IJMRT: Volume (8), Issue 1, 2026, Pages 25-41
International Journal of Multidisciplinary Research Transactions
(A Peer Reviewed Journal)
www.ijmrt.in

I SSN (Print):2663-2381
ISSN(Online):2663-4007
| Regular Article | Open Acess
Aravandi Publications

Robotic Arm Integrated With Object Recognition Using Vision-Based Opencv And CNN

Helan Sophia B[1], Sobhana Vidhyadharsini R S*[2], Dr. Shanmugavalli M[3]

[1,]Student, Department of Instrumentation and Control Engineering, Saranathan College of Engineering, Tamil Nadu, India.
[2]Student, Department of Instrumentation and Control Engineering, Saranathan College of Engineering, Tamil Nadu, India.
[3]Professor, Department of Instrumentation and Control Engineering, Saranathan College of Engineering, Tamil Nadu, India.

Corresponding author(s):
Sobhana Vidhyadharsini R S, Student, Department of Instrumentation and Control Engineering, Saranathan College of Engineering, Tamil Nadu, India.
Email: sobhanarajarajan@gmail.com

DoI: https://doi.org/10.5281/zenodo.18311342

Citation:

Helan Sophia B, Sobhana Vidhyadharsini R S, Dr. Shanmugavalli M (2026). Robotic Arm Integrated With Object Recognition Using Vision-Based Opencv And CNN. International Journal of Multidisciplinary Research Transactions, 8(1), 25–41. https://doi.org/10.5281/zenodo.18311342

Accepted: 19 January 2026 Available online: 20 January 2026

Abstract

This paper depicts about a smart image-recognition system for a robotic arm that combines OpenCV and Convolutional Neural Network (CNN). The objective of this paper is to move objects by using visual input and allow the robotic arm to identify and move objects using visual input. A camera captures real-time images of the environment and serves as the robot's eye. Features are extracted by first processing these images using OpenCV, including tasks such as filtering, contour detection. A trained CNN model then analyses this processed information, classifying and recognizing objects accurately. With the object identified, the robotic arm takes the required action, such as picking or sorting items based on predefined instructions. The high precision and reliability achieved by the integration of OpenCV and CNN allow the system's op eration even in varying lighting and background conditions, reducing the need for human intervention and increasing the efficiency of automated systems. The method proposed is applicable in many fields, from smart manufacturing and industrial automation to medical robotics and agriculture. The paper thus demonstrates how the combination of computer vision

with CNN can make robotic arms more flexible, capable of performing complex tasks and intelligent in changing environments.

Keywords: RoboDK, OpenCV, Convolutional Neural Network, Arduino UNO, USB Camera, Object Recognition.

1. Introduction

Robotic arms are used for prevalently in Industries starting from the 1960's, with their first industrial robot arm. The first robotic arm name is The Unimate robotic arm. In the recent years, they have become an important part of the modern industrial automation systems, which ranges from manufacturing and logistics to the agriculture and healthcare. Nowadays, there is an increasing need for more reliable, independent and intelligent robotic arm which can be used widely. An effective way to make robotic arms more independent and intelligent is to have a more intelligent robotic arm, which are widely used for integration of vision-based systems. The system allows the robot to see its surroundings using a camera and it can be done by understanding it using image recognition and CNN which is used for accurately performing predefined tasks. A basic method of implementing such vision systems can be done by using OpenCV (Open Source Computer Vision Library) along with CNNs (Convolutional Neural Networks).

OpenCV provides a rich array of tools which can be used for image processing, such as filtering, detecting edges, tracking objects and also for recognizing colors, which helps the robotic arm to capture and analyze the images in real-time. The intelligence is required to identify objects accurately by using CNN. CNN is a type deep learning algorithm, which is used for interpreting the processed image, recognizing the patterns to classify the objects. By combining these two technologies for vision-based robotic arms which can be used to perform complex tasks such as sorting, picking and placing objects without human assistance. The goal of this vision-aided robotic arm is to build an automation system that can use visual input to make decisions. A camera captures real-time images, which are cleaned and processed using OpenCV, further analyzed by a CNN, to identify the object. The arm then performs its programmed task, such as picking and arranging items in certain manner.

This approach improves operations speed and reduces errors as the vision system uses real-time visual reference rather than fixed coordinates. It also reduces human intervention in repetitive

tasks or hazardous environments, helping to create a more efficient and safer automation design. A unique advantage of using a CNN based recognition is the flexible adaptability to changing conditions, such as inconsistent lighting, objects being in weird positions and even slight variations in the objects themselves. This is crucial in applications such as healthcare and agriculture, where variations and uncertainties are common.

1.1. Literature Survey

In this research paper, it combines augmented reality, brain-computer interface, and computer vision to control a robotic arm more efficiently. Using EEG signals and the visual inputs, users can be used to select and move objects without switching focus between devices. This system improves accuracy and makes robotic control easier for people with movement disabilities [1].This paper depicts about the robotic system that uses vision-based control to automatically by moving objects with a (2DOF) robotic arm. This system employs deep learning for using the object detection and it uses image-based visual serving for real-time control. It aims to improve the tracking accuracy, response time, and system adaptability while reducing the need for multiple sensors [2].This research focuses on creating an efficient path planning system for an apple-picking robotic arm that can avoid obstacles. It combines the Artificial Potential Field (APF) method with the A* algorithm to find the smooth and safe paths. The study helps improve the robot's ability to pick apples accurately and safely in complex environment [3].This study introduces a 3D robotic arm that can be controlled using both brain signals and computer vision. The system allows users to perform tasks like grasping or drinking by detecting brain activity and recognizing objects visually. It improves accuracy and reduces effort, helping people with movement disabilities interact with their environment more easily [4]. This paper introduces a lightweight model called the TDPPL-Net for detecting the tomatoes and locating picking points in real time. It helps harvesting robots, identify and pick tomatoes accurately using computer vision. The system works efficiently even on the low-cost hardware, making it suitable for real-world agricultural use [5].This research presents a new trajectory planning method for Delta robots to perform smooth and flexible pick-and-place tasks. The approach uses special mathematical curves to create adjustable motion paths for different working conditions. Simulations and experiments show that the method improves accuracy and adaptability in robotic operations [6].This study focuses on improving the speed and efficiency of robotic pick-and-place operations in footwear manufacturing. It uses a decision tree model to identify shoe parts and determine the best order for picking them up. The method helps reduce processing time and prevents possible collisions between robotic arms [7].This research presents a vision-

