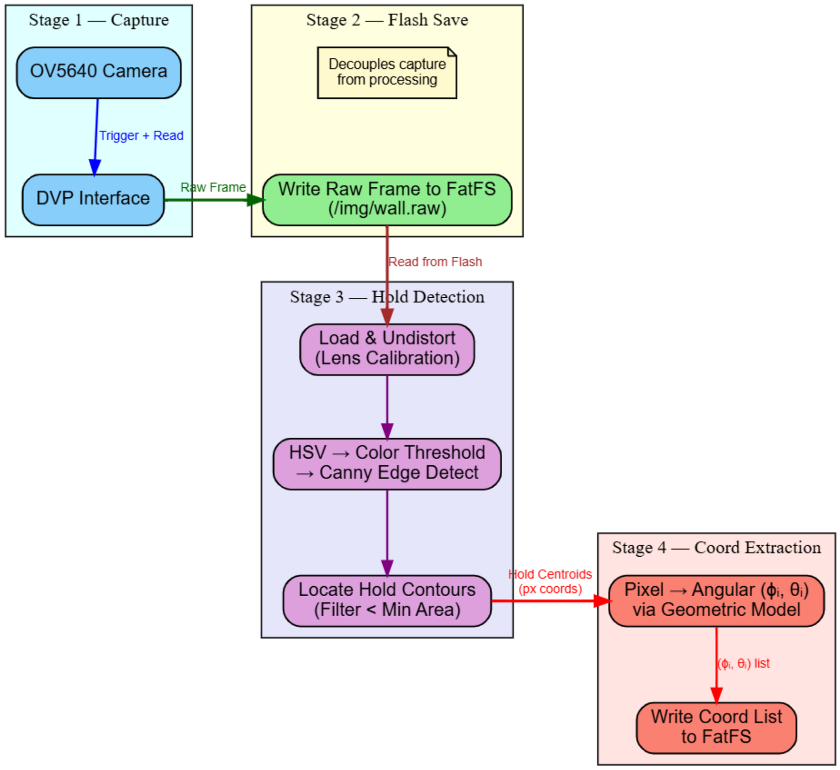

Figure 5: Vision mapping subsystem flowchart.

2.4.3 State Machine

MVP path: IDLE → CAPT → SAVE → PROC → READY. Post-MVP adds WATCH mode for climber detection, to auto-advance the indicator lasers to the next holds when appropriate.

State	Event	Next State	Action
IDLE	`scan_req`	CAPT	Trigger OV5640 frame capture
CAPT	Frame ready	SAVE	Write raw frame to SPI flash image partition
SAVE	Write complete	PROC	Run CV pipeline on flash image
PROC	Holds found ($N \geq 1$)	READY	Convert centroids to (ϕ_i, θ_i); save coord list to flash
PROC	Detection failed	IDLE	Log error; emit `scan_failed`
READY	`scan_req`	CAPT	Re-trigger capture
READY	`get_holds()`	READY	Return angular coord list to caller
Post-MVP: climber detection			
READY	`watch_start(hold_id)`	WATCH	Load target (ϕ, θ) from coord list; start frame loop
WATCH	Frame ready, climber not on hold	WATCH	Continue monitoring
WATCH	Frame ready, climber on hold $\pm\varepsilon$	WATCH	Emit `hold_reached`; load next target
WATCH	`watch_stop`	READY	Stop frame loop

Table 2: Vision mapping subsystem FSM.

2.4.4 Requirements and Verifications

Requirement	Verification
The camera module must capture images at a minimum resolution of 2592×1944 (5 MP) to resolve individual holds at distances up to 4 m.	Capture a test image at maximum resolution and verify the JPEG header reports 2592 × 1944. Visually confirm individual 10 cm holds are distinguishable at 4 m.
The hold detection algorithm must correctly identify at least 90% of holds on a standard spray wall under gym lighting (≥300 lux).	Place 20 reference holds on the test wall. Run the detection pipeline and count detected holds. Repeat 5 times; average detection rate must be ≥90%.
The full wall scan and hold detection pipeline must complete within 30 s of initiating a scan.	Time the pipeline from scan trigger to hold-list output using ESP32-S3 microsecond timestamps. Verify completion ≤30 s in 5 consecutive runs.

