WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Design and Development of Autonomous Robotic Arm Using TinyML

Minal Thawakar[1], Pranav Shende[2], Arun Kashyap[3], Sujal Borkar[4], Krishnam Bilse[5]

[1,2,3,4,5]Artificial Intelligence and Data Science, KDK College of Engineering Nagpur, India
[1]minalthawakar1988@gmail.com, [2]pranav779895@gmail.com, [3]golukashyap7517@gmail.com, [4]borkarsujal13@gmail.com, [5]krishnambilse122@gmail.com

ABSTRACT—

The objective of this research is to develop a cost-effective and autonomous robotic arm system for intelligent object sorting in warehouse-like environments using TinyML. The proposed system is designed to identify labeled packages in real-time using an embedded vision model and perform corresponding pick-and-place operations without human intervention. Emphasis is placed on low power consumption, local processing, and minimal hardware complexity to ensure deployability in constrained settings. The integration of edge-based machine learning with predefined motion control enables responsive and efficient handling of categorized items. This work contributes toward practical automation solutions for small-scale inventory management and demonstrates the feasibility of TinyML in real-time robotic applications.

***Keywords*—TinyML, Robotic Arm, Object Detection, Edge AI, On-Device Learning, Warehouse Automation.**

I. INTRODUCTION

The increasing demand for automation in warehouse management systems has prompted the development of intelligent solutions capable of reducing human intervention, optimizing space utilization, and improving operational efficiency. Among the various technologies driving automation, robotics and machine learning have shown significant potential in enhancing sorting, categorization, and pick-and-place tasks. However, the high computational power requirements of traditional machine learning models often make them impractical for deployment on resource-constrained embedded systems, especially in environments where low cost and low power consumption are critical.

Tiny Machine Learning (TinyML) offers a promising approach for deploying machine learning models directly on embedded devices, allowing real-time processing without the need for cloud-based computation. This research focuses on leveraging TinyML to enable real-time object classification for an autonomous robotic arm used in warehouse sorting. The system integrates a vision-based approach, where an *ESP32-CAM* captures images of labeled boxes, and a *TinyML* model processes these images to classify them as either "apple" or "mango." Based on the classification result, an Arduino Nano controls the movement of the robotic arm to place the object in the appropriate storage space. The integration of *Light Dependent Resistor (LDR***)** sensors ensures that storage spaces are monitored for occupancy, allowing the system to make decisions based on available space.

This paper aims to present a comprehensive solution for a scalable, low-cost, and energy-efficient robotic sorting system suitable for small to medium-scale warehouse environments. By combining edge-based machine learning with a lightweight robotic arm design, the proposed system offers a practical alternative to traditional automated sorting systems.

II. LITERATURE SURVEY

Zhang et al. [1] proposed an automated robotic arm system capable of inventory management, where a camera-based object detection system was employed to classify items. Despite its effectiveness, the reliance on cloud-based computation introduces latency and dependency on a stable network connection.

Tan et al. [2] explored integrating low-cost embedded systems, such as Raspberry Pi and Arduino, to control robotic arms for warehouse tasks. These solutions emphasize cost-efficiency but are still limited by power consumption and computational capacity.

Lobaton et al. [3] demonstrated the feasibility of using TinyML on embedded devices such as Arduino boards for low-power robotic systems. The study showed that machine learning models could be deployed on small microcontrollers without the need for cloud computing.

Saha et al. [4] discussed the integration of ESP32 microcontrollers with TinyML for object detection tasks, emphasizing the potential of low-cost, low-power platforms for real-time machine learning on embedded devices. These advancements make TinyML a promising solution for autonomous robotic systems in warehouse environments where computational resources are limited.

Li et al. [5] utilized the ESP32-CAM module for image classification tasks in a warehouse setting. Their study demonstrated the use of lightweight models like MobileNetV2 and Tiny-YOLO to perform object classification directly on the device with minimal latency, making them suitable for edge devices.

MobileNetV2, a lightweight deep learning model, has been widely used for object classification tasks in embedded systems, as it provides a good balance between accuracy and computational efficiency [6]. These advances in embedded vision systems make it possible to deploy object detection and classification tasks on resource-constrained devices in warehouse automation.

Zhang et al. [7] demonstrated the use of LDR sensors in a sorting system, where sensors monitored available storage space to ensure that the robotic arm only placed items in sections that were not full. This integration is vital for improving the efficiency and responsiveness of autonomous robotic arms in real-world environments.