based system that allows users to control the Dexter ER2 robotic arm using hand gestures. Instead of traditional teach pendants, the robot responds to intuitive finger and hand movements captured by a camera. This method makes robot operation easier, faster, and more user-friendly for various tasks [8].

In this paper, the section 2 represents the Working of the Arduino UNO in the robotic arm and it also gives a detailed working of its block diagram, circuit diagram and its coding. The section 3 of this paper depicts the working of the Robotic Arm using a simulation software called RoboDK and gives its working method in detail. The section 4 discusses about the working of the Object Recognition in the Robotic arm using a USB Camera. The section 5 tells about the results and discussions of this paper. The section 6 represents the conclusion of this paper.

2. Arduino In Robotic Arm

2.1. Block Diagram Of The Robotic Arm

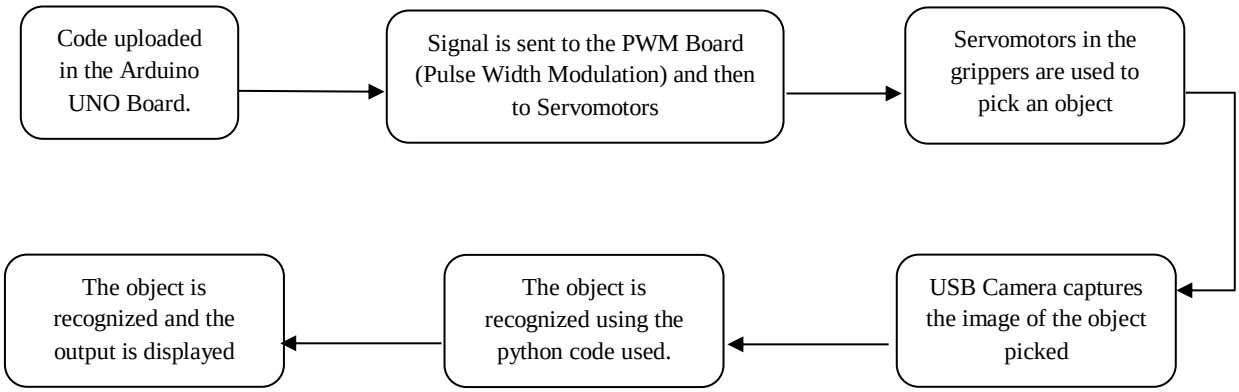

Figure.1. Block Diagram of the Robotic Arm

This block diagram in fig. 1 represents the basic steps in the working of the paper. Here, the code is dumped in the Arduino UNO board. The code must be compiled without any errors and dumped in the UNO Board by selecting the ports. The signal is sent to the PWM Board through the jumper cables. Then, they are sent to the Servomotors to have the movement in the parts of the robotic arm such as base, elbow, arm and gripper. The gripper is used to pick an object, and the USB Camera fitted above the gripper is used for object recognition which can be used for live capture of the image. The code used here is used to detect and recognize the object which has been sent from the USB Camera to the Laptop. They are then identified based on the image's intensity saturation, shape, size etc.

