Real-Time Object Classification: Shape and Size Analysis with Computer Vision on Arduino

Mini Project Report

submitted in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE and **ENGINEERING**

(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

By

K. Surendra	(21341A4226)
V. Uday Kiran	(21341A4255)
V. Ramya	(21341A4257)
K. Surya Venkat	(21341A4225)
V. Ruthvik	(21341A4256)

Under the Guidance of

Dr. CH. Sekhar

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

GMR Institute of Technology

An Autonomous Institute Affiliated to JNTU-GV

GMRIT Training Tomorrow's Engineers Today

May 2024

GMR INSTITUTE OF TECHNOLOGY

Department of Computer Science and Engineering

(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

CERTIFICATE

This is to certify that the project report entitled "**Real-Time Object Classification: Shape and Size Analysis with Computer Vision on Arduino**" is the bonafide record of project work carried out under my supervision by **K. Surendra (21341A4226), V. Uday Kiran (21341A4255), V. Ramya (21341A4257), K. Surya Venkat (21341A4225) and V. Ruthvik (21341A4256)**, during the academic year 2023-2024, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering(Artificial Intelligence & Machine Learning) of Jawaharlal Nehru Technological University, Vizianagaram. The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

Head of the Department

Dr. K. Srividya

Associate Professor & HOD
Department of CSE(AI&ML)
GMRIT, Rajam.

Signature of Project Guide

Dr. CH. Sekhar

Associate Professor
Department of CSE(AI&ML)
GMRIT, Rajam.

ACKNOWLEDGEMENT

It gives us an immense pleasure to express deep sense of gratitude to my guide **Dr. CH. Sekhar**, Associate Professor, Department of CSE-Artificial Intelligence & Machine Learning of whole hearted and invaluable guidance throughout the mini project work. Without his sustained and sincere effort, this project work would not have taken this shape. He encouraged and helped us to overcome various difficulties that we have faced at various stages of our mini project work.

We would like to sincerely thank our Head of the Department, **Dr. K. Srividya**, Associate Professor, Department of CSE-Artificial Intelligence & Machine Learning, for providing all the necessary facilities that led to the successful completion of our project work.

We would like to take this opportunity to thank our beloved Principal, **Dr. C. L. V. R. S. V Prasad**, for providing a great support to us in completing our mini project and for giving us the opportunity of doing the project work.

We would like to thank all the faculty members and the non-teaching staff of the Department of CSE-Artificial Intelligence & Machine Learning for their direct or indirect support for helping us in completion of this mini project work.

K. Surendra (21341A4226)

V. Uday Kiran (21341A4255)

V. Ramya (21341A4257)

K. Surya Venkat (21341A4225)

V. Ruthvik (21341A4256)

ABSTRACT

Now-a-days more people are shopping in online, at the time of festivals and occasions there is a higher demand for products to store in warehouses. This puts extra pressure on employees who have to sort and package items, potentially causing delays in deliveries. The focus of our project is on analyzing the shape and size of objects using computer vision, specifically on the Arduino platform. By incorporating Arduino, we're making this technology more accessible for real-world applications. By combining Arduino with IoT (Internet of Things) technologies, the system can be connected to a network for remote monitoring and control. This facilitates centralized management of multiple warehouse locations. The potential of using data collected by the Arduino-based system to gain insights into key aspects of warehouse operations, allowing for better decision-making and optimization of resources. Some advantages of this project include streamlining warehouse operations, reducing human error in sorting and packaging and reduces time consumption.

Keywords: Object Detection, Object Classification, Computer Vision, Arduino, IoT.

Table of Contents

ABSTRACT i

1.INTRODUCTION **1**

1.1 OPEN CV 1

1.2 OBJECT DETECTION 2

1.3 IOT 3

1.4 E-COMMERCE 4

2.LITERATURE SURVEY **6**

2.1. LITERATURE SURVEY OF DIFFERENT RESEARCH PAPERS 6

2.2 COMPARISION TABLE OF DIFFERENT RESEARCHES 11

3.PROBLEM STATEMENT 17

3.1 EXISTING MODEL 17

3.2 PROPOSED MODEL 17

4.METHODOLOGY AND DESIGN 19

5.REQUIREMENTS SPECIFIED 22

5.1 SOFTWARE 22

5.2 HARDWARE 23

5.3 IOT 26

6.IMPLEMENTATION AND CODE 27

7.RESULTS 35

CONCLUSION

REFERENCES

1. INTRODUCTION

1.1.Open CV

OpenCV, which stands for Open Source Computer Vision Library, is an open-source computer vision and machine learning software library. It provides a wide range of tools and functions for image and video processing, including tasks such as:

- Image Processing: Operations like image filtering, transformation, enhancement, and manipulation.
- Object Detection and Recognition: Identifying objects or specific features within images or video streams.
- Feature Detection and Description: Locating and describing keypoints or features in an image.
- Image Segmentation: Dividing an image into multiple segments to simplify its representation or facilitate analysis.
- Camera Calibration: Determining the intrinsic and extrinsic parameters of a camera to remove distortion and project images accurately.

Machine Learning Integration: OpenCV provides interfaces to various machine learning frameworks, allowing integration of trained models for tasks such as classification, regression, and clustering.

Key concepts and components of OpenCV:

- Mat: The basic data structure in OpenCV is the `Mat` class, which represents a matrix or multi-dimensional array. It is used to store images, with each element representing a pixel value.
- Image Processing: OpenCV provides a rich set of functions for basic image processing tasks, including color space conversion, filtering (e.g., blurring, sharpening), geometric transformations (e.g., resizing, rotation), thresholding, and morphological operations.
- Feature Detection and Description: OpenCV includes algorithms for detecting and describing keypoints in images, such as the popular SIFT (Scale-Invariant Feature

Transform) and SURF (Speeded-Up Robust Features) algorithms. These keypoints can be used for tasks like object recognition and image stitching.

- Object Detection: OpenCV supports various object detection techniques, including Haar cascades and deep learning-based approaches like Single Shot MultiBox Detector (SSD) and You Only Look Once (YOLO). These techniques enable the detection of objects like faces, pedestrians, and vehicles in images or video streams.
- Deep Learning Integration: OpenCV provides integration with deep learning frameworks such as TensorFlow and PyTorch, allowing users to deploy and run pre-trained deep learning models for tasks like image classification, object detection, and semantic segmentation.
- Camera Calibration: OpenCV includes functions for camera calibration, which involves estimating camera parameters like focal length, principal point, and distortion coefficients. Calibration is essential for tasks like 3D reconstruction and augmented reality.
- Video Analysis: OpenCV supports video processing tasks such as video capture, playback, and analysis. It includes functions for reading and writing video files, as well as analyzing video streams frame by frame.

Overall, OpenCV is a powerful tool for a wide range of computer vision tasks, from simple image processing to complex object detection and recognition. Its extensive documentation and active community make it a popular choice for both academic research and industrial applications.

1.2.OBJECT DETECTION

Object detection in computer vision involves locating and classifying objects within an image or a video sequence. It's a fundamental task with numerous applications, from autonomous driving and surveillance to augmented reality and robotics. OpenCV provides various techniques and algorithms for object detection

Object detection using contours is a computer vision technique that identifies and localizes objects within an image by detecting their boundaries. The process involves several key steps.First, the image is converted to grayscale, which simplifies the subsequent analysis

by reducing the complexity from three color channels to one. Next, noise is often reduced using techniques such as Gaussian blurring, which helps to smooth the image and minimize the impact of small artifacts.Following this, edge detection is applied, commonly using the Canny edge detector, to highlight the edges within the image. These edges represent potential contours, or continuous curves that bound or outline the shapes of objects in the image.The contours are then extracted using algorithms such as the OpenCV findContours function, which identifies and organizes the continuous curves of pixels. These contours can be further analyzed to filter out irrelevant ones based on their size, shape, or hierarchy.The resulting contours are used to create bounding boxes or masks around detected objects, enabling localization and classification. This method is particularly effective for detecting well-defined, high-contrast objects but may struggle with complex, overlapping, or low-contrast scenes. Despite these limitations, contour-based object detection remains a robust and efficient technique for many practical applications.