III. PROPOSED METHODOLOGY

The proposed system automates the sorting of boxes into distinct storage spaces based on real-time object classification using TinyML. The methodology is divided into four key sections: System Design, Hardware, Software, and Architecture, each contributing to a modular, scalable, and efficient solution.

A. System Design

The system is engineered to identify the label on incoming boxes (Apple or Mango), process it using a compact ML model on an ESP32-CAM, and execute appropriate robotic arm movement to place the box in its designated bin. Storage occupancy is dynamically tracked using LDR sensors. The aim is to mimic an autonomous warehouse sorting workflow with lightweight, embedded intelligence and efficient motion control.

B. Hardware Components

1) ESP32-CAM: Serves as the vision and intelligence unit of the system. It captures real-time images of boxes placed in front of it and processes them using a TinyML model trained via Edge Impulse.

2) Arduino Nano: The Arduino Nano is dedicated to handling the mechanical operations of the robotic arm. Upon receiving the classification result from the ESP32-CAM, it interprets the label and activates predefined motion logic. This logic determines how the robotic arm should move to perform the pick-and-place operation.

3) ESP32 XIAO: The ESP32 XIAO handles peripheral tasks, including displaying the classification result on a 0.96" OLED screen and monitoring LDR sensors placed in the apple and mango bins to check for space availability.

4) Robotic Arm: Constructed with three servo motors (MG945, MG90S, SG90) and a 5V stepper motor, offering four degrees of freedom and controlled box handling.

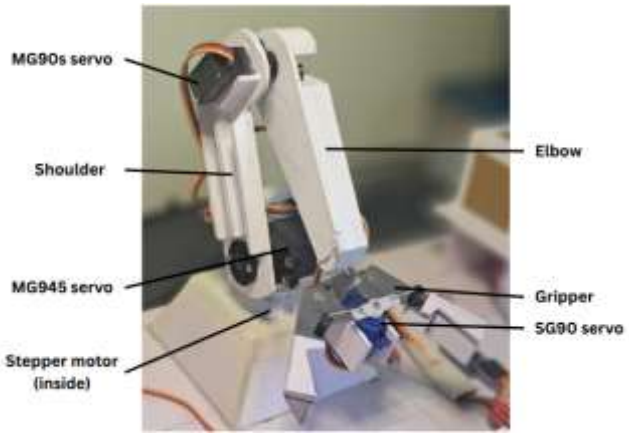

Fig 1. Robotic Arm

5) LDR Sensors: Two LDR (Light Dependent Resistor) sensors are placed at the Apple and Mango storage sections to detect the occupancy status of each bin.

*6) OLED Display: T*he 0.96-inch OLED display module, controlled by the ESP32 XIAO, acts as a visual feedback mechanism. It shows the classification result determined by the TinyML model

7) Power Supply: A 5V 3A supply powers the complete system.

8) Chassis: Entire structure is fabricated using a 3mm sunboard, selected for its water resistance and lightweight properties.

C. Software Components

The system software includes a TinyML model trained using Edge Impulse, which was exported as an Arduino-compatible library and deployed on the ESP32-CAM for real-time image classification. Arduino IDE was used to program the ESP32-CAM, Arduino Nano, and ESP32 XIAO boards. The Nano executes predefined motion logic for the robotic arm, while the XIAO manages OLED display updates and LDR sensor readings. All code was written in embedded C/C++ using Arduino libraries for efficient microcontroller integration.

D. Architecture Adopted

A simple architecture is proposed, following flow diagram (Fig. 2) depicts the architecture used:

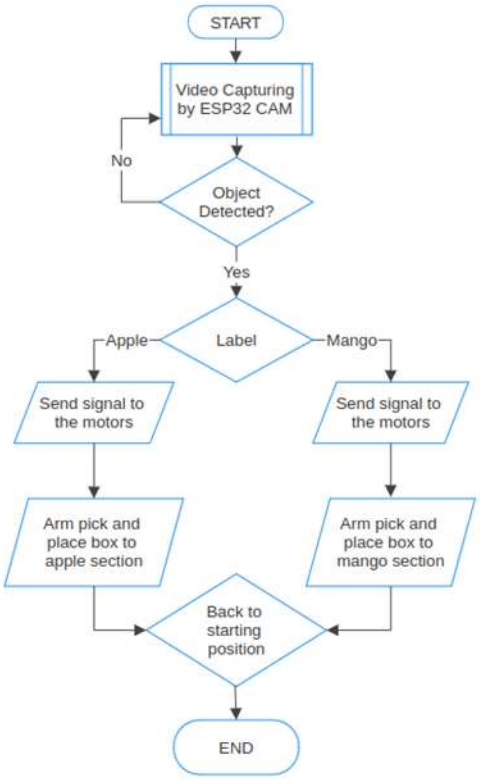

Fig. 2. Flow architecture proposed

IV. RESULT

The following figures depict the proposed system identifying the box label using the TinyML model deployed on the ESP32-CAM (Fig. 4), achieving an inference time of less than 0.7 seconds. Based on the classification output, the robotic arm receives the command via the Arduino Nano and performs the corresponding pick-and-place operation (Fig. 3).

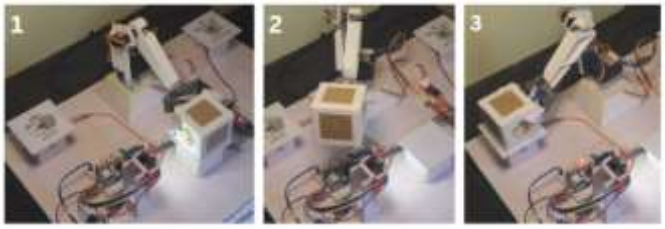

Fig .3. Pick and place operation

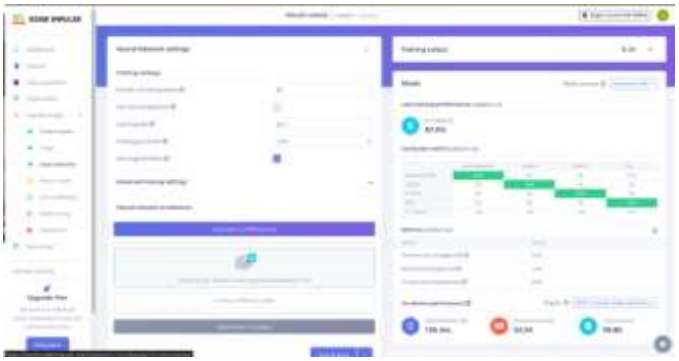

Fig .3. TinyML model development

V. CONCLUSION

This paper presents the design and development of an autonomous robotic arm system integrated with TinyML for intelligent warehouse sorting. By leveraging a lightweight TinyML model deployed on the ESP32-CAM, the system efficiently classifies labeled boxes in real-time and performs accurate pick-and-place operations using a 4-DOF robotic arm. The integration of ESP32 XIAO for peripheral control and LDR-based occupancy detection ensures smooth and reliable functioning. The compact, cost-effective hardware and low-latency classification make the system suitable for small-scale automation in logistics and warehouse environments. Future work may explore dynamic path planning, cloud connectivity, and support for multi-class object detection.

REFERENCES

[1] M. Mazumder, C. Banbury, J. Meyer, P. Warden, and V. J. Reddi, “Few-shot keyword spotting in any language,” arXiv preprint arXiv:2104.01454, 2021.

[2] E. Hardy and F. Badets, “An ultra-low power rnn classifier for always-on voice wake-up detection robust to real-world scenarios,” arXiv preprint arXiv:2103.04792, 2021.

[3] T. Luukkonen, A. Colley, T. Seppänen, and J. Häkkilä, “Cough activated dynamic face visor,” in Augmented Humans Conference 2021, 2021, pp. 295–297.

[4] Hsiao-Yun Tseng, Ching-Hao lai, Shyr-Shen Yu, “An effective license plate detection method for over exposure and complex vehicle images,” International Conference on Convergence and Hybrid Information Technology , 2008, pp. 176-181.

[5] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, “Towards total recall in industrial anomaly detection,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 14 318–14 328.

[6] A. Capotondi, M. Rusci, M. Fariselli, and L. Benini, “Cmix-nn: Mixed low-precision cnn library for memory constrained edge devices,” IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 67, no. 5, pp. 871–875, 2020.

[7] M. Muniswamaiah, T. Agerwala, and C. C. Tappert, “A survey on cloudlets, mobile edge, and fog computing,” in 8th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2021 7th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom). IEEE, Jun 2021, p. 139–142.