Requirement	Verification
Each detected hold coordinate must be accurate to within ± 2 cm in the wall plane.	Compare pipeline-reported hold positions to manually measured ground-truth positions for 10 reference holds. Verify all errors ≤ 2 cm.
During playback, laser spot positions must be detected and reported to the projection subsystem at ≥ 5 Hz with position accuracy ≤ 3 cm.	Command the gimbal to a fixed angle. Log camera-reported laser position at 10 Hz. Verify update rate ≥ 5 Hz and spot localization error ≤ 3 cm against a ruler on the wall.

2.5 Projection Subsystem

2.5.1 Design

The projection subsystem visually marks route holds using laser pointers mounted on 2-axis servo gimbals. Up to four gimbals are each driven by eight PWM signals from the ESP32-S3 (50 Hz, 1–2 ms pulse width). SG90 micro servos provide $\geq 180°$ travel on each axis with a resolution of $\approx 1°$/step and are powered from the 3.3 V rail. Class 2 KY-008 laser modules (<1 mW red, 650 nm) are enabled or disabled via GPIO-driven MOSFETs.

Coordinate Transform. To command the gimbal to illuminate wall position (X_w, Y_w) (in cm, wall-plane coordinates with origin at the wall's lower-left corner), we compute the required azimuth ϕ and elevation θ from the gimbal's known position (x_g, y_g, z_g) relative to the wall:

$$\Delta X = X_w - x_g, \quad \Delta Y = Y_w - y_g, \quad \Delta Z = z_g \tag{1}$$

$$\theta = \arctan\left(\frac{\Delta Y}{\sqrt{\Delta X^2 + \Delta Z^2}}\right) \tag{2}$$

$$\phi = \arctan\left(\frac{\Delta X}{\Delta Z}\right) \tag{3}$$

Angles (ϕ, θ) are converted to PWM pulse widths using the servo's calibrated angle-to-pulse mapping, stored as a lookup table in flash.

Closed-Loop Vision Correction. Because hobby servos do not provide position feedback, we implement a software feedback loop. After each PWM command, the vision subsystem observes the laser spot and returns the 2D wall error $(\varepsilon_X, \varepsilon_Y)$ (in cm). A proportional corrective step is applied:

$$\Delta\phi \leftarrow \Delta\phi - K_p \cdot \varepsilon_X \tag{4}$$

$$\Delta\theta \leftarrow \Delta\theta - K_p \cdot \varepsilon_Y \tag{5}$$

where K_p is a tunable gain. As derived in Section 2.8, two to three iterations reduce the error below the ± 5 cm threshold. The loop runs at 5–10 Hz, limited by camera frame rate.

Open-Loop Fallback (Dead Reckoning). If laser spot detection via the camera proves infeasible (e.g., insufficient spot contrast, processing overhead, or occlusion by the climber), the system operates entirely in Layer 1 using calibrated dead reckoning. At startup each gimbal is driven to a fixed reference angle pointing at a known wall landmark; all subsequent hold coordinates are computed relative to this origin using the geometric model. The angle-to-PWM LUT is manually tuned during a one-time calibration pass and stored on FatFS. This mode requires no camera feedback during playback and serves as the MVP fallback if vision-based correction cannot be validated in time.

