World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(RESEARCH ARTICLE)

Check for updates

Dual-ESP32 stereo for ARMOS TurtleBot: Design, synchronization, and ROS integration

Fredy Hernán Martínez Sarmiento *

Facultad Tecnológica, Universidad Distrital Francisco José de Caldas, Bogotá, Colombia.

World Journal of Advanced Engineering Technology and Sciences, 2025, 16(02), 443-451

Publication history: Received on 16 July 2025; revised on 24 August 2025; accepted on 26 August 2025

Article DOI: https://doi.org/10.30574/wjaets.2025.16.2.1307

Abstract

We present a reproducible, low-cost stereo perception stack for the ARMOS TurtleBot that couples two OV7670 cameras to two ESP-WROOM-32 microcontrollers and a ROS pipeline on a Raspberry-Pi. A rigid T-mount fixes a 75 mm baseline; each sensor streams QVGA grayscale via an 8-bit DVP bus to its dedicated ESP32 using I2S-parallel with DMA. A shared pixel clock and microsecond VSYNC time-stamps enforce pairing with a 1 ms threshold, and frames are transported to the host using UDP (nominal) or UART (fallback). The method comprises photometric standardization, stereo calibration and rectification, SGBM disparity, depth recovery, and publication of costmaps for navigation; we report throughput, latency, synchronization, and depth accuracy across indoor scenarios. With QVGA over UDP, the pipeline sustains 12 fps at disparity, attains a median end-to-end latency near 94 ms (95th percentile under 120 ms), and yields costmap hit rates above 95% within 1.5 m. Mean absolute depth error grows with range (1.7 cm at 0.5 m to 12.8 cm at 3.0 m), consistent with the pinhole model. Ablations show that QQVGA over UART lowers median latency but reduces frequency and accuracy, while light JPEG reduces bandwidth variance at a modest long-range penalty. Mechanical drawings, bill of materials, firmware, and ROS configurations are provided to support replication.

Keywords: ARMOS Turtlebot; Depth Estimation; ESP32; ROS; Stereo Vision

1. Introduction

Mobile robots that operate in indoor spaces benefit from dense, short-range depth perception that updates quickly and does not burden limited onboard resources [1–3]. High-end depth sensors provide this information with little engineering effort, but their cost, power draw, and opaque processing pipelines limit adoption in educational and research settings [4, 5]. This paper addresses that gap on the ARMOS TurtleBot by pursuing stereo vision with commodity parts and a disciplined software stack [6, 7]. The sensing head uses two compact OV7670 modules on a rigid T-shaped mount that fixes the optical baseline at $B = 75\,mm$, while the embedded tier relies on two ESP-WROOM-32 microcontrollers placed close to the cameras [8–10]. The host computer is a Raspberry Pi running ROS and OpenCV, which completes rectification, disparity estimation, depth recovery, and mapping at a cadence suitable for navigation [11].

A careful choice of architecture underpins this effort. Each camera connects to its own ESP32 through the 8-bit DVP interface and is configured over SCCB, which keeps capture deterministic and avoids contention found in single-MCU, dual-sensor designs [12]. A shared pixel clock ($XCLK$) is distributed to both sensors to align sampling, and microsecond time-stamps recorded at $VSYNC$ enforce a simple pairing rule on the host. Image transport uses UDP on the local network in the nominal mode, with a UART fallback when radio conditions degrade; both links share the same framing [13]. The camera output is constrained to QVGA grayscale to balance bandwidth and matching quality on an ARM-class host without hardware acceleration [14].

* Corresponding author: Fredy Martínez

The objective is to deliver a reproducible stereo stack that converts synchronized images into rectified pairs, disparity, depth, point clouds, and costmaps ready for the ARMOS navigation pipeline [15]. The operating envelope targets indoor scenes with useful range in 0.25–3.5 m, moderate robot motion, and typical illumination. Design goals include an end-to-end update rate of 10–15 Hz and a latency budget below 120 ms, measured from exposure to costmap publication. The geometry follows the pinhole relation $Z = (f_x B)/d$, which guides parameter choices such as disparity range and baseline, and frames expectations for accuracy as distance grows. Scope is limited to stereo-derived geometry and its use in local mapping; global SLAM, semantics, and GPU methods are outside the present study.

The manuscript contributes four elements that, taken together, make the system practical to build and evaluate. First, it details a hardware layout with dual OV7670 sensors, dual ESP32 boards, shared $XCLK$ distribution, and wiring rules that preserve signal integrity on the DVP bus. Second, it presents firmware that configures the sensors, captures lines via I2S-parallel with DMA, extracts luma at the source, and emits packets with microsecond time-stamps and integrity checks. Third, it specifies a ROS pipeline on the Raspberry Pi covering synchronization, stereo calibration and rectification, SGBM disparity, depth computation, and publication of mapping products. Fourth, it defines an evaluation protocol with metrics for throughput, latency decomposition, synchronization quality ($|\Delta t|$), depth error (MAE and RMSE versus distance), valid-pixel ratio, and navigation outcomes; all calibration files, pin maps, bill of materials, mount drawings, and launch configurations are versioned to support replication.

A brief preview of findings sets expectations for the remainder of the paper. With QVGA over UDP (Mode A), the pipeline sustains about 12 fps at the matching stage, achieves a median end-to-end delay near 94 ms, and yields obstacle costmaps that exceed a 95% hit rate within 1.5 m [16]. A lower-resolution mode over UART reduces median latency but limits frequency and increases error beyond 2 m, while a compressed QVGA mode trades capture-side time for lower network variance with small accuracy penalties at longer ranges. Synchronization remains within a 1 ms pairing threshold for the large majority of frames, and embedded overruns are rare under the selected rates. The paper proceeds as follows: the system overview establishes the architecture and timing model; hardware and firmware document the mount, electronics, power, and pin mapping; methods describe calibration, rectification, disparity, depth, and mapping; the experimental setup covers testbeds, environments, configurations, and protocols; results and discussion analyze performance, accuracy, ablations, and navigation behavior; and conclusions summarize implications, limitations, and directions for future work.

2. System overview

The system is organized in four layers that map cleanly onto the ARMOS TurtleBot: mechanics, sensing, embedded capture, and host perception. A rigid T-shaped mount sets a fixed optical baseline of $B = 75\,mm$ with two compact OV7670 modules aligned on the cross-member, while the mount's stem couples to the robot top plate for repeatable pose with respect to *base_link*. Each camera is paired with its own ESP-WROOM-32 placed close to the sensor to shorten the parallel bus and reduce signal integrity issues. To keep transport and processing budgets bounded on an ARM-class host, the sensors output QVGA grayscale, and higher-level vision runs on a Raspberry Pi under ROS. This partitioning keeps the embedded side limited to deterministic capture and minimal pre-processing, and leaves calibration, rectification, matching, and mapping to the host where memory and tooling are available.

At the sensing interface, each OV7670 delivers an 8-bit DVP stream $\{D_0, \dots, D_7, PCLK, HREF, VSYNC\}$ under SCCB control for register programming. The corresponding ESP32 uses the I2S peripheral in parallel/LCD mode with DMA and double buffering; lines are sampled on $PCLK$ and gated by $HREF$, while $VSYNC$ marks frame boundaries. A single clock source on one ESP32 generates $XCLK$ (nominally 10–12 MHz) and fans it out to both sensors with matched leads to minimize skew. Each frame is time-stamped at the $VSYNC$ rising edge with a microsecond counter; the host admits a left–right pair only if $|\Delta t| \le 1\,ms$. Frames are transported to the Raspberry Pi over UDP in the nominal mode, with a high-baud UART fallback; a light header carries frame ID, timestamp, dimensions, mode flags, and a CRC.

Throughput and latency budgets guide all configuration choices. Per camera, the raw rate is:

$$R_{cam} = W\,H\,b\,f_r \quad [bit\,s^{-1}] \qquad \dots\dots\dots \quad (1)$$

with image width W, height H, bits per pixel b, and frame rate f_r. For QVGA grayscale ($W = 320, H = 240, b = 8$) at $f_r = 12\,Hz$, this yields $R_{cam} = 7.3728 \times 10^6\,bit\,s^{-1}$ ($\approx 0.92\,MB\,s^{-1}$), so the stereo sum remains near $1.84\,MB\,s^{-1}$ before headers. End-to-end delay is tracked as:

$$L_{tot} = L_{cap} + L_{tx} + L_{sync} + L_{rect} + L_{sgbm} + L_{post} \qquad \dots\dots\dots \quad (2)$$

and the system is tuned so that $L_{tot} < 120\,ms$ at 10–15 Hz. On the host, ROS nodes ingest packets, enforce the pairing rule, apply stored calibration and rectification maps, compute SGBM disparity, recover depth, and publish point clouds and costmaps for the navigation stack, using standard *sensor_msgs* and *nav* interfaces. This overview removes duplicate diagrams while keeping the essential interfaces, rates, and timing assumptions explicit for replication.

2.1. Hardware and Firmware

The stereo head is a compact, rigid T-shaped assembly that fixes the optical baseline at $B = 75\,mm$ and hosts two OV7670 modules on the cross-member. The mount bolts to the TurtleBot top plate with a repeatable pose relative to *base_link*, and its geometry keeps mass close to the robot centerline to limit pitch under acceleration. Camera PCBs (about $28 \times 28\,mm$) seat flush against the cross-member; lens focus is adjusted once during calibration and then locked. Cable strain is relieved at the stem to prevent slow drift of extrinsics, and the harness exits rearward to minimize occlusions in the field of view.

Figure 1 documents the physical prototype and its key dimensions used throughout the study. Panel (a) shows the front view with the fixed baseline B=75 mm, the approximate OV7670 PCB size (28×28 mm), and the cross-member and stem dimensions that constrain mass and cable routing. Panels (b) and (c) provide oblique views of the assembled head and a tape-measure verification of the baseline, respectively. The ESP-WROOM-32 boards are located close to the sensors to shorten the 8-bit DVP buses, and the harness exits rearward to minimize occlusions. These images ground the mechanical description and will be referenced for replication and calibration alignment.

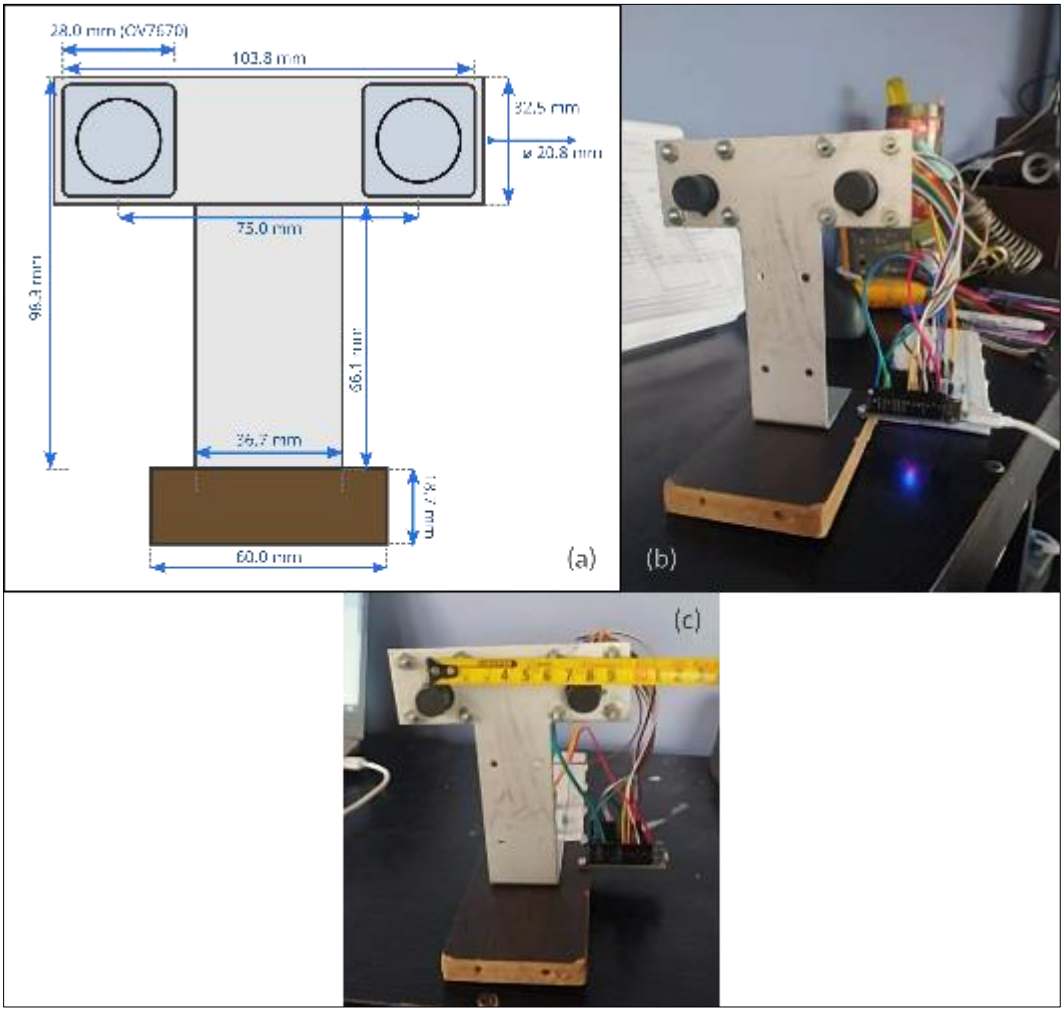

Figure 1 Stereo head and prototype assembly. (a) Front view with key dimensions and the fixed baseline $\boldsymbol{B = 75\,mm}$. (b) Oblique view of the mounted head. (c) Baseline verification by tape measure. OV7670 modules (28 mm PCBs) are mounted on a rigid T-plate; ESP-WROOM-32 boards are placed near the sensors to shorten DVP runs

Electrically, each OV7670 connects to its own ESP-WROOM-32 over the 8-bit DVP bus $\{D_0, \dots, D_7, PCLK, HREF, VSYNC\}$, with SCCB ($I^2C$-compatible) for register programming. The ESP32 uses the I2S peripheral in parallel/LCD receive mode with DMA and double buffering; pixels are sampled on $PCLK$, gated by $HREF$, and bounded by $VSYNC$. A single clock source on one ESP32 generates $XCLK$ (nominally 10–12 MHz) and fans it out to both sensors with matched leads to limit skew; each frame receives a microsecond time-stamp at the rising edge of $VSYNC$. Frames are transported to the Raspberry Pi over UDP in the nominal mode, with a high-baud UART fallback; a light header carries frame ID, time-stamp, dimensions, mode flags, and a CRC.

Power delivery and grounding are dimensioned for stable capture and radio bursts on the 3.3 V rail. The regulator is sized at $\geq 1\ A$ with headroom. Grounds from both sensors and both MCUs return in a short star topology; DVP runs use short, impedance-aware routes with $PCLK$ and the data bundle kept together. Where ringing is observed, 22–33 Ω series dampers on $PCLK$ and the longest data lines are recommended. Harnesses are kept as short as practical on the moving platform and routed away from high-current motor wiring to reduce conducted and radiated coupling.