2.2. Schematic Diagram

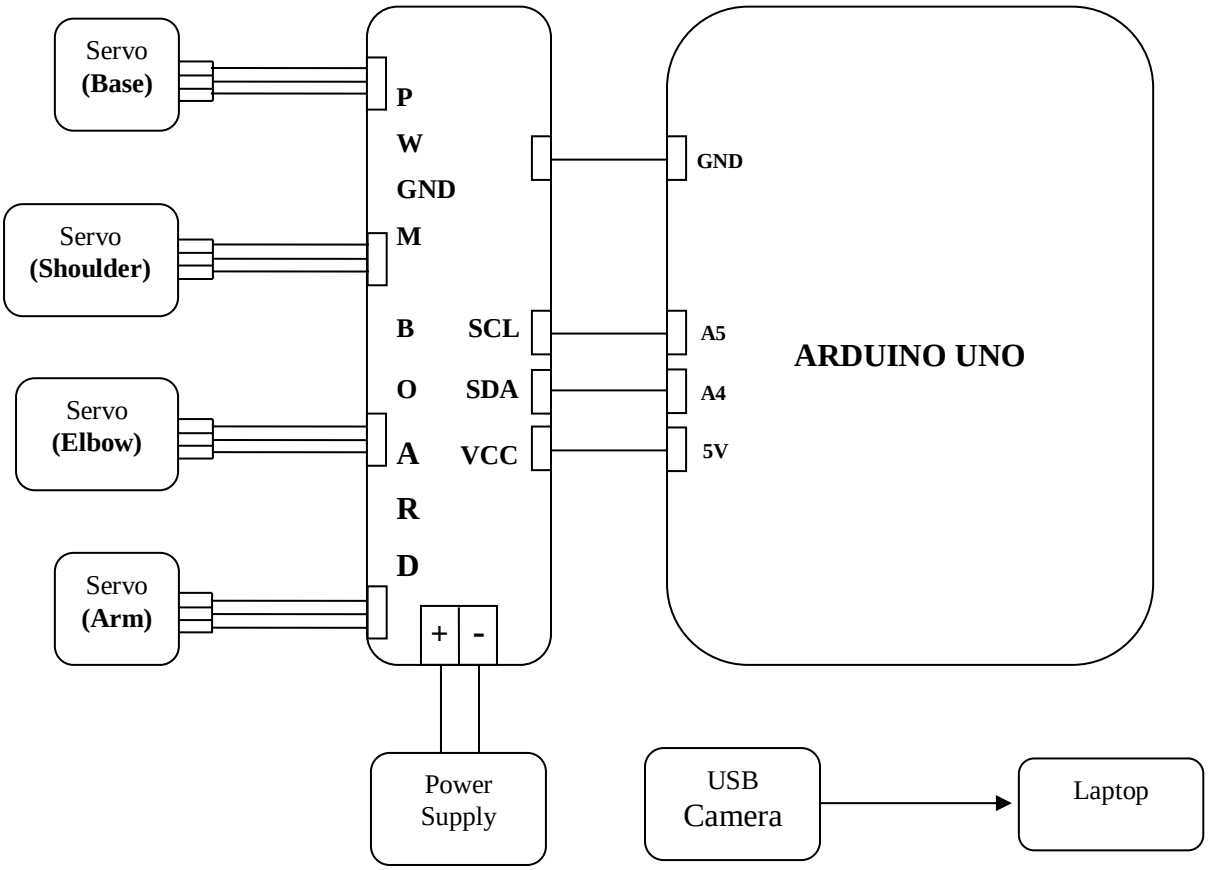

Figure. 2. Schematic Diagram of the Robotic Arm

Here the Circuit Diagram in fig. 2 represents the clear working model of the robotic arm. Here the basic function of the arm is to change the position or move from one place to another and they are used for picking up of objects. Here they use Servomotors which have a 180-degree rotation of any object. These are fitted in the robotic arm's parts, they enable for free movement and rotation of the parts of the robotic arm.

They are then connected to the PWM board which enables easy handling of connection from the servomotors to the Arduino UNO. Then the pins from the PWM board are connected to the Arduino Uno Board. They are connected to the analog pins 4, 5 and to the GND and 5V. The Power supply from the battery is then connected with the PWM Board Power supply boards.

Here the battery sends the power supply, using that the robotic arm works. The Arduino has coding which is responsible for the rotation and movement of the robotic arms. According to the coding dumped in the Arduino board they work, the electrical signals from the Arduino

board is sent to the PWM board. They are used to convert the electrical signal to Mechanical movements. The Robotic arm works with the help of the Servomotors present there. Then, the USB Camera placed in the top of the gripper is used to capture the image of the object and detect the image. The image can be found based on the coding given in the robotic arm. The image is categorized based on the size, shape, structure, colour, intensity, model, etc. Then it gives the output according to the object recognized using the USB Camera and the coding.

2.3. Working Of The Robotic ARM

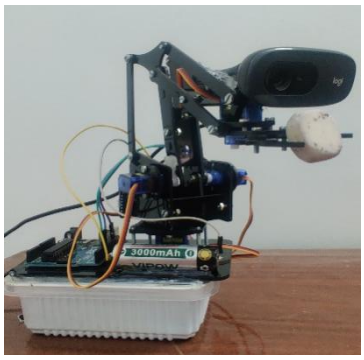

Figure.3. Photo of the Robotic Arm

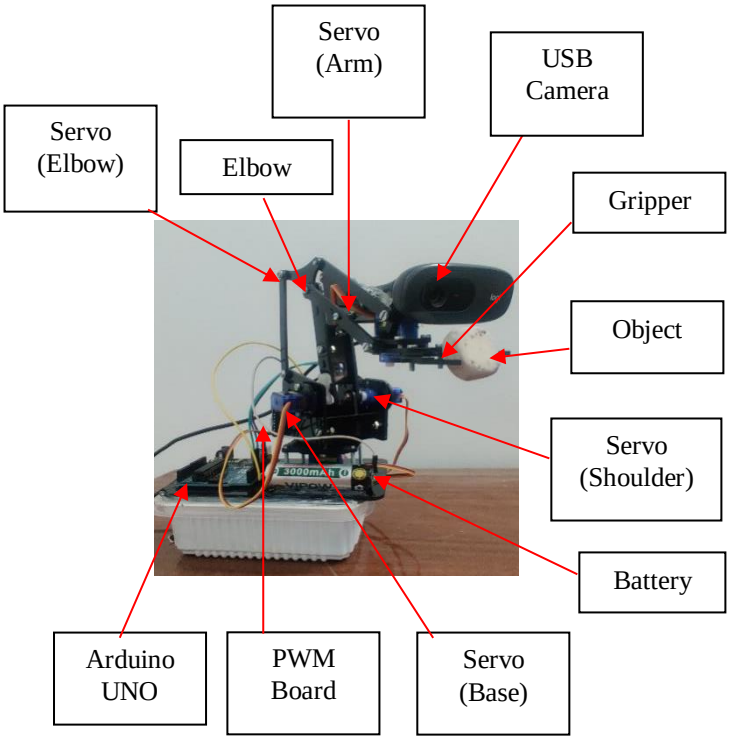

Figure.4. Components of the Robotic Arm

The fig. 3 represents the image of the Robotic arm integrated for object recognition using CNN and OpenCV. The pick and place robotic arm is a system that is used for the automatic lifting, transferring, picking and releasing objects accurately. It has a combination of electronic control, mechanical movement, and software programming in the Arduino UNO. The fig. 4 shows the main parts and the components used in this paper. The main components of this paper includes the Arduino UNO microcontroller, the PCA9685 PWM servo driver board, and servo motors. They work together to perform the pick and place operation in sync. The Arduino UNO plays an important as the control unit. It sends signals to the servo driver board. The PWM board produces a precise Pulse Width Modulation (PWM) signals needed for each servo motor. The PCA9685 board is an I2C-based PWM controller that allows up to 16 servomotors to operate at

the same time by using only two communication pins which are SDA and SCL. This setup makes the system efficient and simplifies wiring compared to connecting directly to Arduino pins. Each servo motor in the robotic arm is linked to a specific joint. Here they have a 4 Degree of Freedom which is also known as 4 DOF.

