ML-Based Object Recognition and Object Picking Robot using ROS

G.A. Rathy[1], P. Sivasankar[2], B. AravindBalaji[3]

[1]Professor EEE, NITTTR, Chennai
[2] Professor ECE, NITTTR, Chennai
[3]Senior Development Engineer, Silicon Technologies, Chennai
rathy@nitttrc.ac.in

Abstract: Machine Learning (ML) based object recognition refers to the use of machine learning techniques to identify and classify objects in images or videos. It plays a crucial role in computer vision, enabling machines to "see" and interpret visual data like humans. The integration of ML and robotics enables robots to perceive, learn, and adapt to their environments dynamically. This synergy enhances robotic capabilities, allowing them to handle complex tasks, make decisions, and interact with humans more effectively. By leveraging ML techniques, particularly Convolutional Neural Networks (CNNs), the system will be trained using YOLO architecture to recognize a diverse range of objects, encompassing various shapes, sizes, and textures. The dataset is meticulously curated and pre-processed to ensure the model receives high-quality input data. The integration with ROS will provide a modular framework, allowing for easy integration of sensors, actuators, and custom control algorithms. This study addresses critical challenges in robotics, such as adaptability in unstructured environments and real-time decision-making. The successful implementation of ML-based object recognition and picking capabilities will not only advance the field of robotics but also have far-reaching implications across Industries, from automated warehousing and manufacturing to assistive healthcare applications. This work marks a significant step towards creating intelligent, adaptable robots capable of complex interactions with their surroundings.

Keywords*: Machine Learning, Object Detection, Robots, Convolution Neural Network(CNN) Data Set and Training*

I. INTRODUCTION

In today's rapidly evolving technological landscape, the integration of Machine Learning (ML) with robotics has paved the way for advanced automation solutions. The integration of Machine Learning and robotics has revolutionized automation, enabling robots to adapt and interact with their environments in increasingly sophisticated ways. The focus is on the development of a cutting-edge robotic system that combines ML-based object recognition with the Robot Operating System (ROS) for seamless control and coordination. The primary goal is to create a versatile platform capable of autonomously identifying objects and efficiently picking them up, demonstrating the potential for advanced automation in various industries.

By leveraging ML techniques, particularly Convolutional Neural Networks (CNNs), the system will be trained using YOLO architecture to recognize a diverse range of objects, encompassing various shapes, sizes, and textures. The dataset is meticulously curated and pre-processed to ensure the model receives high-quality input data. The integration with ROS will provide a modular framework, allowing for easy integration of sensors, actuators, and custom control algorithms.

This study focuses on the development of a robotic system capable of autonomously recognizing objects using ML techniques and efficiently picking them up. The Robot Operating System (ROS) will serve as the foundation for creating a modular and flexible control framework. The successful implementation of ML-based object recognition and picking capabilities will not only advance the field of robotics but also have far-reaching implications across industries, from automated warehousing and manufacturing to assistive healthcare applications [15].

II. REVIEW OF LITERATURE

Ruohuai Sun (2023) introduced a parallel YOLO– deep learning network for collaborative robot target recognition and grasping to enhance the efficiency and precision of visual classification and grasping for collaborative robots [14]. The real-time recognition and grasping network can identify a diverse spectrum of unidentified objects and determine the target type and appropriate capture box using YOLO- deep vision network. The YOLOv3 network uses pre-trained COCO dataset, identifies the object category and position, while the GG-CNN network, trained using the Cornell Grasping dataset, predicts the grasping pose.

Natanael Magno Gomes (2022) uses Reinforcement Learning (RL) to train an Artificial Intelligence (AI) agent to control a Cobot to perform a given pick and-place task, estimating the grasping position without previous knowledge Deep RL applied to a Robotic Pick-and-Place Application [4]. To enable the agent to execute the task, an RGBD camera is used to generate the inputs for the system. An adaptive learning system was implemented to adapt to new situations such as new configurations of robot manipulators and unexpected changes in the environment

Moritz Abdank (2021) proposed a simple colour-based object detection for different robotic tasks, including a linear calibration method, with focus on a specific pick and place use case[1]. A standard RGB-Webcam stream was used to detect a certain colour and obtain information which then was filtered and evaluated based on the predefined conditions of the system. ROS enables the processing of the calibrated camera stream and the interface to the pick and place robot arm.