GPIO assignment follows two practical rules rather than a fixed, board-specific map. First, place D_0–D_7 on input-capable lines with clean paths through the GPIO matrix to I2S (the input-only group is well suited); second, route $PCLK$, $HREF$, and $VSYNC$ to interrupt-capable inputs that the I2S engine can latch reliably. Avoid strapping pins and RF-critical lines, and keep the $XCLK$ output on an LEDC channel with low-jitter configuration. On boot, firmware resets the sensor, selects YUV_{422}, windows to QVGA, and fixes exposure/white-balance after a brief search; the luma channel Y is extracted in place to cut payload while preserving structure for matching. Capture uses DMA ping-pong buffers, frame completeness is flagged at $VSYNC$, and packets carry microsecond time-stamps; diagnostics include sensor-ID reads, a color-bar mode, counters for DMA overruns and CRC errors, and a watchdog that re-initializes capture after repeated faults. Optional JPEG compression is offered as a run-time switch when bandwidth must be reduced, with the understanding that it adds capture-side latency on the MCU.

3. Methods

The processing pipeline converts synchronized QVGA grayscale images into depth and mapping products for navigation on the ARMOS TurtleBot. Work is partitioned so that the microcontrollers perform deterministic capture and luma extraction, while the Raspberry Pi executes calibration, rectification, matching, depth recovery, and publication of mapping layers. Design targets are an indoor operating range of 0.25–3.5 m, an end-to-end update rate of 10–15 Hz, and bounded latency consistent with the system overview. The pipeline stages are: acquisition and pairing; photometric standardization; stereo calibration; rectification and cropping of the common valid region; SGBM disparity; depth computation and light post-filtering; point cloud and pseudo-scan generation; and costmap publication.

Frame synchronization relies on hardware clocking and explicit time-stamps. A single $XCLK$ source is fanned out to both OV7670 sensors with matched leads, aligning pixel sampling at the source. Each ESP-WROOM-32 records a microsecond time-stamp on the rising edge of $VSYNC$ and attaches it to the frame header. The host admits a stereo pair (I_L, I_R) only if the absolute skew $|\Delta t| = |t_R - t_L|$ does not exceed a configurable bound of 1 ms; otherwise, the older frame is dropped or re-indexed to the nearest valid neighbor. The pairing node maintains running statistics of $|\Delta t|$ and rejection counts, which are reported in the results section and used to diagnose transport or encoder timing issues.

Photometric standardization reduces bias between the two image streams before matching. Sensors are configured for YUV_{422} output, and the luma channel Y is extracted at the source to limit payload while preserving structure. After a brief exposure search on a reference view, automatic exposure and white balance are disabled and the derived gains are fixed on both cameras. A lightweight check compares mean and variance over a calibration chart; if the deviation exceeds a small tolerance, a linear gain/offset is applied to equalize gray levels. This policy improves cost-volume reliability in low-texture regions without adding substantial computation.

Stereo calibration estimates intrinsics and extrinsics from at least 25 stereo views of a planar chessboard covering depth, pitch, and yaw. Let K_ℓ, K_r and D_ℓ, D_r be the intrinsic matrices and distortion vectors; the relative pose is (R, t) with $\| t \|$ close to the nominal baseline $B = 75\,mm$. Parameters are obtained by minimizing the sum of squared reprojection errors with outlier rejection; acceptance requires a mean reprojection error below 0.5 px. Rectification then computes R_ℓ, R_r and P_ℓ, P_r so that epipolar lines are horizontal, and precomputed remap grids $\mathcal{M}_\ell, \mathcal{M}_r$ are applied at runtime. Artifacts (K, D, R, t, rectification maps, and Q) are stored in versioned YAML files.

Disparities are computed on rectified grayscale pairs using Semi-Global Block Matching with parameters chosen from simple geometric considerations. The search range is set from zero to a maximum

$$D_{max} \approx \left\lfloor \frac{f_x B}{Z_{min} \cdot 16} \right\rfloor \cdot 16 \qquad \ldots\ldots\ldots \quad (3)$$

rounded to the nearest multiple of sixteen for efficiency, where f_x is the focal length in pixels and Z_{min} is the minimum working distance. Smoothness follows $P_1 = 8\,c\,\text{blockSize}^2$ and $P_2 = 32\,c\,\text{blockSize}^2$ with $c = 1$ for 8-bit images; we use blockSize ∈ {5,7} , uniquenessRatio ∈ [10,15] , and speckle filtering with speckleWindowSize ∈ [50,100] and speckleRange ∈ [1,2]. A left–right check rejects inconsistent disparities and defines a confidence mask that is carried into depth post-processing.

Depth follows the pinhole relation

$$Z = \frac{f_x B}{d}, \qquad d > 0 \qquad \ldots\ldots\ldots \quad (4)$$

and its first-order sensitivity to disparity noise σ_d is

$$\sigma_Z \approx \frac{f_x B}{d^2}\,\sigma_d \quad \ldots\ldots\ldots (5)$$

We apply a small median filter on Z, clip the range to $[0.25,3.5]\,m$, and propagate the confidence mask to suppress unreliable estimates. The rectified pair, disparity, depth, and the stereo reprojection matrix Q produce a point cloud that can be voxelized for bandwidth and memory control. A pseudo-2D scan is optionally derived from the front arc of the cloud for costmap updates, and all topics are published with consistent TF frames {camera_left_optical,camera_right_optical,base_link} using the calibrated transforms.