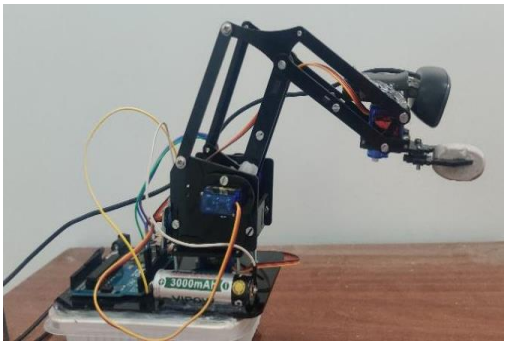

Figure.5. Movement of the Base

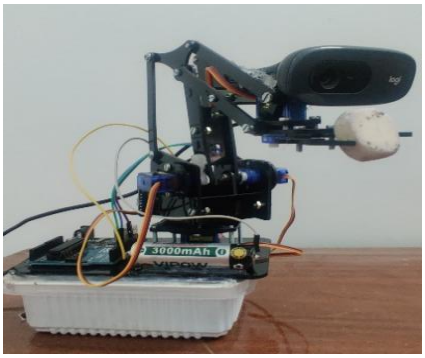

Figure.6. Movement of shoulder and elbow

The base servo in fig. 5 allows the arm to rotate in the left or right direction for horizontal movement. The shoulder and elbow in the fig. 6 servomotor control the vertical position of the arm, it helps the arm to reach different heights and distances. The gripper servo manages the end effectively, so that they open and closes to pick up or release objects easily. Together, these servos replicate the movement of a human arm, performing tasks with flexibility and precision. The process begins with the Arduino establishing communication with the PWM board. Each servo first moves to its home position, ensuring the arm starts in a neutral and balanced way. Once powered on, the program stored in the Arduino's memory runs through instructions to perform the pick and place cycle. The base servo rotates the arm toward the object's location. Then, the shoulder and elbow servos lower the gripper near the object. The gripper servo closes, firmly holding the object using mechanical friction.

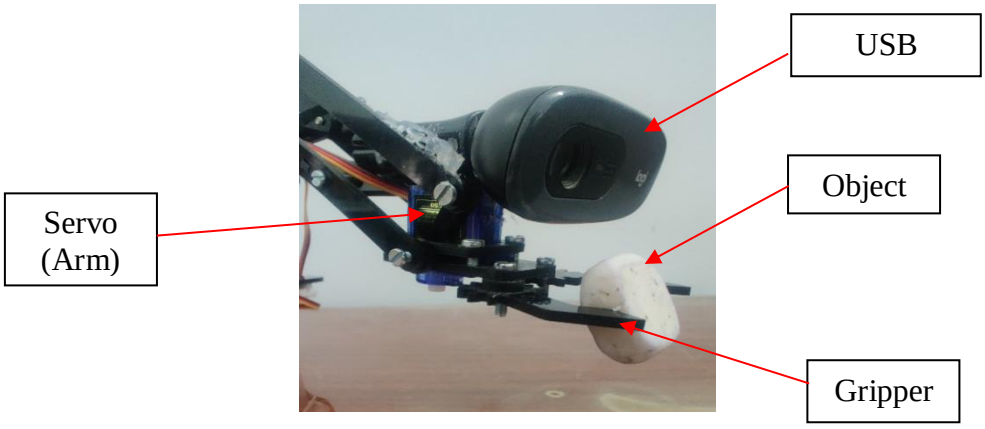

Figure.7. Movement of the Gripper

After gripping the object, the arm lifts it by adjusting the shoulder and its elbow angles. This safely raises the object from the surface. The base servo then rotates to align the arm according to the target location. Once aligned, the shoulder and the elbow servos of the arm are used to lower the controlled movements from the top to bottom. Finally, the gripper servo opens and closes, releasing the object at the right spot. After placing the object, all servos return to their initial normal positions, getting the system ready for the next cycle. This movement can be represented in the fig. 7 the servo operation depends on pulse width modulation. Each servo motor gets a PWM signal made of periodic electrical pulses. The width of these pulses determines the angle of the servo shaft. The PCA9685 module generates these precise signals for each servo channel, ensuring smooth and synchronized movement. The Arduino sends position commands via I2C, while the PWM board manages timing pulses, easing the workload on the microcontroller. Smooth motion and accurate control happen by gradually changing servo angles with small delays. This prevents sudden movements and mechanical strain, enhancing the pick and place task's accuracy. The system's reliability relies on a stable power supply and careful calibration of servo angles during assembly. Logical programming oversees the entire operation, which can improve further by adding sensors or computer vision systems for automatic object detection and sorting.