[8] S. G. Patil, P. Jain, P. Dutta, I. Stoica, and J. Gonzalez, “Poet: Training neural networks on tiny devices with integrated dematerialization and paging,” in International Conference on Machine Learning. PMLR, 2022, pp. 17 573–17 583

[9] C. Profentzas, M. Almgren, and O. Landsiedel, “Minilearn: On-device learning for low-power iot devices,” in Proceedings of the 2022 International Conference on Embedded Wireless Systems and Networks (Linz, Austria)(EWSN 22). Junction Publishing, USA, 2022.

[10] F. Tung and G. Mori, “Similarity-preserving knowledge distillation,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1365– 1374.

[11] D. L. Dutta and S. Bharali, “TinyML meets IoT: A comprehensive

[12] survey,” Internet Things, vol. 16, Dec. 2021, Art. no. 100461.

[13] C. Banbury et al., “MLPerf Tiny benchmark,” in Proc. Adv. Neural Inf.Process. Syst., 2021, 1–15.

[14] P. Warden, “Speech commands: A dataset for limited-vocabularyspeech recognition,” 2018, arXiv:1804.03209.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[16] R. Krishnamoorthi, “Quantizing deep convolutional networks for effi-cient inference: A whitepaper,” 2018, arXiv:1806.08342.

[17] O. Saha, A. Kusupati, H. V. Simhadri, M. Varma, and P. Jain, “RNNPool: Efficient non-linear pooling for RAM constrained infer-ence,” in Proc. Adv. Neural Inf. Process. Syst., vol. 33, 2020, pp. 1–12.

[18] Christos Nikolaos E. Anagnostopoulos, Ioannis E. Anagnostopoulos, Vassili Loumos, and Eleftherios Kayafas, "A License Plate-Recognition Algorithm for Intelligent Transportation System Applications," pp. 377-392, 2006.

[19] Edge Impulse. Accessed: Mar. 3, 2022. [Online]. Available: https://www.edgeimpulse.com/

[20] S. Yao, Y. Zhao, A. Zhang, L. Su, and T. Abdelzaher, “DeepIoT: Compressing deep neural network structures for sensing systems with a compressor-critic framework,” in Proc. 15th ACM Conf. Embedded Netw. Sensor Syst., Nov. 2019, pp. 1–14.

[21] B. Zoph and Q. V. Le, “Neural architecture search with reinforcement learning,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2020, pp. 1–16.

[22] H. Cai, C. Gan, L. Zhu, and S. Han, “TinyTL: Reduce memory, not parameters for efficient on-device learning,” in Proc. Adv. Neural Inf.Process. Syst., vol. 33, 2020, pp. 1–13.

[23] I. Fedorov, R. P. Adams, M. Mattina, and P. N. Whatmough, “SpArSe:Sparse architecture search for CNNs on resource-constrained micro-controllers,” in Proc. Adv. Neural Inf. Process. Syst., vol. 32, 2019,pp. 1–13.

[24] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, “Rethinking the value of network pruning,” in Proc. Int. Conf. Learn. Represent., 2018, pp. 1–21.

[25] J. Yu, A. Lukefahr, D. Palframan, G. Dasika, R. Das, and S. Mahlke, “Scalpel: Customizing DNN pruning to the underlying hardware par-allelism,” in Proc. 44th Annu. Int. Symp. Comput. Archit., Jun. 2017,pp. 548–560.

[26] E. Liberis and N. D. Lane, “Differentiable network pruning to enable smart applications,” in Proc. 4th U.K. Mobile, Wearable Ubiquitous Syst. Res. Symp., 2022, p. 1.

[27] U. Thakker et al., “Compressing RNNs to kilobyte budget for IoT devices using Kronecker products,” ACM J. Emerg. Technol. Comput.Syst., vol. 17, no. 4, pp. 1–18, Jul. 2021.

[28] U. Thakker, P. Whatmough, Z. Liu, M. Mattina, and J. Beu, “Doping: A technique for extreme compression of LSTM models using sparse structured additive matrices,” in Proc. Mach. Learn. Syst., vol. 3, 2021, pp. 533–549.

[29] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size,” 2016, arXiv:1602.07360.

[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton “ImageNet classification with deep convolutional neural networks,” in Proc. Adv. Neural Inf. Process. Syst. (NIPS), vol. 25, Dec. 2012, pp. 1097–1105.

[31] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen,“MobileNetV2: Inverted residuals and linear bottlenecks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018,pp. 4510–4520.

[32] X. Zhang, X. Zhou, M. Lin, and J. Sun, “ShuffleNet: An extremely efficient convolutional neural network for mobile devices,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018,pp. 6848–6856.