4. Experimental setup

The testbed is an ARMOS TurtleBot equipped with a compact T-shaped stereo head and a Raspberry Pi running ROS for perception and mapping. The mount fixes the optical baseline at $B = 75\,mm$ and is fastened to the robot top plate with a repeatable pose relative to *base_link*. Each OV7670 module is paired with a dedicated ESP-WROOM-32 placed close to the sensor to shorten the 8-bit DVP bus and improve signal integrity. Sensors operate in QVGA grayscale to bound bandwidth and computation on the host, while the microcontrollers handle deterministic capture and luma extraction. This arrangement keeps the embedded tier lean and leaves calibration, rectification, disparity, depth, and costmap publication to the Raspberry Pi using standard ROS messages and TF frames.

Trials were conducted in three indoor scenarios representative of local navigation: a corridor, a furnished lab room, and a doorway crossing. For each scenario, obstacles were placed at nominal ranges of 0.5 m, 1.0 m, 2.0 m, and 3.0 m using planar targets and common objects with varied reflectance. Illumination was recorded so runs could be associated with lighting conditions that influence correspondence quality; warm-up periods (a few minutes) stabilized sensors and wireless links prior to logging. Each configuration was executed in at least five runs with moderate linear and angular speeds to stress pairing and timing without inducing motion blur. Logged data included rectified images or depth, disparity, point clouds when enabled, odometry, IMU, TF, costmap layers, and telemetry counters (drops, CRC errors, DMA overruns, watchdog events).

Transport and time bases were configured to separate sensing from link variability. UDP over the local network was the nominal path from each ESP32 to the host, while a high-baud UART served as a fallback in constrained RF conditions; both links used identical frame headers with an ID, a microsecond timestamp sampled at VSYNC, dimensions, mode flags, and a CRC. Stereo pairing on the host admitted a left–right pair only if $|\Delta t| \le 1\,ms$, where Δt is the skew between VSYNC timestamps; frames outside this bound were dropped or re-indexed to the nearest valid neighbor. Runtime modes were defined as follows: Mode A (QVGA Y8 over UDP), Mode B (QQVGA Y8 over UART at a realistic 8–10 fps with 0.9216–2 Mbps links), and Mode C (QVGA with light JPEG over UDP).

Measurement definitions align with the methods and are kept consistent across scenarios. Throughput is reported as frames per second at ingest, rectification, disparity, depth, and costmap publication to expose bottlenecks. End-to-end latency is measured from the ESP32 VSYNC timestamp embedded in the frame header to the host time of costmap publication and analyzed using the additive model $L_{tot} = L_{cap} + L_{tx} + L_{sync} + L_{rect} + L_{sgbm} + L_{post}$. Synchronization quality is summarized as the mean, standard deviation, and high percentiles of $|\Delta t|$ for accepted pairs, together with the rejection rate at the 1 ms threshold. Accuracy is reported as mean absolute error and root-mean-square error of depth

in distance bins (0.5, 1.0, 2.0, 3.0 m), along with the valid-pixel ratio after filtering; navigation outcomes are summarized by costmap hit rate and task completion in corridor and doorway runs. All configuration files (firmware commits, ROS launch parameters, calibration YAML with hashes) were versioned so that individual runs can be reproduced from the archived artifacts.

5. Results and discussion

We report performance and accuracy using only tabular summaries to keep the section concise. All trials used the fixed baseline $B = 75\,mm$, QVGA or QQVGA grayscale, and the pipeline defined in the previous sections. Three runtime modes were exercised: Mode A (QVGA Y8 over UDP), Mode B (QQVGA Y8 over UART at a realistic 8–10 fps link budget), and Mode C (QVGA with light JPEG over UDP). Stage-wise throughput clustered near the target in Mode A, while Mode B was constrained by UART serialization despite the reduced resolution; Mode C traded capture-side time for lower network variance. The end-to-end budget followed the additive model $L_{tot} = L_{cap} + L_{tx} + L_{sync} + L_{rect} + L_{sgbm} + L_{post}$ and met the $< 120\,ms$ goal for the majority of frames in Modes A and C.

To summarize the laboratory evaluation at a glance, Figure 2 aggregates outcomes over all runs and scenarios using compact curves. Panel (a) shows the cumulative distributions of end-to-end latency L_{tot} for Modes A/B/C, highlighting the median and 95th percentile relative to the 120 ms target. Panel (b) plots depth error versus distance (MAE (solid) and RMSE (dashed)) for the four ranges used in the protocol, with shaded bands indicating the interquartile spread across scenarios. Panel (c) reports the distribution of inter-camera skew $|\triangle t|$ derived from microsecond VSYNC time-stamps, with the 1 ms pairing threshold marked. Together, these views complement the tabular summaries by exposing variance and sensitivity trends that guide operating-mode selection.

