Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

Automated Aiming and Tracking System

Devansh Malviya[1], Pranchal Bhadkariya[2], Ansh Jha[3], Saumya Soni[4], Dr. P.P. Bansod[5]

[1,2,3,4] UG Students, Department of Electronics and Instrumentation Engineering
Shri G.S. Institute of Technology and Science, Indore, India

[5]Dean of Academics, Shri G.S. Institute of Techn ology and Science, Indore, India

***Abstract*—In the present time, automation and intelligent systems are becoming an important part of daily life, especially in areas like surveillance, robotics, and smart environments. One of the key challenges in such systems is the ability to detect and continuously track a moving object in real time. This paper presents the design and implementation of a low-cost vision-based object tracking system that combines embedded hardware with distributed artificial intelligence processing.**

Instead of relying on a single device to perform all tasks, the proposed system divides the workload between a Raspberry Pi Zero 2W and an external processing unit. The Raspberry Pi is responsible for capturing video and controlling the mechanical movement of the camera, while the object detection is carried out using a deep learning model on a more powerful system. This approach helps in overcoming the limitations of embedded hardware, especially in terms of computational capability.

A pan-tilt mechanism driven by servo motors is used to physically adjust the camera orientation so that the object remains within the field of view. The system operates in a continuous loop, where frames are captured, processed, and used to generate control commands. Experimental observations show that the system performs reliably under normal indoor conditions, with acceptable response time and tracking stability. The design focuses on simplicity, affordability, and flexibility, making it suitable for academic projects and real-world applications. Additionally, the modular structure allows future improvements such as multi-object tracking, edge AI integration, and enhanced control strategies.

***Keywords*—Object Tracking, Raspberry Pi, Deep Learning, YOLO, Embedded Systems, Computer Vision, Pan-Tilt Mechanism, Real-Time Systems, Distributed Processing.**

1 Introduction

Object tracking is an important concept in modern technology, especially in applications like surveillance systems, autonomous robots, human-computer interaction, and smart monitoring solutions. The ability to detect and follow a moving object accurately in real time is essential for building intelligent systems.

Traditionally, object tracking has been implemented using high-end systems such as PTZ (Pan-Tilt-Zoom) cameras, which provide reliable performance but are expensive and not easily customizable. Because of this, such systems are not always suitable for student projects or low-budget implementations.

With the availability of low-cost hardware such as Raspberry Pi and the advancement in deep learning techniques, it has become possible to design more affordable tracking systems. However, a major challenge arises when trying to run computationally intensive algorithms on embedded devices. As discussed in the dissertation , embedded systems often lack the processing power required for real-time deep learning inference.

To address this limitation, the system proposed in this paper follows a distributed architecture. In this setup:

the embedded device handles real-time operations such as video capture and motor control the external system performs object detection using deep learning

This separation allows each component to perform tasks that suit its capability, resulting in better overall performance.

The main objective of this work is to design a system that is: low cost, capable of real-time tracking, simple to implement, scalable for future improvements. [1], [2]

2 Literature Review

2.1 Traditional Object Detection Techniques

In earlier systems, object detection was carried out using traditional image processing techniques. For instance, Haar Cascade classifiers and Histogram of Oriented Gradients (HOG) are some of the traditional techniques that have been widely employed in earlier architectures. These techniques employed feature detection to detect objects in images. [3]

Although these techniques have been efficient in terms of computational complexity, they have some limitations. For instance, their effectiveness is largely dependent on lighting conditions, background, and object orientation. This implies that they are not effective in a dynamic environment.

2.2 Deep Learning for Object Detection

The introduction of deep learning has significantly improved object detection performance. Convolutional Neural Networks (CNNs) are capable of automatically extracting relevant fea-

Identify applicable funding agency here. If none, delete this.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

tures from images, making detection more robust and efficient [3].