2.5.2 Flowchart

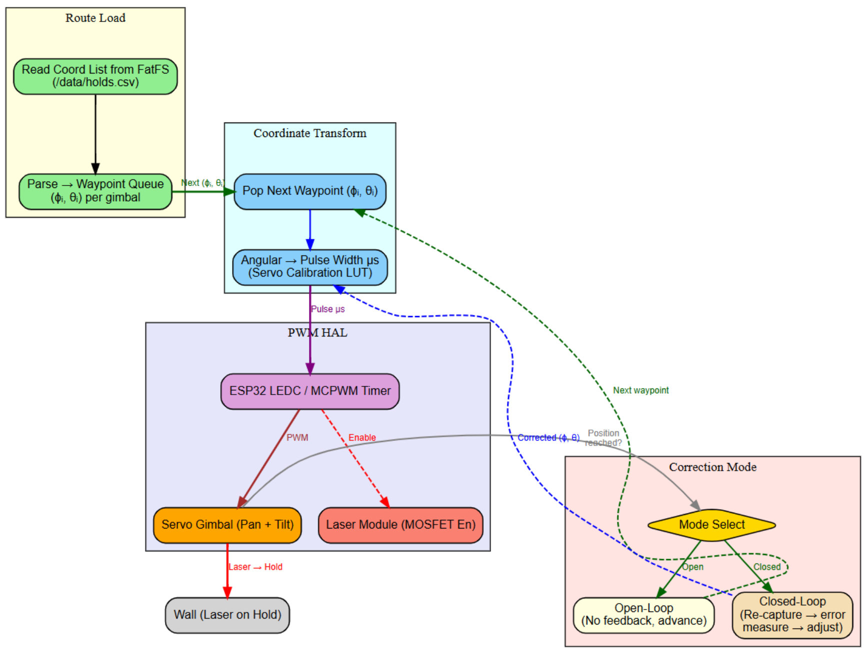

Figure 6: Projection subsystem flowchart.

2.5.3 State Machines

The projection software is organized in three HAL layers plus a route-playback orchestrator. **MVP** uses Layer 1 (open-loop) and the Route Playback FSM. Layer 2 (closed-loop) is activated post-MVP once vision tracking is available.

Layer 0 — PWM HAL. Raw duty-cycle interface to the ESP32-S3 LEDC/MCPWM peripheral. Exposed as a synchronous API to Layer 1; no asynchronous events.

State	Event / Guard	Next State	Action
OFF	`enable(ch, d)`	ON	Configure LEDC timer; write duty `d` to channel `ch`
ON	`set_duty(ch, d)`	ON	Update duty register; applies next PWM period
ON	`disable(ch)`	OFF	Set duty to 0; stop PWM output

Table 4: Layer 0: PWM HAL FSM (per channel).

Layer 1 — Open-Loop Gimbal. Maps target angles (ϕ, θ) to duty cycles via a calibration LUT. **Used in MVP.**

State	**Event / Guard**	**Next State**	**Action**
IDLE	`point(`ϕ`, `θ`)`	MOVE	LUT lookup $\rightarrow (d_{\text{az}}, d_{\text{el}})$; HAL enable+duty; start settle timer
MOVE	Settle timer expired	HOLD	Emit `settled`
MOVE	`point(`ϕ`, `θ`)`	MOVE	Re-lookup; update HAL duties; reset timer
HOLD	`point(`ϕ`, `θ`)`	MOVE	LUT lookup; update HAL duties; restart timer
HOLD	`disable()`	IDLE	HAL disable both axes
MOVE	Settle timeout	FAULT	HAL disable all; emit `fault`
FAULT	`reset()`	IDLE	—

Table 5: Layer 1: open-loop gimbal FSM (per gimbal).

Layer 2 — Closed-Loop Gimbal (post-MVP). Wraps Layer 1; adds vision-feedback correction loop.

State	**Event / Guard**	**Next State**	**Action**
IDLE	`point(`X_w`, `Y_w`)`	CMD	Compute (ϕ, θ); call L1 `point()`
CMD	L1 `settled`	OBS	Request laser spot from vision
OBS	Spot rcvd, $\lvert\varepsilon\rvert > \varepsilon_{\text{thr}}$	CORR	$\phi \leftarrow \phi - K_p\varepsilon_X$; $\theta \leftarrow \theta - K_p\varepsilon_Y$; call L1 `point()`
CORR	L1 `settled`	OBS	Request spot
OBS	Spot rcvd, $\lvert\varepsilon\rvert \leq \varepsilon_{\text{thr}}$	DONE	Hold position
DONE	`point(`X_w`, `Y_w`)`	CMD	Compute new (ϕ, θ); call L1 `point()`
DONE	`disable()`	IDLE	Call L1 `disable()`
OBS	Vision timeout	FAULT	L1 `disable()`; emit `fault`
FAULT	`reset()`	IDLE	—

Table 6: Layer 2: closed-loop gimbal FSM (post-MVP, per gimbal).

