International Journal of Advanced Scientific Research in Engineering Trends and Technology

Volume 1 Issue 5, Nov 2026
www.ijengt.com

A Calibration-Aware Association Framework for YOLO Detections and Ultrasonic Scan Events in Embedded Monitoring Systems

Swathi HY, Baladev DR, Animesh M, Manukrishna, and Samyak
Department of Artificial Intelligence and Machine Learning
Malnad College of Engineering, Hassan, India
Email: hys@mcehassan.ac.in

***Abstract*—**A webcam and an ultrasonic scanner can provide two useful views of the same area: the camera shows what an object looks like, while the ultrasonic sensor estimates how far away a reflecting surface is. However, the two readings are collected independently, so they cannot automatically be treated as measurements of the same object. This paper presents a calibration-aware approach for matching YOLO detections from a webcam with measurements from a servo-mounted HC-SR04 sensor. The approach uses both image position and time. The camera is calibrated first, including its position in relation to the ultrasonic scanner. Each valid range measurement is then represented as a point using the measured distance and servo angle. After the point is transformed into the camera coordinate system, its expected image location can be compared with the bounding boxes produced by YOLO. A visual detection is considered as a possible match only when it falls within an allowed image region and comes from a frame captured close to the scan event. A combined score ranks the candidates. When the evidence is weak or more than one candidate is plausible, the system records the event as unmatched or ambiguous instead of forcing a match. The paper describes the required calibration, event data, matching procedure, error handling, and a test plan for a system built around an Arduino UNO, HC-SR04 sensor, servo motor, webcam, Python, OpenCV, and YOLO. The method is proposed for further implementation and testing. Since no calibrated measurements or labelled test dataset were provided, this paper does not report association accuracy or timing results.

***Index Terms*—**Arduino, camera calibration, embedded vision, object detection, sensor association, temporal synchronization, ultrasonic ranging, YOLO.

I. INTRODUCTION

Many low-cost monitoring systems use more than one sensor because each device provides a different kind of information. A webcam captures the appearance of objects, whereas an ultrasonic sensor estimates distance from the time taken by a sound pulse to travel to a surface and return. When a webcam is used with a servo-mounted HC-SR04, the application can show visual object labels alongside measurements taken at different angles. For these two outputs to describe the same object, however, the software must establish a reliable link between them.

Simply running the camera and ultrasonic scanner at the same time does not establish that link. The webcam captures frames continuously, but the ultrasonic sensor takes a reading only after the servo reaches a position and the sensor receives an echo. The devices may also be mounted at different locations and may have different fields of view and response delays. As a result, choosing the latest camera detection and attaching it to the latest range value can produce an incorrect object-distance pair.

This paper examines how those two data streams can be matched reliably. Rather than describing the complete radar-style prototype again, it focuses on a calibration and matching procedure that uses image geometry as well as timestamps. The basic idea is to use a valid ultrasonic reading to estimate where the reflecting point should appear in the camera image. Detections from a frame captured near the time of that reading can then be checked against the estimated location.

The proposed method is designed as an extension to the existing Arduino-based setup. The Arduino remains responsible for moving the servo and measuring ultrasonic distance. The host computer handles camera calibration, YOLO inference, coordinate conversion, matching, and record storage. Keeping these tasks on separate devices allows the Arduino to concentrate on hardware control while the computer performs the more demanding vision processing.

The paper has four main elements. It defines a common event format for camera and ultrasonic observations, describes how calibration can project a scan point into the camera image, and combines spatial and timing checks to decide whether two observations should be linked. It also outlines a repeatable test procedure and suitable measures for evaluating matching quality, timing, and range measurements. The work is presented as a proposed method; experimental validation is still required.

II. BACKGROUND AND RELATED WORK

A. Camera Calibration and Geometric Projection

A camera converts points in the three-dimensional scene into positions on a two-dimensional image. Its intrinsic parameters include the focal lengths and the principal point, while its extrinsic parameters describe its position and orientation relative to another coordinate frame. Zhang's planar calibration method estimates camera parameters from images of a calibration pattern taken at different orientations [1]. OpenCV includes functions that can be used to calculate and apply these parameters [2].

B. Temporal Alignment of Sensor Measurements

Sensor readings are not always produced at the same rate. A camera may capture several frames while another sensor produces only one measurement, and the devices may also use clocks that are not perfectly aligned. Kelly and Sukhatme discuss the importance of estimating timing differences between sensor streams, noting that spatial calibration alone does not account for delays [3]. Approximate-time synchronization methods used in robotics provide another way to pair measurements whose timestamps are close, even when they are not identical [4]. These principles form the basis of the time check proposed in this paper.