Two primary approaches are commonly used in object detection:

- **Two-stage detectors:**
  - Examples: R-CNN, Faster R-CNN
  - Provide high detection accuracy
  - Have slower processing speed due to multi-stage computation [9]
- **Single-stage detectors:**
  - Example: YOLO (You Only Look Once)
  - Offer faster processing speeds
  - Suitable for real-time applications [4], [5]

YOLO is widely used due to the following advantages:

- Processes the entire image in a single pass
- Achieves a good balance between speed and accuracy

2.3 Embedded Systems in Vision Applications

Embedded systems such as the Raspberry Pi are widely used in vision-based applications due to their practical advantages [7].

Key advantages include:

- Low cost
- Compact size
- Easy interfacing with sensors and peripherals

However, these systems also have certain limitations:

- Limited CPU processing power
- Restricted memory capacity
- Difficulty in efficiently running deep learning models

2.4 Distributed System Approach

To overcome hardware constraints, distributed systems are employed for efficient task management and performance optimization [2].

In such systems:

- **Embedded Device:** Responsible for sensing and actuation tasks
- **External System:** Handles computationally intensive processes

The key benefits of this approach include:

- Improved overall system performance
- Reduced computational load on the embedded device
- Enhanced scalability for future upgrades

However, certain challenges are associated with distributed systems:

- Communication latency between system components
- Synchronization issues during real-time operation

2.5 Identified Research Gap

Based on the literature review, several research gaps have been identified:

- Lack of low-cost solutions capable of real-time performance
- Inefficient utilization of embedded systems for intelligent applications
- Limited integration between AI-based detection and control systems

The proposed system addresses these challenges through the following approaches:

- Implementation of distributed AI processing
- Integration of embedded control mechanisms
- Use of efficient communication protocols for data exchange

3 System Design and Methodology

3.1 System Architecture

The system is divided into four major modules:

- Vision acquisition module
- Processing module
- Communication module
- Control and actuation module

The overall system operates in a continuous loop, consisting of the following steps:

- Capture video frame
- Send frame to the processing unit
- Detect the object
- Calculate the object position, generate control signal, adjust camera position accordingly

In order to effectively manage real-time object tracking tasks, the suggested system architecture employs a distributed and modular design approach. A structured data exchange mechanism allows each module to function independently while maintaining synchronized communication. Video frames are continuously captured by the vision acquisition module and sent to the external processing unit, which uses sophisticated object detection algorithms. Control signals are created and sent back to the embedded system based on the position of the detected object. The camera dynamically modifies its orientation to keep the target in the frame thanks to this closed-loop interaction. In addition to increasing system scalability and flexibility, the modular architecture enables the future integration of sophisticated features like multi-object tracking and on-device inference.Additionally, the computational load on the embedded platform is greatly decreased by employing a distributed processing strategy.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

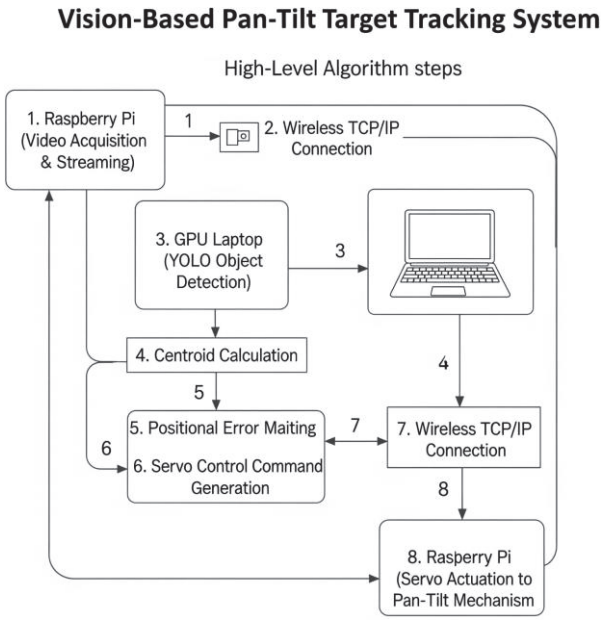

Fig. 1: System Architecture of Vision-Based Tracking System

3.2 Hardware Components

3.2.1 Raspberry Pi Zero 2W

The Raspberry Pi Zero 2W serves as the central controller of the system.