2.4. Code In Arduino UNO

In this paper, we use Embedded C code for the working of the arm. The code is dumped in the Arduino Uno Board using the Arduino IDE software. Here we use Adafruit library for easy working of the robotic arm. To dump the code, we have to select the Board as “Arduino UNO” from the Tools option present in the software. After the Selection of the board, we have to select the port from the same tools section. Before that we have to connect the Arduino UNO board to the software present in the Laptop using the Data Transfer Cable. After they are connected we have to first compile the program to check if there is error present in the coding. If there is no error we can easily upload the code to the board. The port for uploading the code must be selected from the tools section present at the top of the coding page. After the selection of port, we can easily upload the code to the Arduino UNO board and this can be done using the data transfer cable. The code for the movement of the parts of the robotic arm. They also used for the rotation and change of direction of the base, shoulders, elbow and arm of the robotic arm. Here they give code for the Grippers to open and grasp the object placed. Here the code gives command to the gripper to catch the object, to the elbow and the shoulders to move in vertical

directions like front and back movement, and to the base servo for the movement of the arm in horizontal direction like right and left. The code is also used for Arduino Board also to send the signals form it to the PWM board and finally it converts them for the robotic arm to work.

3. Robotic Arm System Simulation

The below figures represents the simulation of vision-based robotic arm system developed using RoboDK software to demonstrate automated object detection, sorting, and placement. The *fig. 8* shows the overview of the simulation of the robotic arm. The setup includes two UR10 robotic arms, conveyor system, camera sensors, and pallets for handling objects. The aim of this simulation is to show how industrial robots can be integrated with vision systems to perform pick-and-place operations efficiently and accurately.

The simulation of the robotic arm is developed by the RoboDK software which is one of the easiest software to do the paper. This helps us to perform the manual operations done in the paper by using a software. They are very easy to use and are very cost effective. The type of the robotic arm can be chosen easily and perform its operations. Here the conveyor system is used to perform the function for moving the objects while the robotic arm can be used for pick and place of it.

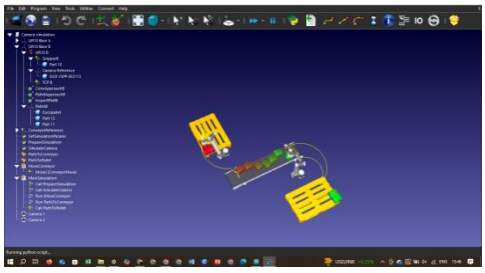

Fig. 8 Overview of the Simulation

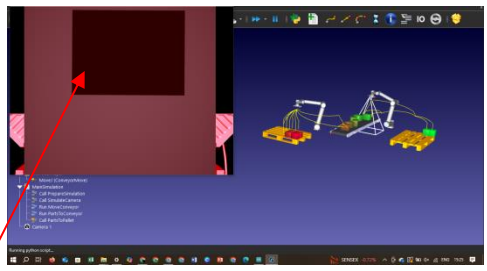

Fig. 9 Output of the Simulation

Output

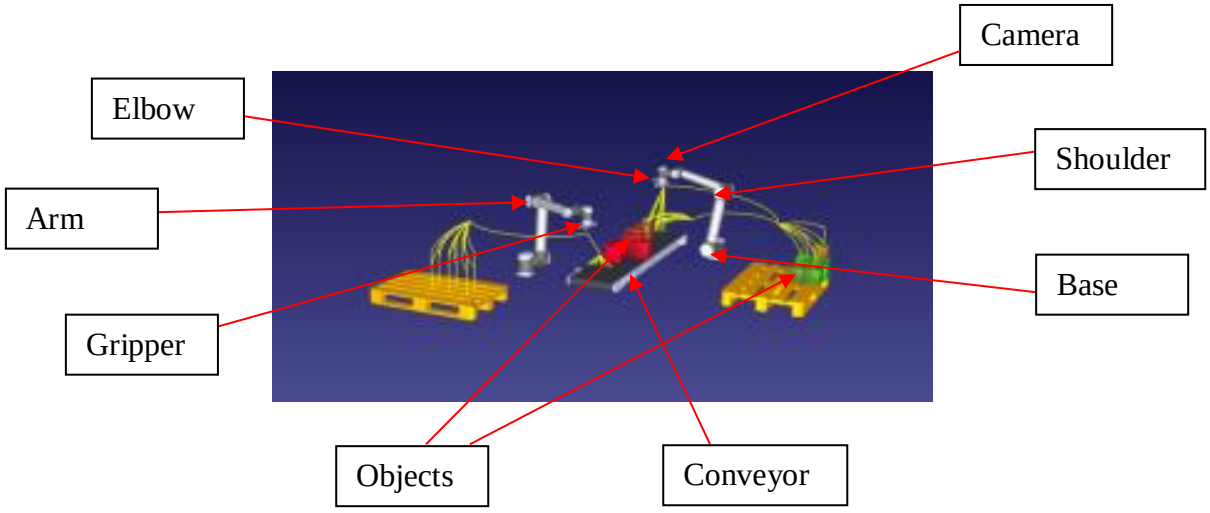

Fig. 10 Components used in Simulation

3.1. System Overview Of The Robotic ARM

The fig. 10 represents the components used in the project. The workspace consists of two UR10 robotic arms placed near a conveyor belt. One robotic arm is responsible for picking objects from the conveyor, while the second arm places them onto the pallet. A camera system (using a simulated SICK VSPP sensor) is positioned to capture real-time images of objects moving on the conveyor. The camera identifies the position, orientation, and type of objects using image processing algorithms, which guide the robot's motion.

3.2. Simulation Process

The first process in this simulation is to detect the object. The camera is used for continuous monitoring of the conveyor using the camera simulation function. It is used for identification and detection of each part's location and orientation using vision based input. The second process of the simulation is to have the movement of the conveyor. The conveyor is used to move the object from one place to another using the script known as the move conveyor script. When this object reaches the pick-up point of the conveyor, the system pauses for some time for having an inspection in it.