C. Object Detection and Ultrasonic Ranging

YOLO detects objects in an image and returns information such as class labels, confidence scores, and bounding boxes [5]. This tells the system what may be visible and where it appears in the frame, but it does not directly provide physical distance or indicate whether an object is dangerous. The HC-SR04 provides a different measurement: it estimates the distance to a reflecting surface from the round-trip travel time of an ultrasonic pulse [6]. It cannot identify the object's class. The two devices therefore provide complementary information, but an additional matching step is needed before their outputs can be presented as measurements of one object.

D. Research Gap

The existing radar-style prototype brings webcam-based recognition and ultrasonic scanning into one application. Its

 ©IJENGT: All Rights are Reserved

design also identifies an important limitation: the camera label and ultrasonic range should be treated as separate observations until their geometry and timing have been checked. This paper concentrates on that issue. It describes how an ultrasonic return can be projected into the camera image and how the software can classify a possible match as accepted, rejected, or uncertain. The proposed work does not change the YOLO detector or the HC-SR04 ranging method; it focuses on deciding when their outputs can reasonably be linked.

III. PROBLEM DEFINITION AND SYSTEM MODEL

Each ultrasonic reading is stored as an event $U_k = (\theta_k, d_k, t^u_k, q_k)$. Here, θ_k is the servo angle, d_k is the measured distance, t^u_k is the time of the reading, and q_k indicates whether the echo measurement is valid. A camera event is written as $C_j = (I_j, t^c_j, D_j)$, where I_j is the captured frame, t^c_j is its capture time, and D_j contains the YOLO detections found in that frame. Each detection includes a class label, a confidence score, and a bounding box.

The task is to decide whether a detection in D_j could correspond to the ultrasonic event U_k. Neither a matching class label nor a close timestamp is enough on its own. A match is accepted only when the ultrasonic reading is valid, the detection meets the selected confidence threshold, the frame falls within the permitted time window, and the projected scan point is consistent with the candidate bounding box.

For every scan event, the method returns one of three outcomes: associated, unmatched, or ambiguous. Keeping these outcomes separate is important. Automatically choosing the closest detection may attach a distance to the wrong object. Allowing the system to leave a reading unmatched or ambiguous preserves uncertainty and makes the event available for later review.

IV. PROPOSED CALIBRATION-AWARE ASSOCIATION METHOD

A. Hardware and Software Arrangement

The proposed setup uses the same core hardware: an Arduino UNO, a servo motor, and an HC-SR04 ultrasonic sensor. A laptop webcam supplies the video frames. The Arduino sends the servo angle and measured distance to the host computer, together with a sequence number and, where possible, a timestamp from the microcontroller. Python reads the serial data, obtains frames through OpenCV, and runs the selected YOLO model. The association module then uses the calibration information to compare the camera and sensor observations. Processing can continue to display the radar-style scan, while a CSV file keeps a record of each event and its matching result.

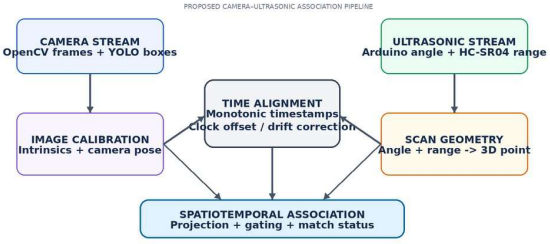


Fig. 1. Proposed processing flow for matching camera detections with ultrasonic scan events using calibration and timestamps.

B. Camera and Scanner Calibration

The system should be calibrated before the matching procedure is used. First, a planar calibration pattern is shown to the webcam from several orientations. These images are used to estimate the camera's intrinsic matrix K and lens-distortion coefficients. The position and orientation of the camera relative to the ultrasonic scanner must then be estimated using a shared calibration setup or known reference points. This relationship is represented by a rotation matrix R_{cs} and a translation vector t_{cs}. Together, they describe how a point in the scanner coordinate frame is expressed in the camera coordinate frame.