1.3.IOT

The Internet of Things (IoT) is a revolutionary paradigm that refers to the network of interconnected devices embedded with sensors, actuators, and software, enabling them to collect, exchange, and act on data without human intervention. It's a transformative technology with applications across various domains, from smart homes and cities to industrial automation and healthcare. Here's an in-depth overview of IoT:

Key Concepts:

- Connected Devices: IoT encompasses a vast array of devices, including sensors, actuators, wearables, appliances, vehicles, industrial machines, and infrastructure components, all connected to the internet.
- Data Collection: IoT devices collect data from their environment through sensors measuring parameters like temperature, humidity, motion, light, pressure, and more.
- Data Communication: Collected data is transmitted over networks (e.g., Wi-Fi, cellular, Bluetooth, LPWAN) to centralized servers, cloud platforms, or other devices for processing and analysis.

- Data Processing and Analysis: IoT platforms process and analyze the collected data to derive insights, detect patterns, and make informed decisions using techniques such as machine learning, artificial intelligence, and data analytics.
- Actuation: IoT devices can also act upon the data they collect by controlling actuators to perform actions, such as adjusting environmental conditions, activating alarms, or triggering responses in other connected devices.
- Device Layer: This layer consists of IoT devices equipped with sensors and actuators for data collection and actuation.
- Communication Layer: IoT devices communicate with each other and with backend systems through various communication protocols and networks.
- Middleware Layer: Middleware facilitates communication, data processing, and integration between IoT devices and backend systems.
- Application Layer: This layer includes IoT applications and services that utilize data from IoT devices to provide value-added functionalities and solutions.
- Communication Protocols: IoT devices use protocols like MQTT, CoAP, HTTP, and WebSocket for efficient and reliable data exchange.
- Interoperability Standards: Standards like OPC UA, MQTT, and DDS promote interoperability and seamless integration between heterogeneous IoT devices and systems.
- Security: Security is a critical aspect of IoT, encompassing authentication, encryption, access control, secure bootstrapping, firmware updates, and intrusion detection to protect IoT ecosystems from cyber threats.

1.4.E-Commerce

Certainly! E-commerce, short for electronic commerce, refers to the buying and selling of goods and services over the internet. It has become a significant part of the global economy, revolutionizing the way businesses operate and consumers shop. Here's a comprehensive overview of e-commerce:

History and Evolution: E-commerce traces its roots back to the 1970s with the development of electronic data interchange (EDI), which allowed businesses to exchange documents

electronically. The rise of the internet in the 1990s led to the emergence of online marketplaces and the first e-commerce websites, enabling consumers to purchase goods and services online. Over the years, advancements in technology, improvements in internet infrastructure, and changes in consumer behavior have fueled the growth of e-commerce, making it a dominant force in retail and commerce worldwide.

Types of E-commerce :

- Business-to-Consumer (B2C): In B2C e-commerce, businesses sell products or services directly to consumers through online storefronts, marketplaces, or digital platforms.
- Business-to-Business (B2B): B2B e-commerce involves transactions between businesses, where one business sells products or services to another business through online marketplaces, procurement portals, or electronic trading platforms.
- Consumer-to-Consumer (C2C): C2C e-commerce facilitates transactions between individual consumers, often through online auction sites, classifieds platforms, or peer-to-peer marketplaces.
- Consumer-to-Business (C2B): C2B e-commerce occurs when individual consumers sell products or services to businesses, such as freelancers offering their skills or expertise to companies through online platforms.

2. LITERATURE SURVEY

2.1. LITERATURE SURVEY OF DIFFERENT RESEARCH PAPERS

Sachin, C., Manasa, N., Sharma, V., & AA, N. K. (2019, November). Vegetable classification using you only look once algorithm. In *2019 International Conference on Cutting-edge Technologies in Engineering (ICon-CuTE)* (pp. 101-107). IEEE.

The paper discusses the use of the You Only Look Once (YOLO) algorithm for vegetable detection and classification, achieving an accuracy of 61.6 percent . The YOLO model is compared with other neural network models for object detection, showing better performance, accuracy, and processing time .YOLO is different from region-based algorithms as it uses a single convolution network for class probabilities and bounding box details.A solution for detection of hidden objects for classification of threat is presented, achieving a precision rate of 88.89.

Kakde, Y., Bothe, N., & Paul, A. (2019). Real life implementation of object detection and classification using deep learning and robotic arm. *Proceedings of Recent Advances in Interdisciplinary Trends in Engineering & Applications (RAITEA)*.

The paper mentions the use of 1000 images of apples and 800 images of oranges to train the object detection model. The classifier yielded an accuracy of up to 99.22%. The paper highlights the combination of deep learning concepts with Arduino programming as a framework that can be used to solve real-life problems. The paper suggests using a white background when capturing images to avoid background interference in object detection.

Kale, V. R., & Kulkarni, V. A. (2013). Object sorting system using robotic arm. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, *2*(7), 3400-3407.

The paper presents a smart approach for a real time inspection and selection of objects in continuous flow. The project involve sensors that senses the object's colour, size and sends the signal to the microcontroller. The microcontroller sends signal to circuit

which drives the various motors of the robotic arm to grip the object and place it in the specified location.

Poda, X., & Qirici, O. (2018). Shape Detection and Classification Using OpenCV and Arduino Uno. *RTA-CSIT*, *2280*, 128-136.

This project was developed in three main directions. First direction focuses on image processing techniques in order to detect and classify the regular shapes present in an image. The second direction focuses on how the output produced in the first stage will be transmitted in Arduino and how Arduino will interact with this output. The third direction is the construction of a hardware platform in which all the results obtained will be combined and will solidify the purpose of this research.

Abdullah-Al-Noman, M., Eva, A. N., Yeahyea, T. B., & Khan, R. (2022). Computer vision-based robotic arm for object color, shape, and size detection. *Journal of Robotics and Control (JRC)*, *3*(2), 180-186.

The proposed system demonstrates 80% accuracy in detecting colored objects and 100% accuracy in real-time size and shape recognition. The paper presents a computer vision-based robotic arm that can select and arrange objects based on their color, shape, and size. It utilizes the PixyCMU camera sensor for color detection and OpenCV image processing libraries in Python for shape and size identification.

Singh, T., Dhaytadak, D., Kadam, P., & Sapkal, R. J. (2016). Object sorting by robotic arm using image processing. *International Research Journal of Engineering and Technology (IRJET)*, *3*(4), 2395-0056.

The paper addresses the common hindrance faced by small and large scale industries, such as shortage of time and workers, leading to inefficient manufacturing. The use of robotics is proposed as a solution to achieve better production and income. The robotic arm, controlled by a microcontroller, is used to sort objects based on faults like missing drill holes or improper shape. The combination of image processing and the robotic arm enables quick sorting without the need for continuous surveillance, thereby increasing industry growth.

Sriramguru, A., Vishwa, A., Yogeshwaran, D., & Yogessh, B. S. (2020). Arduino based Automatic Material Sorting Machine Using Open CV and Python 2.7 IDLE.

The paper discusses the development of an automatic material sorting machine using image processing techniques and Arduino UNO microcontroller. Another related study mentioned in the literature survey is the development of an automated sorting machine for plastic recycling. The system uses shape analysis methods and achieved a high sorting accuracy of around 95% for different types of bottles. The paper also mentions the use of CATIA software for creating a model of the sorting machine and the analysis of the conveyor belt and frame using Ansys software. The material selection for the conveyor belt is Polyurethane.