The third part of the simulation process is to do the pick and place operation. The detected object from the conveyor is detected by UR10 arm. The motion path is predefined by using the Approach B and Inspect Part B routines, ensuring smooth and accurate arm movement. The

fourth process of this simulation is the process of palletizing. The second UR10 arm performs palletizing placing the picked parts on their respective pallet positions (Parts to Pallet routine). Each object is positioned based on its detected data.

The last process of the simulation is to perform the process of simulation execution. The simulation is executed using the main program sequence called as the Main Simulation. This includes commands such as Prepare Simulation which is used for Initializing the workspace and robot positions, Simulate Camera to Activate the vision system for object detection., then Run Parts To Conveyor which is used to Start the conveyor motion., and finally to Call the Parts To Pallet which Executes the robotic placement sequence.

3.3. Outcome

The outcome of the simulation successfully demonstrates a fully automated robotic Arm using vision system. The Fig. 9 shows the output of the simulation of the project. The robot is used for identifying, picking, and placing different objects accurately without human intervention. It can be used to showcase the efficiency by combining computer vision and robotic motion control that can be applied in industrial automation, material handling, and smart manufacturing environments.

4. Object Recognition In Robotic Arm

4.1. Flowchart

The flow chart in the fig. 11 tells about the working of a vision-based object recognition system integrated with a robotic arm. The process starts by doing the system initialization, where the camera, pretrained CNN model, and communication with the Arduino are set up. The camera then captures the image frame from the environment, which is preprocessed by resizing, colour conversion, and normalization to match the input requirements of the convolutional neural network. The CNN performs the feature extraction by identifying the visual patterns such as edges, shapes, and textures, and then it classifies the object by comparing these features with learned patterns from training data. After classification, a decision is done to check whether the detected object matches the predefined target object.

If the target object is not detected, the system continues capturing and analysing new frames in a continuous loop. If the target object is detected, a control command is sent from the Python program to the Arduino through serial communication. The Arduino receives the command and

activates the servo motors to perform the required movement of the robotic arm, such as a pick-and-place operation. Thus, the flowchart represents the complete integration of image capture, CNN–based object recognition, decision making, and hardware actuation in an intelligent robotic system.