Route Playback Orchestrator (MVP). Drives gimbals through a time-series of holds at a fixed dwell time T_{dwell}. MVP: advances on dwell timer (climber follows device pace). Full: advances on `hold_reached` from vision subsystem (climber on hold $\pm\varepsilon$). Route is a list of angular coordinate pairs (ϕ_i, θ_i) loaded from SPI flash.

State	Event / Guard	Next State	Action
IDLE	`start(route)`	PROJ	$i \leftarrow 0$; load (ϕ_0, θ_0); `point()`; laser on; start dwell timer
PROJ	Dwell exp, $i < N-1$ [MVP]	PROJ	$i \leftarrow i+1$; load (ϕ_i, θ_i); `point()`; reset timer
PROJ	Dwell exp, $i = N-1$ [MVP]	IDLE	Laser off; emit `route_done`
PROJ	`hold_reached`, $i < N-1$ [full]	PROJ	$i \leftarrow i+1$; load (ϕ_i, θ_i); `point()`
PROJ	`hold_reached`, $i = N-1$ [full]	IDLE	Laser off; emit `route_done`
PROJ	`pause()`	PAUSE	Stop dwell timer
PAUSE	`resume()`	PROJ	Restart dwell timer
PAUSE	`stop()`	IDLE	Laser off
(any)	Gimbal fault	IDLE	Laser off; emit `error`

Table 7: Route playback orchestrator FSM (MVP). T_{dwell} configurable.

2.5.4 Requirements and Verifications

Requirement	Verification
Each servo gimbal must provide $\geq$180° of rotation on each axis with an open-loop positioning resolution of $\leq$1°.	Command the servo across its full range in 1° steps using a calibration jig with a protractor. Verify mechanical travel $\geq$180° and no missed steps.
Each laser module must be Class 2 (<1 mW continuous output at 650 nm) to comply with IEC 60825-1.	Measure laser optical power with a calibrated photodiode power meter. Verify output <1 mW.
Each laser must produce a visible spot of $\geq$1 cm diameter at 4 m to unambiguously identify the targeted hold.	Project each laser onto a wall at 4 m and measure spot diameter with a ruler. Verify $\geq$1 cm.
The servo transition time between any two target holds must be <500 ms.	Command a worst-case angle change ($\geq$90° on both axes simultaneously) and measure elapsed time from PWM update to mechanical settle using oscilloscope and high-speed camera. Verify <500 ms.

Requirement	Verification
After closed-loop correction, the laser spot must be within ± 5 cm of the target hold at 3 m projection distance.	Command the gimbal to 20 randomly chosen wall coordinates. After correction loop convergence, measure residual error with a ruler. Verify all 20 errors ≤ 5 cm.

2.6 User Interface Subsystem

2.6.1 Design

The user interface subsystem enables climbers to manage routes via a web application served directly by the ESP32-S3. The ESP32-S3 runs an HTTP server using the ESP-IDF networking stack. Static HTML/CSS/JavaScript files for the web interface are stored in the SPI flash partition. Route data (angular coordinate lists for selected holds, route names) is stored as JSON files on a FatFS volume mounted on the integrated 16 MB QSPI flash of the ESP32-S3-WROOM-1-N16R8. The device stores a maximum of 5 routes concurrently (e.g. `/routes/r0.json` . . . `/routes/r4.json`). Static web assets are also served from FatFS. The ESP32-S3 operates as a Wi-Fi access point in standalone mode, enabling use without external network infrastructure.

The web interface provides four primary workflows:

1. **Scan:** Triggers the vision mapping pipeline and displays the detected hold map as an SVG overlay on a wall thumbnail.
2. **Create:** Allows the user to click holds on the displayed map to build a route and assign it a name.
3. **Save:** Serializes the route and writes it to the flash store via a POST request.
4. **Replay:** Selects a saved route and commands the projection subsystem to illuminate holds sequentially.

2.6.2 Flowchart

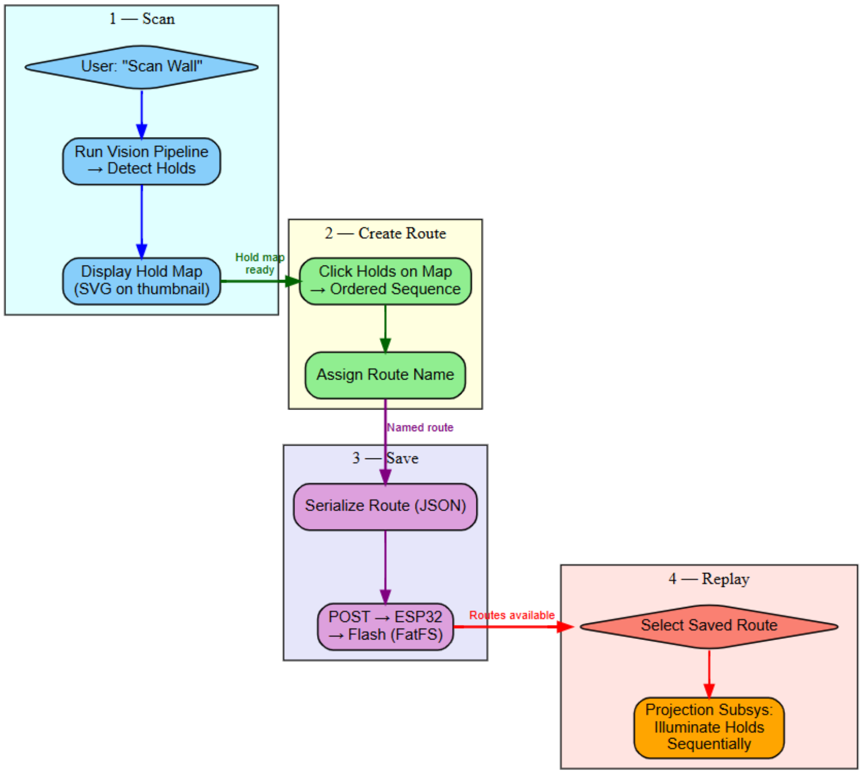

Figure 7: User interface subsystem flowchart.

2.6.3 State Machine

The HTTP server is event-driven; FSM state tracks what the system is currently doing. MVP requires IDLE, SCAN, EDIT, and PLAY states. Max 5 routes on flash.

State	Event / Guard	Next State	Action
INIT	Boot complete	IDLE	Start Wi-Fi AP; start HTTP server
IDLE	`POST /scan`	SCAN	Invoke `vision::scan()`
SCAN	Scan complete	IDLE	Cache hold map; respond 200 + hold list
SCAN	Scan failed	IDLE	Respond 500
IDLE	`POST /route/create`	EDIT	Open empty route buffer
EDIT	`POST /route/hold/{id}`	EDIT	Append hold to buffer
EDIT	`POST /route/save`	IDLE	Write route to flash; respond 200; respond 409 if 5-route limit reached
EDIT	`POST /route/discard`	IDLE	Clear buffer; respond 200
IDLE	`POST /route/{id}/replay`	PLAY	Call `playback::start(route)`; respond 202
PLAY	Route complete	IDLE	Notify client
PLAY	`POST /route/stop`	IDLE	Call `playback::stop()`; respond 200
(any)	Hardware fault	FAULT	Laser off; respond 503 on pending requests
FAULT	`POST /reset`	INIT	Restart subsystems

Table 9: User interface subsystem FSM.

2.6.4 Requirements and Verifications

Requirement	Verification
The ESP32-S3 HTTP server must respond to client GET and POST requests within 100 ms under normal operating conditions.	Use a browser dev-tools network panel or `curl` with timing to measure round-trip HTTP response time for 20 requests. Verify all ≤100 ms.
The system must support storage of at least 50 routes, each containing up to 30 holds, on SPI flash.	Create 50 routes of 30 holds each via the API, then verify all are retrievable and correct. Confirm SPI flash usage does not exceed partition size.
The web interface must be accessible from Chrome, Safari, and Firefox without app installation.	Load the interface on each browser. Verify all four workflows (scan, create, save, replay) function correctly.