Fan, Z., Li, Z., & Li, W. (2018). Object Detection and Sorting by Using a Global Texture-Shape 3D Feature Descriptor. ***arXiv preprint arXiv:1802.01116.***

The paper presents a global texture-shape 3D feature descriptor for object recognition and sorting in a sorting system. The proposed descriptor is an extension of the clustered viewpoint feature histogram (CVFH) with texture information, which improves the recognition performance for objects with similar geometrical information. The paper utilizes the Microsoft Kinect camera to capture point cloud images of real scenes and presents a method of segmentation as a preprocessing step for utilizing the proposed descriptor for object recognition.

Kumar, R., Lal, S., Kumar, S., & Chand, P. (2014, November). Object detection and recognition for a pick and place robot. In ***Asia-Pacific world congress on computer science and engineering*** **(pp. 1-7). IEEE.**

The paper focuses on the development of an image processing algorithm for object detection and recognition in a pick and place robotic arm system. The algorithm involves a feature extraction algorithm and two classifiers for object recognition and detection. The feature extraction algorithm achieved an accuracy of 83.6443% . The classifier achieved an accuracy of 89.33% upon cross-validation and 82.7162% for the overall system performance.

Baballe, M. A., Bello, M. I., Abdulkadir, Z., & Garbadiso, G. B. (2022). Study on Cabot's Arms for Color, Shape, and Size Detection. ***Global Journal of Research in Engineering & Computer Sciences*****,** ***2*****(2), 48-52.**

The paper discusses the use of robots in various fields, such as military combat and industrial production, to minimize labor costs and perform risky tasks. It highlights the use of computer vision-based robotic arms for tasks like picking and placing products with precision. The authors propose a computer vision-based automated system for sorting colors, detecting shape and size, and picking and placing objects using a robotic arm controlled by an Arduino Mega and servo motors. The system aims to reduce labor costs and increase productivity in industries. The paper mentions the challenges associated with safety approval for robots, as changes in tasks or tools may require a new safety certification. It also highlights the advantages of collaborative robots (Cabot's), such as flexibility, safety features, cost-effectiveness, and the ability to perform tasks faster with two arms.

Fadhil, A. T., Abbar, K. A., & Qusay, A. M. (2020, February). Computer Vision-Based System for Classification and Sorting Color Objects. In ***IOP Conference Series: Materials Science and Engineering*** **(Vol. 745, No. 1, p. 012030). IOP Publishing.**

The paper presents a mechatronic sorting system that uses image processing and a robotic arm to classify and sort color objects based on their color and geometry. The system uses a web camera to capture real-time images of the objects on a conveyor belt and processes the information using image processing techniques in EmguCV and Visual Studio software. The Canny edge detection algorithm is used for shape identification in image processing.

Intisar, M., Khan, M. M., Islam, M. R., & Masud, M. (2021). Computer Vision Based Robotic Arm Controlled Using Interactive GUI. ***Intelligent Automation & Soft Computing*****,** ***27*****(2).**

The paper presents the design and implementation of a robotic vision system operated using an interactive Graphical User Interface (GUI) application. The system is designed to be user-friendly, allowing beginners to operate the device with minimal

instruction. The application allows users to filter objects based on color, shape, and size using a Hue-Saturation-Value (HSV) mode color detection algorithm, shape detection algorithm, and size determining algorithm.

SNEHA, S., & NAYANA, R. DESIGN AND IMPLEMENTATION OF ROBOT ASSISTED ARDUINO BASED OBJECT RECOGNITION AND SORTING.

The research paper integrates advanced sensors to detect objects on conveyor belts and computer vision algorithms for color identification, enhancing automation efficiency. The system addresses challenges like synchronizing robotic arms with conveyor systems, requiring precise timing and control for seamless operation. The system utilizes Arduino for creating an intelligent robot capable of identifying object colors and sorting them accurately, showcasing precision and flexibility in handling tasks. The success of the system opens doors to new possibilities in streamlining workflows, reducing human intervention, and increasing overall efficiency, hinting at the potential for more sophisticated and adaptable robotic systems in the future. The system employs a CNN for accurate object detection, leveraging deep learning and image processing for efficient object recognition and handling.

Lahoti, J., Sn, J., Krishna, M. V., Prasad, M., Rajeshwari, B. S., Mysore, N., & Nayak, J. S. (2024). Multi-class waste segregation using computer vision and robotic arm. *PeerJ Computer Science*, *10*, e1957.

Various studies have focused on waste segregation using computer vision and robotic arms, showcasing advancements in the field. Zhou et al. (2021) developed a dataset for trash detection and utilized YOLOV4 with GhostNet for improved accuracy and efficiency. Padalkar, Pathak Stynes (2021) used Scaled-Yolov and EfficientDet for plastic waste sorting, achieving high accuracy with Scaled-Yolov4-CSP. Sai Sushanth, Jenila Livingston Agnel Livingston (2021) automated waste segregation with CNN and emphasized the importance of dataset quality for model accuracy. Sheth et al. (2010) automated sorting tasks using machine vision, a robotic arm, and a conveyor belt, highlighting speed and cost advantages. Thanawala, Sarin Verma (2020) proposed a voice-controlled robotic arm for medical waste segregation, integrating speech-to-text and waste detection modules.

Kumar, V., Wang, Q., Minghua, W., Rizwan, S., Shaikh, S. M., & Liu, X. (2018, April). Computer vision based object grasping 6DoF robotic arm using picamera. In *2018 4th International Conference on Control, Automation and Robotics (ICCAR)* (pp. 111-115). IEEE.

The paper presents a project integrating a 3D vision system with a robotic arm for object grasping and manipulation tasks. The system uses a camera and computer vision algorithms to detect objects, recognize deformations, and control spatial coordination for precise object manipulation. The system utilizes Arduino for creating an intelligent robot capable of identifying object colors and sorting them accurately, showcasing precision and flexibility in handling tasks.

Maity, S., Chakraborty, T., Pandey, R., & Sarkar, H. YOLO (YOU ONLY LOOK ONCE) ALGORITHM-BASED AUTOMATIC WASTE CLASSIFICATION SYSTEM.

The paper discusses various algorithms used for image classification, including sliding object detection, RNN, Fast RCNN, Faster RCNN, SVMs, and the YOLO algorithm. The algorithm provides real-time object detection, making it valuable in various domains like autonomous driving and surveillance systems. The paper mentions the use of Convolutional Neural Networks (CNN) in computer vision tasks. The integration of the YOLO algorithm in the waste classification system enables real-time object detection for efficient waste segregation.