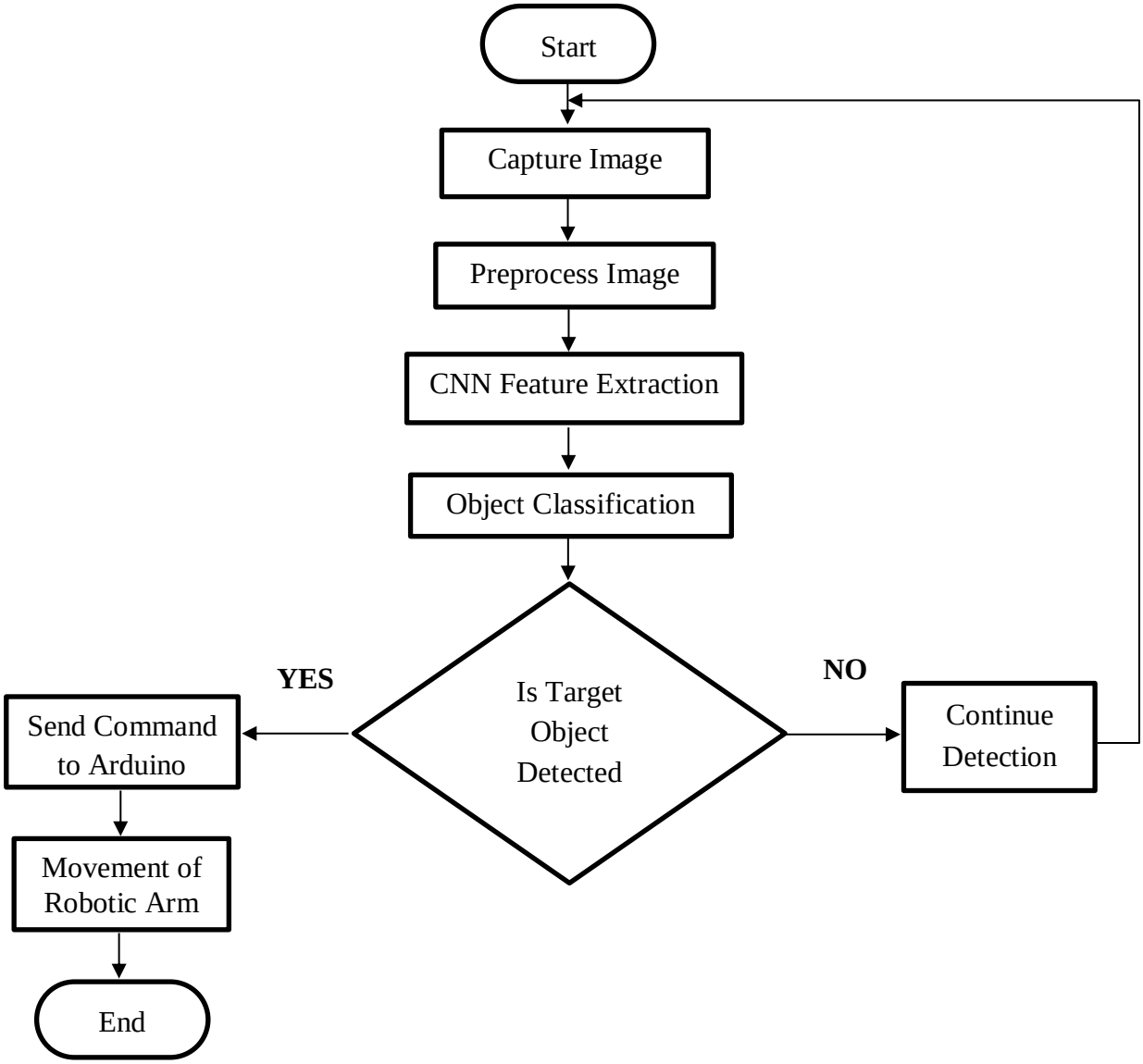

Figure.11. Flowchart of the Object Recognition in Robotic Arm

4.2. Coding In Object Recognition And Explanation

The object recognition code works by using the computer vision and pretrained convolutional neural networks (CNNs). Another important goal of the paper is used for identification of the objects present in an image or video frame and display their names along with confidence values. The code used here is used for implementing an object recognition in a part of the paper using a pretrained convolutional neural network (CNN), along with OpenCV for image and video processing. The program works in two modes: static image recognition and real-time object recognition using a USB camera.

First the Libraries for image recognition are installed. TensorFlow library is used for loading and running it, while the OpenCV is used for reading images and to capture video from a camera. NumPy Library is used for array manipulation, and Matplotlib Library is used for displaying the images. Here, the preprocess_input () prepares images in the format expected by the model and the decode_predictions () converts the model's numerical output into human-readable class names and confidence scores.

The MobileNetV2(weights="image net") is a code in it which is used for loading a pretrained CNN that has a trained ImageNet dataset. It consists of more than one million images across 1000 object categories. Since, the model is already trained, no training process is required.

During real-time, the image is captured, the camera continuously captures the frames of the image in a fixed rate. Each frame are set as an individual image and are passed to the code of object recognition pipeline. To continue this process the image pre-processing is needed to be done. First, Resizing is done in every frame of the image and they are resized to 244 x 244 pixels to get the required input. Then, the second part is for Colour Conversion. Here the OpenCV is used to recognize the image in BGR (Blue, Green and Red) Colour format to get the proper image for input. According to the colour format the image is converted.

The next important step in the Image Preprocessing is Batch Dimension Addition. Here the Images captured and processed are sent to the CNN Dataset and then, they are split into batches. Even for a single part of an image processing must be done and it must be reshaped to fit in the dataset. The final step in this step is Normalization. Here, the pixels are normalized and are distributed for getting the output.

There are four parts of layers in which the image is sent. These layers can also be known as the network architecture. The main four layers are early layers, Middle detect pattern layer, Deeper detect high-level feature layer, and the Final Layer is used to show the output for the image of the object captured in the USB Camera. The Early Layers are used in the Edges and Corners of the image. The Middle layer is used to find the shape and texture of the image. The deep layer is used to identify the parts of the object. The final layer is used to show the probabilities of the output of the image captured.

After the step of preprocessing, the image is passed to the CNN by using the predict () function. The output of the model is a vector of probability values, where each value here represents the likelihood of the image of the object which will be belonging to a specific class. The decode_predictions() function maps these numerical probabilities to human-readable object labels (for example, “plastic cover”, “laptop”, or “tap”) with their confidence scores. The system typically selects and prefers the first 3 prediction based on its probability.

The detected object name and confidence percentage are overlaid on the video frame using OpenCV. This visual feedback allows the user to see what object the system is identifying in real time. The system captures an image using a camera, preprocesses it, and passes it through a pretrained CNN Model. The network extracts features and classifies the object based on learned patterns from the ImageNet dataset, providing accurate object recognition in real time.

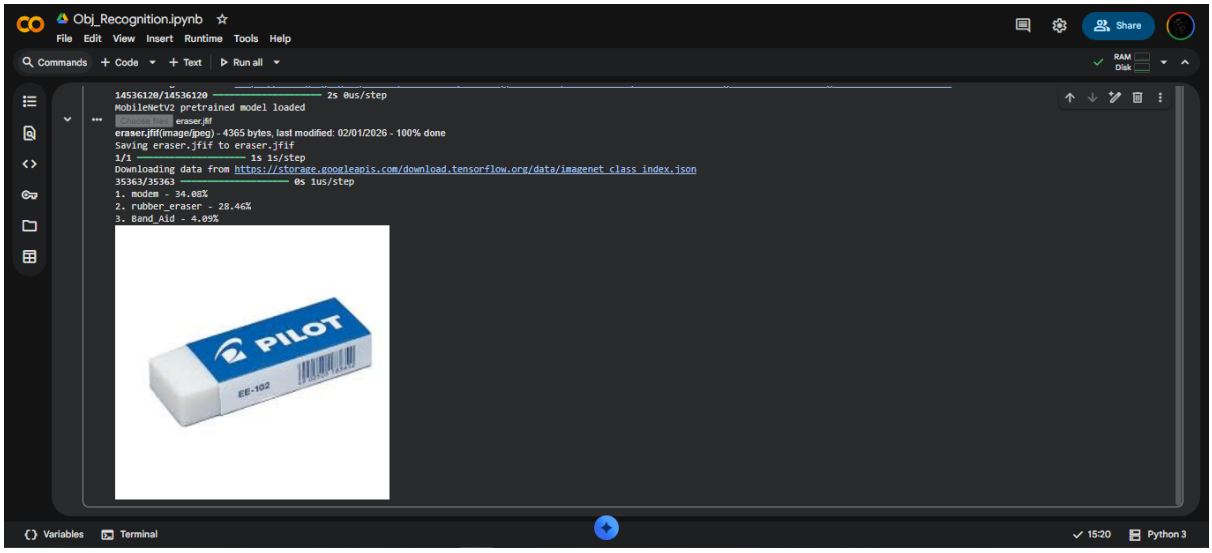

Figure.12. Output of object Recognition in Robotic Arm

Communication between Python and Arduino can be achieved by using the serial communication over USB. Python uses the pyserial library to send data through a specific COM port in the Arduino UNO at a defined baud rate. The most common Baud rate used here is 9600. When Python sends a command, it is transmitted as a byte stream through the USB cable. In the Arduino, the microcontroller continuously listens to the incoming serial data. The baud rate used in Arduino must match the baud rate used in Python for ensuring correct data reception.