Requirement	Verification
The Wi-Fi link must remain stable at distances up to 10 m from the device in a typical gym environment.	Connect a client device at 10 m with $\geq$2 walls between the device and client. Perform 20 HTTP transactions; verify $\leq$1 failure.
End-to-end latency from route-replay command input to first laser actuation must be <200 ms.	Instrument the firmware with microsecond timestamps at HTTP request receipt and first PWM output. Measure latency for 10 replay commands; verify all $\leq$200 ms.

2.7 System Integration

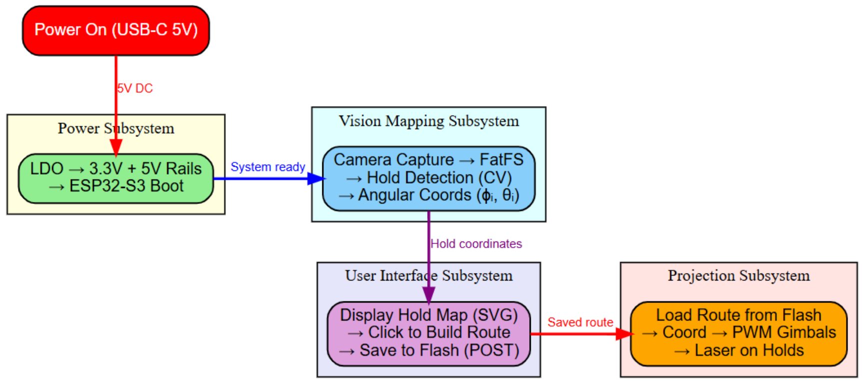

Figure 8: End-to-end system integration flowchart.

2.8 Tolerance Analysis

The most critical tolerance (not related to power integrity) we have identified in the BetaSpray system is the angular accuracy of the servo-actuated laser gimbals, as a pointing error placing the laser on the wrong hold renders the system ineffective in its stated purpose.

Geometric Model. Consider a laser gimbal at floor level, horizontal distance d from the wall, pointing at a hold at height h. The required elevation angle is:

$$\theta = \arctan\left(\frac{h}{d}\right)$$

A small servo angular error $\Delta\theta$ (rad) produces a wall positional error:

$$\begin{aligned} x(\theta) &= d\tan\theta \\ \frac{\Delta x}{\Delta\theta} &= d \cdot \sec^2\theta \\ \frac{\Delta x}{\Delta\theta} &= \frac{d}{\cos^2\theta} \\ \Delta x &= \frac{d}{\cos^2\theta} \cdot \Delta\theta \end{aligned}$$

Worst-Case Open-Loop Error. For $d = 3\,\text{m}$ and $h = 4\,\text{m}$ (near the top of a standard spray wall), $\theta = \arctan(4/3) \approx 53.1°$. The SG90 servo has a rated accuracy of $\approx \pm 1°$ ($\Delta\theta = 0.0175\,\text{rad}$):

$$\Delta x = \frac{3}{\cos^2(53.1°)} \cdot 0.0175 \approx \frac{3}{0.360} \cdot 0.0175 \approx 14.6\,\text{cm} \tag{6}$$

This exceeds the ± 5 cm requirement. At a lower angle ($h = 2$ m, $\theta \approx 33.7°$) the error reduces to $\approx$6.2 cm—still above spec.

Two-Stage Mitigation.

1. **Calibration mapping:** After wall scan, the system projects each laser onto four known reference points and records the actual versus commanded angle. A correction lookup table reduces the effective servo error to $\approx \pm 0.5°$.

2. **Closed-loop vision feedback:** During playback the camera observes the laser spot and computes positional error $(\varepsilon_X, \varepsilon_Y)$. A proportional controller applies corrective PWM adjustments, converging in 2–3 iterations. Assuming the vision subsystem measures positions to within ± 3 cm, the corrected servo error after convergence is $\Delta\theta_{\text{corr}} \approx 0.003$ rad:

$$\Delta x_{\text{final}} = \frac{3}{0.360} \cdot 0.003 \approx 2.5\,\text{cm} \quad (\leq \pm 5\,\text{cm} \checkmark) \tag{7}$$

The calibration procedure adds approximately 30 s to setup time; the real-time feedback loop operates at 5–10 Hz with no perceptible latency during route playback.

Dead-Reckoning Fallback Accuracy. If vision-based correction (Stage 2) is unavailable, Stage 1 calibration alone leaves a residual error of $\approx \pm 0.5°$ ($\Delta\theta = 0.00873$ rad). At the worst-case angle:

$$\Delta x_{\text{cal}} = \frac{3}{0.360} \cdot 0.00873 \approx 7.3\,\text{cm} \tag{8}$$

This exceeds the ± 5 cm spec but is usable as a degraded-mode fallback, particularly at lower wall angles. Iterative manual LUT tuning beyond the four-reference-point calibration can further reduce this error.

3 Cost and Schedule

3.1 Parts

Description	Manufacturer	Part #	Qty	Cost
ESP32-S3-DevKitC-1 (prototyping)	Espressif	ESP32-S3-DevKitC-1	1	$10.00
ESP32-S3 module (PCB)	Espressif	ESP32-S3-WROOM-1U-N16R8	1	$7.00
OV5640 camera module	Arducam	B0272	1	$25.00
3.3 V LDO regulator (LM3940IT)	TI	LM3940IT-3.3	1	$1.71
Class 2 laser diode module (KY-008)	Generic	KY-008	4	$8.00
Micro servo motor (SG90)[2]	Tower Pro	SG90	8	$16.00
2-axis pan-tilt bracket	Generic	—	4	~$20.00
USB-C receptacle	GCT	USB4105-GF-A	1	$0.80
Bulk electrolytic cap 100 μF	Panasonic	ECE-A1HKA101	4	$2.00
Ceramic cap 100 nF	Not relevant	—	20	$1.00
Parts subtotal				**~$91.51**

3.2 Schedule

Week	Tasks	Responsible	Deliverable / Milestone
1–2	Finalize PCB schematic (all subsystems); KiCad DRC clean; submit first PCB order round	Prakhar (vision schematic), Ingi (power), Max (servo)	PCB order submitted
3	Design document draft; tolerance analysis extension; FSM diagrams for each subsystem	All	Design Document due

[2]We are considering better motors, per discussion with E-Shop staff

Week	Tasks	Responsible	Deliverable / Milestone
4	Receive PCB Rev 1; bring up power rails; flash ESP32-S3 bootloader; verify camera DVP communication	Prakhar, Ingi	Power-on test passing
5	Implement OV5640 driver; capture test frames at full resolution; verify SCCB register config	Prakhar	Camera streaming at 2592×1944
6	Implement CV pipeline (HSV threshold, Canny, homography); hold detection on test wall	Prakhar, Max	$\geq$90% hold detection on 20-hold test set
7	Calibrate servo gimbals; implement coordinate transform; open-loop pointing test	Max, Ingi	Lasers pointing to $\pm$15 cm (pre-correction)
8	Implement closed-loop vision correction; integrate with CV pipeline; verify $\pm$5 cm	All	Closed-loop accuracy $\leq$5 cm
9	Implement HTTP server; route create/save/replay API; web frontend	Ingi, Max	Route playback via browser
10	System integration; end-to-end latency test ($<$200 ms); flash storage stress test (50 routes)	All	Mock Demo passing all R&V
11	PCB Rev 2 (bug fixes from Rev 1); re-run R&V suite	All	PCB Rev 2 order
12	Final demo preparation; enclosure assembly; safety verification (laser power measurement)	All	Final Demo
13	Final report writing; lab notebook completion	All	Final Report due

4 Societal Impact, Standards, Ethics, and Safety

4.1 Public Health, Safety, and Welfare

BetaSpray has the potential to make climbing training more accessible and systematic. By enabling easy route creation and sharing, the system lowers the barrier for beginner climbers who may otherwise struggle to identify appropriate routes on an unmarked spray wall. Consistent route recall supports reliable progression tracking, which is particularly valuable for climbers in training or recovery programs. The environmental impact of the device is minimal: it uses low-power components, and produces no waste during normal operation.