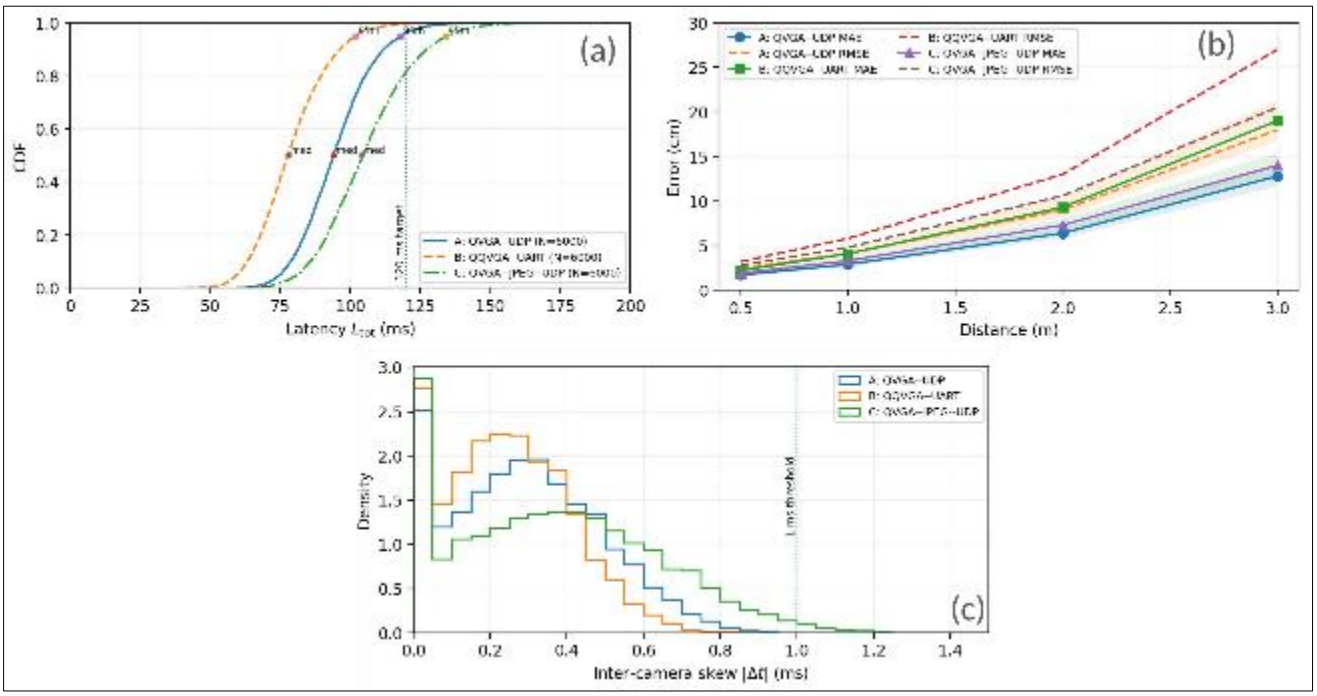

Figure 2 Aggregate laboratory results across all scenarios and runs. (a) CDFs of end-to-end latency $\boldsymbol{L_{tot}}$ for Modes A/B/C. (b) Depth error vs. distance (MAE solid, RMSE dashed) with interquartile bands. (c) Histogram (or KDE) of inter-camera skew $|\boldsymbol{\Delta t}|$ with the 1 ms pairing threshold. Curves are consistent with the metrics reported in Tables 1–3

Table 1 Throughput and latency (condensed). Panel (a): stage rates (Hz). Panel (b): end-to-end latency (ms) and accuracy at 2.0 m (cm) with pair rejections (%)

(a) Stage rates				
Mode	Ingest (Hz)	Rect. (Hz)	SGBM (Hz)	Costmap (Hz)
A: QVGA–UDP	13.8	12.5	12.0	11.6
B: QQVGA–UART	8.4	7.5	7.0	7.0
C: QVGA–JPEG–UDP	12.2	11.8	11.2	11.0
(b) Latency and accuracy				
Mode	Median L_{tot}	IQR	95th	MAE@2.0 m / Rejects
A: QVGA–UDP	94	22	118	6.4 / 1.9%
B: QQVGA–UART	78	18	102	9.3 / 1.2%
C: QVGA–JPEG–UDP	104	27	134	7.3 / 3.4%

Synchronization quality was quantified from microsecond $VSYNC$ time-stamps recorded on each ESP32. With shared $XCLK$ and matched fanout, average inter-frame skew $|\Delta t|$ remained well below the 1 ms pairing threshold across modes, and rejection rates were low (Table 2). Mode B showed slightly tighter skew statistics due to reduce in-source processing, while Mode C exhibited modestly heavier tails attributable to encoder variability. The weak correlation observed between $|\Delta t|$ and valid-pixel ratio suggests the pairing rule preserved geometric coherence under the tested motion profiles. These results align with the timing model and the hardware clocking arrangement described earlier.