Its primary functions include:

- Capturing video frames from the camera module
- Sending data to the external processing system
- Receiving control commands from the processing unit
- Generating PWM signals for servo control

Key advantages of using the Raspberry Pi Zero 2W include:

- Low cost
- Compact size
- Ease of integration with various hardware components

Apart from its fundamental features, the Raspberry Pi Zero 2W's low power consumption and effective processing power make it a dependable platform for embedded vision applications. The board facilitates easy integration with sensors and external devices by supporting multiple communication interfaces, including SPI, I2C, and UART. Real-time communication with the external processing unit requires wireless data transmission, which is made possible by its integrated Wi-Fi module. Additionally, development and debugging procedures are made easier by the abundance of software libraries and community support. The Raspberry Pi Zero 2W is a good option for creating scalable and reasonably priced real-time tracking systems because of these features.

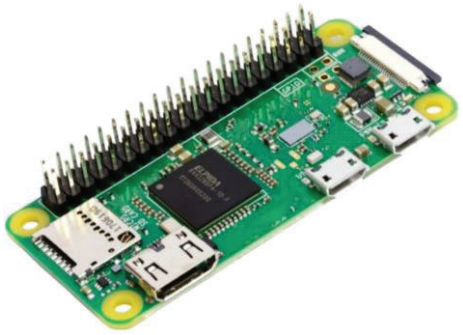

Fig. 2: Raspberry Pi Zero 2W

3.2.2 Camera Module

The camera module connected to the Raspberry Pi Zero 2W is responsible for capturing continuous video streams, which serve as input for the object detection system. Typically, the Raspberry Pi Camera Module (such as Camera Module v2 or HQ Camera) is used, which supports high-resolution imaging and stable video capture.

The camera interfaces with the Raspberry Pi through the CSI (Camera Serial Interface), enabling fast data transfer with minimal latency. It is capable of capturing video at various resolutions and frame rates, allowing flexibility based on application requirements.

.

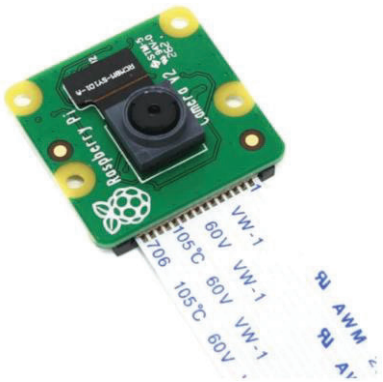

Fig. 3: Camera Module

3.2.3 Pan-Tilt Mechanism

The pan-tilt mechanism enables controlled movement of the camera, allowing it to track objects dynamically in both horizontal and vertical directions.

The main components of the mechanism include:

- Two servo motors
- Mounting structure for camera support

The system provides the following capabilities:

- **Pan:** Horizontal rotation of the camera
- **Tilt:** Vertical rotation of the camera

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

However, certain challenges are associated with the mechanism:

- Mechanical jitter during movement
- Limited precision in positioning

Fig. 4: Pan-Tilt Mechanism

3.2.4 Hardware Specification

Component	Specification
Raspberry Pi Zero 2W	Quad-core ARM Cortex-A53 @1GHz, 512MB RAM
Camera Module	CSI Interface, HD video capture
Servo Motors (SG90)	PWM controlled, 0°–180° rotation
Pan-Tilt Mechanism	2-DOF mechanical bracket
Communication	TCP/IP over Wi-Fi
Power Supply	5V DC
External Processing Unit	CPU/GPU based system

TABLE 1: **Component Specification**

3.3 Software Architecture

3.3.1 Embedded Software

Running on Raspberry Pi:

- Captures video
- Sends frames
- Receives commands
- Controls motors

Python is used because of its simplicity and available libraries.

3.3.2 AI Processing Software

The AI processing software operates on an external system and performs the following functions:

- Receives image frames from the camera module
- Processes images for enhancement and feature extraction
- Detects objects using advanced algorithms
- Calculates position and spatial coordinates of detected objects
- Sends control commands to the connected system

GPU acceleration is utilized to achieve real-time performance and ensure efficient processing of high-resolution data streams [4], [5].

3.4 Communication Protocol

The system uses TCP communication to ensure reliable data transfer between components.