The Arduino program is written in Embedded C Language. They use serial functions such as Serial.available () and Serial.read () to check for and read incoming commands. Once a command is received, the Arduino UNO compares it with predefined values. Each command corresponds to a specific motor action of the robotic arm. For example, when the Arduino receives the 'P' command, it executes a function that controls the servo motors to perform a pick-and-place operation. The Arduino does not perform any image processing or object recognition, it only focuses on executing physical movements based on the received instructions. This system successfully works with the embedded systems by combining Python-based object recognition and Arduino-based motor control. Python handles computationally intensive tasks such as CNN inference, while Arduino executes real-time hardware actions. Serial communication acts as a reliable bridge between the two, making the system suitable for intelligent robotic applications such as automated sorting, pick-and-place robots, and smart manufacturing systems. The output of the paper is shown in the fig. 12

5. Results and Discussion

In this paper elaborates the working and implementation of this paper using CNN and OpenCV. The paper combines the real-time image acquisition, object classification using the captured image, and finally for a microcontroller used for hardware control operations. Image Processing is done using the python program while the hardware control movements of the robotic arm that can be done with the help of the Embedded C Program which is dumped in the Arduino UNO. In other words, high-level programming is done using python and the low-level programming is done using the Embedded C. The output of the paper is represented by the fig. 12

A Pretrained model is used for the identification of objects which is done using the Convolutional Neural Network. This has better accuracy and training for the paper. It helps in resizing, colour conversion, normalization of an image. It helps in eliminating high risk factors to the paper. The USB Camera is used to enable the continuous real-time image acquisition of

the object which is picked by the robotic arm. Communication between the vision module and the embedded controller can be achieved using the serial communication, by providing simple and robust interface between them for command transmission. The microcontroller interprets the received commands and generated appropriate control signals to drive the servo motors of the robotic arm. This modular system architecture effectively separates computationally intensive tasks from real-time hardware control, thereby improving system efficiency, scalability, and reliability.

6. Conclusion

In this paper, the vision-based robotic arm system provides more intelligent applications. It is useful by integrating real-time image processing with embedded system control and it enables accurate object recognition and autonomous robotic actions. Pre trained model helps to easily eliminate the need for the complex model training. This helps us to have a cost-effective and efficient model for working. The serial communication enables a simple and robust interface between software and hardware control. It also provides more reliable hardware control of the robotic arm. The architecture of the company allows us to have easy upgrades, scalability, maintenance and enabling future expansion without more complex designs. It also helps us to reduce the manual labour and increase the productivity. Overall, the paper represents a more practical, flexible, and efficient solution for the robotic arm integrated with object recognition using CNN and OpenCV, making it easy use of applications in industrial automation, smart automation, smart manufacturing and easy material handling system.

Acknowledgement
The authors have no acknowledgements to declare.

Funding
This study has not received any funding from any institution/agency.

Conflict of Interest/Competing Interests
No conflict of interest.

Data Availability
The raw data supporting the findings of this research paper will be made available by the authors upon a reasonable request.

REFERENCES

[1]. Xiaogang Chen, Member, IEEE, Xiaoshan Huang, Yijun Wang, Member, IEEE, and Xiaorong Gao, Member, IEEE " Combination of Augmented Reality BasedBrain-Computer Interface and Computer Vision for High-Level Control of a Robotic Arm",2020.

[2]. Umesh Kumar Sahu[1], Mebin K. S., Abhinav K., Muhammed Muzammil P, Ankur Jaiswal1 Umesh Kumar Yadav2 & Santanu Kumar Dash, "Autonomous object tracking with vision based control using a 2DOF robotic arm",2025.

[3]. Min Zhuang, Ge Li, And Kexin Ding "Obstacle Avoidance Path Planning for Apple Picking Robotic Arm Incorporating Artificial Potential Field and A* Algorithm",2023.

[4]. Yajun Zhou, Tianyou Yu, Member, IEEE, Wei Gao, Weichen Huang, Zilin Lu, Qiyun Huang, and Yuanqing Li, Fellow, IEEE "Shared Three-Dimensional Robotic Arm Control Based on Asynchronous BCI and Computer Vision" Emb Leee transactions on neural Bystems and Rehabilitation Engineering, vol. 31, 2023.

[5]. Chengyuan Song, Kai Wang, Chao Wang , Yanan Tian ,Xinjie Wei, CuijianLi ,QilinAn,And Jian Song "TDPPL-Net A Lightweight Real-Time Tomato Detection and Picking Point Localization Model for Harvesting Robots",2023.

[6]. TingtingSu, Xu Liang, GuotaoLi, Member, IEEE, and Zeng-Guang Hou, Fellow, IEEE "Trajectory Planning for Interactive Pick-and-Place Operations of Delta Robots", 2025.

[7]. Jorge Borrell Mendz, Carlos Perez-Vidal, (Member, IEEE), Jose Vincent SegurHeres, And Juan Jose Perez-Hernandez "Robotic Pick and Place Time Optimization Application to Footwear Production", 2020.

[8]. Abhishek S.1 , Yash S. Jogi 1 , Umesh Kumar Sahu 1 , Santanu Kumar Dash, And Umesh Kumar Yadav, "Teach Pendant at Fingertips Intuitive Vision-Based Gesture-Driven Control of Dexter ER2 Robotic Arm",2025.