2.2. Comparison table of different researches

Table 2.1

	Title	year	Objectives	Limitations	Advantages	Performance metrics	Algorithm
Reference 1	Vegetable classification using you only look	2019	The aim is to develop a faster and smarter way to identify and classify different green	The algorithm may fail to detect objects in new or modified configurations and	Faster and smarter way to detect and classify vegetables	Accuracy:61.6%	You Only Look Once (YOLO)

	once algorithm		vegetables based on shape, size, and color.	aspect ratios, as it primarily learns from the data it was trained on.	based on shape, size, and color.		
Reference 2	Real life implementation of object detection and classification using deep learning and robotic arm	2019	To automatically detect and classify different fruits using a camera and a convolutional neural network (CNN) model.	Does not discuss the scalability of the proposed system to handle a larger number of object classes.	Framework that combines the simplicity and extensibility of Arduino Uno with the power of deep learning.	Accuracy:89.2%	Convolutional Neural Networks (CNN)
Reference 3	Object sorting system using robotic arm	2013	A smart approach for a real time inspection and selection of objects in continuous flow.	Loss of pixel data upon resizing images due to raster formats.	The robotic vision system offers increased efficiency and time-saving benefits through automation.	Accuracy:72.9%	Open CV
Reference 4	Shape Detection and Classification Using OpenCV and Arduino Uno	2018	To analyze the vulnerability of smallholder farmers to climate change and assess the effectiveness of adaptation strategies in mitigating these impacts.	The study does not extensively explore the gender-specific impacts of climate change on agricultural productivity and food security, which could be a significant factor in vulnerability.	The study provides a comprehensive analysis of the impact of climate change on agricultural productivity and food security in developing countries.	-	Contours in CV
Reference 5	Computer vision-based robotic arm	2022	To sort the different objects based on color, shape and	The paper lacks detailed discussion on the		Accuracy:80%	Open cv

	for object color, shape, and size detection		size using robotic arm.	challenges faced during the experiments.	-		
Reference 6	Object sorting by robotic arm using image processing	2016	To separate faulty and correct objects using a robotic arm and image processing techniques.	External lighting conditions can introduce ambiguity in image capture, affecting the accuracy of object sorting.	The system exhibits a high extent of intellect in its automated operations, enhancing efficiency and decision-making	-	Binary Segmentation Algorithm
Reference 7	Arduino based Automatic Material Sorting Machine Using Open CV and Python 2.7 IDLE	2020	The objective of the automatic material sorting machine is to sort different kinds of materials based on their shape and color using image processing techniques.	The machine is limited to sorting materials based on their shape and color using image processing techniques. It may not be suitable for sorting materials based on other characteristics such as size or weight.	The machine has been successfully tested with different materials such as cardboard, aluminum, and pumice stone, demonstrating its ability to handle a variety of materials.	Accuracy:88.8%	Gaussian blur, median blur, and histogram equalization, Harris corner detection
Reference 8	Object Detection and Sorting by Using a Global Texture-Shape 3D Feature Descriptor	2018	The paper aims to present a global texture-shape 3D feature descriptor for object recognition and sorting tasks in robotic systems.	The classifier trained by the proposed Color-CVFH descriptor may not perform well for certain object categories, such as bottle0 and cup, which	The proposed global texture-shape 3D feature descriptor improves object recognition and sorting	Accuracy:83%	You Only Look Once (YOLO)

				share similarities in shape and color.	tasks in robotic systems.		
Reference 9	Object detection and recognition for a pick and place robot	2014	To develop image processing techniques for object sorting tasks using feature extraction and classification algorithms, with a focus on real-time testing on a SCORBOT ER-4U robotic arm platform	Loss of pixel data upon resizing images due to raster formats	Developed a robust image processing algorithm for object detection and recognition in a pick and place robot application	Accuracy:82.7%	Artificial Neural Networks (ANN)
Reference 10	Study on Cabot's Arms for Color, Shape, and Size Detection	2022	The objective is to develop a computer vision-based automated system for sorting colors, detecting shapes, and sizes, aiming to reduce labor costs and increase productivity in industries.	Cabots operate at slower speeds for safety, which may not be suitable for applications requiring high speeds, impacting cycle times.	Computer vision-based systems enhance productivity and efficiency in industries by reducing labor costs and improving accuracy.	-	Color interpolation algorithm, automatic seeded region growing and instance-based learning
Reference 11	Computer Vision-Based System for Classificatio n and Sorting Color Objects	2020	The system aims to automate the sorting process by identifying, manipulating, selecting, and sorting objects in real-time using a robotic arm with 5 degrees of freedom	Suitable lighting is required for good image capturing, which can impact the effectiveness of the image processing program	The mechatronic sorting system achieves high accuracy and low cost in the sorting process through the integration of computer	Accuracy:85%	Canny edge detection algorithm

					vision and a robotic arm		
Reference 12	Computer Vision Based Robotic Arm Controlled Using Interactive GUI	2021	The objective of the robotic vision system is to allow users to determine their desired object for the robotic arm to pick up and place into a target location.	The robotic arm system lacks a feedback mechanism for dropped objects and relies on user observation.	The robotic vision system offers increased efficiency and time-saving benefits through automation.	Accuracy: 94%	The system utilizes contouring as a computer vision technique to identify objects based on color, shape, and size
Reference 13	DESIGN AND IMPLEMENTATION OF ROBOT ASSISTED ARDUINO BASED OBJECT RECOGNITION AND SORTING	2023	The system is designed to reduce human intervention in tedious sorting tasks, enhance automation, and improve productivity in material handling processes	The system's limitations may include constraints in handling objects beyond the specified color range or shape recognition capabilities	Efficient sorting based on color coding, reducing errors and streamlining manufacturing processes	Accuracy: 92%	computer vision algorithms for precise identification and manipulation of colored objects, ensuring accurate sorting on the conveyor belt
Reference 14	Multi-class waste segregation using computer vision and robotic arm	2023	The research aims to introduce a multi-class garbage segregation system utilizing the YOLOv5 object detection model to classify dry waste categories and	The limited range in the training dataset, lacking variations in color, texture, and background, contributes to frequent incorrect classifications,	The system's speed and compactness are unique, with a bespoke YOLOv5 model connected to a 5DOF robotic arm for precise	Accuracy: 89%	YOLOv5 object detection model

			segregate them into respective bins	especially for paper.	garbage sorting.		
Reference 15	Computer Vision Based Object Grasping 6DoF Robotic Arm Using Pin camera	2018	The project aims to design a robust robotic arm capable of lifting, carrying, and unloading objects at specific locations	The vision system requires a separate computing hardware capable of processing complex vision algorithms.	The system can detect objects, mark their positions, and manipulate them towards desired locations efficiently.	Accuracy: 90%	one for red color detection and the other for identifying object positions, enabling efficient object manipulation.
Reference 16	YOLO (YOU ONLY LOOK ONCE) ALGORITHM-BASED AUTOMATIC WASTE CLASSIFICATION SYSTEM	2023	The research aims to design and implement an automated waste management system utilizing the YOLO algorithm and computer vision techniques to enhance waste sorting efficiency and accuracy.	Manual waste sorting processes are time-consuming, error-prone, and expose workers to health risks, motivating the need for automated solutions.	Potential replacement of traditional waste disposal methods by streamlining waste management and for recycling purposes.	Accuracy: 95%	YOLO (You Only Look Once)

This table provides a comparative analysis of various research papers, evaluating their objectives, methodologies (including algorithms used), strengths and weaknesses, and performance metrics.

3. PROBLEM STATEMENT

Develop an automated object sorting system using an Arduino microcontroller and a robotic arm for industrial applications, focusing on shape and size recognition.

3.1 Existing Model

Industrial object sorting systems have seen a rise in **deep learning models** for shape recognition. These models boast impressive capabilities, particularly in handling intricate shapes with variations and achieving near-perfect accuracy rates in controlled settings. Their ability to continuously adapt by retraining on new data makes them ideal for evolving sorting requirements. However, these advantages come at a cost. Deep learning models require significant computational resources, both for training and real-time operation. This makes them less suitable for low-powered systems like Arduino, often found in smaller scale industrial settings. Furthermore, developing and training deep learning models involve substantial time and expertise. The need for large, labeled datasets adds another layer of complexity. Finally, the hardware and software infrastructure required for deep learning can be expensive, potentially limiting accessibility for smaller businesses or applications with tighter budgets.

3.2 Proposed Model

This project proposes a novel approach using **contour-based recognition** with an Arduino microcontroller and robotic arm for shape sorting. Contour analysis focuses on the object's outline (contour) to determine its shape. This approach offers several advantages compared to deep learning models. Firstly, contour analysis is a simpler technique that doesn't require pre-training data. This significantly reduces development time and complexity, making it ideal for implementation on an Arduino platform. Additionally, contour analysis is computationally efficient, allowing for real-time object processing on the Arduino, leading to faster sorting speeds. This translates to a more cost-effective solution - perfect for smaller industries or applications with limited budgets where affordability is a key concern. However, it's important to note that contour-based recognition works best for objects with well-defined shapes (cubes, cylinders,

spheres) and in controlled environments with consistent lighting and minimal background clutter. In these specific scenarios, contour analysis offers a reliable and efficient alternative to deep learning models for object sorting tasks.

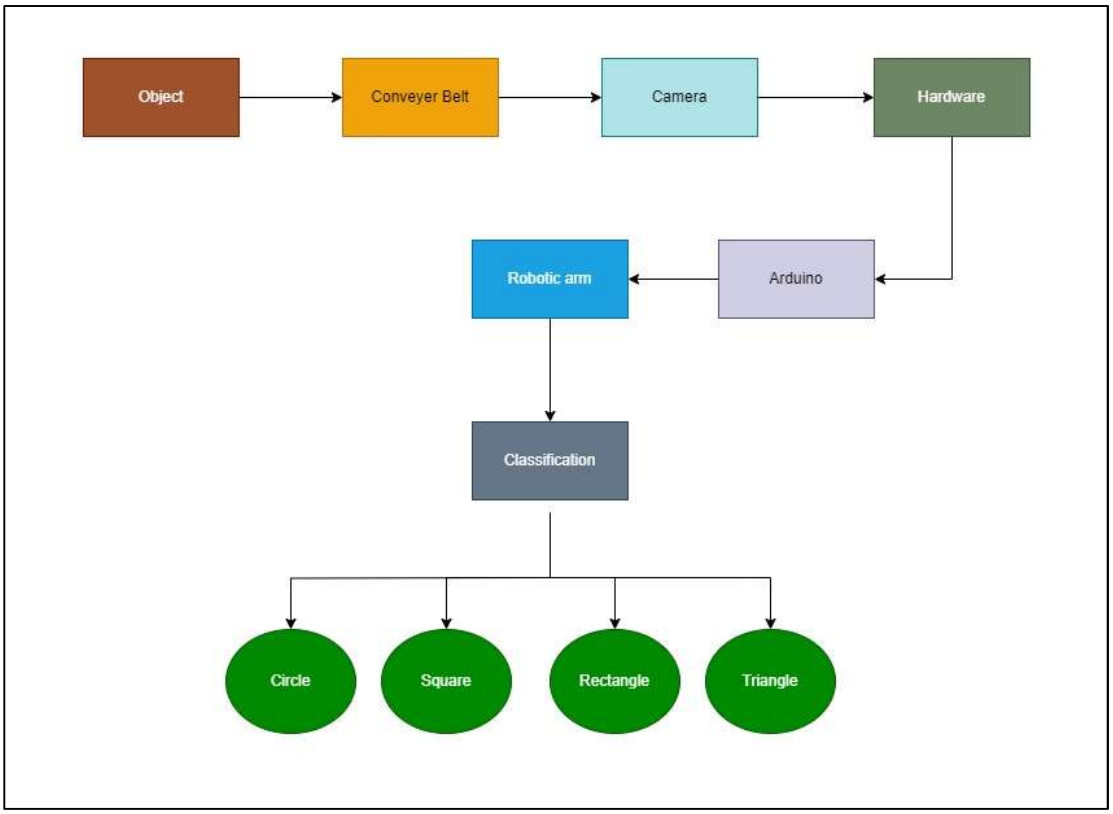

Fig 3.2.1. : Architecture of the proposed model.

The above figure is a block diagram that outlines the architecture of an object sorting system based on shape and size. It uses a conveyor belt to transport objects past two cameras for image capture. Then, a computer with OpenCV software analyzes the captured frames to classify objects by shape (triangle, rectangle, square, circle) and size.

4. METHODOLOGY & DESIGN

System Overview

1. Conveyor Belt System

Purpose: The conveyor belt system moves objects under the camera for shape detection.

Components:

- **Conveyor Belt:** A continuous loop that transports objects.
- **Motor:** Drives the conveyor belt, ensuring continuous movement.

2. Camera and Computer Vision System

Purpose: To capture images of objects on the conveyor belt and process these images to detect the shape of the objects.

Components:

- **Camera:** Positioned above the conveyor belt to capture images of the objects.
- **Computer or Raspberry Pi:** Runs image processing algorithms.

Image Processing Steps:

1. **Image Capture:** The camera captures images.
2. **Preprocessing:** The image is converted to grayscale and blurred to reduce noise.
3. **Edge Detection:** Canny edge detection algorithm is applied to identify object boundaries.
4. **Shape Detection:** Contours are detected and analyzed to determine the shape of the object (e.g., triangle, square, circle).

3. Communication Interface

Purpose: To transfer the detected shape information from the computer vision system to the Arduino.

Components:

- **Serial Communication (e.g., pySerial):** Used to send data from the computer (running the vision algorithms) to the Arduino. The computer encodes the shape information and transmits it via a serial connection.

4. Arduino and Robotic Arm

Purpose: To classify the objects by controlling the robotic arm based on the received shape information.

Components:

- **Arduino Board:** Receives shape data from the computer and controls the robotic arm.
- **Robotic Arm:** Moves to predefined positions to classify the objects based on shape.

Arduino Program:

1. **Serial Read:** Arduino reads the shape data from the serial connection.
2. **Shape-Based Classification:** Arduino interprets the shape data and moves the robotic arm to classify the object. Different shapes correspond to different positions or actions of the robotic arm.
3. **Servo Control:** The robotic arm, typically controlled by servos, is programmed to move to specific angles to place objects into designated bins or areas based on their shape.

Integration and Workflow

1. **Object Movement:** Objects move along the conveyor belt.
2. **Image Capturing:** The camera captures the image.
3. **Image Processing:** The captured image is processed to detect the shape.
4. **Data Transfer:** Detected shape information is sent to the Arduino via serial communication.

5. **Classification:** The Arduino controls the robotic arm to classify the object based on the received shape information.

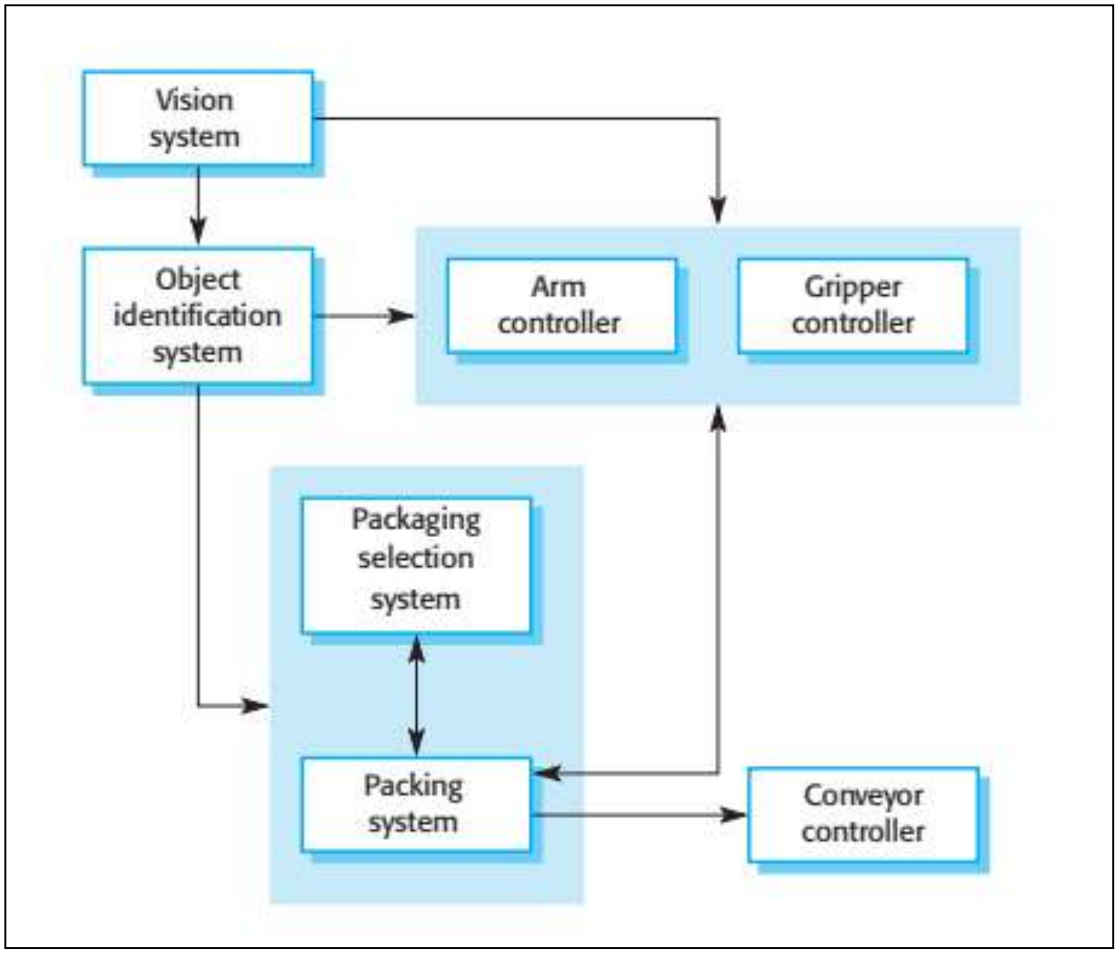

Fig 4.1 : The Architecture of a packing robot control system.

The above figure is a block diagram outlines the workflow of a packing robot control system, where components like vision system, object identification, and gripper controller collaborate to automate the packing process.

5. REQUIREMENTS SPECIFIED

5.1. SOFTWARE

OpenCV (cv2) Library

- **Image Processing:** OpenCV (cv2) is a powerful library that provides extensive functions for image processing. In your code, you leverage OpenCV's functionalities for various tasks:
    - **Color Conversion (cv2.cvtColor):** This function converts the captured image from BGR (Blue, Green, Red) color format, which is how cameras typically capture color information, to grayscale for easier shape analysis.
    - **Blurring (cv2.GaussianBlur):** This function applies a Gaussian blur to the grayscale image, which helps reduce noise and improve edge detection in the next step.
    - **Edge Detection (cv2.Canny):** This function identifies edges within the image. Edges are helpful for contour detection, which is crucial for isolating objects in the frame.
    - **Contour Detection (cv2.findContours):** This function finds and retrieves the boundaries of objects (shapes) within the image.
- **Shape Classification:** OpenCV doesn't have built-in functions for shape classification, but you've implemented your own logic within the classify_objects function using:
    - **Contour Area and Perimeter:** These measurements are used to differentiate between large and small objects.
    - **Shape Approximation (cv2.approxPolyDP):** This function simplifies the contour by approximating it with fewer vertices while preserving its overall shape. The number of vertices (corners) is then used to classify triangles and quadrilaterals.

- **Bounding Rect (cv2.boundingRect):** This function retrieves a rectangle that encompasses the detected contour. You use the rectangle's width and height to calculate the aspect ratio for square/rectangle differentiation.
- **Circularity Calculation:** This calculation estimates how closely a shape resembles a circle. It's helpful in distinguishing circles from other shapes.

Python Programming Language

- **Scripting Language:** Python is a versatile scripting language known for its readability and beginner-friendliness. It allows you to write concise and well-structured code for controlling the image processing pipeline and interacting with hardware like cameras.
- **OpenCV Integration:** Python integrates seamlessly with OpenCV libraries, making it a popular choice for computer vision projects. You can import OpenCV using import cv2 and then access its functionalities throughout your code.

1) **Video Streaming Libraries:** If your camera setup involves streaming video over a network (IP camera) instead of direct USB capture, you might need additional libraries like v4l2-python or picamera to handle the video stream. These libraries provide functionalities to open and access video streams from network cameras.
2) **Serial Communication Library (pySerial):** If you intend to connect your Python program to an Arduino to control the conveyor belt or sorting mechanisms based on the detected object's shape and size, you'll need a serial communication library like pySerial. This library enables communication between Python and Arduino by establishing a serial connection over USB or other communication ports.

5.2.HARDWARE

Processing Unit

- **Computer:** The computer acts as the brain of your object sorting system. It runs the Python code with OpenCV libraries to process the video input from the cameras,

perform shape and size classification, and potentially control the sorting mechanism (if implemented).

- **Processing Power:** The processing power required depends on the complexity of your project. Here's a breakdown:
  - **Basic Setup:** For a simple system with a low-resolution camera (e.g., 640x480) and low frame rate (e.g., 10 frames per second), a low-power computer like a Raspberry Pi can handle the processing tasks.
  - **More Demanding Setup:** If you're using higher resolution cameras (e.g., 1920x1080), higher frame rates (e.g., 30 frames per second), or plan to implement more complex image processing algorithms, a computer with a dedicated graphics processing unit (GPU) is recommended. GPUs can significantly accelerate image processing tasks compared to a CPU alone.

Input Devices

- **Cameras (2):** You've configured your code to use two cameras (cap1 and cap2). These cameras capture video footage (individual frames) of the objects on the conveyor belt.
  - **Resolution and Frame Rate:** The resolution (number of pixels) and frame rate (number of frames captured per second) of the cameras impact the processing power required and the amount of data your program needs to handle. For a basic setup, standard webcams with a resolution of 640x480 and a frame rate of 30 fps might be sufficient. However, if you need to capture finer details or objects move quickly on the conveyor belt, you might need higher resolution cameras with faster frame rates.
  - **Lighting:** Proper and consistent lighting is essential for accurate image processing. Depending on your environment, you might need additional lighting fixtures to ensure the objects on the conveyor belt are well-lit and have good contrast for the cameras to capture clear images.

Output and Control

- **Conveyor Belt:** A conveyor belt is a motorized belt that continuously transports the objects past the cameras for image capture and subsequent sorting based on their shape and size. The conveyor belt usually requires a separate motor and power supply for operation. The speed of the conveyor belt should be calibrated to ensure objects are presented to the cameras clearly and consistently for analysis.
- **Sorting Mechanism:** For a fully automated sorting system, you might implement a mechanism controlled by the computer to separate objects based on their classified shapes and sizes. This mechanism could involve various components depending on your design:
    - **Gates:** Doors or dividers that open and close to direct objects to different chutes or bins based on their classification.
    - **Diverters:** Moving sections of the conveyor belt that can be controlled to divert objects to different paths based on their properties.
    - **Other Mechanisms:** Depending on the object types and sorting requirements, you might explore other creative mechanisms like air jets or robotic arms for more complex sorting tasks.
- **Arduino:** If you want to integrate an Arduino into your system, it acts as a microcontroller that can be programmed to control the conveyor belt speed, trigger the sorting mechanism based on signals from the computer program, and potentially interact with additional sensors or actuators. You'll need the following additional components for Arduino integration:
    - **Arduino Board:** The physical Arduino board itself, which is a programmable microcontroller.
    - **Sensors:** Depending on your sorting mechanism design, you might use sensors like limit switches to detect object presence or photoelectric sensors to differentiate object colors (if color is also a sorting criteria).
    - **Actuators:** These are components controlled by the Arduino to perform actions based on the computer program's instructions and sensor data. In

your case, the actuators could be the motor controlling the conveyor belt speed or solenoids controlling gates in the sorting mechanism.

- **Wires and Connections:** Electrical wires and connectors are needed to establish communication between the Arduino board, the computer, sensors, and actuators.

5.3. IOT

Hardware Considerations

- **Connectivity Module:** To connect your system to a network, you'll likely need an additional hardware module like a Wi-Fi shield or an Ethernet shield for the Arduino (if you're using one). These modules enable the Arduino to communicate over a network.
- **Cloud Storage (Optional):** If you plan to store and access data about the sorted objects (e.g., number of objects of each shape sorted per hour), you might consider cloud storage services.

Software Considerations

- **Networking Library:** The Arduino programming environment might require additional libraries to handle network communication protocols like Wi-Fi or Ethernet (depending on your chosen connectivity module).
- **Cloud API (Optional):** If you're using cloud storage, you'll need to interact with the cloud service's API (Application Programming Interface) using libraries or code specific to that cloud platform.

6. IMPLEMENTATION & CODE

Project Implementation Steps

Setup Conveyor Belt

1. **Assemble Conveyor Belt**:
    - Ensure all mechanical parts of the conveyor belt are securely fastened and aligned.
    - Check the tension of the belt to ensure smooth operation without slipping or stalling.
2. **Install Motor and Controller**:
    - Connect the motor to a suitable power source and controller to drive the conveyor belt.
    - Test the motor and adjust settings to ensure smooth and consistent movement of the belt.

Mount Camera and Configure Image Capture

1. **Mount the Camera**:
    - Securely mount the camera above the conveyor belt at an appropriate height and angle to capture clear images of the objects passing underneath.
2. **Lighting Setup**:
    - Set up consistent and adequate lighting to avoid shadows and reflections on the objects.
    - Use diffused lighting if possible to ensure even illumination across the conveyor belt.
3. **Calibrate Camera**:
    - Use a calibration pattern, such as a checkerboard, to adjust the camera settings for clear, distortion-free images.
    - Ensure the camera is properly focused and aligned to capture the entire width of the conveyor belt.

Develop and Test Computer Vision Code

1. **Install Required Libraries**:
   - Install OpenCV and any other necessary libraries in your Python environment.

     - import cv2

2. **Write Image Capture Code**:
   - Develop code to capture images from the camera.

     - cap = cv2.VideoCapture(0)
     - link = http://192.168.55.107:8080/video
     - cap.open(link)

3. **Shape Detection Code**:
   - Develop code to preprocess images, detect contours, and classify shapes.

     - def classify_objects(contour,current):

```
area = cv2.contourArea(contour)

perimeter = cv2.arcLength(contour, True)

approx = cv2.approxPolyDP(contour, 0.04 * perimeter, True)

vertices = len(approx)

if area > 1000:

    if vertices == 3:

        return "Triangle"

    elif vertices == 4:

        x, y, w, h = cv2.boundingRect(contour)
```

```
                aspect_ratio = float(w) / h
                if 0.95 <= aspect_ratio <= 1.05:
                    return "Square"
                else:
                    return "Rectangle"
            else:
                circularity = 4 * 3.1415 * area / (perimeter * perimeter)
                if circularity >= 0.7:
                    return "Circle"
        return current
```

4. **Test and Validate**:
    - Test the vision code with different objects to ensure accurate shape detection.

```
➢ while True:
    ret, frame = cap.read()
    if not ret:
        break
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    blurred = cv2.GaussianBlur(gray, (5, 5), 0)
    edged = cv2.Canny(blurred, 50, 150)
    contours,_=cv2.findContours(edged.copy(),cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```

```
        for c in contours:
            shape = detect_shape(c)
            M = cv2.moments(c)
            if M['m00'] > 0:
                cX = int((M['m10'] / M['m00']))
                cY = int((M['m01'] / M['m00']))
                cv2.drawContours(frame, [c], -1, (0, 255, 0), 2)
                cv2.putText(frame, shape, (cX, cY), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2)
        cv2.imshow('Frame', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    cap.release()
    cv2.destroyAllWindows()
```

Setup Serial Communication

1. **Establish Serial Communication**:
   - Connect the computer to the Arduino via a USB cable.
   - Use the pyserial library in Python for serial communication.
     - import serial import time arduino = serial.Serial('COM3', 9600)
     - time.sleep(2)
2. **Send Data to Arduino**:
   - Modify the vision code to send shape data to the Arduino.
     - arduino.write(shape.encode())

3. **Test Communication**:
    - Write a simple Arduino sketch to read and print serial data.

```
void setup() {
    Serial.begin(9600);
    }
    void loop() {
    if (Serial.available() > 0) {
        String shape = Serial.readStringUntil('\n');
        Serial.println(shape);
    }
     }
```

Program Arduino and Test Robotic Arm

1. **Write Arduino Code**:
    - Develop code to control the robotic arm based on received shape data.

```
#include <Servo.h>
Servo servo1;
Servo servo2;
Servo servo3;
void setup() {
  Serial.begin(9600);
  servo1.attach(9);
```

```
    servo2.attach(10);
    servo3.attach(11);
}
void loop() {
    if (Serial.available() > 0) {
        String shape = Serial.readStringUntil('\n');
        if (shape == "triangle") {
            moveToTriangleBin();
        } else if (shape == "rectangle") {
            moveToRectangleBin();
        } else if (shape == "circle") {
            moveToCircleBin();
        }
    }
}
void moveToTriangleBin() {
    servo1.write(90);
    delay(1000);
    servo2.write(45);
```