4.2 Applicable Standards

- **IEC 60825-1:2014**—Safety of laser products. BetaSpray uses Class 2 (<1 mW) lasers, which comply with this standard. Class 2 lasers are safe for accidental short-duration exposure due to the natural blink reflex (≈0.25 s).
- **FDA 21 CFR Part 1040.10**—Performance standards for laser products. All laser modules will be verified to comply before use.
- **IEEE 802.11**—The Wi-Fi communication standard implemented by the ESP32-S3 for the user interface subsystem.
- **IEC 61000-4**—Electromagnetic compatibility (EMC). The PCB layout follows best practices for EMC (solid ground planes, short high-frequency traces, bypass capacitors) which should minimize radiated emissions from PWM switching.

4.3 IEEE/ACM Code of Ethics

We will adhere to the IEEE Code of Ethics [8] throughout development. In accordance with IEEE Code 1.1, we hold paramount the safety and well-being of the public; this is reflected in our selection of Class 2 lasers and the software interlock that disables all lasers when no valid wall target is loaded, preventing uncontrolled emission. We will accurately represent our design capabilities and document limitations transparently in the final report (IEEE Code 3.1). All open-source libraries (ESP-IDF, OpenCV) and third-party hardware designs used in this project will be documented and credited accordingly (IEEE Code 7.8).

4.4 Electrical and Mechanical Safety

- The device is powered from a 5 V USB-C wall adapter; no mains voltage is present inside the enclosure. All exposed connectors are low-voltage SELV circuits.
- All PCB traces carrying >0.5 A are sized per IPC-2221 for the maximum expected current.

- The enclosure has no sharp external edges and includes rubber feet to prevent displacement on gym floors. Since the device operates in an environment where people are physically active, the enclosure must withstand incidental contact without exposing internal electronics.

- Laser apertures in the enclosure are sized to limit the accessible emission angle, further reducing stray beam exposure risk.

- All development, soldering, PCB assembly, and high-current testing will be conducted in designated ECE 445 laboratory areas in compliance with ECEB lab safety standards [9].

References

[1] International Federation of Sport Climbing, "IFSC Rules 2024," *ifsc-climbing.org*. Accessed: Feb. 12, 2026.

[2] Moon Climbing, "MoonBoard," *moonboard.com*. Accessed: Feb. 12, 2026.

[3] International Electrotechnical Commission, "IEC 60825-1: Safety of laser products — Part 1: Equipment classification and requirements," Ed. 3.0, 2014.

[4] `https://documentation.espressif.com/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf`. Accessed: Feb. 23, 2026.

[5] Espressif Systems, "ESP32-S3 Hardware Design Guidelines" `https://docs.espressif.com/projects/esp-hardware-design-guidelines/en/latest/esp32s3/esp-hardware-design-guidelines-en-master-esp32s3.pdf`. Accessed: Feb. 23, 2026.

[6] Texas Instruments, "LM3940 1-A Low-Dropout Regulator for 5-V to 3.3-V Conversion" `https://www.ti.com/lit/ds/symlink/lm3940.pdf?ts=1772126424154`. Accessed: Feb. 20, 2026.

[7] ArduCam, "AArducamOV5640 for Toradex Apalis and Verdin Product Families", `https://docs.toradex.com/115939-toradex-ov5640-datasheet-b0533.pdf`. Accessed: Feb. 23, 2026.

[8] IEEE, "IEEE Code of Ethics," *ieee.org*, `https://www.ieee.org/about/corporate/governance/p7-8.html`. Accessed: Feb. 12, 2026.

[9] ECE 445 Safety Guidelines, "Safety — ECE 445 Senior Design Laboratory," `https://courses.grainger.illinois.edu/ece445/guidelines/safety.asp`. Accessed: Feb. 12, 2026.