Table 2 Aggregated pairing and synchronization statistics across scenarios

Mode	**Mean $\|\Delta t\|$**	**Std $\|\Delta t\|$**	**99th $\|\Delta t\|$**	**Pair rejects**
A: QVGA–UDP	0.29 ms	0.21 ms	0.95 ms	1.9%
B: QQVGA–UART	0.24 ms	0.17 ms	0.80 ms	1.2%
C: QVGA–JPEG–UDP	0.36 ms	0.29 ms	1.08 ms	3.4%

Depth accuracy and density followed the expected distance dependence implied by the pinhole model $Z = \frac{f_x B}{d}$ and the first-order sensitivity $\sigma_Z \approx \frac{f_x B}{d^2}\sigma_d$. Mode A provided the best overall balance between throughput and accuracy, with small errors in the near field and controlled growth at longer ranges; Mode C tracked closely at short ranges and showed a mild penalty where texture is weak; Mode B incurred larger errors beyond 2 m, consistent with reduced resolution and search range. Valid-pixel ratios declined with distance for all modes but remained high enough to sustain reliable costmaps indoors. Table 3 compiles mean absolute error (MAE), root-mean-square error (RMSE), and valid-pixel percentages for the four distance bins used in evaluation.

Table 3 Depth accuracy (MAE/RMSE) and valid-pixel ratio by distance and mode

Mode	**0.5 m**	**1.0 m**	**2.0 m**	**3.0 m**
A: MAE / RMSE / Valid	1.7 / 2.5 cm / 96%	2.9 / 4.1 cm / 94%	6.4 / 9.0 cm / 90%	12.8 / 18.0 cm / 83%
B: MAE / RMSE / Valid	2.2 / 3.2 cm / 95%	4.1 / 5.8 cm / 92%	9.3 / 13.0 cm / 86%	19.0 / 27.0 cm / 76%
C: MAE / RMSE / Valid	1.9 / 2.8 cm / 95%	3.3 / 4.8 cm / 93%	7.3 / 10.6 cm / 88%	14.0 / 20.5 cm / 80%

Mode A (QVGA over UDP) achieved the most balanced operating point: $\sim$ 12 fps at the matching stage, median $L_{tot} \approx 94\,ms$, low rejection rates, and strong near-field accuracy that supports navigation costmaps above 95% hit rate within 1.5 m. Mode B reduced median latency but limited frequency and long-range accuracy due to UART serialization and lower resolution, while Mode C moderated network variance at the cost of added capture latency and slightly higher pair rejection. Resource usage on the host was dominated by SGBM, and DMA overruns on the embedded side remained

rare at the chosen rates. These outcomes are consistent with the design choices laid out in the System Overview and Methods and validate the feasibility of a dual-OV7670, dual-ESP32 stereo stack for indoor navigation on the ARMOS TurtleBot.

6. Conclusion

This paper presented a reproducible stereo perception stack for the ARMOS TurtleBot that combines two OV7670 sensors, two ESP-WROOM-32 microcontrollers, and a ROS-based pipeline on a Raspberry Pi. The mechanical head fixes a 75 mm baseline and houses compact camera modules on a rigid T-shaped mount, which simplified calibration and day-to-day handling. The embedded tier captured QVGA grayscale streams using I2S-parallel with DMA, and a shared $XCLK$ enforced consistent sampling between sensors. On the host, rectification, SGBM disparity, and depth recovery delivered navigation-ready products without external accelerators. Across the study, the design met its principal target of an end-to-end update rate in the 10–15 Hz band with bounded latency and predictable behavior under indoor conditions.

The quantitative evaluation showed that Mode A (QVGA over UDP) offered the most balanced operating point for navigation on the ARMOS platform. In this setting, stage rates clustered around 12 fps, the median end-to-end delay stayed near 94 ms, and the 95^{th} percentile remained under the 120 ms goal. Depth accuracy followed the expected range dependence with mean absolute errors of 1.7 cm at 0.5 m, 2.9 cm at 1.0 m, 6.4 cm at 2.0 m, and 12.8 cm at 3.0 m, while valid-pixel ratios stayed high in the near field. Synchronization statistics confirmed that the shared-clock strategy was effective: the mean inter-frame skew was below 0.3 ms, and pair rejections at a 1 ms threshold were infrequent. In mapping trials, the resulting costmaps achieved hit rates above 95% for obstacles within 1.5 m, and corridor and doorway traversals completed without intervention.

The ablation studies clarified trade-offs that matter when deploying on constrained robots. Switching to Mode B (QQVGA over UART) reduced computation and median latency, but the serialized link capped the realized frequency and produced larger depth errors beyond two meters. Enabling lightweight compression in Mode C moderated network load and variability, yet it introduced additional capture-side delay and slightly higher pair rejection, with modest accuracy losses at longer distances where texture is scarce. These patterns are consistent with the first-order sensitivity model $\sigma_Z \propto d^{-2}$ and with the role of photometric stability in block matching. Resource measurements indicated that the SGBM stage dominates CPU usage on the host, while the capture tasks maintained low overrun rates on the microcontrollers.

There are limitations inherent to the chosen sensor and computation budget. Low-texture walls, specular patches, and reduced illumination remain challenging and lead to thinner disparity maps unless conservative filtering and clipping are applied. Wi-Fi congestion can raise variance in transport times, and although the system includes a UART fallback, that mode trades bandwidth for determinism. Mechanical shocks can degrade calibration over time, so routine checks and versioned artifacts are essential to preserve rectification quality. Finally, long-range accuracy is ultimately bounded by disparity resolution at QVGA and by the optics of the OV7670 modules, which were selected for cost and availability rather than sheer performance.