Humans have a surprising ability for pattern recognition and object identification, in order to mimic this ability different algorithms have been proposed based on deep learning. There are a lot of algorithms for computational vision based on convolutional neural networks, one of the most advanced techniques proposed to classify objects in real time is the You Only Look Once (YOLO) algorithm. This study uses Raspberry pi 4with servo motor control shield as the controller

III.METHODOLOGY

The methodology of this work focuses on developing an object detection system that integrates Raspberry Pi 4, machine learning and neural networks, specifically convolutional neural networks (CNNs), to achieve high accuracy and efficiency [2] [13]. The process begins with data collection and pre-processing, followed by ML Model Selection and Training, ROS Setup and Configuration and concludes with Feedback and Monitoring.

Processor and motor control shields

The Raspberry Pi 4 has a Broadcom BCM2711 processor. This is a 64-bit ARM Cortex-A72 (ARMv8-A) quad-core processor, which allows for parallel processing and improved multi-tasking at 1.5 GHz, providing better performance compared to previous models like the Raspberry Pi 3.The Raspberry Pi 4 uses the VideoCore VI GPU, a powerful graphics processor capable of supporting 4K video output.It can drive two 4K HDMI displays at 60Hz (with dual micro-HDMI

ports). It offers various RAM con**fig**urations, including 2 GB, 4 GB, and 8 GB of LPDDR4-3200 SDRAM. The increased RAM options make it more capable of running memory-intensive applications.

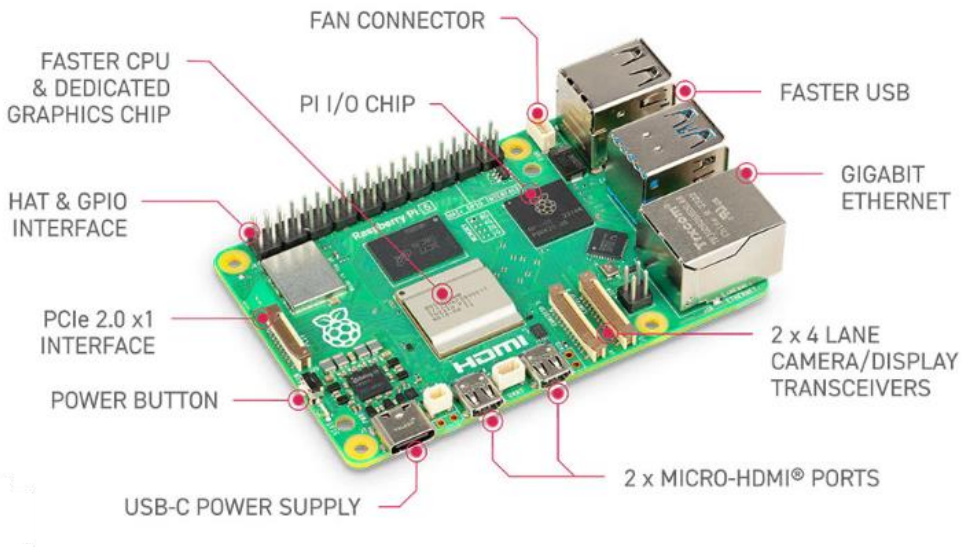

Fig. 1. Shows Raspberry pi 4 with features

I/O and Connectivity: USB Ports: 2 x USB 3.0 ports, 2 x USB 2.0 ports, Ethernet: Gigabit Ethernet for faster networking, Wi-Fi: 802.11ac Wi-Fi for wireless connectivity, Bluetooth: Bluetooth 5.0 for connecting to Bluetooth devices, Storage: MicroSD slot for storage, with support for faster SD cards for quicker boot and data access.

The Servo & Motor Control Shields are designed to simplify the process of controlling motors and servos in robotics and automation projects. These shields come with integrated motor driver circuits (e.g., L298N or similar), allowing the control of forward, reverse, and speed of DC motors. PWM (Pulse Width Modulation): Speed control of DC motors is typically achieved through PWM signals. The shield can generate PWM signals to adjust the speed of the motor by controlling the voltage. Motor Voltage: Typically supports motors running on 5V to 12V or higher.