```
    delay(1000);
    servo3.write(10);
    delay(1000);
}
void moveToRectangleBin() {
    servo1.write(120);
    delay(1000);
    servo2.write(60);
    delay(1000);
    servo3.write(10);
    delay(1000);
}
void moveToCircleBin() {
    servo1.write(150);
    delay(1000);
    servo2.write(75);
    delay(1000);
    servo3.write(10);
    delay(1000);
```

}

2. **Test Robotic Arm Movements**:
    - Test the robotic arm movements to ensure they are accurate and smooth.
    - Calibrate as necessary to ensure precise positioning.

Integration

1. **Combine All Components**:
    - Run the conveyor belt, vision system, and robotic arm together.
    - Ensure the camera captures images as objects pass, the vision system detects shapes, and the Arduino receives and acts on the shape data.
2. **Test End-to-End System**:
    - Place various objects on the conveyor belt and observe the entire process.
    - Make adjustments to timing, positioning, and code to ensure smooth operation.
3. **Fine-Tuning**:
    - Adjust lighting, camera settings, vision parameters, and robotic arm calibration for optimal performance.
    - Ensure consistent and reliable operation through multiple test runs.

7. RESULTS

This section discusses the system designed for this project; it shows the results acquired by testing the system; analyze the results to contextualize them. The completed system consists of the robotic arm, the user interface, and the frame, consisting of the camera mount and the workspace for the robotic arm.Hardware part of the system is shown in Fig 7.1

Fig 7.1 : Hardware of the System

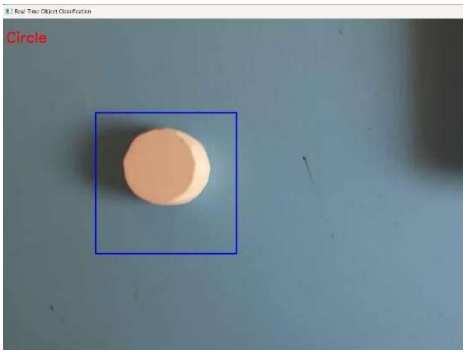

Fig 7.2 : Classification of Circle

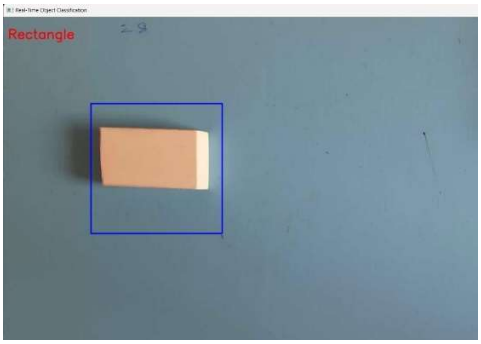

Fig 7.3 : Classification of Rectangle

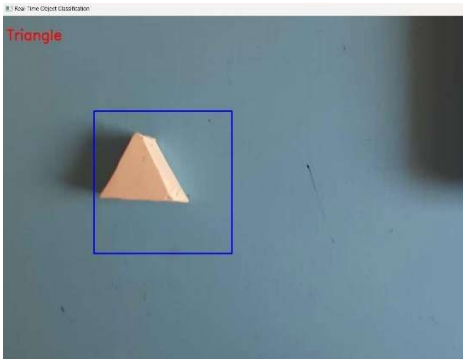

Fig 7.4 : Classification of Triangle

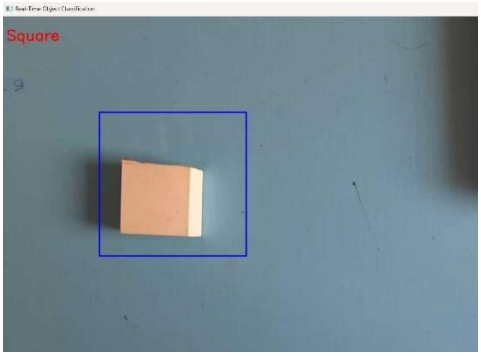

Fig 7.5 : Classification of Square

The above four images describes the Classification of different shapes like Square,Rectangle,Circle,Triangle.

CONCLUSION:

This project provides a foundation for developing a practical, cost-effective object sorting system suitable for specific needs in various industries. By addressing the limitations and exploring further advancements, this approach has the potential to streamline sorting processes, improve efficiency, and minimize human error in warehouse operations and other applications. This project's impact extends beyond developing a single object sorting system. The successful implementation of contour-based recognition with Arduino opens doors for several advancements in various industries:

- **Accessibility for Small Businesses:** By demonstrating the viability of a low-cost and easy-to-implement sorting solution using Arduino, this project empowers smaller businesses and startups to automate sorting tasks that were previously cost-prohibitive with traditional deep learning methods. This can lead to increased efficiency and productivity in their warehouse operations.
- **Streamlining Workflows:** The real-time processing capabilities of the system enable faster sorting speeds, potentially reducing bottlenecks and expediting delivery times in warehouses and fulfillment centers. This translates to improved customer satisfaction and potentially lower operational costs.
- **Reduced Human Error:** Automating sorting tasks minimizes human error associated with manual sorting, leading to improved accuracy and consistency. This can be particularly beneficial in industries with strict quality control requirements.
- **Foundation for Future Advancements:** The project serves as a stepping stone for further development in object sorting technologies. By exploring the integration of additional sensors and machine learning techniques, the system's capabilities can be expanded to handle a wider range of objects and complexities, paving the way for even more sophisticated sorting solutions.

REFERENCES

Journal Article referencing:

[1] Sachin, C., Manasa, N., Sharma, V., & AA, N. K. (2019, November). Vegetable classification using you only look once algorithm. In *2019 International Conference on Cutting-edge Technologies in Engineering (ICon-CuTE)* (pp. 101-107). IEEE.

[2] Kakde, Y., Bothe, N., & Paul, A. (2019). Real life implementation of object detection and classification using deep learning and robotic arm. *Proceedings of Recent Advances in Interdisciplinary Trends in Engineering & Applications (RAITEA)*.

[3] Kale, V. R., & Kulkarni, V. A. (2013). Object sorting system using robotic arm. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, *2*(7), 3400-3407.

[4] Poda, X., & Qirici, O. (2018). Shape Detection and Classification Using OpenCV and Arduino Uno. *RTA-CSIT*, *2280*, 128-136.

[5] Abdullah-Al-Noman, M., Eva, A. N., Yeahyea, T. B., & Khan, R. (2022). Computer vision-based robotic arm for object color, shape, and size detection. *Journal of Robotics and Control (JRC)*, *3*(2), 180-186.

[6] Singh, T., Dhaytadak, D., Kadam, P., & Sapkal, R. J. (2016). Object sorting by robotic arm using image processing. *International Research Journal of Engineering and Technology (IRJET)*, *3*(4), 2395-0056.

[7] Sriramguru, A., Vishwa, A., Yogeshwaran, D., & Yogessh, B. S. (2020). Arduino based Automatic Material Sorting Machine Using Open CV and Python 2.7 IDLE.

[8] Fan, Z., Li, Z., & Li, W. (2018). Object Detection and Sorting by Using a Global Texture-Shape 3D Feature Descriptor. *arXiv preprint arXiv:1802.01116*.

[9] Kumar, R., Lal, S., Kumar, S., & Chand, P. (2014, November). Object detection and recognition for a pick and place robot. In *Asia-Pacific world congress on computer science and engineering* (pp. 1-7). IEEE.

[10] Baballe, M. A., Bello, M. I., Abdulkadir, Z., & Garbadiso, G. B. (2022). Study on Cabot's Arms for Color, Shape, and Size Detection. *Global Journal of Research in Engineering & Computer Sciences*, *2*(2), 48-52.

[11] Fadhil, A. T., Abbar, K. A., & Qusay, A. M. (2020, February). Computer Vision-Based System for Classification and Sorting Color Objects. In *IOP Conference Series: Materials Science and Engineering* (Vol. 745, No. 1, p. 012030). IOP Publishing.

[12] Intisar, M., Khan, M. M., Islam, M. R., & Masud, M. (2021). Computer Vision Based Robotic Arm Controlled Using Interactive GUI. *Intelligent Automation & Soft Computing*, *27*(2).

[13] SNEHA, S., & NAYANA, R. DESIGN AND IMPLEMENTATION OF ROBOT ASSISTED ARDUINO BASED OBJECT RECOGNITION AND SORTING.

[14] Lahoti, J., Sn, J., Krishna, M. V., Prasad, M., Rajeshwari, B. S., Mysore, N., & Nayak, J. S. (2024). Multi-class waste segregation using computer vision and robotic arm. PeerJ Computer Science, 10, e1957.

[15] Kumar, V., Wang, Q., Minghua, W., Rizwan, S., Shaikh, S. M., & Liu, X. (2018, April). Computer vision based object grasping 6DoF robotic arm using picamera. In 2018 4th International Conference on Control, Automation and Robotics (ICCAR) (pp. 111-115). IEEE.

[16] Maity, S., Chakraborty, T., Pandey, R., & Sarkar, H. YOLO (YOU ONLY LOOK ONCE) ALGORITHM-BASED AUTOMATIC WASTE CLASSIFICATION SYSTEM.

Book referencing:

[1] **Computer Vision: Algorithms and Applications** by Richard Szeliski (2011)

[2] **Arduino Robotics** by Laurent Schwartz (2015)

Referencing of an Article:

[1] **Real-Time Object Detection and Classification with OpenCV**
https://medium.com/@chen-yu/real-time-object-tracking-and-classification-with-opencv-and-densenet-43d39f875096

[2] **Shape Feature Extraction and Classification Using OpenCV**
https://domino.ai/blog/feature-extraction-and-image-classification-using-deep-neural-networks

[3] **Building an Automated Sorting System with Arduino and OpenCV**
https://randomnerdtutorials.com/