For a scan angle θ_k and a valid range d_k, the reflecting point is approximated in the scanner's horizontal scan plane by the coordinate vector:

$$P_{s,k} = [d_k \sin(\theta_k), 0, d_k \cos(\theta_k)]^T \quad (1)$$

The chosen coordinate convention defines the scanner's forward direction as the positive z-axis and the horizontal direction as the x-axis. The point is transformed into the camera frame using:

$$P_{c,k} = R_{cs} P_{s,k} + t_{cs} \quad (2)$$

If the point lies in front of the camera, the intrinsic matrix projects it to an image location. With $P_{c,k} = [X_c, Y_c, Z_c]^T$, the ideal pixel coordinate is obtained from:

$$s [u_k, v_k, 1]^T = K P_{c,k} \quad (3)$$

C. Timestamp Alignment

The system should compare measurements using the times at which they were captured, not the order in which the software finishes processing them. For this reason, the Arduino message should include a sequence number and a millisecond timestamp. The host computer records a monotonic timestamp whenever it captures a webcam frame. Several synchronization exchanges can be used to estimate the offset and relative clock-rate difference between the Arduino and host clocks. A simple affine conversion is:

$$\hat{t}^u_k = a\, t^u_k + b \quad (4)$$

where a represents relative clock-rate adjustment and b represents the offset. The adjusted scan time $\hat{t}^u_k$ is compared with the camera frame time t^c_j. The temporal difference is:

$$\Delta t_{kj} = |t^c_j - \hat{t}^u_k| \quad (5)$$

A frame is considered for matching only when its time difference from the adjusted scan timestamp is within the selected window τ_t. If the Arduino firmware cannot be changed to send its own timestamp, the host can temporarily record the time at which each serial message arrives. This is less reliable because transmission and buffering add delay. The timing variation should therefore be measured and reported; the message-receipt time should not be treated as the exact moment when the ultrasonic reading was taken.

D. Spatial Gating and Candidate Selection

For each frame that passes the time check, YOLO returns a group of bounding boxes. For detection i, let (x_i, y_i) represent the centre of the box, w_i and h_i its width and height, c_i its class label, and p_i its confidence score. The projected ultrasonic point (u_k, v_k) is compared with each box. One way to express the distance between the point and the box centre is the normalized image-space residual:

$$e_{ik} = \sqrt{[((u_k - x_i)/(w_i/2))^2 + ((v_k - y_i)/(h_i/2))^2]} \quad (6)$$

A projected point that falls inside a box, or close to it, produces a smaller residual. The candidate is kept only when its confidence is at least τ_d and its spatial residual does not exceed τ_p. Here, τ_d is the detector-confidence threshold and τ_p is the permitted spatial difference. Both values should be selected through testing and recorded so that the evaluation can be repeated.

For the remaining candidates, a normalized cost combines the temporal and spatial residuals:

$$J_{ikj} = \lambda_t (\Delta t_{kj}/\tau_t) + \lambda_p (e_{ik}/\tau_p) \quad (7)$$

The weights λ_t and λ_p are non-negative and should add up to one. A candidate with the lowest cost can be selected only if it has already passed both the time and spatial checks. If no candidate passes, the event is recorded as unmatched. If two candidates have

©IJENGT: All Rights are Reserved

very similar costs, or their boxes overlap in a way that makes the choice unclear, the event is marked ambiguous unless another validated rule can resolve it. The cost is used to rank candidates; it is not the YOLO confidence score and should not be described as the probability that a match is correct.

E. Event Output and Record Structure

Each decision should be recorded with enough detail for another person to review how it was made. The suggested CSV fields include the scan sequence number, Arduino timestamp, adjusted scan time, camera-frame timestamp, servo angle, measured range, projected image coordinate, detected object label, YOLO confidence, bounding-box coordinates, time difference, spatial residual, matching cost, calibration version, and final association status. The file should include unmatched and ambiguous events as well as accepted matches. Recording all outcomes gives a more complete picture of system behaviour than saving only successful matches.

V. ASSOCIATION WORKFLOW

The method follows the sequence below. Each stage checks its input before passing data to the next stage, so that incomplete serial messages, invalid echoes, missing camera frames, or uncertain matches are not accidentally treated as valid object-distance pairs.

1) Initialize: Load the YOLO model and camera calibration data, open the webcam and serial connection, and check that both data sources are available.

2) Acquire: Read a valid scan record from the Arduino and capture a webcam frame with its acquisition timestamp.

3) Align time: Convert the Arduino timestamp to the host clock and keep only frames that fall within the configured time window.

4) Project: Use the servo angle and measured range to form a point in the scanner coordinate frame, transform it into the camera frame, and project it onto the image.

5) Detect and filter: Run YOLO on the selected frame, remove detections below the confidence threshold, and check whether the remaining boxes are compatible with the projected point.

6) Decide: Rank the eligible detections using the combined cost and label the event as associated, unmatched, or ambiguous.

7) Display and save: Refresh the user interface and add the complete event to the CSV file. These tasks should run without interrupting camera capture or serial reading.

VI. IMPLEMENTATION CONSIDERATIONS

The Arduino should send each scan measurement as a complete serial record with a consistent delimiter and line ending. A practical record can contain the sequence number, timestamp, servo angle, range, and a flag indicating whether the reading is valid. The order of these fields must be the same in the Arduino firmware and the Python parser. On the computer, incomplete records, non-numeric values, angles outside the configured range, and invalid distance values should be rejected. The firmware should also use a timeout so that a missing echo does not leave the scan waiting indefinitely.

On the host computer, serial reading, camera capture, YOLO inference, and file writing should not block one another. A small queue can be used to pass timestamped events between these tasks. The software should record queue size and dropped frames, since a busy computer may end up processing an old frame after a newer scan event has arrived. Matching should be based on the recorded timestamps rather than on whichever detection happens to be visible at that moment.

Calibration is valid only for the camera setup used to collect it. A change in image resolution, digital zoom, focus, or camera position may affect the results and require calibration to be repeated. The camera-to-scanner transform must also remain fixed during operation. Before each test session, the operator should check a few known reference points and confirm that they appear in the expected image locations.

VII. EXPERIMENTAL DESIGN AND PERFORMANCE METRICS

The method should be tested under controlled conditions before it is described as validated. The test area should contain labelled objects placed at known positions, with their reference distances measured independently. Objects should be presented at several servo angles and ranges. For every trial, the camera frame, ultrasonic reading, actual target identity, and matching decision should be saved. This makes it possible to check whether the method links the correct image detection to the correct range event.

Testing can begin with one object in view and then move to scenes containing two or more objects. The conditions should also include changes in lighting, partial obstruction of an object, targets near the edge of the webcam's view, and different delays between frame capture and ultrasonic measurement. Separate reliability tests should examine serial delays, dropped frames, missing echoes, and temporary loss of the camera connection.

A. Baseline Comparisons

The proposed method should be compared with two simpler alternatives under the same test conditions. The first alternative matches observations using time alone, without geometric calibration. The second uses image-space position but does not apply a timestamp check. The third configuration uses both spatial and temporal checks, as described in this paper. Running all three methods on the same recorded dataset will make the comparison fairer than conducting a different physical trial for each method.

B. Association Metrics

Precision and recall should be calculated by comparing the system's decisions with pairs that have been checked manually. A true positive is a correct match between a camera detection and an ultrasonic event. A false positive is a match between observations that refer to different targets, while a false negative is a valid pair that the method does not identify. The metrics are defined as follows:

$$\text{Precision} = TP/(TP + FP), \quad \text{Recall} = TP/(TP + FN) \quad (8)$$

The false-association rate should also be reported. It is the number of incorrect accepted matches divided by the total number of accepted matches. The time difference between paired observations can be summarized using its median and an upper percentile. The image-space residual should be reported in pixels or in normalized bounding-box units. Together, these measures show both how often the method finds a match and how closely the paired observations agree.

C. Range and Runtime Metrics

The distance readings should be evaluated separately using a reference measuring instrument. For N valid samples, the mean absolute error (MAE) is calculated as:

$$MAE = (1/N)\ \Sigma\ |d_i - r_i| \quad (9)$$

In this expression, d_i is the HC-SR04 reading and r_i is the reference distance. Runtime testing should also record frame-processing time, scan-event processing time, total association latency, queue depth, dropped frames, and successful CSV writes. Results should be reported for each test condition. The report should also state the YOLO model version, input resolution, host-computer specifications, confidence threshold, calibration

©IJENGT: All Rights are Reserved

procedure, and matching thresholds so that the experiment can be repeated.

TABLE I. RESULTS TO BE COMPLETED AFTER CONTROLLED TESTING

Metric	Definition / reported value
Association precision	TP / (TP + FP) — [Measure]
Association recall	TP / (TP + FN) — [Measure]
False-association rate	Incorrect accepted matches / all accepted matches — [Measure]
Median time residual	Median $\|\Delta t\|$ — [Measure] ms
Median spatial residual	Median normalized image residual — [Measure]
Ultrasonic MAE	Mean $\|d - r\|$ — [Measure] cm
Association latency	Frame acquisition to match decision — [Measure] ms

VIII. DISCUSSION

The main value of this framework is that it makes the link between camera detections and ultrasonic measurements something the software checks, rather than something it assumes. A label is not attached to a distance simply because both appear on the screen at the same time. The system first checks whether the measured scan point could fall within the detected image region and whether the two observations were captured close enough together. By allowing unmatched and ambiguous outcomes, it also avoids presenting uncertain pairings as confirmed results.

The quality of the result depends heavily on calibration. If the camera's intrinsic parameters are inaccurate, the projected point may appear in the wrong part of the image. An incorrect camera-to-scanner transformation can cause a similar mismatch. Timing errors become especially important when an object is moving or when the servo takes a long time to complete a scan. For this reason, the spatial and temporal thresholds should be based on measured calibration and timing variation, not selected simply to make the display look responsive.

There are also limits to what the matching test can establish. The HC-SR04 receives echoes over a finite acoustic beam, so the reflecting surface may not be located exactly at the projected point. A YOLO bounding box may contain more than one object or include part of the background. The point-to-box check is therefore a practical way to filter possible matches, but it cannot prove that the camera and ultrasonic sensor observed precisely the same physical point. More advanced approaches could use several scan samples, track objects across frames, or apply probabilistic data association. Each option would need to be implemented and tested separately.

IX. LIMITATIONS AND FUTURE WORK

This paper describes a proposed method and a plan for evaluating it; it does not report completed experimental measurements. The materials available for the project do not include the camera's intrinsic parameters, the camera-to-scanner transformation, timestamp calibration results, labelled object-to-range pairs, or runtime logs. It is therefore not possible to claim a measured association accuracy, a reduction in latency, or an improvement in distance error. The projection also treats each range-and-angle reading as a point in the scanner's scan plane. Large, irregular, or angled surfaces may not follow this approximation.

The next stage is to add timestamps to the Arduino serial messages, calibrate the camera and its position relative to the scanner, and collect labelled observations across different objects, distances, angles, and lighting conditions. The combined method can then be compared with the time-only and geometry-only alternatives. Further work may explore object tracking, a model of the ultrasonic beam width, calibration-uncertainty estimates, and a time window adjusted to the measured system delay. These extensions should be included in the final research results only after they have been implemented and evaluated.

X. CONCLUSION

This paper has described a calibration-aware method for linking YOLO detections with measurements from a servo-mounted ultrasonic scanner. Each scan event is represented by its servo angle, measured range, and timestamp. The corresponding point is transformed from the scanner coordinate frame into the camera coordinate frame and projected onto the image. Detections from a suitably timed frame are then checked against that projected location. The matching rule considers both spatial and temporal consistency, along with the configured YOLO confidence threshold. When the available evidence is insufficient, the event remains unmatched or is marked ambiguous.

The method addresses a common difficulty in systems that combine a webcam with an ultrasonic scanner: operating both devices together does not guarantee that a visual label belongs to a particular distance reading. Recording calibration information, timestamps, matching residuals, and the final decision makes the association process easier to inspect and evaluate. Further testing is still needed before the framework can be described as experimentally validated or used to support numerical performance claims.

REFERENCES

[1] Z. Zhang, “A flexible new technique for camera calibration,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 11, pp. 1330–1334, Nov. 2000, doi: 10.1109/34.888718.

[2] OpenCV, “Camera Calibration and 3D Reconstruction,” OpenCV Documentation. [Online]. Available: https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html. [Accessed: Nov. 2, 2026].

[3] J. Kelly and G. S. Sukhatme, “On determining the time delay between inertial and visual sensor measurements,” IEEE Transactions on Robotics, vol. 30, no. 6, pp. 1514–1523, Dec. 2014, doi: 10.1109/TRO.2014.2343073.

[4] ROS 2, “message_filters: ApproximateTime Policy,” ROS 2 Documentation. [Online]. Available: https://github.com/ros2/message_filters. [Accessed: Nov. 2, 2026].

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[6] “HC-SR04 Ultrasonic Ranging Module,” SparkFun Electronics, technical datasheet. [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf. [Accessed: Nov. 2, 2026].

©IJENGT: All Rights are Reserved