Data Collection and Pre-processing

A diverse dataset of objects of interest is be collected, encompassing a range of shapes, sizes, and textures. Also, data augmentation is used to further diversify the dataset by applying simple image manipulation such as cropping, blurring and other techniques on the existing images to increase the images in the dataset. Data pre-processing steps be undertaken, including resizing, normalization, and augmentation. This ensures the ML model receives consistent and high-quality input.

ML Model Selection and Training

A Convolutional Neural Network (CNN) based YOLO V5 architecture is be employed for object recognition, owing to its effectiveness in image-related tasks. The chosen CNN model is trained on the pre-processed dataset using a suitable ML framework called PyTorch. This phase involves the iterative process of forward and backward passes to optimize the model's parameters [6] [10].

ROS Setup and Configuration

The ROS environment is configured to support the specific hardware components of the robotic system, including 2D camera, actuators such as servo motors for the manipulator arm and other human machine interfaces. Custom ROS nodes is developed to facilitate seamless communication between the ML model and the robot's control software [3][7].

Object Recognition Module
The trained ML model is deployed on the robot's onboard computer, enabling it to process images captured by the robot's cameras in real time. The ML-based recognition module annotates the video feed with labels indicating recognized objects[5][11].

Object Localization
Using the ML model's predictions, the system determines the spatial coordinates of the recognized objects relative to the robot's coordinate system [12].

Motion Planning
The robot's control system generates a suitable trajectory for the end-effector to approach and pick up the identified object. This involves calculating joint positions and velocities. This can be implemented using ROS package called move it, the inverse kinematics solver used for this manipulator is Kinematics and Dynamics Library (KDL) which is a powerful library for solving both kinematic and dynamic problems in robotic systems, with a focus on the efficient computation of forward and inverse kinematics, as well as dynamics modelling[8][9].

Object Manipulation
Control algorithms is implemented to ensure the robot's end-effector grasps and holds the object securely.

Feedback and Monitoring
Visual and servo feedback mechanisms is implemented to provide real-time information regarding the success or failure of object recognition and picking tasks.

IV. PROPOSED SYSTEM

The proposed system includes robotic system capable of autonomously recognizing objects using ML techniques and efficiently picking them up. The Robot Operating System (ROS) serve as the basis for creating a modular and flexible control framework. The successful operation of ML-based object recognition and picking is implemented using Raspberry pi 4.

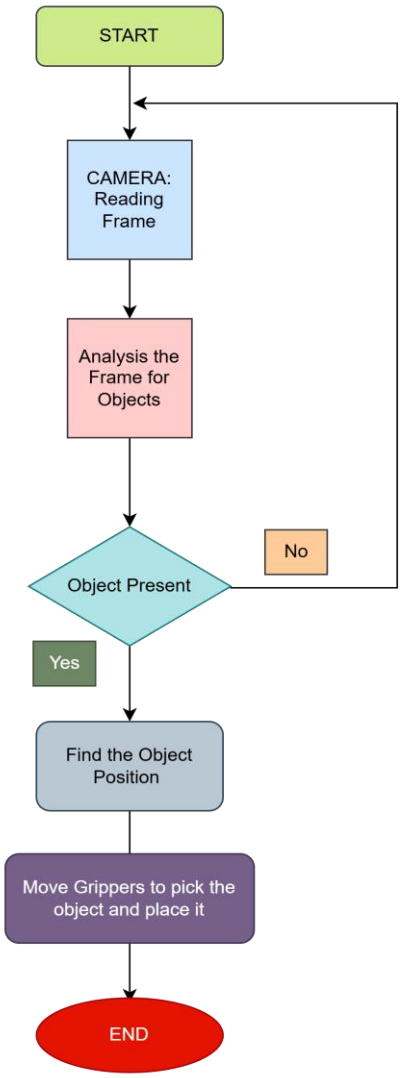

Fig. 2. Flow chart of the Proposed System

Figure 2 shows the flow chart of the proposed object detection system using machine learning technique. The camera captures the image of size as 300 x 300 pixels. Based on the captured image, the boundary box is estimated to pick up the object. After capturing the image, the image is analysed for the presence of the object. If the object is present, the position of the detected object is optimized. Unless the capture process will be repeated.