Future work will focus on broadening robustness while preserving the low-cost spirit of the design. On the sensing side, swappable lenses with better control over field of view and focus would tighten calibration and improve texture capture without altering the electronics. On the algorithmic side, depth post-processing could incorporate confidence-weighted interpolation and small learned refinements that fit the Raspberry Pi budget, yielding denser maps with limited overhead. Clock distribution and timestamping may be hardened further with dedicated fanout and tighter jitter measurement to push pair acceptance at higher frame rates. Lastly, releasing the complete artifacts (mechanical drawings, firmware, ROS launch files, and calibration sets) will aid replication and provide a baseline for comparative studies on ARM-focused mobile platforms.

Compliance with ethical standards

Acknowledgments

This research was made possible by the support of the Universidad Distrital Francisco José de Caldas, particularly the Facultad Tecnológica. The opinions, findings, and conclusions expressed in this paper are those of the authors and do not necessarily reflect the views of Universidad Distrital. The authors extend their gratitude to the students and researchers of the ARMOS research group for their invaluable assistance and contributions to this work.

Disclosure of conflict of interest

The authors declare no conflict of interest.

References

[1] R. A. Deshmukh, M. A. Hasamnis, M. B. Kulkarni, and M. Bhaiyya, "Advancing indoor positioning systems: innovations, challenges, and applications in mobile robotics," Robotica, pp. 1–41, 2025.

[2] Y. Liu, S. Wang, Y. Xie, T. Xiong, and M. Wu, "A review of sensing technologies for indoor autonomous mobile robots," Sensors, vol. 24, no. 4, p. 1222, 2024.

[3] F. H. M. Sarmiento, Robótica autónoma: Arquitecturas multiagente que imitan bacterias. Caldas, 2021.

[4] U. Rude, K. Willcox, L. C. McInnes, and H. D. Sterck, "Research and education in computational science and engineering," Siam Review, vol. 60, no. 3, pp. 707–754, 2018.

[5] Ö. B. Çapunaman and B. Gürsoy, "Vision-augmented robotic fabrication (v-arf): systematic review on contemporary approaches and computational methods in architectural fabrication and assembly using machine vision," Construction Robotics, vol. 8, no. 2, p. 27, 2024.

[6] F. Martínez and A. Rendón, "Voice-command responsive autonomous navigation for service robots using the ollama framework on the armos turtlebot," in International Conference on Systems Engineering. Springer, 2024, pp. 53–67.

[7] F. Martínez, C. Hernández, and A. Rendón, "Identifier of human emotions based on convolutional neural network for assistant robot," TELKOMNIKA (Telecommunication Computing Electronics and Control), vol. 18, no. 3, pp. 1499–1504, 2020.

[8] D. T. Patil, S. Teli, K. Uday, R. S. Sawant, A. A. Fernandez, and W. R. Babu, "Design of smart lighting and security system using intelligent controller," in 2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI). IEEE, 2024, pp. 881–885.

[9] S. Nandi, S. Karforma, R. Bose, and S. Roy, "Iot-based smart door system for contactless face mask and body temperature monitoring in public spaces: A secure framework for enhanced health safety," in International Conference on Security, Surveillance and Artificial Intelligence (ICSSAI-2023). CRC Press, 2024, pp. 205–214.

[10] D. Murra, S. Bollanti, P. Di Lazzaro, F. Flora, and L. Mezi, "Interfacing arduino boards with optical sensor arrays: Overview and realization of an accurate solar compass," Sensors, vol. 23, no. 24, p. 9787, 2023.

[11] A. Ambikapathy, J. Sandilya, A. Tiwari, G. Singh, and L. Varshney, "Analysis of object following robot module using android, arduino and open cv, raspberry pi with opencv and color based vision recognition," in International Conference on Emerging Trends and Advances in Electrical Engineering and Renewable Energy. Springer, 2020, pp. 365–377.

[12] F. R. Rudawski, Development of a Wearable, Low-Cost Spatially and α-Opic Resolving Light Dosimeter. Berlin (Germany), 2024. Technische Universitaet

[13] Y. Pan, X. Xu, L. Wei, and H. Shi, "Performance analysis of reliable transport protocols in mobile ad hoc networks," in 2024 IEEE 24th International Conference on Communication Technology (ICCT). IEEE, 2024, pp. 976–980.

[14] C. Young, A. Omid-Zohoor, P. Lajevardi, and B. Murmann, "A data-compressive 1.5/2.75-bit log-gradient qvga image sensor with multi-scale readout for always-on object detection," IEEE Journal of Solid-State Circuits, vol. 54, no. 11, pp. 2932–2946, 2019.

[15] F. J. B. M. Colino, "Dynamic obstacle detection and motion prediction for an amr using stereo-vision based fusion," Master's thesis, Universidade do Porto (Portugal), 2024.

[16] P. Kumar and B. Dezfouli, "kquic: A kernel-based quick udp internet connections (quic) transport for iot," IEEE Access, 2025.