There are two primary data flows in the system:

- **Forward Direction:** Transmission of video frames from the camera module to the processing unit
- **Reverse Direction:** Transmission of control commands from the processing unit to the hardware system

TCP ensures that all data packets are delivered without loss and in the correct sequence, thereby maintaining system reliability and synchronization [2].

3.5 Control Strategy

The control system determines the positional error by calculating the difference between the detected object position and the center of the frame.

Based on this error, the system performs the following actions:

- Adjusts servo motor angles to align the object with the center of the frame
- Applies smoothing techniques to the movement to ensure gradual transitions

This approach minimizes sudden jerks and significantly improves tracking stability and overall system performance.

4 Implementation and Result

4.1 System Implementation

The system was implemented using the following components:

- Raspberry Pi Zero 2W
- Camera module
- Servo motors
- External processing system

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

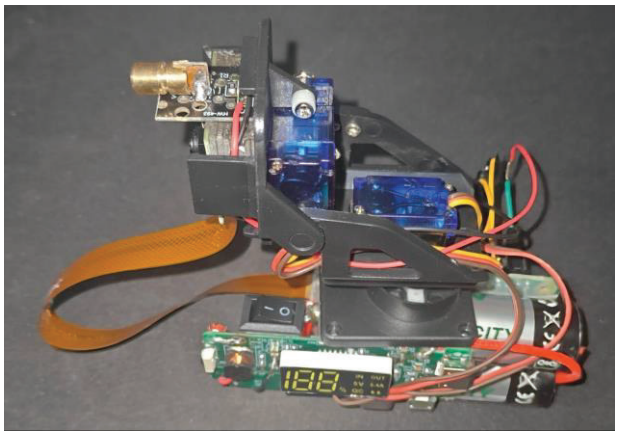

Fig. 5: Hardware Setup

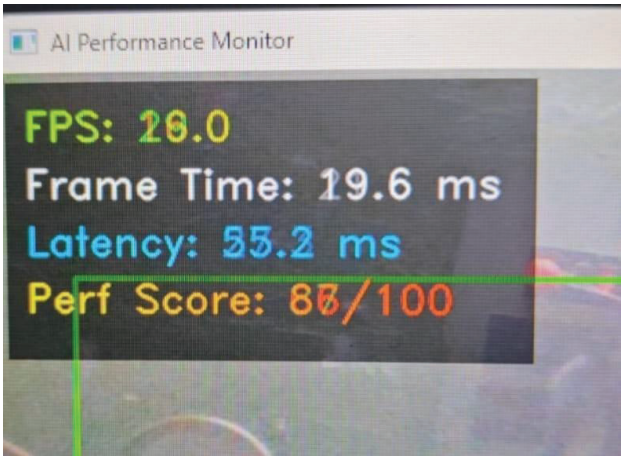

Fig. 6: Tracking Parameters

4.2 Performance Evaluation

The system was tested under indoor environmental conditions to evaluate its performance and reliability.

The key results obtained are as follows:

- Tracking performance was stable throughout operation
- Response time was within acceptable limits for real-time applications
- Detection accuracy was observed to be satisfactory

Metric	Minimum	Maximum	Average
Frames Per Second	23.39	32.93	28.29
Frame Time (ms)	11.67	27.64	16.87
Latency (ms)	30.51	44.43	36.53
Performance Score	73.40	90.00	83.48

TABLE 2: System Performance Metrics

4.3 Observation

During testing, several observations were made regarding system performance:

- Distributed processing significantly improved overall efficiency
- The system was able to handle moderate motion effectively
- Servo smoothing enhanced stability and reduced abrupt movements

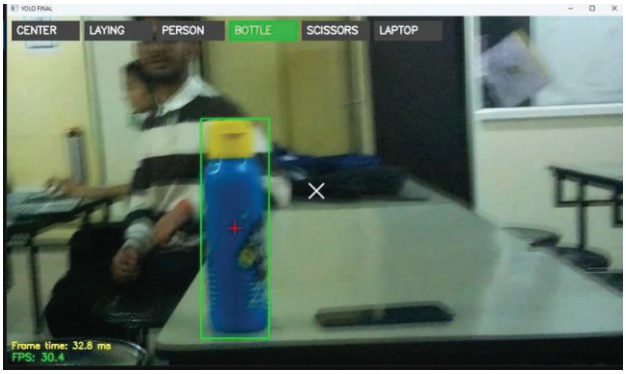

Fig. 7: Real Time Tracking

4.4 Challenges Faced

Some issues were observed during system operation:

- Communication delay between system components
- Sensitivity to varying lighting conditions
- Minor inaccuracies in servo motor positioning

These issues were mitigated through the following measures:

- Calibration of sensors and actuators
- Parameter tuning for optimized performance
- Enhancement of control algorithms and logic

5 Discussion

The system demonstrates that effective object tracking can be achieved without the need for expensive hardware. By adopting a distributed processing approach, the system efficiently balances computational load and control operations, resulting in improved performance and scalability.

Compared to traditional systems, the proposed solution offers the following advantages:

- Lower overall cost of implementation
- Easier setup and deployment

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 03 , March - 2026

- Greater flexibility for modifications and future enhancements

6 Conclusion

This paper presents the design and implementation of a vision-based object tracking system that utilizes a distributed processing technique. The system was able to effectively integrate an embedded system and an external processing system to address the limitations associated with embedded systems. The system utilizes the Raspberry Pi Zero 2W to perform real-time tasks, and it utilizes an external system that performs object detection through deep learning.

The system utilizes a pan-tilt mechanism that enables it to track moving objects. The camera is adjusted to focus on moving objects. This enhances the system's applicability compared to other monitoring systems. During testing, the system was able to perform adequately in normal conditions. The system's response time and accuracy were reasonable. The system's ability to utilize distributed processing was effective in enhancing performance and cost. The system's ability to perform adequately without incurring additional costs is attributed to its ability to utilize distributed processing. The system's modularity enhances its flexibility and ability to be modified. However, certain limitations were also experienced, which include sensitivity to lighting conditions, minor communication delays, and inaccuracies in the mechanical movement of the servo motors. This shows that despite the good performance of the system, there is still room for improvement.

Generally, it is evident that the proposed system is capable of tracking an object in real time with the aid of affordable components.

7 Future Scope

Although the system is working well in this form, there are several improvements that can be made to the system in order to improve the performance and extend the capabilities of the system. First and foremost, the system can be made independent of external systems by incorporating on-device processing with more advanced embedded systems or AI accelerators. This will minimize the communication delays that might occur in the system.

Another improvement that can be made to the system is the implementation of multi-object tracking. The system is currently designed to track a single object. However, if the system is made capable of tracking multiple objects, it can be used in several real-world scenarios.

The control mechanism can be made even better by implementing advanced control mechanisms such as PID control. The PID control mechanism can be used to make the pan-tilt mechanism move smoothly. Further improvements can be made in the area of robustness by making the system less sensitive to environmental factors such as changes in lighting and background noise. This can be done by training the detection models with different datasets or by adding image enhancement features.

The system can be further developed with features such as remote monitoring via the web or mobile devices. Hardware upgrades can also be made to the system to improve overall performance.

With these improvements in place, the system can be developed into a more robust and flexible system that can be used in various applications.

References

[1] T. Teixeira, F. Gouveia, and J. Vieira, "Lightweight Embedded Vision System for Real-Time Tracking," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 5, pp. 1600–1605, 2010.

[2] A. Kouris and T. Stathaki, "Efficient Real-Time Object Tracking using Deep Learning and Embedded Systems," *IEEE Access*, vol. 7, pp. 110191–110202, 2019.

[3] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Net-works," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2012.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[5] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[6] W. Liu et al., "SSD: Single Shot MultiBox Detector," in *European Conference on Computer Vision (ECCV)*, 2016.

[7] Raspberry Pi Foundation, "Raspberry Pi Documentation," [Online]. Available: https://www.raspberrypi.org/documentation

[8] OpenCV, "Open Source Computer Vision Library," [Online]. Available: https://opencv.org/

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[10] A. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv preprint arXiv:1704.04861*, 2017.