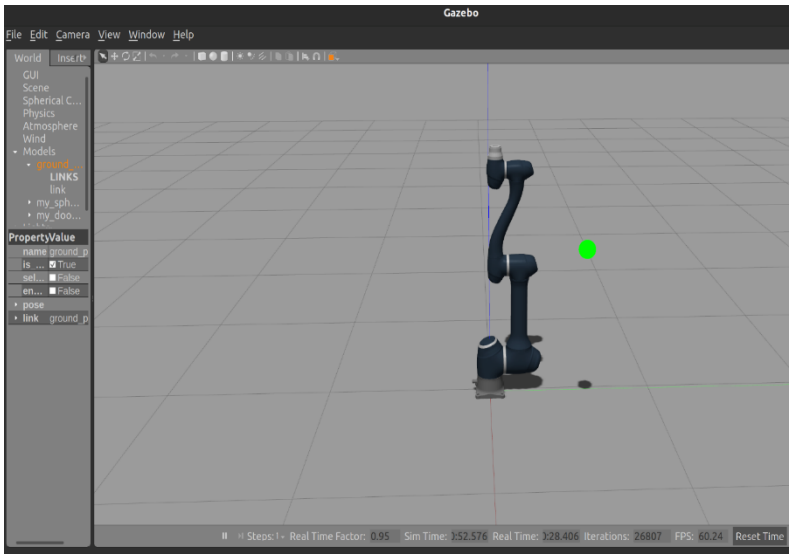

Fig. 3 shows the view of robot simulation using gazebo 3D robot simulation software

Fig. 4 shows the Hardware implementation of Robot manipulator arm with Raspberry Pi 4 and camera

V. PERFORMANCE MATRICES

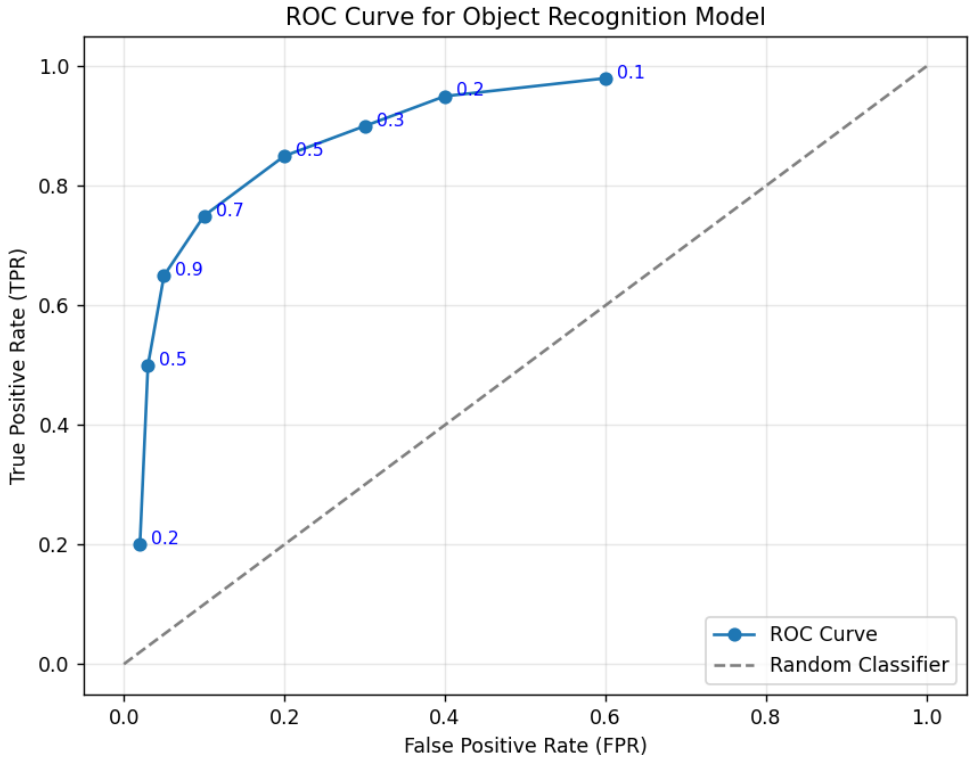

Fig. 5. ROC Curve of proposed object detection model

This ROC curve illustrates the performance of an object detection model. The curve demonstrates the trade-off between the True Positive Rate (TPR) and False Positive Rate (FPR) across different decision thresholds. The model performs significantly better than a random classifier (represented by the diagonal dashed line), as its ROC curve is consistently above this baseline. A high TPR with a relatively low FPR at most thresholds indicates that the model is effective at correctly detecting objects while minimizing false alarms. The curve's shape, particularly the steep ascent near the lower FPR values, highlights the model's strong ability to differentiate between true positives and negatives, affirming its reliability in object detection tasks. The ROC curve is preferred for performance evaluation in tasks like object detection because it provides a comprehensive view of a model's discriminative ability, especially when classifying between true positives and false positives. Unlike metrics such as accuracy, which can be misleading for imbalanced datasets, or precision and recall, which depend on fixed thresholds, the ROC curve evaluates performance across all thresholds. This makes it ideal for highlighting trade-offs between detecting true objects (high TPR) and minimizing false alarms (low FPR).

VI. OUTPUT OF THE PROPOSED SYSTEM

The system provides the following outputs:

- A real-time video feed with annotated object recognition results.
- ROS messages indicating recognized objects and their respective positions.
- Control commands for the robot's manipulator arm to execute picking tasks.

- Status updates indicating the success or failure of object recognition and picking operations.

VII. Conclusion

This study demonstrates the successful integration of ML-based object recognition with a robotic system using ROS, showcasing the potential for sophisticated automation in real-world applications. The ROC curve used for performance evaluation, proves that this model is best fit for object detection using machine learning , especially when classifying between true positives and false positives The developed system lays the foundation for more advanced robotic solutions in industries such as logistics, manufacturing, and healthcare. Future work may involve refining the ML model's accuracy, optimizing motion planning algorithms, and expanding the object dataset for enhanced versatility and adaptability.

REFERENCES

1. Abdank, M., et al. (2021). Using colour-based object detection for pick and place applications. *Annals of DAAAM & Proceedings, 32,* 536+. Gale Academic OneFile. https://doi.org/10.2507/32nd.daaam.proceedings.077
2. Dai, J., Li, Y., He, K., & Sun, J. (2016). R-FCN: Object detection via region-based fully convolutional networks. *Advances in Neural Information Processing Systems (NIPS)*, Barcelona, Spain.
3. Fu, K. S. (1987). *Robotics: Controlling, sensing, vision and intelligence* (International edition). McGraw-Hill.
4. Gomes, N., Martins, F., Lima, J., & Wörtche, H. (2022). Reinforcement learning for collaborative robots pick-and-place applications: A case study. *Automation, 3*(1). https://doi.org/10.3390/automation3010011
5. Gowsikraja, P., Thevakumaresh, T., Raveena, M., Santhiya, J., & Vaishali, A. R. (2022). Object detection using Haar cascade machine learning. *International Journal of Creative Research Thoughts (IJCRT), 10*(6), [Page Numbers].
6. Howard, A. G., Zhu, M., et al. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint*, arXiv:1704.04861.
7. Huang, J., Wang, X., Liu, D., & Cui, Y. (2012). A new method for solving inverse kinematics of an industrial robot. *2012 International Conference on Computer Science and Electronics Engineering*, 3.
8. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, USA.
9. Hock, O., & Sedo, J. (2018). Inverse kinematics using transportation method for robotic arm. *ELEKTRO 2018 Conference Proceedings*,
10. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA.
11. Kumar, S., Balyan, A., & Chawla, M. (2017). Object detection and recognition in images. *International Journal of Engineering Development and Research (IJEDR), 5*(4). ISSN: 2321-9939.

12. Prasad, S., & Sinha, S. (2011). Real-time object detection and tracking in an unknown environment. *World Congress on Information and Communication Technologies*.
13. Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. *arXiv preprint*, arXiv:1506.01497.
14. Sun, R., Wu, C., Zhao, X., Zhao, B., & Jiang, Y. (2024). Object recognition and grasping for collaborative robots based on vision. *Sensors*, *24*(195). https://doi.org/10.3390/s24010195
15. Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA.